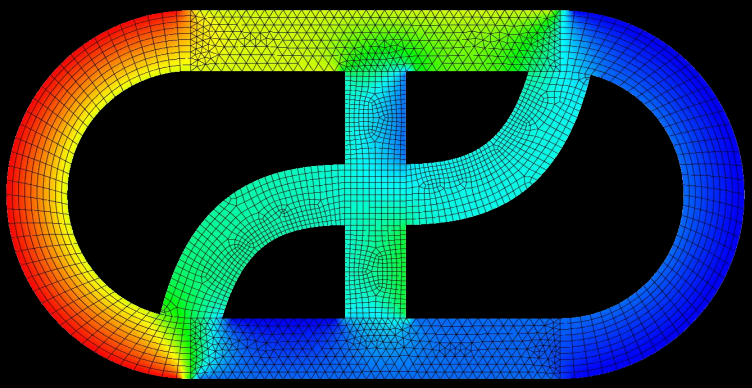


CFD EXPERTS

Simulate the Future

WWW.CFDEXPERTS.NET

Ansys 2021 R2
Engineering What's Ahead.

Ansys Workbench Scripting Guide

Ansys
ANSYS, Inc.
Southpointe
2600 Ansys Drive
Canonsburg, PA 15317
ansysinfo@ansys.com
http://www.ansys.com
(T) 724-746-3304
(F) 724-514-9494

Release 2021 R2
July 2021

ANSYS, Inc. and Ansys Europe, Ltd. are UL registered ISO 9001: 2015 companies.

Copyright and Trademark Information

Disclaimer Notice

U.S. Government Rights

Third-Party Software

Published in the U.S.A.

Table of Contents

List of Figures

List of Tables

Scripting Overview

Ansys Workbench offers the ability to record the actions you perform via the GUI, which is referred to as *journaling*. Journals are recorded as Python-based scripts. You can modify these scripts or create new ones, which is referred to as *scripting*. Together, these capabilities allow you to quickly and easily replay analyses you've already run via recorded journals, as well as to extend functionality, automate repetitive analyses, and run analyses in batch mode via scripting.

Note:

- For the purposes of this documentation, Ansys Workbench refers to all applications and tools running in the Workbench environment, including the Project tab and Parameter Manager. If features apply only to specific applications or tools, this is indicated.
- Journaling and scripting uses a decimal point as the decimal separator and a comma as the separator for list items regardless of locale settings. The GUI will display values and lists using the current locale settings.

Journaling and Scripting Capabilities

Most actions performed via the GUI are journaled. Some examples of actions that are not journaled include:

- GUI-only actions, such as:
  - Interrupting a solve operation
  - Launching help (including Quick Help and Sidebar Help)
  - Running the **View Solver Output** option from VistaTF's **Solution** cell
- Actions taken in some data-integrated applications as described in Scripting and Data-Integrated Applications (p. 7)
- Some graphics scene actions, such as hide body and rotation.

Journaling

A journal is a record of all operations that have modified data during your session. Based on your preference settings, a journal of your full session can be automatically saved to a location that you specify. For more information, see Setting Journaling Preferences (p. 2). You can also choose to record part of a session to a journal file, capturing a specific set of actions. Playing back the journal will re-create the recorded actions exactly. Journaling and scripting tools (including recording and playback) are available through the **File > Scripting** menu in Ansys Workbench.

Journaled sessions can be used to restore work after a crash. Journals are platform-independent and portable, subject to file location consistency between accounts. For information on file path handling within journals and scripts, see File Path Handling in Ansys Workbench (p. 15).

Setting Journaling Preferences

You can set journaling preferences such as automatically producing a journal for every project, the default directory where journals are to be written, and how long to keep a journal file:

1. In Ansys Workbench, select **Tools > Options > Journals and Logs**.
2. To have Ansys Workbench automatically write journal files, select **Record Journal Files**.
3. Specify the default location where journal files are to be written.

   This is the location that the file browser will open in automatically when you choose to begin recording a journal. You will still be able to browse to a different location before saving a particular journal.
4. Specify the number of days to keep a journal file.
5. Specify how long (in seconds) to pause between each command when running a journal file.
6. Click **OK** to save your settings.

Recording a Journal Manually

You can record a journal manually:

1. Launch Ansys Workbench.
2. Select **File > Scripting > Record Journal**.
3. Specify the name and location of the journal file and click **Save**.
4. Use the GUI to work through your analysis as you normally would.
5. Select **File > Scripting > Stop Recording Journal**.

   A message appears informing you that you will stop recording.
6. Click **OK**.

Using the Command Window

You can use the command window to invoke commands, access data entity properties, and invoke data entity and data container methods interactively, one at a time:

1. Select **File > Scripting > Open Command Window**.
2. Enter the commands you want to run, one at a time.

As you enter each command, the appropriate action will occur in the Ansys Workbench database and, if applicable, in the GUI.

Command Window Usage

While recording a journal, Ansys Workbench creates a number of variables for the object references that contain the data in your project. For example, consider the following lines from a journal:

```
template1 = GetTemplate(TemplateName="Thermal")
system1 = template1.CreateSystem()
```

In this journal, *template1* and *system1* are the variables for the references to the associated data objects. The variables are used within the journal to access the properties and methods of the objects. These variables are created and recorded specifically for replaying the journal, and they are not immediately accessible from within the command window. However, when working in the command window, you may want to use these variables. Doing so can aid in manually examining the details of your project or assist in creating scripts based on the journal. To use these variables, execute the command ImportJournalVariables (p. 1324) in the command window to make the variable definitions from the currently recorded journal available in the command window. You should be aware of the following points when using the `ImportJournalVariables()` command:

- The variable definitions are based on those in the currently recorded journal. By default, this journal is the automatically recorded journal controlled by user preferences. For more information, see Setting Journaling Preferences (p. 2). If you have manually started a journal recording of part of your session, the variable definitions are taken from the manually recorded journal. For more information, see Recording a Journal Manually (p. 2).
- If you have any manually defined variables of the same name as any journal variables, your variables will be overwritten by the journal variables.
- Changing the definition of a journal variable in the command window after executing `ImportJournalVariables()` does not affect the definition of the variable in the currently recorded journal.
- The `ImportJournalVariables()` command can be executed multiple times in a session and will update the variables based on the currently recorded journal.

Command Window Navigation

The command window uses the Python programming language to interpret and invoke commands or other operations. In addition, you can use numerous keyboard shortcuts to facilitate your window interaction.

Text Cursor Keyboard Keys

When typing a command or statement, the following special keys are available for moving the text cursor:

Key	Action
Left Arrow	Moves the cursor back one character
Right Arrow	Moves the cursor forward one character
Ctrl + Left Arrow	Moves the cursor back one word

Key	Action
Ctrl + Right Arrow	Moves the cursor forward one word
Home	Moves the cursor to the beginning of the line
End	Moves the cursor to the end of the line

Copy and Paste Keyboard Keys

You can copy text from the command window to the clipboard or paste text from the clipboard as input to the command window. The following keys allow you to copy and paste text:

Key	Action
Ctrl+**C**/**Ctrl**+**Insert**	Copies selected text from the command window to the clipboard. Text copied from the command window is first selected (highlighted) using the mouse.
Ctrl+**V**/**Shift**+**Insert**	Pastes the text found on the clipboard into the input area of the command window. If multiple lines of text are pasted, the lines must be one or more complete Python statements.

Command History

The command window maintains a history of commands or statements that you enter so you can easily recall a previously entered command or statement and invoke it again without re-typing it. You could also make some modifications to it before invoking it again.

The following keys allow you to access the command history:

Key	Action
Up Arrow/**Page Up**	Recalls the previously entered command, and the command before that if the key is pressed again
Down Arrow/**Page Down**	Recalls the next command in the history list

Command Completion

The command window provides a command-completion (tab-completion) feature to automatically complete partially typed variables and commands to save tedious typing.

Type one or more characters and press the **Tab** key once or multiple times to see the defined variables and commands that have names beginning with the characters you typed. Entering an object variable name with the dot (.) and then pressing the **Tab** key will cycle through the defined properties and methods for that object. Entering one or more characters after the dot will restrict the completion results to just those properties and methods that start with those characters.

The following keys allow you to access the command completion:

Key	Action
Tab	Completes the current text on the command line with any variable, property, command, or method name that is valid in the current context. Press **Tab** repeatedly to cycle forward through possible completions, if available.
Shift+**Tab**	Same as **Tab** but cycles backwards through possible completions.

Scripting

A script is a set of instructions to be issued to Ansys Workbench. The script can be a modified journal, or it can be a completely new set of instructions that you write directly.

Ansys Workbench uses an object-based approach. For scripting, some knowledge of object oriented programming and the Python language is advantageous. For more information on using scripting, see Using Scripting in Ansys Workbench (p. 13).

Ansys Workbench scripting is based on IronPython 2.6. Before attempting to write or modify Ansys Workbench scripts, you should be familiar with using this version of Python.

IronPython is well integrated with the rest of the .NET Framework (on Windows) and Mono CLR (on Linux) and makes all related libraries easily available to Python programmers while maintaining compatibility with the Python language. For more information on IronPython, see http://ironpython.net/.

IronPython is generally compatible with existing standard Python scripts. However, not all C-based Python library modules are available under IronPython, as discussed on the IronPython website.

For more information on Python, including a standard language reference, see http://www.python.org/.

For a complete list of Ansys-published data containers (p. 37), namespaced commands (p. 1277), and data types (p. 1353), see the reference material later in this document.

Scripting Backwards Compatibility

If you want to run a script that was created for a previous version of Ansys Workbench, you must insert a SetScriptVersion (p. 1332) command at the beginning of the script file indicating the version for which the script was initially created. In addition, if you copy commands from an older script and past them into thecommand window, you must enter the `SetScriptVersion` command before pasting the script commands into the window. As long as you use the `SetScriptVersion` command, any scripts created for previous versions should run correctly, except for those issues listed in Known Issues and Limitations (p. 35)

Note:

When you run an older script, the original script file will not be changed. However, if you paste commands from an older script into the command window and you are journaling your session, any outdated commands in the journal are replaced by the updated commands.

Command Line Execution of Ansys Workbench

Ansys Workbench can be executed from the operating system command line and accepts a number of command line arguments to facilitate automation and the replay of scripts. The following command can be used to run Ansys Workbench from the command line:

```
ANSYS_INSTALL_PATH/v212/Framework/bin/PLATFORM/runwb2
```

Where *PLATFORM* is one of the following (as applicable to the application or tool that you are using):

Win64

Linux64

For example, to run Ansys Workbench from the default installation location on a Windows 64-bit system, the command would be:

```
C:\Program Files\ANSYS Inc\V212\Framework\bin\win64\runwb2
```

The following table describes the command line arguments that can be used to control Ansys Workbench file operations and execution behavior at start-up.

Argument	Operation
-B	Run Ansys Workbench in batch mode. In this mode, the user interface is not displayed and a console window is opened. The functionality of the console window is the same as the Ansys Workbench command window.
-R *WorkbenchScriptFile*	Replay the specified Ansys Workbench script file on start-up. If specified in conjunction with –B, Ansys Workbench will start in batch mode, execute the specified script, and shut down at the completion of script execution.
-I	Run Ansys Workbench in interactive mode. This is typically the default, but if specified in conjunction with –B, both the user interface and console window are opened.
-X	Run Ansys Workbench interactively and then exit upon completion of script execution. Typically used in conjunction with –R.
-F *WorkbenchProjectFile*	Load the specified Ansys Workbench project file on start-up.
-E *command*	Execute the specified Ansys Workbench scripting command on start-up. You can issue multiple commands, separated with a semicolon (;), or specify this argument multiple times and the commands will be executed in order.

Console Window

The console window is the same as the command window but is present when running in batch mode to provide a way of working directly with commands outside of the user interface.

Playing a Journal or Script

You can play a journal or script interactively:

1. Select **File** > **Scripting** > **Run Script File**.

2. Select the journal or script file to be played back and click **Open**.

The recorded actions will be performed.

To use the command line to play a journal or script, see Command Line Execution of Ansys Workbench (p. 6).

Scripting and Data-Integrated Applications

From the **Project Schematic**, you can interact with applications that are native to Ansys Workbench (called workspaces), and you can launch applications that are data-integrated. Native workspaces are built entirely on the Ansys Workbench framework and can use all of its services. Examples of native workspaces include the Project Schematic, Engineering Data, and Design Exploration.

Data-integrated applications are created independently of the Ansys Workbench framework but have been extended so they can be driven by the **Project Schematic** and share key data and parameters with Ansys Workbench and Ansys Workbench-enabled applications. Data-integrated applications include the Mechanical APDL application, Ansys Fluent, Ansys CFX, DesignModeler, and the Mechanical application.

The difference between native workspaces and data-integrated applications is an important consideration for Ansys Workbench journaling and scripting. All operations that modify the data associated with a native workspace are journaled and can be fully automated with Ansys Workbench scripting. For data-integrated applications, only those operations initiated from the **Project Schematic** are journaled. Such operations include system updates and data transfers. Operations performed within data-integrated application are not necessarily journaled or controlled by Ansys Workbench scripting. For example, steps to construct geometry in Mechanical APDL or solution methods in Ansys Fluent are not journaled.

Although data-integrated applications do not fully support Ansys Workbench scripting, many of them have their own native scripting language which is accessible through the Ansys Workbench scripting interface. For more information, see Table 1: Scripting Support for Data-Integrated Applications (p. 8). For example, Mechanical APDL is based on the powerful Ansys Parametric Design Language (APDL), and APDL commands can be directly incorporated within an Ansys Workbench script.

You use `SendCommand` to pass native scripting commands to data-integrated applications. You can insert `SendCommand` calls into your Ansys Workbench scripts to drive data-integrated applications. However, data-integrated applications do not necessarily record operations in the Ansys Workbench journal. Most scriptable data-integrated applications have an independent journal where native commands are recorded.

In Table 1: Scripting Support for Data-Integrated Applications (p. 8), a **Yes** in the **Supports Scripting with SendCommand** column indicates that the data-integrated application can be driven from an Ansys Workbench script by manually inserting `SendCommand` calls.

A **Yes** in the **Supports Journaling with SendCommand** column indicates that the data-integrated application records its operations with `SendCommand` and that the state of the application can be restored from the Ansys Workbench journal itself. To learn more about `SendCommand`, see Mechan-

ical APDL and Sending Commands to Integrated Applications (p. 27) and the detailed description of the method in Part , Data Containers (p. 37).

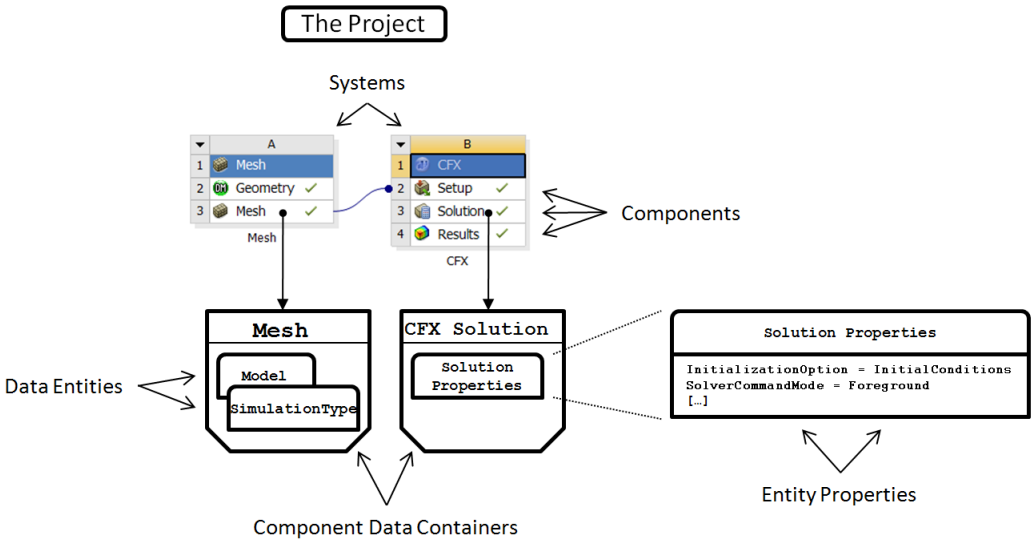
Table 1: Scripting Support for Data-Integrated Applications

Data-Integrated Applications	Native Scripting Language	Supports Scripting with SendCommand	Supports Journaling with SendCommand
Mechanical APDL	APDL	Yes	
Mechanical	JScript	Yes	
CFX	CCL	Yes	Yes
Fluent	Scheme	Yes	Yes
Aqwa	JScript	Yes	
Autodyn	Not Available		
CFD-Post	CCL	Yes	Yes
DesignModeler	JScript	Yes	
Meshing	JScript	Yes	
Polyflow	Not Available		
IcePak	Not Available		
ICEM CFD	TCL	Yes	No

Ansys Workbench Project and Data Model Concepts

Project Elements

You should understand the following terms and concepts when working with Ansys Workbench scripting. The following image depicts the relationship between the objects in a project.



Project

The project is the full collection of systems, components, data, and their connections that you create to achieve an overall CAE goal.

System

A system is a collection of components that together provide workflow to achieve an engineering simulation goal, such as completing a static structural analysis or a steady state fluid flow simulation. Systems may also contain a single component to complete an analysis sub-task, for example, meshing a geometric assembly.

Component

A component includes a collection of data and a data editor that work together to achieve a CAE-related task. A system may include several components which together to define a simulation workflow. A data editor can be an Ansys Workbench-enabled application like Ansys Fluent or Mechanical, or a native Ansys Workbench workspace like Engineering Data or Design Exploration. In the Ansys Workbench user interface, a component is depicted as a cell in the **Project Schematic**.

In an abstract sense, a component receives data from external or upstream sources, enables you to edit data local to the container, and then performs operations to produce output data. For example, a meshing component may receive geometry data from an upstream source, enable you define meshing attributes, and then generate a mesh for a downstream analysis.

In addition to managing the data and data editor required for a CAE task, a component also provides common services to control the input and output data from the component. For example, the component allows its output data to be transferred to other components.

Data Container

A data container (p. 37) includes data unique to an individual component as well as the services to manage and manipulate it. Although most components have a single data container, they may have more than one.

Services which create, retrieve, or modify a component's local data are provided by the data container, while services to control the transfer of data into and out of the component are provide by the component.

Data Entity

A data entity is a data structure defined within the data container. A data container often employs several data entities. A data entity is similar to a class in an object-oriented programming language. It defines member data (or properties) and methods that can be called to perform operations on the data.

Data Reference

A data reference is a handle to an instantiated data entity. From a data entity definition, an object can be created, and the data reference provides access to that object. In an Ansys Workbench journal, data references are assigned to variables. The variables are then used to access the properties and methods of the data entity. Consider the following example:

```
system = GetSystem(Name="My Analysis")
system.Update()
```

The first line queries for a previously instantiated system data entity named **`My Analysis`**. The `GetSystem` method returns a data reference to the specified system object and assigns it to the variable *`system`*. Using this variable, the second line calls a method supported by the system data entity named **`Update`**. This method updates the state of the system referenced by the variable, which is named **`My Analysis`**.

Data Container Reference

Similar to data references, a data container reference is a handle to an instantiated data container object.

Scripting Interface Concepts

Object

An object encapsulates a combination of data and services that act on the data. In the Ansys Workbench scripting interface, data entities, data containers, or components are all examples of objects. Access to these objects is provided by data references and data container references.

Property

A object's data is referred to as its properties. In the Ansys Workbench scripting interface, only objects derived from data entities have properties. A property is defined by its name, type, and value. Property types include:

- Boolean
- String
- Numerical types (Integer, Real)
- Physical quantities
- Data references and data container references
- Collections of the above types (Lists, Dictionaries)

An object's properties are accessed by applying the dot operator on the corresponding reference to that object. In the following example, **`parameter1`** is a data reference to a parameter object. The **`Expression`** property of this object is set to 10.

```
parameter1.Expression = "10"
```

Method

A method is a command that is bound to a data entity or data container object. Methods can change property values, create or delete properties, or invoke complex calculations and operations. Methods may require arguments to further specify the action to be taken, and they may return an object as a result of that action. Like properties, an object's methods are accessed by applying the dot operator on a corresponding data reference. In the following example, **`parameter1`** is a data reference to a parameter object. The method **`SetQuantityUnits`** is called to set the object's units to meters.

```
parameter1.SetQuantityUnits("m")
```

To see a list of all methods and queries in a component, execute **dir(*component*)** from the Workbench script console.

Argument

An argument is a variable that can be passed to a method when invoked. An argument has a type and share most of the same types as properties.

Ansys Workbench methods use *named arguments*, where each argument is given a name that is used to distinguish it from other arguments in the method. Because the arguments are named, their order is inconsequential.

Arguments can be required or optional. All required arguments must be specified or the method will fail. Optional arguments may be omitted and typically take a default value.

In the following example, the density property of a material is set to 8500 kg/m3. The named arguments are **Variables** and **Values**, which are set to **"Density"** and **"8500 [kg m^-3]"** respectively.

```
property1.SetData(Variables="Density", Values="8500 [kg m^-3]")
```

Because Ansys Workbench uses named arguments, the following form of the command is also acceptable.

```
property1.SetData(Values="8500 [kg m^-3]", Variables="Density")
```

Query

A query is a method which typically returns a data or container reference which can then be used to further interrogate or modify the project. Queries by themselves do not change the state of the project. Like methods, queries may require specifying arguments.

Namespaced Commands

Namespaced commands (p. 1277) are commands that are not bound to a particular object but have been grouped in a namespace with other commands having similar context. Like methods, commands perform some action, often require arguments, and may return an object as a result of the action. In the following example, the Update command is called in the **Project** namespace to update all components in a project by refreshing the input data and performing local calculations:

```
Project.Update()
```

Apps and Templates

Ansys Workbench uses template apps to create the project, system, and component elements described earlier.

Apps that create templates are analogous to document templates provided for Microsoft Word. For example, you can pick a Word report template that contains the formatting and sections for a type of report that you want to create. You then create a document from that template and fill out your specific content within that document to produce the report. Comparably, you may pick a system template in Ansys Workbench that describes the components, data, and relationships needed to execute a particular

type of CAE analysis. You then create a system from that template, enter data, and perform operations within that system to complete the analysis.

Templates facilitate and automate creation of project elements for a specific purpose. However, these templates do not preclude you from manually (and flexibly) building up a project from the constituent elements to achieve the same goal.

Project Template

A project template defines a set of system templates and the connections between them that can be used to perform a multi-disciplinary CAE analysis task (such as Thermal-Stress or One-Way FSI).

The project template can be used to create specific instances of systems and components within the project (such as the custom templates in the Ansys Workbench **Toolbox**).

System Template

A system template contains the information to create an Ansys Workbench System designed for a particular simulation objective. Ansys Workbench provides system templates for many types of standard analyses, including static structural, fluid flow, explicit dynamics, steady-state thermal, and others. The analysis systems listed in the Ansys Workbench **Toolbox** are all examples of system templates. All analyses performed in Ansys Workbench begin by referencing a system template.

Component Template

The component template includes the allowed input/output data types, internal data, and key commands associated with a specific component (such as Geometry).

Using Scripting in Ansys Workbench

Ansys Workbench scripting follows an object-based approach, where interaction with data is defined in terms of objects. Objects have properties that can be used to get or set data values, and methods that use or modify the data. In terms of the Ansys Workbench concepts described in Ansys Workbench Project and Data Model Concepts (p. 8), objects are references to data entities in the data model, and the methods are commands and queries which operate on those entities.

For example, a parameter is a data entity that has properties such as description, value, and expression, and has methods that operate on the parameter to (for example) delete it or determine which other parts of the data model are associated with the parameter.

Ansys Workbench Objects

Ansys Workbench objects fall into two general categories: data containers and data entities. The basic process for operating on these objects is:

1. **Query for either a data container or data entity objects.** Queries are implemented as methods on both data container and data entity objects, and they most often start with *Get* (such as `GetFiles`, `GetSystem`, `GetComponent`). These methods return references to objects that are assigned to variables. The variables can then be used to access object properties and methods. To illustrate this process, consider the Design of Experiments (DOE) data container that includes a data entity for the DOE model, which in turn contains a data entity for the input parameters. To query the input parameter object, first query the DOE model from the data container, and then query the input parameter from the DOE model:

   ```
   DOEModel = DOEDataContainer.GetModel()
   InputParameter = DOEModel.GetParameter(Name="P1")
   ```

   In most instances, the variable for a data reference will be reused later in the journal or script so that queries do not need to be re-executed. However, there are some instances (such as System Coupling variables), where re-executing the query can clarify the context that is used to access the object. In those situations, journal variables are not reused and queries are generated each time an object is referenced.

2. **Interrogate and modify object properties.** If the query returns a reference to a data entity, you can interrogate its properties and modify those that are not identified as **Read Only** in the reference section (p. 37) of this guide. Properties are accessed by appending a dot and the property name to the variable assigned to the object reference. For example, once a reference to an input parameter is obtained, you can modify its classification (or Nature Property) to reflect that it is a continuous or discrete parameter.

   ```
   inputParameter.Nature = "NatureContinuous"
   ```

3. **Call methods on objects.** In addition to properties, most objects provide methods that operate on internal data. To call a method on an object, append a dot, the method name, and a comma-separated

argument list in closed parentheses. A method's required and optional arguments are also documented in the reference section (p. 37) of this guide. Continuing the input parameter example, you can specify a restricted set of manufacturable values for a parameter by calling the `AddLevels` method on the input parameter object and by constructing a list of values and assigning it to the **`Levels`** argument.

```
inputParameter.AddLevels( Levels=["65", "70", "75", "80"])
```

Typical examples are provided in Usage Examples (p. 16). These examples demonstrate how to query objects and how to invoke methods on those objects for the desired result. Refer to the reference section (p. 37) in this guide for a complete list of all available data container objects and their respective data entities, methods, properties, and arguments.

Units

Many properties and variable values in Ansys Workbench represent physical quantities, which include both a value and a unit in their definition. For more information, see Expressions, Quantities, and Units in the *Workbench User's Guide*.

The assignments of these quantities are journaled using a *`"Value [Unit]"`* string syntax. This method ensures that all available information is recorded in the journal. However, strings may be inconvenient to work with when writing or modifying scripts that assign quantities.

Specifying Quantities Without Units

As a convenience, Ansys Workbench allows the assignment of a quantity using just the numeric value. When units are omitted, the unit is assumed to be the unit for that quantity in the current project unit system. Although this method is more convenient, you must ensure that the value being supplied is consistent with the current units.

Setting an Entity Property

When setting an entity property that refers to a quantity, the property assignment can be done as a numeric value, and the units will be taken from the project unit system. For example, after setting the Inlet Mass Flow in a VistaTF setup, the following would be recorded in the journal:

```
vistaTFSetup1 = setup1.GetSetupEntity()
vistaTFSetup1.MassFlow = "0.5 [kg s^-1]"
```

When entering this command or writing script, you can use a direct numeric value:

```
vistaTFSetup1 = setup1.GetSetupEntity()
SetProjectUnitSystem(UnitSystemName="SI")
vistaTFSetup1.MassFlow = "0.3 [lbm s^-1]" # Units explicitly provided
print vistaTFSetup1.MassFlow
>>> 0.3 [lbm s^-1]
vistaTFSetup1.MassFlow = 0.3 # Units are taken from the project unit system
print vistaTFSetup1.MassFlow
>>> 0.3 [kg s^-1]
```

Setting Quantity in Variable Data Tables

The same principles apply when setting variables in Material Property data tables (used primarily in Engineering Data). For example, after selecting a material and changing the density to 9000 [kg m^-3], the following would be recorded in the journal:

```
material1= eda1.GetMaterial(Name="Structural Steel")
materialProperty1= material1.GetProperty(Name="Density")
materialProperty1.SetData(
    SheetName="Density",
    Variables=["Density"],
    Values=[["9000 [kg m^-3]"]])
```

When writing a script, for convenience, you can omit the units, and they will be taken from the current project unit system. You can also omit the list brackets because only single values are being specified. A condensed version of the above command that is valid when playing back a script is:

```
material1= eda1.GetMaterial(Name="Structural Steel")
materialProperty1= material1.GetProperty(Name="Density")
materialProperty1.SetData(
    Variables="Density",
    Values=9000)
```

A more complex example shows the creation of a temperature-dependent property using a script:

```
# Temperatures in degrees Fahrenheit
temperatures = [200,400,600,800,1000]
# Coefficient of Thermal Expansion in F^-1
alphas = [6.3e-6, 7.0e-6, 7.46e-6, 7.8e-6, 8.04e-4]

# Change to an appropriate unit system
#(US Customary, which has an internal tag of "BIN_STANDARD")
SetProjectUnitSystem(UnitSystemName="BIN_STANDARD")

# Create a new instance of engineering data and
# access the Coefficient of Thermal Expansion property of Structural Steel
EDAtemplate = GetTemplate(TemplateName="EngData")
system = EDAtemplate.CreateSystem()
eda = system.GetContainer(ComponentName="Engineering Data")
steel = eda.GetMaterial(Name="Structural Steel")
alpha = steel.GetProperty(Name="Coefficient of Thermal Expansion")

# Set the property data according to the provided data
alpha.SetData(Variables=["Temperature","Coefficient of Thermal Expansion"],
              Values = [temperatures, alphas])
```

File Path Handling in Ansys Workbench

Ansys Workbench uses specific conventions for file or directory paths that are recorded to an Ansys Workbench journal, thus improving the portability of these paths between operating systems and user directories.

Handling Slashes as File Path Separators

Ansys Workbench uses the following conventions when a forward slash or a backslash is used as a path separator in a journal or script:

- When a journal is written, all backslashes that occur in command arguments representing a path are converted to forward slashes.

- When a command argument containing a path is read, all slashes are internally converted to the current platform-appropriate slash.

Handling of Absolute and Relative Paths

To ensure a robust recording and playback of scripts and journals, Ansys Workbench always records file and directory paths using a full (absolute) path. However, to facilitate portability of journals to different locations on a file system, a *user path root* is used to dynamically record and construct the absolute path.

The following conventions and capabilities are used with the user path root.

- The default value of the user path root is taken from the **Default Folder for Permanent Files** preference. Changing this preference will take effect in your next session.
- When a path is recorded to the journal and the start of the path matches the user path root, then the path is recorded using `AbsUserPathName("RelativePathName")`. This function constructs an absolute path based on the supplied relative path and the current user path root.
- You can access or change the current value of the user path root within a script using the following commands:
  - **`pathRoot = GetUserPathRoot()`**
  - **`SetUserPathRoot(DirectoryPath="NewPathRoot")`**

  `SetUserPathRoot` does not change the **Default Folder for Permanent Files** preference but simply overrides the path root setting in the current session.

Example

Assume that the current value of the user path root is `C:\Users\myUser\Projects`. The command is:

```
Open(FilePath=r"C:\Users\myUser\Projects\proj1.wbpj")
```

This would be journaled as:

```
Open(FilePath=AbsUserPathName("proj1.wbpj"))
```

You can override the current user path root to control the base location of files in your script:

```
SetUserPathRoot(DirectoryPath = "C:/Users/myUser1/Projects")
Open(FilePath=AbsUserPathName("proj1.wbpj")) # Read project from first location
SetUserPathRoot(DirectoryPath = "C:/Users/myUser2/Projects")
Open(FilePath=AbsUserPathName("proj1.wbpj")) # Read project from second location
```

Usage Examples

Several examples are provided here to demonstrate some typical scripting uses. All of the files used by these examples are located in *ANSYS_INSTALL_PATH*`\v212\commonfiles\examples\Scripting\ScriptingGuideExamples.zip`. To use these example files, unzip them into the `My Docu-`

ments directory. All necessary subdirectories will be created, allowing these sample journals to run correctly.

Automatically Update all Projects Affected by a Design Modification

You have performed a set of analyses on a design with an initial geometry file. These analyses have been saved as a number of different Ansys Workbench projects. The design has been modified, and you have been provided with a new geometry file that represents the updated design. After you extract the `ScriptingGuideExamples.zip` file, you will find the files for this example in the `Design_Modification` directory.

To automate the update of all affected projects, you would write a script that does the following:

1. Finds all Ansys Workbench projects within a specified directory.
2. For each project that has been found:
   - Replaces the original geometry with the new geometry file in any system in the project.
   - Updates the project and reports any errors from the update.
   - If the update was successful, reports the values of key output parameters under the modified geometry.
   - Saves the modified project with the new geometry file to a new directory.

Although the analysis specifics are secondary for the purposes of this example, a CFD analysis of a blood mixing device is used. This device attempts to maximize mixing of two blood streams while minimizing flow vorticity (which is an indicator of blood damage potential). Three projects involving a coarse mesh model, a fine mesh model, and an asymmetric flow condition were created. The parameters of interest are pressure drop, average and maximum vorticity, and mixing factor of blood stream 1 at the exit.

Figure 1: Mixing in base geometry

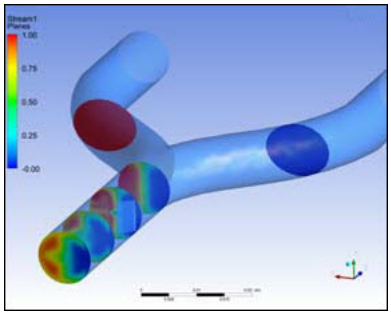
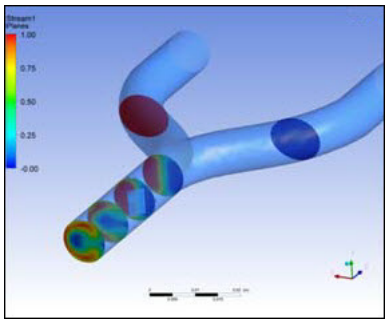


Figure 2: Mixing in modified geometry



Workbench Script

The script for this example follows. Each line is numbered for reference in the discussion that follows.

```
1  # import the 'os' module, which provides a portable way of using operating system dependent functionality
2  import os
3
4  # Helper function to write parameters to the log file
5  def writeParams(logFile):
6      for param in Parameters.GetAllParameters():
7          prmString = "  " + param.Name + ": " + param.DisplayText + " = " + param.Value.ToString()
8          logFile.write(prmString + "\n")
9      logFile.flush()
10
11 workDir = "ScriptingGuideExamples/Design_Modification/"
12
13 # Define the original and target directories
14 origDir = AbsUserPathName(workDir + "Original")
15 newDir = AbsUserPathName(workDir + "Modified")
16
17 # Define new geometry file
18 newGeom = AbsUserPathName(workDir + "Geometry/bloodMix2.agdb")
19
20 # Open a log file to record script progress
21 logFile = open(AbsUserPathName(workDir + "GeometryReplace.log"),"w")
22
23 # Create the new directory if necessary
24 if not os.path.exists(newDir):
25     os.makedirs(newDir)
26
27 # Find all the projects in the original directory
28 projList = []
29 for fileName in os.listdir(origDir):
30     if fileName.endswith(".wbpj"):
31         projList.append(fileName)
32
33 # Process each project one at a time
34 for projFile in projList:
35
36     # Open the project into Workbench and clear any starting messages
37     projPath = os.path.join(origDir, projFile)
38     logFile.write("Processing project in " + projPath + "\n")
39     Open(FilePath=projPath)
40     ClearMessages()
41
42     # Output project parameter values before update
43     logFile.write("Parameter values in original project:\n")
44     writeParams(logFile)
45
46     # Walk through each system in the project and replace the geometry file (if there is a geometry Component)
47     for system in GetAllSystems():
48         try:
49             geometry = system.GetContainer(ComponentName="Geometry")
```

```
50          except:
51              logFile.write("No geometry to replace in system " + system.DisplayText + "\n")
52          else:
53              geometry.SetFile(FilePath=newGeom)
54
55      # Update the project
56      try:
57          Update()
58      except:
59          logFile.write("update failed")
60
61      # If the project has been successfully updated, write out the new parameter values
62      # If not, write any project messages
63      if IsProjectUpToDate():
64          logFile.write("Parameter values in revised project:\n")
65          writeParams(logFile)
66      else:
67          logFile.write("ERROR: Project not successfully updated. The following messages were found:\n")
68          for msg in GetMessages():
69              msgString = "   " + msg.DateTimeStamp.ToString() + " " + msg.MessageType + ": " + msg.Summary + "\
70              logFile.write(msgString + "\n")
71
72      # In any case, save the modified project to the new directory
73      projPath = os.path.join(newDir, projFile)
74      Save(FilePath=projPath)
75
76      logFile.write("\n")
77      # End of project loop
78
76  logFile.close()
79
```

Log File

The log file generated by this script should look like this:

```
Processing project in C:\Users\neUser\Demo/ScriptExample1/Original/AsymmetricFlow.wbpj
Parameter values in original project:
  P1: PressureDropCoeff = 34.2088
  P2: mixing = 0.217835
  P5: maxVorticity = 3939.22 [s^-1]
  P4: aveVorticity = 27.4697 [s^-1]
Parameter values in revised project:
  P1: PressureDropCoeff = 34.0394
  P2: mixing = 0.276673
  P5: maxVorticity = 3939.22 [s^-1]
  P4: aveVorticity = 27.4034 [s^-1]

Processing project in C:\Users\neUser\Demo/ScriptExample1/Original/BaseAnalysis.wbpj
Parameter values in original project:
  P1: PressureDropCoeff = 30.04
  P2: mixing = 0.248514
  P5: maxVorticity = 3939.22 [s^-1]
  P4: aveVorticity = 36.8447 [s^-1]
Parameter values in revised project:
  P1: PressureDropCoeff = 30.3782
  P2: mixing = 0.288321
  P5: maxVorticity = 3939.22 [s^-1]
  P4: aveVorticity = 36.6682 [s^-1]

Processing project in C:\Users\neUser\Demo/ScriptExample1/Original/FineAnalysis.wbpj
Parameter values in original project:
  P1: PressureDropCoeff = 29.0038
  P2: mixing = 0.266388
  P5: maxVorticity = 3939.22 [s^-1]
  P4: aveVorticity = 29.2073 [s^-1]
Parameter values in revised project:
  P1: PressureDropCoeff = 30.7209
  P2: mixing = 0.295078
```

```
P5: maxVorticity = 3939.22 [s^-1]
P4: aveVorticity = 37.5408 [s^-1]
```

Discussion

This example demonstrates a number of the typical programming constructs and Ansys Workbench functionality that will be employed in the creation of scripts. The following discussion refers to the specified line numbers of the example script to illustrate some of these concepts and constructs.

Lines 1-2

The `import` keyword is used to include Python modules that enhance functionality available within the script. The '**`os`**' module used in this script provides capabilities for interacting with operating system services. Refer to the Python Standard Library documentation for details on all available modules.

Lines 4-9

A common pattern is to encapsulate repeated operations in Python function definitions and then call these functions later in the script. Because the script is processed sequentially, the function must be defined before it is used. This example defines a function `writeParams()` to loop through all the parameters in the current project and write information to a provided file object. When concatenating data into a string for output, all non-String data types must first be explicitly converted to their string representation using the `str(`*`data`*`)` function or `.ToString()` method.

Lines 11-18

It is recommend that you use the `AbsUserPathName` function when working with directory or file paths to promote portability of the path between user accounts and operating systems. See File Path Handling in Ansys Workbench (p. 15) for further details.

Lines 20-21

Standard Python functionality is used to open and write output to a log file throughout this script.

Lines 23-31

This section also employs standard Python to loop through all file names in a directory. All those file names that end with the `.wbpj` extension are added to a list of project files for further processing.

Lines 33-40

The script now processes each project in turn. At the beginning of the loop, the full path to the project file is generated from the provided directory name and project file name in a platform-independent manner using Python's `os.path.join` function. The project is then opened and messages are cleared so that later message processing will focus on new messages generated during the update. Note that the Python is case-sensitive. The `Open` command is part of the Ansys Workbench `Project` namespace and is used to open an Ansys Workbench project, while the `open` command is provide by Python and is used to open any file and create a Python file object.

Lines 42-44

The `writeParams` function defined earlier in the script is used to record parameter values to the log file before project modification.

Lines 46-47

The `GetAllSystems()` query is used to provide references to all systems in the project.

Lines 48-53

Because Geometry components are of specific interest, the system's `GetContainer()` query is used to try to get a reference to the Geometry data. A Python try/except/else construct is used to handle the success or failure of the query.

An alternate approach here would be to walk through each component in the system and then use information about the component to decide which to process. That approach is demonstrated in a subsequent example.

Lines 55-59

An Ansys Workbench project update is used to recompute all aspects of the project affected by the geometry change. This step is the most computationally expensive part of the script. A Python try/except construct is used to handle the success or failure of the update and prevent a premature exit of the script.

Lines 61-70

After the update operation has completed, the script checks the whole project to see if it has been successfully updated. If the project update was successful, the script calls the `writeParams` function again to output updated parameter information to the log file. If the update was not successful, the script records any project messages that were generated during the update.

Lines 66-70

Similar to the method used to process parameters, the script loops through all messages in the project and write key information about each message to the log file.

Lines 72-77

A file path for the project within the desired directory for the modified projects is generated, and the Ansys Workbench **Save** command is called to save the modified project. After saving, the loop repeats for the next project in the list.

Interaction with Files in a Project

In this example, you will look at how you can interact and query specific files that are associated with components in a project. In this example, you wish to know how and where a specific geometry file has been used within any Ansys Workbench project that you have within your directory. After you extract the `ScriptingGuideExamples.zip` file, you will find the files for this example in the `Project_File_Search` directory.

To automate this search, you will write a script that performs the following operations. Two different methods that accomplish the same task are demonstrated to illustrate different approaches to project and file navigation.

1. Starting from a given location, recursively search through all subdirectories to find any Ansys Workbench projects.

2. For each project, looks for a specified geometry file in using one of two methods:

   a. Get a flat list of all files in the project, and look for the desired file name.

   b. Walk through all systems and their components, and query each component to get the list of files registered to it. Then look if the component is using a file of the specified name. This method is more involved but provides more detailed information (such as the specific system or component using the file). It also gives access to the Ansys Workbench file references to provide detailed information like file size and last modification time.

Workbench Script

The script for this example follows. Each line is numbered for reference in the discussion that follows.

```
1  # import the 'os' module, which provides a portable way of using operating system dependent functionality
2  import os
3
4  # A helper function to find all Workbench projects within a specified directory and add them to a list.
5  # This recursivly calls itself to process subdirectories.
6  def findProjectFiles(searchDir, fileList):
7      print "Searching in %s" % searchDir
8      for dirEntry in os.listdir(searchDir):
9          fullName = os.path.join(searchDir,dirEntry)
10         if dirEntry.endswith(".wbpj"):
11             # Store the full path to the project file
12             projList.append(fullName)
13         if os.path.isdir(fullName):
14             findProjectFiles(fullName, fileList)
15
16
17 # Define starting directory to find project files.
18 # Here we'll look everywhere within the user's project path root directory.
19 searchDir = AbsUserPathName("ScriptingGuideExamples")
20
21 # Define file name of interest
22 targetFile = "bloodMix1.agdb"
23
24 # Recursively find all WB2 projects within the search directory
25 projList = []
26 findProjectFiles(searchDir, projList)
27
28 # Open a log file to record script progress
29 logFile = open(AbsUserPathName("ScriptingGuideExamples/Project_File_Search/FindFileInProjects.log"),"w")
30
31 for projFile in projList:
32     try:
33         Open(FilePath=projFile)
34     except Exception as ex:
35         logFile.write("Error opening %s, %s\n" % (projFile, ex))
36         continue
37
38     # Method 1: Search the list of all project files.
39     # This method is simpler, but not as much file information is readily available
40     for fileName in GetAllFiles():
41         if fileName.find(targetFile)> -1:
42   logFile.write("--\n")
43          fileStr = "File %s found in project %s\n"
44          logFile.write(fileStr % (fileName, projFile))
```

```
45          logFile.flush()
46
47      # Method 2: Walk through the systems and components, to find any that are using the file.
48      # This method is more complex, but provides detailed information through systems,
49      # components and file references.  It's also a useful example of general
50      # System & Component navigation.
51
52      # Loop over all systems in the project
53      for system in GetAllSystems():
54          # Loop over the all components in the system
55          for component in system.Components:
56              container = component.DataContainer
57              # Loop over all file data references associated with the container for the component
58              for compFile in container.GetFiles():
59                  if compFile.FileName.find(targetFile) > -1:
60        logFile.write("--\n")
61                      sysStr = "Target file found in component %s of system named %s in %s. Details:\n"
62                      fileStr = "   %s, Size %s bytes, Modified %s\n"
63                      logFile.write(sysStr % (component.DisplayText, system.DisplayText, projFile))
64                      logFile.write(fileStr % (compFile.Location, compFile.Size, compFile.LastModifiedTime))
65                      logFile.flush()
66
67  logFile.close()
68
```

Log File

The log file generated by this script should look like this:

```
--
Target file found in component Geometry of system named Static Structural in
     C:\Users\neUser\DemoProjects\pipe1.wbpj. Details:
   E:\data\Models\pipe_demo\pipe.x_t, Size 6934 bytes, Modified 06/07/2009 11:50:53 AM
--
File E:\data\Models\pipe_demo\pipe.x_t found in project C:\Users\neUser\DemoProjects\pipe1.wbpj
--
Target file found in component Geometry of system named Static Structural in
     C:\Users\neUser\Working\pipeDemo.wbpj. Details:
   E:\data\Models\pipe_demo\pipe.x_t, Size 6934 bytes, Modified 06/07/2009 11:50:53 AM
--
File E:\data\Models\pipe_demo\pipe.x_t found in project C:\Users\neUser\Working\pipeDemo.wbpj
```

Discussion

This example demonstrates how to navigate through systems and components in a project, as well as useful queries and data entities for working with files. The following discussion refers to the specified line numbers of the example script to illustrate some of these concepts and constructs. Discussion points from earlier examples will not be repeated here.

Lines 4-14

This function finds all project files within a specified directory. The Python `os.listdir` function returns all file and directory names in the specified location. If the name ends with `.wbpj`, Ansys Workbench stores the full path to a list of projects. If the name is a directory, Ansys Workbench recursively calls the function to search the subdirectory.

Lines 32-36

Here, the script demonstrates exception handling to catch any errors that result from opening the project file and report the error to the log file. The continue statement skips to the next entry in the project file loop.

Lines 38-45

This method uses the `GetAllFiles()` query to get a flat list of all files used in the project and looks at each entry to see if contains the target file name. If so, the script records the full path to the target file and the project file to the log.

Line 53

The `GetAllSystems()` query is used to loop over all systems in the project.

Line 55

The `Components` property in the system is accessed to loop over the set of components in the system.

Line 56

The `DataContainer` property is used to access the data within the component.

Line 58

The `GetFiles()` method on a container is used to return a list of file references that have been associated to that container.

Lines 59-65

The script looks at the `FileName` property of each file reference to see if it contains the target file name. If so, other properties of the file reference, system, and component are used to record information about the file and its location within the project to the log file.

Material Properties and Tabular Data

This example demonstrates scripting interaction with Engineering Data to access, create, and modify material property data. In this example, you have experimental total strain/stress data in a text file that you wish to use to define **Multilinear Isotropic Hardening** property data for a material. After you extract the `ScriptingGuideExamples.zip` file, you will find the files for this example in the `Material_Tabular_Data` directory.

As an additional consideration, the **Multilinear Isotropic Hardening** property data in Engineering Data is defined in terms of plastic strain. The script must first convert the total strain data to plastic strain, using the relationship:

`Plastic Strain = Total Strain – Stress/Young’s Modulus`

The above consideration will be used to demonstrate calculations based on physical quantities and units.

To automate property creation, you will write a script that performs the following operations:

1. Create a system for a Static Structural analysis and access the data container for Engineering Data.

2. Load material data for **Polyethylene**. By default, this material does not include property data for **Multilinear Isotropic Hardening**.

3. Read experimental data from a text file and generate lists of data for the necessary variables (converting **`Total Strain`** to **`Plastic Strain`**)

4. Create the **Multilinear Isotropic Hardening** property within **Polyethylene** and set its data table.

Sample Data File

The sample data file to be used in this example follows:

```
#
# Stress Stain Data for the Material Properties scripting example.
#
# The data is Total Strain (in m m^-1), Stress (in MPa)
#
7.33E-02, 80.6
1.80E-01, 88.0
6.30E-01, 142.5
7.53E-01, 168.0
8.70E-01, 187.0
```

Workbench Script

The script for this example follows. Each line is numbered for reference in the discussion that follows.

```
1  workDir = "ScriptingGuideExamples\Material_Tabular_Data/"
2
3  # Create an Engineering Data system and access its data container
4  template1 = GetTemplate(
5      TemplateName="Static Structural",
6      Solver="ANSYS")
7  system1 = template1.CreateSystem()
8  engineeringData1 = system1.GetContainer(ComponentName="Engineering Data")
9
10 # Import Polyethylene
11 poly = engineeringData1.ImportMaterial(
12     Name="Polyethylene",
13     Source="General_Materials.xml")
14
15 # Initialize lists for variable values
16 temperature = []
17 strain = []
18 stress = []
19
20 # Get the value of Young's Modulus for use in the Total/Plastic strain calculation.
21 # Material Property data is always returned as a Quantity.
22 elasticity = poly.GetProperty(Name="Elasticity")
23 E = elasticity.GetData(Variables="Young's Modulus")
24
25 # Read the data file and create lists for Temperature, Strain and Stress values
26 # We must convert Total Strain in the data file into Plastic Strain
27 fileName = AbsUserPathName(workDir + "StressStrainData.txt")
28 dataFile = open(fileName,"r")
29 for line in dataFile:
30     # Skip comment lines
31     if line.startswith("#"):
32        continue
33
34     # Split at the comma to get strain, stress
35     (thisStrain, thisStress) = line.split(",")
36
37     # Convert the data into Quantities with Units.
38     thisStrain = Quantity(float(thisStrain),"m m^-1")
39     thisStress = Quantity(float(thisStress),"MPa")
40
41     # Append data to the variable lists (converting total strain to plastic strain)
42     temperature.append(0)
43     calcStrain = thisStrain - thisStress/E
```

```
44     if calcStrain.Value < 1e-4:
45         calcStrain = Quantity(0.0,"m m^-1")
46     strain.append(calcStrain)
47     stress.append(thisStress)
48
49 # Create the Multilinear Isotropic Hardening property and set the data for it
50 miso = poly.CreateProperty(
51     Name="Isotropic Hardening",
52     Definition="Multilinear")
53 miso.SetData(
54     SheetName="Isotropic Hardening",
55     SheetQualifiers={"Definition Method": "Multilinear"},
56     Variables = ["Temperature","Plastic Strain","Stress"],
57     Values = [temperature, strain, stress])
58
59 # Save the project
60 Save(
61     FilePath=AbsUserPathName(workDir + "TabularData.wbpj"),
62     Overwrite=True)
```

Discussion

This example demonstrates interaction with Engineering Data to get, create, and set material properties. It also demonstrates calculations involving quantities and units. The following discussion refers to the specified line numbers of the example script to illustrate some of these concepts and constructs. Discussion points from earlier examples will not be repeated here.

Lines 1-13

To set the project up to work with Polyethylene, the script creates a new Static Structural system, access its Engineering Data container, and load material data from the General Materials library.

Lines 15-18

Interaction with tabular data is done in terms of the variables that make up the data table. Each variable represents a column in the table, with a list of data for the variable. Here the script initializes three lists for temperature, strain and stress data. **Multilinear Isotropic Hardening** property data can be temperature-dependent, and temperature is a required variable in the data table. Because the data is not temperature-dependent, you will use the same value of temperature (0 [C]) at each strain/stress point.

Lines 20-23

Because the calculation for plastic strain depends on Young's Modulus, you use the `GetData()` method to get the Young's Modulus value from the elasticity property. Within Engineering Data, all property values are stored and returned as quantities, which include both a numeric value and a unit. See Lines 35-41 for further discussion on quantities and units.

Lines 25-32

Here the script opens the text data file and reads each line one at a time. If the line begins with a comment character '#', the script continues to the next line in the file.

Lines 34-35

The Python `split()` function is used to break the comma delimited line and return the separate values into variables for strain and stress.

Lines 37-39

In this example, you are creating explicit unit-based quantities for stress and strain values to ensure dimensional and unit consistency in later calculations.

Material property data (or any other quantity data) can be set either as a quantity (with value and units) or as a simple numeric value. For more information, see Specifying Quantities Without Units (p. 14). If a numeric value is supplied, the units are assumed to be the same as the current project units for the variable. It is the script writer's responsibility to ensure the numeric data and project units are consistent.

Line 43-45

Because temperature is a simple constant value, temperature is specirfied as a numeric value (without units), and the units are taken from the current project unit system. The script appends 0 into the list of temperature data for this strain/stress pair.

Line 46

Here the script calculates plastic strain based on the available quantities and appends it to the variable list. Mathematical operations involving quantities enforce dimensional consistency and automatically perform unit conversion as necessary to generate a consistent result.

Line 47

Stress data is also appended to the appropriate list.

Lines 50-52

A new property for **Multilinear Isotropic Hardening** is created within **Polyethylene**.

Lines 53-57

The `SetData()` method is used to set single value or tabular data for material properties. Some properties can contain more than one data table, so the **`SheetName`** and **`SheetQualifiers`** arguments are used to specify the exact data table to be accessed. The **`Variables`** argument specifies one or more variables in the table to be set. The **`Values`** argument specifies the values that correspond to each variable.

Lines 59-62

Finally the script saves the project, overwriting if one already exists.

Mechanical APDL and Sending Commands to Integrated Applications

As discussed in Scripting and Data-Integrated Applications (p. 7), Ansys Workbench can interact with the native scripting language of many of its integrated applications. This example demonstrates scripting interaction with Mechanical APDL and the use of the `SendCommand` method to pass APDL commands to the editor. After you extract the `ScriptingGuideExamples.zip` file, you will find the files for this example in the `Sending_Commands` directory.

The case under consideration is a simple stress analysis of a bar that is defined in a Mechanical APDL input file. The bar dimensions and output displacement have been parameterized within Mechanical

APDL. In this example, you wish to write a script that automates this analysis for a number of different bar lengths and write an Ansys `.cdb` file for each case for future processing.

To automate this process, you will write an Ansys Workbench script that performs the following operations:

1. Creates a system for Mechanical APDL analysis and loads the specified Ansys input file.
2. Publishes some of the parameters from the input file to Ansys Workbench.
3. Loops through a list of desired bar lengths and, for each value, does the following:
    a. Sets the length parameter value.
    b. Updates the project to compute displacement for the new length.
    c. Sends APDL commands to Mechanical APDL to write the `.cdb` file to a desired location.

Sample Data File

The sample data file to be used in this example follows:

```
! set parameters
xlen=3
ylen=4
zlen=7

! define model
/prep7
block,,xlen,,ylen,,zlen
et,1,185
mp,ex,1,1e6
mp,prxy,1,0.3
vmesh,all
nsel,s,loc,z,0
d,all,all
nsel,s,loc,y,ylen
sf,all,pres,125
alls
fini

! obtain the solution
/solution
solve
fini


! retrieve results
/post1
! get the max stress at the fixed end
nsel,s,loc,x,xlen
nsel,r,loc,y,ylen
nsel,r,loc,z,0
*GET,out_my_node_stress,NODE,1,NXTH
*GET,out_seqv,NODE,out_my_node_stress,S,EQV
! get the max displacement at the free end
nsel,s,loc,x,xlen
nsel,r,loc,y,ylen
nsel,r,loc,z,zlen
*GET,out_my_node_def,NODE,1,NXTH
*GET,out_uy,NODE,out_my_node_def,U,Y
alls
fini
```

Workbench Script

The script for this example follows. Each line is numbered for reference in the discussion that follows.

```
1  # Import the 'os' module, which provides a portable way of using
2  # operating system dependent functionality
3  import os
4
5  workDir = "ScriptingGuideExamples/Sending_Commands/"
6
7  # Specify the Mechanical APDL Input file to be processed
8  inputFile = AbsUserPathName(workDir + "bar.dat")
9
10 # Provide a list of bar length (Z) values to solve and write CDB files for.
11
12 zValues = [3,5,12,15]
13
14 # Open a log file to record script progress
15 logFile = open(AbsUserPathName(workDir + "my_bar_script.log"),"w")
16
17 # Start a new project and create the Mechanical APDL system
18 Reset()
19 template1 = GetTemplate(TemplateName="Mechanical APDL")
20 system1 = template1.CreateSystem()
21
22 # Read the input file into the Mechanical APDL Setup
23 setup1 = system1.GetContainer(ComponentName="Setup")
24 mapdlInputFile1 = setup1.AddInputFile(FilePath=inputFile)
25
26
27 # Create Workbench parameters from two of the Mechanical APDL parameters
28 # in the input file
29 mapdlInputFile1.PublishMapdlParameter(Name="ZLEN")
30 parameter1 = Parameters.GetParameter(Name="P1")
31
32 mapdlInputFile1.PublishMapdlParameter(
33     Name="OUT_UY",
34     IsDirectOutput=True)
35 parameter2 = Parameters.GetParameter(Name="P2")
36
37 # Save the initial project definition.
38 Save(
39     FilePath=AbsUserPathName(workDir + "myBar.wbpj"),
40     Overwrite=True)
41
42 # Loop through all provided bar lengths
43 for zVal in zValues:
44
45     # Set the Z (length) parameter expression
46     parameter1.Expression = str(zVal)
47     logFile.write("Updating for z = %s\n" % zVal)
48
49     # Update the project for the new parameter value, and report
50     # success or failure to the log file.
51     try:
52         Update()
53     except:
54         logFile.write("  Update failed.\n")
55     else:
56         logFile.write("  Update succeeded. UY = %s\n" % parameter2.Value)
57
58     # Generate the name of the CDB file to save
59     cdbName = os.path.join(GetUserFilesDirectory(), "my_bar_" + str(zVal) + ".cdb")
60     cdbName = cdbName.replace("\\","/")
61
62     # Delete the cdb file if it already exists, to prevent
63     # Mechanical APDL from promting us about overwrite.
64     if os.path.exists(cdbName):
65         os.remove(cdbName)
66
```

```
67     # Generate the APDL command to save the CDB file and send it.
68          apdlCmd = "cdwr,db,%s" % cdbName
69          setup1.Edit(
70              Interactive=False,
71              LoadInputFiles=True)
72
73     setup1.SendCommand(Command=apdlCmd)
74     logFile.write("  CDB written to %s\n" % cdbName)
75          setup1.Exit()
76
77 # Save the final project state.
78 Save()
79 logFile.close()
```

Log File

The log file generated by this script should look like this:

```
Change all the forward slashes (\) with backslashes (/)
Updating for z = 3
  Update succeeded. UY = -0.00083352729
  CDB written to C:\Users\neUser\Demo\ScriptExample4\myBar_files\user_files\my_bar_3.cdb
Updating for z = 5
  Update succeeded. UY = -0.00304172391
  CDB written to C:\Users\neUser\Demo\ScriptExample4\myBar_files\user_files\my_bar_5.cdb
Updating for z = 0
  Update failed.
  CDB written to C:\Users\neUser\Demo\ScriptExample4\myBar_files\user_files\my_bar_0.cdb
Updating for z = 12
  Update succeeded. UY = -0.0504979576
  CDB written to C:\Users\neUser\Demo\ScriptExample4\myBar_files\user_files\my_bar_12.cdb
Updating for z = 15
  Update succeeded. UY = -0.121175623
  CDB written to C:\Users\neUser\Demo\ScriptExample4\myBar_files\user_files\my_bar_15.cdb
```

Discussion

This example demonstrates interaction with Mechanical APDL to operate on an existing Ansys input file and send APDL commands. The following discussion refers to the specified line numbers of the example script to illustrate some of these concepts and constructs. Discussion points from earlier examples will not be repeated here.

Lines 1-15

The initial lines of the script import useful modules, define controlling variables and create a log file to record script progress.

Lines 17-20

Here the script starts a new project and creates a new Mechanical APDL system.

Lines 22-24

The `AddInputFile` method on the Mechanical APDL Setup container reads and processes the specified input file.

Lines 27-35

The input file contains a number of values that can be parameterized. Here the script promotes two of them to be parameters that are controlled and displayed at the Ansys Workbench level.

The script accesses the **ZLEN** parameter as an input to set the bar length and the **OUT_UY** parameter as an output to track maximum Y displacement.

Lines 40

The script saves the project to a permanent location. Until the project is saved, all files and project directories are based on a temporary location. By saving the project, the script can more accurately report the directory that holds the `.cdb` files. The script will still function if the project is not first saved, but it will report the temporary location for the project files.

Lines 42-47

The script starts the loop for the desired bar length values and sets the Z parameter appropriately. The **Expression** for a parameter is a string as it can support complex functions. The Python `str()` function is used to convert the Z value to a string.

Lines 49-56

The script calls the project `Update()` command to recalculate the project based on the new length parameter value and uses Python exception handling to report the success or failure of the update. The value of output displacement is reported on successful update.

Lines 58-60

The project `user_files` directory is used to store the `.cdb` files for each case. Here the script generates the desired `.cdb` file path based on the `GetUserFilesDirectory()` query and uses `os.path.join` to combine that with the desired file name.

Mechanical APDL typically uses forward slashes in file paths by convention, so line 62 of the script converts any backslashes to forward slashes.

Line 62-65

Mechanical APDL interactively prompts you to overwrite a `.cdb` file if one already exists, so the script ensures that no `.cdb` file of the desired name is present.

Lines 67-75

The generated APDL command string writes the `.cdb` file to the desired file path and uses the `SendCommand` method to execute it. While this example just passes a single line command, the command string can contain multiple commands separated by newlines.

Lines 77-79

The script saves the final project state and closes the log file.

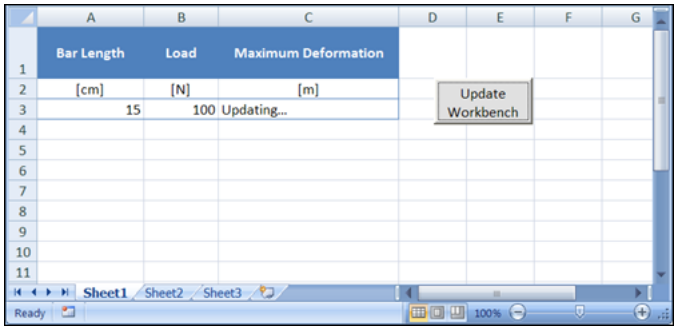
Updating a Workbench Project and Parameters from Excel

This example demonstrates the mechanics of interacting with Microsoft Excel from within an Ansys Workbench script, and connecting Ansys Workbench operations to UI Control events issues by Microsoft Excel (or other similar interfaces). The physics in this example is a simple cantilever beam, where the beam length and load are input parameters, and the resulting deflection is an output parameter. The case is defined within Ansys Workbench as a Static Structural system with the appropriate parameters

created. After you extract the `ScriptingGuideExamples.zip` file, you will find the files for this example in the `Excel_Parameter_Scripting` directory.

Of primary interest in this example is that user interaction with this project will be done via Microsoft Excel, where you will:

1. Set the input parameter values within an Excel Workbook.
2. Click the **Update Workbench** button in Excel.
3. See an **Updating...** message in Excel as the calculation proceeds.



The output parameter value are updated within the Workbook when the calculation is complete.

When the Ansys Workbench script below is executed, it opens the necessary Ansys Workbench project and Excel Workbook, and then it establishes an event connection so that an Ansys Workbench script function is executed at the press of the button within Excel.

Workbench Script

The script for this example follows. Each line is numbered for reference in the discussion that follows.

```
1  # IronPython imports to enable Excel interop
2  import clr
3  clr.AddReference("Microsoft.Office.Interop.Excel")
4  import Microsoft.Office.Interop.Excel as Excel
5
6  workingDir = AbsUserPathName("ScriptingGuideExamples/Excel_Parameter_Scripting/")
7
8  def updateHandler():
9
10    # Define key ranges in the Workbook
11    lengthCell = worksheet.Range["A3"]
12    loadCell = worksheet.Range["B3"]
13    defCell = worksheet.Range["C3"]
14
15    # Get the Workbench Parameters
16    lengthParam = Parameters.GetParameter(Name="P1")
17    loadParam = Parameters.GetParameter(Name="P2")
18    defParam = Parameters.GetParameter(Name="P3")
19
20    # Assign values to the input parameters
21    lengthParam.Expression = lengthCell.Value2.ToString()
22    loadParam.Expression = loadCell.Value2.ToString() + " [N]"
23
24    # Mark the deformation parameter as updating in the workbook
25    defCell.Value2="Updating..."
26
27    # Run the project update
28    Update()
29
```

```
30    # Update the workbook value from the WB parameter
31    defCell.Value2 = defParam.Value
32
33
34 # Open the Workbench Project
35 Open(FilePath = workingDir + "ExcelParameterScripting.wbpj")
36
37 # Open Excel and the workbook
38 ex = Excel.ApplicationClass()
39 ex.Visible = True
40 workbook = ex.Workbooks.Open(workingDir + "ParameterExample.xlsx")
41 worksheet=workbook.ActiveSheet
42
43 #Apply the update handler to the workbook button
44 OLEbutton = worksheet.OLEObjects("CommandButton1")
45 OLEbutton.Object.CLICK += updateHandler
46
```

Discussion

This example demonstrates a number of the typical programming constructs necessary for Ansys Workbench scripting to interact with the Common Language Runtime (CLR) API exposed by Microsoft Excel and similar applications.

Lines 1-4

The **clr** module is imported to enable IronPython to load and interact with CLR modules from other applications. In this example a **clr** reference to the Microsoft Excel Interop assembly is added and then the module is imported as the Excel namespace.

Line 6

The working directory is set, so that the script can load the Project and Excel Workbook from the specified location. When completing this example, this line will need to be modified to reflect your working location.

Lines 8-31

An `updateHandler()` function is created that performs all the necessary actions to interact with Excel and update the Ansys Workbench project. This function is connected to the `CLICK` event of the Excel button on line 45 of this script.

Lines 10-13

Specific cells of interest in the Workbook are created as named references in the script to facilitate later use of these cells. In this instance, the cells that hold the values of the two input parameters and one output parameter are given named references.

Lines 15-18

References to the three parameters exposed by the Static Structural system are also assigned to variables for later use.

Lines 20-22

The **Expression** property defining the input parameters are set based on the values of the associated cells in the Excel Workbook. The **Value2** property is used because it has simpler interaction when working with a single cell value.

Lines 24-25

The script sets the value of the output parameter cell in the workbook to **Updating...** while the calculation proceeds.

Lines 27-28

The `Update()` command is executed to update the project based on the new input parameter values.

Lines 30-31

When the update is complete, the output cell value in the Workbook is updated with the parameter value in the project.

Note the reference to `defparam.Value.Value`. The `Value` property of a parameter can have different types, including **`Numeric`**, **`String`**, **`Boolean`**, and **`Quantity`**. In this instance, the type is **`Quantity`**, which in turn has properties for `Value` and `Unit`. Therefore `defparam.Value.Value` is the numeric part of the quantity that is the parameter's current value.

Lines 34-35

The lines following the `updateHandler()` definition are those first executed when the script executes. The first operation is to load the Workbench project containing the parameterized Static Structural analysis.

Lines 37-39

Using the imported Excel namespace, an instance of the Excel application is created and made visible.

Lines 40-41

The script opens the Excel workbook that contains the parameter table and **Update** button and gets a reference to the primary (active) worksheet in the book.

Lines 43-44

You get a reference to the named OLE button (CommandButton1) that is present in the worksheet. This button was added to the worksheet by inserting the control within Excel, but no other macros or code associated with this button is required in the workbook.

Line 45

The `updateHandler()` function is added as an event handler to the `CLICK` event on the command button. Whenever the button is clicked, the associated Ansys Workbench script function is executed.

Known Issues and Limitations

A listing of known Journaling and Scripting issues and limitations.

General

- No known issues or limitations.

Data Containers

Ansys.InjectionMoldingData.Addin.Addin:SetupContainer

Ansys.InjectionMoldingData.Addin.Addin:SetupContainer

No details are provided for this entry.

Data Entities

SetupDataObject

No details are provided for this entry.

Properties

DisplayText

The general property that defines the user-visible name of an entity. This property is defined for all data entities but is used only in those entities that present a label in the user interface.

Type string (p. 1438)

Read Only No

FotDataFile

No details are provided for this entry.

Type string (p. 1438)

Read Only No

IstDataFile

No details are provided for this entry.

Type string (p. 1438)

Read Only No

MeshDataFile

No details are provided for this entry.

Type string (p. 1438)

Read Only No

rigidTransformationThetaXY

No details are provided for this entry.

Type Quantity (p. 1422)

Read Only No

rigidTransformationThetaYZ

No details are provided for this entry.

Type Quantity (p. 1422)

Read Only No

rigidTransformationThetaZX

No details are provided for this entry.

Type Quantity (p. 1422)

Read Only No

rigidTransformationX

No details are provided for this entry.

Type Quantity (p. 1422)

Read Only No

rigidTransformationY

No details are provided for this entry.

Type Quantity (p. 1422)

Read Only No

rigidTransformationZ

No details are provided for this entry.

Type Quantity (p. 1422)

Read Only No

VolFracDataFile

No details are provided for this entry.

Type string (p. 1438)

Read Only No

Ansys.Sherlock.Addin:Model

Ansys.Sherlock.Addin:Model

No details are provided for this entry.

Methods

Edit

No details are provided for this entry.

ImportProject

Import a Model into Sherlock

Required Arguments

ProjectFilePath Path to the Model file

Type string (p. 1438)

SyncProject

Sync Sherlock Projects into WB

Data Entities

ComponentDataObject

Main data object to be referenced from the container

Properties

DisplayText

The general property that defines the user-visible name of an entity. This property is defined for all data entities but is used only in those entities that present a label in the user interface.

Type string (p. 1438)

Read Only No

ImportedResults

No details are provided for this entry.

Type Dictionary (p. 1375)<string (p. 1438), List (p. 1400)<ImportAnalysis (p. 1391)>>

Read Only No

Ansys.Sherlock.Addin:Result

Ansys.Sherlock.Addin:Result

No details are provided for this entry.

Data Entities

ComponentDataObject

Main data object to be referenced from the container

Properties

DisplayText

The general property that defines the user-visible name of an entity. This property is defined for all data entities but is used only in those entities that present a label in the user interface.

Type string (p. 1438)

Read Only No

ImportedResults

No details are provided for this entry.

Type Dictionary (p. 1375)<string (p. 1438), List (p. 1400)<ImportAnalysis (p. 1391)>>

Read Only No

Ansys.Sherlock.Addin:Setup

Ansys.Sherlock.Addin:Setup

No details are provided for this entry.

Data Entities

ComponentDataObject

Main data object to be referenced from the container

Properties

DisplayText

The general property that defines the user-visible name of an entity. This property is defined for all data entities but is used only in those entities that present a label in the user interface.

Type string (p. 1438)

Read Only No

ImportedResults

No details are provided for this entry.

Type Dictionary (p. 1375)<string (p. 1438), List (p. 1400)<ImportAnalysis (p. 1391)>>

Read Only No

AQWA

AQWA Model

This container holds Model data for an instance of AQWA.

Methods

Edit

Opens the AQWA editor to allow modification of AQWA Setup data or viewing of the AQWA Results.

Exit

Exits the AQWA editor.

GetModel

Get the DataReference for the Model data entity.

Return DataReference for the Model data entity.

Type DataReference (p. 1371)

SendCommand

Executes one or more JScript commands in the AQWA editor.

Required Arguments

Command The command to execute in the AQWA editor.

Type string (p. 1438)

Example

To execute some arbitrary command (in this case, causing a dialog box to appear) in the AQWA editor:

```
model1.SendCommand(Command="WBScript.Out(\"My Text\",true);" )
```

If the AQWA editor is not open SendCommand will open it, run the command and then close it. Consider calling model1.Edit()to open the editor before using SendCommand if you do not want it to close.

Data Entities

AqwaModel

Model data entity.

Properties

AnalysisType

Entity Analysis Type setting.

Type string (p. 1438)

Read Only Yes

AQDBDatabaseFilesWritten

Indicates if aqwa .aqdb database files have been written for this system.

Type bool (p. 1360)

Read Only No

DisplayText

The general property that defines the user-visible name of an entity. This property is defined for all data entities but is used only in those entities that present a label in the user interface.

Type string (p. 1438)

Read Only No

GeometrySelected

Specifies if this data entity has Geometry data available.

Type bool (p. 1360)

Read Only No

PhysicsType

Entity Physics Type setting.

Type string (p. 1438)

Read Only Yes

ProjectName

The system string identifier.

Type string (p. 1438)

Read Only No

SolverType

Entity Solver Type setting.

Type string (p. 1438)

Read Only Yes

AQWA Results

This container holds Results data for an instance of AQWA.

Methods

Edit

Opens the AQWA editor to allow modification of AQWA Setup data or viewing of the AQWA Results.

Exit

Exits the AQWA editor.

GetResults

Get the DataReference for the Results data entity.

Return DataReference for the Results data entity.

Type DataReference (p. 1371)

SendCommand

Executes one or more JScript commands in the AQWA editor.

Required Arguments

Command The command to execute in the AQWA editor.

Type string (p. 1438)

Example

To execute some arbitrary command (in this case, causing a dialog box to appear) in the AQWA editor:

```
model1.SendCommand(Command="WBScript.Out(\"My Text\",true);" )
```

If the AQWA editor is not open SendCommand will open it, run the command and then close it. Consider calling model1.Edit()to open the editor before using SendCommand if you do not want it to close.

Data Entities

AqwaResults

Results data entity.

Properties

AnalysisType

Entity Analysis Type setting.

Type string (p. 1438)

Read Only Yes

DisplayText

The general property that defines the user-visible name of an entity. This property is defined for all data entities but is used only in those entities that present a label in the user interface.

Type string (p. 1438)

Read Only No

PhysicsType

Entity Physics Type setting.

Type string (p. 1438)

Read Only Yes

SolverType

Entity Solver Type setting.

Type string (p. 1438)

Read Only Yes

AQWA Setup

This container holds Setup data for an instance of AQWA.

Methods

Edit

Opens the AQWA editor to allow modification of AQWA Setup data or viewing of the AQWA Results.

Exit

Exits the AQWA editor.

GetSetup

Get the DataReference for the Setup data entity.

Return DataReference for the Setup data entity.

Type DataReference (p. 1371)

SendCommand

Executes one or more JScript commands in the AQWA editor.

Required Arguments

Command The command to execute in the AQWA editor.

Type string (p. 1438)

Example

To execute some arbitrary command (in this case, causing a dialog box to appear) in the AQWA editor:

```
model1.SendCommand(Command="WBScript.Out(\"My Text\",true);" )
```

If the AQWA editor is not open SendCommand will open it, run the command and then close it. Consider calling model1.Edit()to open the editor before using SendCommand if you do not want it to close.

Data Entities

AqwaSetup

Setup data entity.

Properties

AnalysisType

Entity Analysis Type setting.

Type string (p. 1438)

Read Only Yes

DisplayText

The general property that defines the user-visible name of an entity. This property is defined for all data entities but is used only in those entities that present a label in the user interface.

Type string (p. 1438)

Read Only No

PhysicsType

Entity Physics Type setting.

Type string (p. 1438)

Read Only Yes

SolverType

Entity Solver Type setting.

Type string (p. 1438)

Read Only Yes

StaticOnly

Used to control if Hydrodynamic Diffraction analysis solve is full or hydrostatic only when run from Workbench. If True then only Hydrostatics are solved.

Type bool (p. 1360)

Read Only No

AQWA Solution

This container holds Solution data for an instance of AQWA.

Methods

Edit

Opens the AQWA editor to allow modification of AQWA Setup data or viewing of the AQWA Results.

Exit

Exits the AQWA editor.

GetSolution

Get the DataReference for the Solution data entity.

Return DataReference for the Solution data entity.

Type DataReference (p. 1371)

SendCommand

Executes one or more JScript commands in the AQWA editor.

Required Arguments

Command The command to execute in the AQWA editor.

Type string (p. 1438)

Example

To execute some arbitrary command (in this case, causing a dialog box to appear) in the AQWA editor:

```
model1.SendCommand(Command="WBScript.Out(\"My Text\",true);" )
```

If the AQWA editor is not open SendCommand will open it, run the command and then close it. Consider calling model1.Edit()to open the editor before using SendCommand if you do not want it to close.

Data Entities

AqwaSolution

Solution data entity.

Properties

AnalysisType

Entity Analysis Type setting.

Type string (p. 1438)

Read Only Yes

DisplayText

The general property that defines the user-visible name of an entity. This property is defined for all data entities but is used only in those entities that present a label in the user interface.

Type string (p. 1438)

Read Only No

PhysicsType

Entity Physics Type setting.

Type string (p. 1438)

Read Only Yes

SolverType

Entity Solver Type setting.

Type string (p. 1438)

Read Only Yes

AUTODYN

AUTODYN Analysis

This container holds Results data for an instance of AUTODYN.

Methods

GetAutodynAnalysis

Returns a reference to an AutodynAnalysis data entity within the container.

Return A reference to the AutodynAnalysis data entity.

Type DataReference (p. 1371)

AUTODYN Setup

This container holds Solution data for an instance of AUTODYN.

Methods

Edit

Opens the AUTODYN editor for pre-processing, solving and post-processing. If there is an input file associated with the system the editor will load that file when it opens, otherwise the editor will create a new model.

Exit

Closes the AUTODYN editor. If there is any unsaved data in the editor it will be saved.

GetAutodynSetup

Returns a reference to an AutodynSetup data entity within the container.

Return A reference to the AutodynSetup data entity.

Type DataReference (p. 1371)

Import

Specifies the AUTODYN input file (*.ad), to be associated with the system. The specified file is copied to the systems working directory and registered with workbench.

Required Arguments

FilePath The location of the AUTODYN input file (*.ad). If the name is blank any currently associated input file will be removed.

Type string (p. 1438)

SetUserExecutable

Sets the location of the user created executable to be used when the editor is opened.

Required Arguments

ExecutablePath The full path and name of the executable to use when pre-processing, solving and post-processing.

Type string (p. 1438)

Data Entities

AutodynSetup

This holds the properties of the setup component.

Properties

Directory

The working directory for the editor.

Type string (p. 1438)

Read Only No

DisplayText

The general property that defines the user-visible name of an entity. This property is defined for all data entities but is used only in those entities that present a label in the user interface.

Type string (p. 1438)

Read Only No

FileName

The name of the associated input file.

Type string (p. 1438)

Read Only No

strUserExecutable

The location of the user defined execuatable to use.

Type string (p. 1438)

Read Only Yes

CFD Results

CFD Results

This container holds Results and post-processing data for a CFD simulation.

Methods

Edit

Opens the CFD-Post editor to allow modification of Results data.

This command will open the editor only if one is not already open on this component. If this component's editor is already open and in interactive mode, then it will be raised to the front.

Optional Arguments

Interactive Run the editor in interactive mode if True, or in no GUI mode if False.

If not specified, the editor runs in interactive mode.

Type bool (p. 1360)

Default Value True

Exit

Exits the editor.

Any changes made in this editor will be retained on exit. These changes are made permanent by a Project Save, and will be discarded in the event of closing the project without saving.

If no editor is open on the component in question, this command will have no effect.

GetCFDResults

Returns the Data Entity which contains user settings and properties for the Results container.

Return The Data Entity containing settings for this component.

Type DataReference (p. 1371)

SendCommand

Sends commands to the editor for this component using CFX Command Language (CCL) syntax. If the editor for this component is not open, it will be launched before the commands are sent and subsequently closed. In this mode, component data is loaded and saved as if calling Edit(Interactive=False) and Exit around the SendCommand invocation.

The instructions must be CFX Command Language session commands that are valid for the editor in question.

Required Arguments

Command Valid CFX Command Language (CCL) commands

Type string (p. 1438)

Data Entities

CFDResults

Entity which manages settings and data for the CFD Results component.

Properties

ClearState

Specifies whether the state should be cleared when the Results cell is updated. This is only used if a report has been selected or a session script has been specified to run on update.

Type bool (p. 1360)

Read Only No

CustomReportTemplate

Specifies the file path of the custom report template to load if LoadReport is set to 'Custom'. It is recommended that the template file be located in the user_files directory of the project so that it is available if the project is archived and moved to another system.

Type string (p. 1438)

Read Only No

DisplayText

The general property that defines the user-visible name of an entity. This property is defined for all data entities but is used only in those entities that present a label in the user interface.

Type string (p. 1438)

Read Only No

GenerateReport

Specifies whether to regenerate a report during component update.

The report will be generated according to the report definition within CFD-Post. Its name and location can be controlled by the ReportName and ReportLocationDirectory advanced user properties on this entity.

Type bool (p. 1360)

Read Only No

LoadReport

Report to load when Results cell is edited or updated.

Type PostReportNamesType (p. 1418)

Read Only No

MRESLoadOptionsMsg

This property has no effect and is simply used for presentation purposes.

Type string (p. 1438)

Read Only No

PostStateFile

CFD-Post State file location.

Do not modify this property directly.

Type DataReference (p. 1371)

Read Only No

ReportLocationDirectory

Specifies a custom directory in which to place the report files.

This is an advanced user property to be used in conjunction with the GenerateReport and ReportName properties. By default this property is empty, in which case the project's user_files directory is used. If a directory path is specified, the directory must exist at the time of component update.

Type string (p. 1438)

Read Only No

ReportName

Specifies the name of the report to be generated during component update. By default the report name is "Report".

This is an advanced user property to be used in conjunction with the GenerateReport and ReportLocationDirectory properties.

Type string (p. 1438)

Read Only No

RunSessionScript

Specifies whether a CFD-Post session script will be run as part of the component update.

If enabled, the UpdateScriptCCL property on this entity should be set to contain the session script text.

Type bool (p. 1360)

Read Only No

StateInitializationFile

Contains CFX Command Language script for CFD-Post, to be executed as part of the component update. This can be used to toggle regeneration of exported data, animation files, etc. In this scenario, the script will only need to be run when loading the results into Post at a time when the Results cell is in the Unfulfilled state (i.e. has never been opened yet).

This is an advanced user feature, and caution should be exercised. CFD-Post session files are capable of performing file activity and changing state in ways that are not supported in Workbench sessions. Limit your session script to activities which will not corrupt or circumvent Workbench project data management.

Type string (p. 1438)

Read Only No

UpdateScriptCCL

Contains CFX Command Language script for CFD-Post, to be executed as part of the component update. This can be used to toggle regeneration of exported data, animation files, etc.

This is an advanced user feature, and caution should be exercised. CFD-Post session files are capable of performing file activity and changing state in ways that are not supported in Workbench sessions. Limit your session script to activities which will not corrupt or circumvent Workbench project data management.

Type string (p. 1438)

Read Only No

CFX

CFX Setup

This container holds Setup data for an instance of CFX-Pre.

Methods

BuildSystemCouplingParticipant

Command wrapper to build the System Coupling Participant object from XML

Return Returns a valid SystemCouplingParticipant object.

Type DataReference (p. 1371)

Required Arguments

Xml Source Xml to build the SystemCouplingParticipant from.

Type string (p. 1438)

Edit

Opens the CFX-Pre editor to allow modification of CFX Setup data.

This command will open the editor only if one is not already open on this component. If this component's editor is already open and in interactive mode, then it will be raised to the front.

Optional Arguments

Interactive Run the editor in interactive mode if True, or in no GUI mode if False.

If not specified, the editor runs in interactive mode.

Type bool (p. 1360)

Default Value True

Exit

Exits the editor.

Any changes made in this editor will be retained on exit. These changes are made permanent by a Project Save, and will be discarded in the event of closing the project without saving.

If no editor is open on the component in question, this command will have no effect.

GetCFXSetupProperties

Returns the Data Entity which contains user settings and properties for the Setup container.

Return The Data Entity containing settings for this component.

Type DataReference (p. 1371)

Import

Imports CFX Setup data into the CFX-Pre editor from an existing CFX Case file.

The Case file's contents will be imported into a new case file within the project, managed by the Setup component. This operation is not valid if the component already contains Setup data.

Required Arguments

FilePath Path of CFX Case File to be imported.

Type string (p. 1438)

ReinitializeContainer

This command reinitialises the selected container, resetting the local data and using the cached initialisation CCL commands

SendCommand

Sends commands to the editor for this component using CFX Command Language (CCL) syntax. If the editor for this component is not open, it will be launched before the commands are sent and subsequently closed. In this mode, component data is loaded and saved as if calling Edit(Interactive=False) and Exit around the SendCommand invocation.

The instructions must be CFX Command Language session commands that are valid for the editor in question.

Required Arguments

Command Valid CFX Command Language (CCL) commands

Type string (p. 1438)

SetInitCCL

Sets the initialisation commands for this component using CFX Command Language (CCL) syntax. On setting the commands they may be applied to the editor in question using ApplyInitCCLCommand.

The instructions must be CFX Command Language session commands that are valid for the editor in question.

Required Arguments

Command Valid CFX Command Language (CCL) commands

Type string (p. 1438)

SuppressPhysicsErrors

Suppresses physics validation errors for the current CFX Setup component.

Normally, a CFX Setup component will not permit an update to proceed while validation errors exist in the case. However, in certain special situations, it may be that particular validation errors are tolerable and the case should be solvable.

This command can be used on CFX Setup components to bring the component from Attention Required to Up-to-Date, allowing the downstream Solution component to be updated.

Data Entities

CFXSetupProperties

Entity which manages settings and data for the CFX Setup component.

Properties

AllowAssemblyMeshImport

This is a beta option. Allow the user to import the assembly meshes

Type bool (p. 1360)

Read Only No

CaseFile

CFX-Pre Case file location.

Do not modify this property directly.

Type DataReference (p. 1371)

Read Only No

DisplayText

The general property that defines the user-visible name of an entity. This property is defined for all data entities but is used only in those entities that present a label in the user interface.

Type string (p. 1438)

Read Only No

PhysicsStatus

Physics state of the CFX-Pre editor, specifying validation errors, warnings and information on the current case.

This property is updated by CFX-Pre only.

Type string (p. 1438)

Read Only No

SolverInputExtendedFiles

Extended CFX Solver Input Files for multiconfig cases

Do not modify this property directly.

Type DataReferenceSet (p. 1371)

Read Only No

SolverInputFile

CFX-Solver Input file location.

Do not modify this property directly.

Type DataReference (p. 1371)

Read Only No

CFX Solution

This container holds data for a CFX Solution.

Methods

ClearCachedSolutionData

This command is used to clear cached solution data, optionally from just the current design point or from all design points

Required Arguments

ClearAllDPs If true, cached data from all design points will be removed. Otherwise, only cached data from the current design point will be removed.

Type bool (p. 1360)

ClearOldSolutionData

Every time the solver is run for a Solution container, a new set of solution files are generated and the old ones are kept around. This command is used to clear all solution data files from past solver runs except for the last one.

Required Arguments

KeepReferenced If false, all old solution data will be removed. Otherwise, only data not referenced in the latest solution will be removed.

Type bool (p. 1360)

DisplayMonitors

Displays run history and monitors for the most recent Solution update.

CFX-Solver Manager is used for displaying monitors, and (if necessary) will be launched by executing this command.

Edit

Open the CFX-Solver Manager editor to allow modification of the run definition for the CFX Solution component.

Note that CFX-SolverManager is a GUI-only component that is not scriptable. This command will not be journaled when invoked through the GUI. For batch operations, omit this command and use SetExecutionControl. If execution control is specified in the Setup component, it may be necessary to use this command with the appropriate value for the ExecutionControlOption parameter. Alternatively, the ExecutionControlOption property of the Solution component can be modified directly in the script to ensure no execution control conflicts are detected.

Optional Arguments

ExecutionControlOption Specifies how to handle conflicts in execution control if they exist.

Default	use On Execution Control Conflict property setting
UseSetupExecutionControl	Edit Run Defintion using execution control from the Setup container
UseSetupExecutionControlAlways	change On Execution Control Conflict property to 'Use Setup Cell Execution Control' and proceed with Edit Run Defintion
UseSolutionExecutionControl	Edit Run Defintion using execution control from the Solution container
UseSolutionExecutionControlAlways	change On Execution Control Conflict property to 'Use Solution Cell Execution Con-

trol' and proceed with Edit Run Defintion

Type ExecutionControlConflictOptions (p. 1383)

Default Value Default

EditDefFileInCommandEditor

Opens the CFX Command Editor to allow modification of the physics in the specified CFX Results or Solver Input file.

This command will simply launch the CCL Command Editor with the provided file path. No further Workbench interaction with the CCL Command Editor is possible, and no further activity of this component is journaled. Do not use this command as part of any batch workflow.

Required Arguments

FilePath Path of CFX Solver Input File, or Results File, to be edited

Type string (p. 1438)

Exit

Exits the editor.

Any changes made in this editor will be retained on exit. These changes are made permanent by a Project Save, and will be discarded in the event of closing the project without saving.

If no editor is open on the component in question, this command will have no effect.

GetCFXSolutionProperties

Returns the Data Entity which contains user settings and properties for the Solution container.

Return The Data Entity containing settings for this component.

Type DataReference (p. 1371)

GetComponentSettingsForRsmDpUpdate

This query is used to obtain the ComponentSettingsForRsmDpUpdate object for Journaling and Scripting

GetSolutionSettings

This query is used to obtain the component solve settings object for Journaling and Scripting

Import

Imports CFX Solution data from an existing CFX Results file.

The Results file, as well as all files associated with it (such as transient or multi-configuration files, if they exist), will be copied into the project as the generated data for the Solution component.

This command is intended for use when no Setup has been defined. If no data has been provided to an upstream Setup, or to any upstream Geometry or Mesh components (if they exist), then all of these components will automatically be deleted from the system as a result of this command.

Required Arguments

FilePath Path of CFX-Solver Results file to be imported.

Type string (p. 1438)

MarkUpToDate

Accept an interrupted Solution as Up-to-Date.

The specified Solution component should be in Interrupted state. As a result of this command, the Solution will be marked Up-to-Date.

SetExecutionControl

Sets the CFX-Solver settings (in the form of Execution Control CCL) for a Solution component.

These settings will form the basis of a run definition for the Solution component. However, certain settings, such as the Solver Input File, are always overridden during Update. Other settings, such as all Initialization settings, can be overridden by the Solution component's Initialization Option property as well by as the presence of initialization data provided by other components on the project schematic.

The CCL argument must specify valid Execution Control CCL.

Required Arguments

CCL CFX Command Language defining Execution Control for the CFX-Solver.

Type string (p. 1438)

SwitchToBackgroundMode

Switch the Update in progress into background mode. This will enable operations that are not allowed during an Update in foreground mode (e.g. Project Save).

This command is not normally useful in a script. Journals may record the invocation of this command after an Update invoke, as the result of GUI activity while the Update is in progress. However, replay of these journals will always wait for the Update invoke to complete before invoking the next command, rendering this step ineffectual.

Data Entities

CFXSolutionProperties

Entity which manages settings and data for the CFX Solution component.

Properties

CacheSolutionData

Flag to determine if the latest solution data for each retained design point is to be cached.

Type bool (p. 1360)

Read Only No

DisplayText

The general property that defines the user-visible name of an entity. This property is defined for all data entities but is used only in those entities that present a label in the user interface.

Type string (p. 1438)

Read Only No

ExecutionControlOption

Specifies how to handle conflicts in execution control between this entity and the upstream Setup cell. It is only used if execution control has been saved in this entity and the execution control in the DEF file, provided by the upstream Setup cell, has changed since the last run.

IssueWarning	In the case of a conflict, a message is displayed and the process is aborted.
UseExecutionControlFromSetup	In the case of a conflict, the execution control provided in the DEF file is used.
UseExecutionControlFromSolution	In the case of a conflict, the execution control saved in the Solution container is used.

Type ExecutionControlSource (p. 1383)

Read Only No

InitializationOption

Specifies the way a Solution will be initialized during Update.

Available options:

CurrentSolutionData	Initialize the solution from the current Solution component data (if the data is present and is appropriate).
InitialConditions	Always initialize the solution from defined initial conditions.

In cases where no existing solution data is present, this property has no effect.

The default behavior is "CurrentSolutionData"; however this default can be changed in Options->CFX->Default Initialization Option

Type InitializationOption (p. 1393)

Read Only No

KeepLatestSolutionOnly

Flag to clear all old solution data after the component update.

Type bool (p. 1360)

Read Only No

LoadMResOptions

Specifies how multi-configuration solution data should be treated by a Results component when opened in CFD-Post.

This option has no effect if the Solution component does not contain a results set with multiple configurations.

Available options:

AllConfigsSingleCase	All configurations for this set of results should be loaded and treated as a single case.
AllConfigsSeparateCases	All configurations for this set of results should be loaded, but each configuration should be treated as a separate case.
LastConfigOnly	Only the last configuration should be loaded.

The default behavior is "LastConfigOnly".

Type MResOptions (p. 1407)

Read Only No

ResultsFile

Current CFX-Solver Results File location.

Type DataReference (p. 1371)

Read Only Yes

SolverArguments

Additional Solver Arguments

Type string (p. 1438)

Read Only No

SolverPath

Path to a custom Solver Executable

Type string (p. 1438)

Read Only No

ComponentSolveSettingsForAddin

This class contains the solve settings for Addin solution components to use

Properties

ConfiguredQueue

A configured queue used for new RSM architecture

Type string (p. 1438)

Read Only No

DisplayText

The general property that defines the user-visible name of an entity. This property is defined for all data entities but is used only in those entities that present a label in the user interface.

Type string (p. 1438)

Read Only No

EnablePolling

whether enable polling during remote solve

Type bool (p. 1360)

Read Only No

ExecutionMode

execution mode

Type string (p. 1438)

Read Only No

NumberOfProcesses

number of processes for parallel solve

Type int (p. 1394)

Read Only No

PollingInterval

polling internal during remote solve

Type Quantity (p. 1422)

Read Only No

RsmJobName

RSM job name

Type string (p. 1438)

Read Only No

RsmQueueDetails

Configured queue details

Type RsmQueueDetails (p. 1427)

Read Only No

UpdateOption

Update mode

Type JobRunMode (p. 1397)

Read Only No

Design Exploration

DX Direct Optimization

This container holds data for an instance of Parameter Direct Optimization.

Methods

ApproveGeneratedData

Approve generated data after nonparametric changes by keeping DX results. This command is recursive and applied with any component dependent on the first component.

Example

The following example shows how to approve generated data.

```
system1 = GetSystem(Name="RSO")
designofExperiment1 = system1.GetContainer(ComponentName="Design of Experiment")
designofExperiment1.ApproveGeneratedData()
```

GetModel

Get the DataReference of the Model. An exception is thrown if the entity is not found.

Return The DataReference of the Model.

Type DataReference (p. 1371)

Example

The following example shows how the user can get a Model to change one of its properties.

```
system1 = GetSystem(Name="RSO")
designofExperiment1 = system1.GetContainer(ComponentName="Design of Experiment")
dOEModel1 = designofExperiment1.GetModel()
dOEModel1.DOEType = "eDOETYPE_OSFD"
```

Data Entities

AmoOptimization

Entity which performs and manages the DesignXplorer Optimization Method Component

Properties

Converged

Convergence state

Type bool (p. 1360)

Read Only Yes

CrossoverProbability

Maximum Allowable Pareto Percentage

Type double (p. 1378)

Read Only No

CurrentParetoPercentage

Pareto Percentage

Type double (p. 1378)

Read Only Yes

CurrentStabilityPercentage

Stability Percentage

Type double (p. 1378)

Read Only Yes

DisplayText

The general property that defines the user-visible name of an entity. This property is defined for all data entities but is used only in those entities that present a label in the user interface.

Type string (p. 1438)

Read Only No

MaximumNumberOfPermutations

Maximum number of permutations for discrete / manufacturable values.

Type int (p. 1394)

Read Only Yes

MaxNumCandidates

Maximum Number of Candidates

Type int (p. 1394)

Read Only No

MaxNumCycles

Maximum Number Of Cycles for OSFD algorithm.

Type int (p. 1394)

Read Only No

MaxNumIterations

Maximum Number of Iterations

Type int (p. 1394)

Read Only No

MutationProbability

Maximum Allowable Pareto Percentage

Type double (p. 1378)

Read Only No

NumberOfEvaluations

Number of Evaluations

Type int (p. 1394)

Read Only Yes

NumberOfFailures

Number of Failures

Type int (p. 1394)

Read Only Yes

NumberOfInitialSamples

Number of Initial Samples

Type int (p. 1394)

Read Only No

NumCandidates

Number of Candidates

Type int (p. 1394)

Read Only Yes

NumIterations

Current Iteration

Type int (p. 1394)

Read Only Yes

NumSamplesPerIter

Number of Samples Per Iteration

Type int (p. 1394)

Read Only No

ParetoPercentage

Maximum Allowable Pareto Percentage

Type double (p. 1378)

Read Only No

RandomGeneratorSeed

LHS Seed

Type int (p. 1394)

Read Only No

SampleSetSize

Size of Generated Sample Set

Type int (p. 1394)

Read Only Yes

StabilityCriterion

Maximum Allowable Pareto Percentage

Type double (p. 1378)

Read Only No

TypeOfInitialSampling

Type Of Initial Sampling

Type TypeOfInitialSampling (p. 1445)

Read Only No

AsoOptimization

Entity which performs and manages the DesignXplorer Optimization Method Component

Properties

Converged

Convergence state

Type bool (p. 1360)

Read Only Yes

ConvergenceTolerance

Convergence Tolerance

Type double (p. 1378)

Read Only No

DisplayText

The general property that defines the user-visible name of an entity. This property is defined for all data entities but is used only in those entities that present a label in the user interface.

Type string (p. 1438)

Read Only No

MaximumNumberOfPermutations

Maximum number of permutations for discrete / manufacturable values.

Type int (p. 1394)

Read Only Yes

MaxNumberReductionPerIteration

Maximun Number of Domain Reduction per Iteration

Type int (p. 1394)

Read Only No

MaxNumCandidates

Maximum Number of Candidates

Type int (p. 1394)

Read Only No

MaxNumCycles

Maximum Number Of Cycles for OSFD algorithm.

Type int (p. 1394)

Read Only No

MaxNumDomainReductions

Maximum Number of Domain Reductions

Type int (p. 1394)

Read Only No

MaxNumEvaluations

Maximum Number of Evaluations

Type int (p. 1394)

Read Only No

NumberOfEvaluations

Number of Evaluations

Type int (p. 1394)

Read Only Yes

NumberOfFailures

Number of Failures

Type int (p. 1394)

Read Only Yes

NumberOfInitialSamples

Number of LHS Initial Samples.

Type int (p. 1394)

Read Only No

NumberOfScreeningSamples

Number of Screening Samples for the Adaptive Single-Objective optimization method.

Type int (p. 1394)

Read Only No

NumberOfStartingPoints

Number of Starting Points for the Adaptive Single-Objective optimization method.

Type int (p. 1394)

Read Only No

NumCandidates

Number of Candidates

Type int (p. 1394)

Read Only Yes

PercentageOfDomainReductions

Percentage of Domain Reductions

Type double (p. 1378)

Read Only No

RandomGeneratorSeed

LHS Seed

Type int (p. 1394)

Read Only No

SampleSetSize

Size of Generated Sample Set

Type int (p. 1394)

Read Only Yes

CustomConstraintProperties

Entity which wraps and manages the external Parameter Definition for the Optimization Criterion with an external Optimization Method

Properties

No Properties.

CustomObjectiveProperties

Entity which wraps and manages the external Objective Definition for the Optimization Criterion with an external Optimization Method

Properties

No Properties.

CustomParameterProperties

Entity which wraps and manages the external Parameter Definition for the Input Parameter with an external Optimization Method

Properties

No Properties.

DiscreteLevel

The data entity which describes a Discrete Level of an Input Parameter.

Properties

DisplayText

The general property that defines the user-visible name of an entity. This property is defined for all data entities but is used only in those entities that present a label in the user interface.

Type string (p. 1438)

Read Only No

Index

Zero-based Index of the discrete level in the list of levels of the owning parameter.

Type int (p. 1394)

Read Only No

Value

Value of the DiscreteLevel.

Type Object (p. 1409)

Read Only No

Methods

SetValue

Sets the value of a discrete level entity. A discrete level can have an integer value (e.g. a number if holes, a number of turns, etc) or a string value (e.g. a material name or a geometry file name).

Required Arguments

Value Value set to the discrete level entity.

Type Object (p. 1409)

Example

The following example shows how to retrieve a discrete level from an input parameter and then change its value.

```
container = system1.GetContainer(ComponentName="Design of Experiment")
model = container.GetModel()
DiscreteInputParameter = model.GetParameter(Name="P2")
level1 = DiscreteInputParameter.GetDiscreteLevel(Name="Level 1")
level1.SetValue( Value="2500" )
```

InputParameter

The data entity which describes an Input Parameter in DesignXplorer.

Properties

Attribute1

First editable attribute of the distribution for an uncertainty parameter. The nature of the attribute depends on the distribution type. For instance, the first attribute of a Normal distribution is the Mean value.

Type double (p. 1378)

Read Only No

Attribute2

Second editable attribute of the distribution for an uncertainty parameter. The nature of the attribute depends on the distribution type. For instance, the second attribute of a Normal distribution is the Standard Deviation value. Some distribution type do not have a second attribute.

Type double (p. 1378)

Read Only No

ConstantValue

Constant value of the Parameter when it is disabled.

Type Object (p. 1409)

Read Only No

CustomDefinitionList

Custom Definition List

Type List (p. 1400)<DataReference (p. 1371)>

Read Only No

DiscreteLevels

List of the discrete levels.

Type List (p. 1400)<DataReference (p. 1371)>

Read Only Yes

DisplayText

The general property that defines the user-visible name of an entity. This property is defined for all data entities but is used only in those entities that present a label in the user interface.

Type string (p. 1438)

Read Only No

DistributionLowerBound

Distribution lower bound of the variation range for an uncertainty Parameter.

Type double (p. 1378)

Read Only No

DistributionType

Distribution type for an uncertainty parameter.

Type DistributionType (p. 1377)

Read Only No

DistributionUpperBound

Distribution upper bound of the variation range for an uncertainty Parameter.

Type double (p. 1378)

Read Only No

Enabled

True if the Parameter is enabled for the current study.

Type bool (p. 1360)

Read Only No

Kurtosis

Kurtosis value of the distribution for an uncertainty parameter.

Type double (p. 1378)

Read Only Yes

LowerBound

Lower bound of the variation range for a Continuous Parameter.

Type double (p. 1378)

Read Only No

Mean

Mean value of the distribution for an uncertainty parameter.

Type double (p. 1378)

Read Only Yes

Nature

Nature of the Parameter.

Type ParameterNature (p. 1413)

Read Only No

NumberOfLevels

Number of levels if the parameter nature is Discrete, or the parameter nature is Continuous and the UseManufacturableValues property is set to True.

Type int (p. 1394)

Read Only Yes

Skewness

Skewness value of the distribution for an uncertainty parameter.

Type double (p. 1378)

Read Only Yes

StandardDeviation

Standard deviation value of the distribution for an uncertainty parameter.

Type double (p. 1378)

Read Only Yes

Type

Type of the Parameter, either a DesignVariable in a GDO context, or an UncertaintyVariable in a SixSigma Analysis context.

Type SimulationType (p. 1432)

Read Only Yes

Units

Units

Type string (p. 1438)

Read Only Yes

UpperBound

Upper bound of the variation range for a Continuous Parameter.

Type double (p. 1378)

Read Only No

UseManufacturableValues

True to restrict the variation of the parameter to defined Manufacturable Values.

Type bool (p. 1360)

Read Only No

Methods

AddDiscreteLevel

Adds a Discrete Level entity on a discrete input parameter. A discrete level can have an integer value (e.g. a number of holes, a number of turns, etc) or a string value (e.g. a material name or a geometry file name). The command has optional arguments to specify the Name of the level and its Index in the list of levels of the parameter. By default, the new level is added to the end of the list. The various discrete levels of an input parameter represent independent configurations of the project, processed in the order of their creation.

Return The created entity.

Type DataReference (p. 1371)

Required Arguments

Value The value of the discrete level. Value can be an integer or a string.

Type Object (p. 1409)

Optional Arguments

DisplayText DisplayText of the created entity. If not specified, a default name of the form "Level [#]" is used.

Type string (p. 1438)

Index The position of the new level in the list of discrete levels of the parameter. Index is zero-based. If it is not specified, the new level is appended to the list.

Type int (p. 1394)

Default Value -1

Example

The following example shows how to add new discrete levels on a discrete input parameter. The third level is inserted between the two others.

```
container = system1.GetContainer(ComponentName="Design of Experiment")
model = container.GetModel()
DiscreteInputParameter = model.GetParameter(Name="P2")
DiscreteInputParameter.AddDiscreteLevel(Value=3000, DisplayText="Three thousand turns")
DiscreteInputParameter.AddDiscreteLevel(Value=2000, DisplayText="Two thousand turns")
DiscreteInputParameter.AddDiscreteLevel(Value=5000, Index="1")
```

AddLevels

Adds a list of levels to a continuous input parameter. Each level is a quantity or a real number corresponding to a manufacturable value. The list of levels forms a restriction filter used when post-processing the input parameter. If levels are added outside of the variation range, the lower and upper bounds are adjusted accordingly.

Required Arguments

Levels List of added levels.

Type List (p. 1400)<string (p. 1438)>

Optional Arguments

Overwrite True in order to overwrite the existing levels, False by default.

Type bool (p. 1360)

Example

The following example shows how to overwrite the manufacturable values of an input parameter and how to define an additional value later.

```
container = system1.GetContainer(ComponentName="Design of Experiment")
model = container.GetModel()
InputParameter = model.GetParameter(Name="P1")
InputParameter.UseManufacturableValues = True
InputParameter.AddLevels(Levels=["0.3 [mm]", "0.5 [mm]", "1e-3 [m]"], Overwrite=True)
InputParameter.AddLevels(Levels="7 [mm]")
```

CreateOptimizationCriterion

Creates an OptimizationCriterion entity associated to a parameter.

Return The DataReference of the OptimizationCriterion.

Type DataReference (p. 1371)

Required Arguments

Parameter The parameter on which the criterion is created.

Type DataReference (p. 1371)

Example

The following example shows how to create an OptimizationCriterion entity.

```
container = system1.GetContainer(ComponentName="Optimization")
model = container.GetModel()
parameter1 = model.GetParameter(Name="P1")
optimizationCriterion = parameter1.CreateOptimizationCriterion()
```

DeleteDiscreteLevels

Deletes a list of levels from a discrete input parameter.

Required Arguments

DiscreteLevels List of the DiscreteLevel entities to delete.

Type List (p. 1400)<DataReference (p. 1371)>

Example

The following example shows how to add and then delete one or more levels from a discrete input parameter.

```
container = system1.GetContainer(ComponentName="Design of Experiment")
model = container.GetModel()
DiscreteInputParameter = model.GetParameter(Name="P1")
```

```
level1 = DiscreteInputParameter.AddDiscreteLevel(Value=3000, DisplayText="Three thousand turns")
level2 = DiscreteInputParameter.AddDiscreteLevel(Value=2000, DisplayText="Two thousand turns")
level3 = DiscreteInputParameter.AddDiscreteLevel(Value=5000, Index="1")
DiscreteInputParameter.DeleteDiscreteLevels(DiscreteLevels=[level1, level2])
```

DeleteLevels

Deletes a list of levels from a continuous input parameter.

Required Arguments

Indices Indices of the items to remove from the levels list

Type List (p. 1400)<int (p. 1394)>

Example

The following example shows how to add and then delete one or more levels from a continuous input parameter for which the UseManufacturableValues property is set to True.

```
container = system1.GetContainer(ComponentName="Design of Experiment")
model = container.GetModel()
InputParameter = model.GetParameter(Name="P1")
InputParameter.UseManufacturableValues = True
InputParameter.AddLevels(Levels=["0.3 [mmm]", "0.5 [mm]", "1e-3 [m]"], Overwrite=True)
InputParameter.DeleteLevels( Indices=[0, 1] )
```

ExportData

Export the data of a DesignXplorer entity to a csv file. The entity can be one of the DesignXplorer chart entities, a ParametricTable or an InputParameter entity.

Required Arguments

FileName The exported file name.

Type string (p. 1438)

Optional Arguments

AppendMode True to append to an existing csv file, False to overwrite it.

Type bool (p. 1360)

Default Value False

Example

The following example shows how the user can export the table of Design Points and then the Parameters Parallel chart of a Design of Experiments component.

```
container = system1.GetContainer(ComponentName="Design of Experiment")
model = container.GetModel()
parametricTable = model.GetParametricTable(Name="DesignPoints")
```

```
parametricTable.ExportData(FileName="doe.csv")
chart = model.GetChart(Name="DesignPointsParallel")
chart.ExportData(FileName="doe.csv", AppendMode=True)
```

GetCustomParameterDefinition

Get the DataReference of a CustomParameterProperties entity associated with an Input Parameter. An exception is thrown if the entity is not found.

Return The DataReference of the CustomParameterProperties.

Type DataReference (p. 1371)

Required Arguments

ExtensionName Extension's Name.

Type string (p. 1438)

Example

The following example shows how the user can get a CustomParameterProperties.

```
system1 = GetSystem(Name="DOP")
optimization1 = system1.GetContainer(ComponentName="Optimization")
optimizationModel1 = optimization1.GetModel()
inputParameter1 = optimizationModel1.GetParameter(Name="P1")
customParameterDefinition1 = inputParameter1.GetCustomParameterDefinition(ExtensionName="UncertaintyParame
customParameterDefinition1.Distribution = "Triangular"
```

GetDiscreteLevel

Get a discrete level by name from an input parameter. The parameter's full and ordered list of discrete levels is available as its "DiscreteLevels" property.

Return The DataReference of the discrete level entity.

Type DataReference (p. 1371)

Required Arguments

Name The name of the discrete level.

Type string (p. 1438)

Example

The following example shows how the user can retrieve a discrete level of a discrete input parameter by its name.

```
system1 = GetSystem(Name="RSO")
designofExperiment1 = system1.GetContainer(ComponentName="Design of Experiment")
dOEModel1 = designofExperiment1.GetModel()
DiscreteInputParameter1 = dOEModel1.GetParameter(Name="P1")
```

```
level = DiscreteInputParameter1.GetDiscreteLevel(Name="Level 1")
```

GetOptimizationCriterion

Get the DataReference of the OptimizationCriterion associated with a Parameter. An exception is thrown if the entity is not found.

Return The DataReference of the OptimizationCriterion.

Type DataReference (p. 1371)

Required Arguments

Parameter Parent Parameter.

Type DataReference (p. 1371)

Example

The following example shows how the user can get a OptimizationCriterion to change one of its properties.

```
system1 = GetSystem(Name="RSO")
optimization1 = system1.GetContainer(ComponentName="Optimization")
optimizationModel1 = optimization1.GetModel()
parameter3 = optimizationModel1.GetParameter(Name="P3")
optimizationCriterion = parameter3.GetOptimizationCriterion()
optimizationCriterion.ObjectiveType = "eGT_MinimumPossible"
```

GetParameterStatistics

Get the DataReference of the ParameterStatistics entity associated with a Parameter.

Return The DataReference of the ParameterStatistics entity.

Type DataReference (p. 1371)

Example

The following example shows how the user can get a ParameterStatistics entity and examine its Mean property.

```
system1 = GetSystem(Name="SSA")
sixSigmaAnalysis1 = system1.GetContainer(ComponentName="Six Sigma Analysis")
sixSigmaModel1 = sixSigmaAnalysis1.GetModel()
parameter4 = sixSigmaModel1.GetParameter(Name="P4")
parameterStatistics1 = parameter4.GetParameterStatistics()
mean = parameterStatistics1.Mean
```

MisqpOptimization

Entity which performs and manages the DesignXplorer Optimization Method Component

Properties

Converged

Convergence state

Type bool (p. 1360)

Read Only Yes

DerivativeApproximationType

DerivativeApproximationType

Type DerivativeApproximationType (p. 1374)

Read Only No

DisplayText

The general property that defines the user-visible name of an entity. This property is defined for all data entities but is used only in those entities that present a label in the user interface.

Type string (p. 1438)

Read Only No

InitialFiniteDifferenceDelta

Relative Gradient Step

Type double (p. 1378)

Read Only No

MaxConvPercentage

Allowable Convergence Percentage

Type double (p. 1378)

Read Only No

MaximumNumberOfPermutations

Maximum number of permutations for discrete / manufacturable values.

Type int (p. 1394)

Read Only Yes

MaxNumCandidates

Maximum Number of Candidates

Type int (p. 1394)

Read Only No

MaxNumIterations

Maximum Number of Iterations

Type int (p. 1394)

Read Only No

NumberOfEvaluations

Number of Evaluations

Type int (p. 1394)

Read Only Yes

NumberOfFailures

Number of Failures

Type int (p. 1394)

Read Only Yes

NumCandidates

Number of Candidates

Type int (p. 1394)

Read Only Yes

NumIterations

Current Iteration

Type int (p. 1394)

Read Only Yes

SampleSetSize

Size of Generated Sample Set

Type int (p. 1394)

Read Only Yes

MOGAOptimization

Entity which performs and manages the DesignXplorer Optimization Method Component

Properties

Converged

Convergence state

Type bool (p. 1360)

Read Only Yes

CrossoverProbability

Maximum Allowable Pareto Percentage

Type double (p. 1378)

Read Only No

CurrentParetoPercentage

Pareto Percentage

Type double (p. 1378)

Read Only Yes

CurrentStabilityPercentage

Stability Percentage

Type double (p. 1378)

Read Only Yes

DisplayText

The general property that defines the user-visible name of an entity. This property is defined for all data entities but is used only in those entities that present a label in the user interface.

Type string (p. 1438)

Read Only No

MaximumNumberOfPermutations

Maximum number of permutations for discrete / manufacturable values.

Type int (p. 1394)

Read Only Yes

MaxNumCandidates

Maximum Number of Candidates

Type int (p. 1394)

Read Only No

MaxNumCycles

Maximum Number Of Cycles for OSFD algorithm.

Type int (p. 1394)

Read Only No

MaxNumIterations

Maximum Number of Iterations

Type int (p. 1394)

Read Only No

MutationProbability

Maximum Allowable Pareto Percentage

Type double (p. 1378)

Read Only No

NumberOfEvaluations

Number of Evaluations

Type int (p. 1394)

Read Only Yes

NumberOfFailures

Number of Failures

Type int (p. 1394)

Read Only Yes

NumberOfInitialSamples

Number of Initial Samples.

Type int (p. 1394)

Read Only No

NumCandidates

Number of Candidates

Type int (p. 1394)

Read Only Yes

NumIterations

Current Iteration

Type int (p. 1394)

Read Only Yes

NumSamplesPerIter

Number of Samples Per Iteration

Type int (p. 1394)

Read Only No

ParetoPercentage

Maximum Allowable Pareto Percentage

Type double (p. 1378)

Read Only No

RandomGeneratorSeed

LHS Seed

Type int (p. 1394)

Read Only No

SampleSetSize

Size of Generated Sample Set

Type int (p. 1394)

Read Only Yes

StabilityCriterion

Stability Criterion

Type double (p. 1378)

Read Only No

TypeOfInitialSampling

Type Of Initial Sampling

Type TypeOfInitialSampling (p. 1445)

Read Only No

NlpqlOptimization

Entity which performs and manages the DesignXplorer Optimization Method Component

Properties

Converged

Convergence state

Type bool (p. 1360)

Read Only Yes

DerivativeApproximationType

DerivativeApproximationType

Type DerivativeApproximationType (p. 1374)

Read Only No

DisplayText

The general property that defines the user-visible name of an entity. This property is defined for all data entities but is used only in those entities that present a label in the user interface.

Type string (p. 1438)

Read Only No

InitialFiniteDifferenceDelta

Relative Gradient Step

Type double (p. 1378)

Read Only No

MaxConvPercentage

Allowable Convergence Percentage

Type double (p. 1378)

Read Only No

MaximumNumberOfPermutations

Maximum number of permutations for discrete / manufacturable values.

Type int (p. 1394)

Read Only Yes

MaxNumCandidates

Maximum Number of Candidates

Type int (p. 1394)

Read Only No

MaxNumIterations

Maximum Number of Iterations

Type int (p. 1394)

Read Only No

NumberOfEvaluations

Number of Evaluations

Type int (p. 1394)

Read Only Yes

NumberOfFailures

Number of Failures

Type int (p. 1394)

Read Only Yes

NumCandidates

Number of Candidates

Type int (p. 1394)

Read Only Yes

NumIterations

Current Iteration

Type int (p. 1394)

Read Only Yes

SampleSetSize

Size of Generated Sample Set

Type int (p. 1394)

Read Only Yes

OptimizationCriterion

The data entity which describes the objective and constraint associated with a Parameter for an Optimization Study.

Properties

AllowedValues

Allowed Values Type of parameter if Parameter is a continuous input parameter.

Type AllowedContinuousValues (p. 1354)

Read Only No

ConstraintFirstValue

Constraint First Value used when ConstraintType is eGI_LessThanTarget, eGI_GreaterThanTarget, eGI_NearTarget or eGI_InsideBounds.

Type double (p. 1378)

Read Only No

ConstraintHandling

Constraint Handling

Type ConstraintHandlingType (p. 1367)

Read Only No

ConstraintImportance

Importance of the constraint when multiple objectives are defined.

Type ImportanceLevel (p. 1391)

Read Only No

ConstraintSecondValue

Constraint Second Value used when ConstraintType is eGI_InsideBounds.

Type double (p. 1378)

Read Only No

ConstraintType

Constraint type for the Parameter to be optimized.

Type ConstraintType (p. 1367)

Read Only No

CustomConstraintDefinitionList

List of custom constraint definitions.

Type List (p. 1400)<DataReference (p. 1371)>

Read Only No

CustomObjectiveDefinitionList

List of custom objective definitions.

Type List (p. 1400)<DataReference (p. 1371)>

Read Only No

FeasibleTolerance

Tolerance associated to the constraint value.

Type double (p. 1378)

Read Only No

FitnessInitialValue

The initial value when tolerance settings are on.

Type double (p. 1378)

Read Only No

GridInterval

Grid Interval of the parameter if Parameter is a continuous input parameter and the allowed values type is Snap To Grid.

Type double (p. 1378)

Read Only No

HistoryChart

DataReference of the associated history chart entity.

Type DataReference (p. 1371)

Read Only No

LowerBound

Lower Bound of the variation range if Parameter is an input parameter.

Type double (p. 1378)

Read Only No

ObjectiveImportance

Importance of the objective when multiple objectives are defined.

Type ImportanceLevel (p. 1391)

Read Only No

ObjectiveTargetValue

Objective Target Value if ObjectiveType is SeekTarget.

Type double (p. 1378)

Read Only No

ObjectiveType

Objective type for the Parameter to be optimized.

Type GoalType (p. 1389)

Read Only No

StartingValue

The value of the parameter in the starting point if Parameter is an input parameter and the optimization method uses a starting point.

Type double (p. 1378)

Read Only No

TargetTolerance

Tolerance associated to the Target value.

Type double (p. 1378)

Read Only No

Units

Unit of the Parameter.

Type string (p. 1438)

Read Only Yes

UpperBound

Upper Bound of the variation range if Parameter is an input parameter.

Type double (p. 1378)

Read Only No

Methods

GetChart

Get the DataReference of the HistoryChart associated with an OptimizationCriterion. An exception is thrown if the entity is not found.

Return The DataReference of the OptimizationCriterion.

Type DataReference (p. 1371)

Example

The following example shows how the user can get a HistoryChart to export its data as .csv file.

```
system = GetSystem(Name="RSO")
optimization = system.GetContainer(ComponentName="Optimization")
optimizationModel = optimization.GetModel()
parameter = optimizationModel.GetParameter(Name="P3")
criterion = parameter.GetOptimizationCriterion()
historyChart = criterion.GetChart()
historyChart.ExportData(FileName="D:/Temp/HistoryChart.csv")
```

GetCustomConstraintDefinition

Get the DataReference of a CustomConstraintProperties entity associated with an Optimization Criterion. An exception is thrown if the entity is not found.

Return The DataReference of the CustomConstraintProperties.

Type DataReference (p. 1371)

Required Arguments

ExtensionName Extension's Name.

Type string (p. 1438)

Example

The following example shows how the user can get a CustomConstraintProperties.

```
system1 = GetSystem(Name="DOP")
optimization1 = system1.GetContainer(ComponentName="Optimization")
optimizationModel1 = optimization1.GetModel()
outputParameter1 = optimizationModel1.GetParameter(Name="P4")
optimizationCriterion1 = outputParameter1.CreateOptimizationCriterion()
customConstraintDefinition1 = optimizationCriterion1.GetCustomConstraintDefinition(ExtensionName="CommonCo
customConstraintDefinition1.FeasibleTolerance = 0.02
```

GetCustomObjectiveDefinition

Get the DataReference of a CustomObjectiveProperties entity associated with an Optimization Criterion. An exception is thrown if the entity is not found.

Return The DataReference of the CustomObjectiveProperties.

Type DataReference (p. 1371)

Required Arguments

ExtensionName Extension's Name.

Type string (p. 1438)

Example

The following example shows how the user can get a CustomObjectiveProperties.

```
system1 = GetSystem(Name="DOP")
optimization1 = system1.GetContainer(ComponentName="Optimization")
optimizationModel1 = optimization1.GetModel()
outputParameter1 = optimizationModel1.GetParameter(Name="P4")
optimizationCriterion1 = outputParameter1.CreateOptimizationCriterion()
customObjectiveDefinition1 = optimizationCriterion1.GetCustomObjectiveDefinition(ExtensionName="CommonObje
customObjectiveDefinition1.RelativeObjectiveWeight = 0.9
```

OptimizationMethod

Entity which wraps and manages the external Optimization Method for the Optimization component

Properties

No Properties.

OptimizationModel

Entity which performs and manages the DesignXplorer Optimization Component

Properties

DisplayText

The general property that defines the user-visible name of an entity. This property is defined for all data entities but is used only in those entities that present a label in the user interface.

Type string (p. 1438)

Read Only No

EstimatedNumberofEvaluations

Estimated Number of Evaluations for the automated optimization method.

Type int (p. 1394)

Read Only Yes

ExportDesignPoints

If True and PreserveDesignPoints is True as well, export project for each preserved Design Point.

Type bool (p. 1360)

Read Only No

Method

Optimization Method

Type DataReference (p. 1371)

Read Only No

MethodName

Type of the optimization

Type string (p. 1438)

Read Only No

MethodSelection

Type of the optimization method selection.

Type OptimizationMethodSelection (p. 1410)

Read Only No

NumberOfRetries

Indicates the number of times DX will try to update the failed design points.

Type int (p. 1394)

Read Only No

PreserveDesignPoints

If True, preserve the Design Points at the project level after the component Update.

Type bool (p. 1360)

Read Only No

RetainDesignPoints

If True and PreserveDesignPoints is True as well, retain data for each preserved Design Point.

Type bool (p. 1360)

Read Only No

RetryDelay

Indicates how much time will elapse between tries. This option is only applicable when NumberOfRetries is greater than 0, otherwise it has no effect.

Type Quantity (p. 1422)

Read Only No

RunTimeIndex

Type of the Run Time Index to select automatically the optimization method.

Type RunTimeIndex (p. 1428)

Read Only No

ToleranceSettings

If True, enable fitness terms at Optimization Criterion level.

Type bool (p. 1360)

Read Only No

VerifyCandidatePoints

If True, verifies the candidates by a design points update.

Type bool (p. 1360)

Read Only No

Methods

CreateChart

Creates a Chart entity. The chart must be updated once after its creation by invoking its Update method. Then changes to its properties will update the chart automatically.

Return The DataReference of the Chart.

Type DataReference (p. 1371)

Required Arguments

ChartType Type of chart to be created. The possible values depend on the type of Model. For instance, a ResponseSurface model accepts Spider, LocalSensitivity and Response chart while a CorrelationModel accepts CorrelationMatrix, DeterminationMatrix and CorrelationScatter charts.

Type ChartType (p. 1363)

Optional Arguments

DisplayText Displayed name of the chart. If not specified, a default name is applied, depending on the value of ChartType.

Type string (p. 1438)

Example

The following example shows how to create a chart.

```
container = system1.GetContainer(ComponentName="Optimization")
model = container.GetModel()
TradeoffChart = model.CreateChart(ChartType="eChartTradeoff")
TradeoffChart.Update()
```

CreateCustomCandidatePoint

Creates a custom candidate point. The Expressions dictionary can be used to specify a value or a quantity for some or all of the input parameters. In the context of a response surface based optimization, the output values are evaluated from the response surface.

Return The created entity.

Type DataReference (p. 1371)

Optional Arguments

DisplayText DisplayText of the created entity. If not specified, a default name of the form "Custom Candidate Point [#]" is used.

Type string (p. 1438)

Expressions The values for each input parameter. If not specified, each parameter is initialized to its current value.

Type IDictionary (p. 1375)<DataReference (p. 1371), string (p. 1438)>

Example

The following example shows how to create a CustomCandidatePoint entity on an Optimization model based on a ResponseSurface.

```
container = system1.GetContainer(ComponentName="Optimization")
model = container.GetModel()
inputParameter1 = model.GetParameter(Name="P1")
inputParameter2 = model.GetParameter(Name="P2")
customCandidatePoint = model.CreateCustomCandidatePoint( DisplayText="Existing Design",
                  Expressions={inputParameter1: "2.01", inputParameter2: "15.5"})
```

CreateParameterRelationship

Creates a Parameter Relationship

Return The DataReference of the ParameterRelationship.

Type DataReference (p. 1371)

Optional Arguments

LeftExpression Left Expression.

Type string (p. 1438)

RightExpression Right Expression.

Type string (p. 1438)

Type Parameter Relationship Type.

Type ParameterRelationshipType (p. 1413)

Default Value ePRT_LessThanOrEqualTo

Example

The following example shows how to create an ParameterRelationship entity.

```
container = system1.GetContainer(ComponentName="Optimization")
model = container.GetModel()
parameterRelationship = model.CreateParameterRelationship(LeftExpression="P1+P2",RightExpression="P3",Type
parameterRelationship2 = model.CreateParameterRelationship(LeftExpression="P4+P5",RightExpression="10[inch
```

DeleteCharts

Deletes a list of Chart entities.

Required Arguments

Charts List of Chart entities to delete.

Type List (p. 1400)<DataReference (p. 1371)>

Example

The following example shows how to delete existing charts.

```
container = system1.GetContainer(ComponentName="Optimization")
model = container.GetModel()
chart1 = model.GetChart(Name="TradeoffChart 1")
chart2 = model.GetChart(Name="SamplesChart 1")
model.DeleteCharts(Charts=[chart1, chart2])
```

DeleteCustomCandidatePoints

Deletes a list of CustomCandidatePoint entities.

Required Arguments

CustomCandidatePoints CustomCandidatePoint entities to delete.

Type List (p. 1400)<DataReference (p. 1371)>

Example

The following example shows how to delete existing CustomCandidatePoint entities from an optimization model.

```
container = system1.GetContainer(ComponentName="Optimization")
model = container.GetModel()
customCandidatePoint1 = model.GetCustomCandidatePoint(Name="CustomCandidatePoint 1")
customCandidatePoint2 = model.GetCustomCandidatePoint(Name="CustomCandidatePoint 2")
model.DeleteCustomCandidatePoints(CustomCandidatePoints=[customCandidatePoint1, customCandidatePoint2])
```

DeleteObjectives

Deletes a list of OptimizationCriterion entities.

Required Arguments

OptimizationCriteria List of Objective entities to delete.

Type List (p. 1400)<DataReference (p. 1371)>

Example

The following example shows how to delete existing OptimizationCriterion entities.

```
container = system1.GetContainer(ComponentName="Optimization")
model = container.GetModel()
```

```
parameter1 = model.GetParameter(Name="P1")
optimizationCriterion1 = parameter1.GetOptimizationCriterion()
parameter3 = model.GetParameter(Name="P3")
optimizationCriterion3 = parameter3.GetOptimizationCriterion()
model.DeleteOptimizationCriteria(OptimizationCriteria=[optimizationCriterion1, optimizationCriterion3])
```

DeleteParameterRelationships

Deletes a list of ParameterRelationship entities.

Required Arguments

ParameterRelationships DataReferences of the entities to delete

Type List (p. 1400)<DataReference (p. 1371)>

Example

The following example shows how to delete existing ParameterRelationships.

```
container = system1.GetContainer(ComponentName="Optimization")
model = container.GetModel()
pr1 = model.GetParameterRelationship(Name="ParameterRelationShip 1")
pr2 = model.GetParameterRelationship(Name="ParameterRelationShip 2")
pr3 = model.GetParameterRelationship(Name="ParameterRelationShip 3")
model.DeleteParameterRelationships(ParameterRelationships=[pr1, pr2, pr3])
```

DuplicateChart

Duplicates a Chart entity. The chart must be updated once after its creation by invoking its Update method. Then changes to its properties will update the chart automatically.

Return The DataReference of the Chart.

Type DataReference (p. 1371)

Required Arguments

Chart The source chart to duplicate.

Type DataReference (p. 1371)

Optional Arguments

DisplayText Displayed name of the chart. If not specified, a default name is applied, depending on the value of ChartType.

Type string (p. 1438)

TargetModel The model on which the duplicated chart is created. If this parameter is not set, the model of the source chart is used.

Type DataReference (p. 1371)

TargetResponsePoint Parent TargetResponsePoint of the chart. If this parameter is not set, the response point of the source chart is used if applicable.

Type DataReference (p. 1371)

Example

The following example shows how to duplicate a chart.

```
designofExperiment1 = system1.GetContainer(ComponentName="Design of Experiment")
dOEModel1 = designofExperiment1.GetModel()
designPointsCurvesChart1 = dOEModel1.GetChart(Name="DesignPointsCurves")
chart1 = dOEModel1.DuplicateChart(Chart=designPointsCurvesChart1)
chart1.Update()
```

ExportData

Export the data of model's entities to a csv file. These entities can be candidate points, custom candidate points, charts, parametric tables or input parameters.

Required Arguments

Entities An optional list of entities containing data to be exported in the same file.

Type List (p. 1400)<DataReference (p. 1371)>

FileName The exported file name.

Type string (p. 1438)

Optional Arguments

AppendMode True to append to an existing csv file, False to overwrite it.

Type bool (p. 1360)

Default Value False

Example

The following example shows how the user can export the candidate points of an optimization component.

```
container = system1.GetContainer(ComponentName="Optimization")
model = container.GetModel()
candidates = model.GetCandidatePoints()
model.ExportData(FileName="doe.csv", Entities=candidates)
```

GetCandidatePoint

Get the DataReference of a CandidatePoint entity. An exception is thrown if the entity is not found.

Return The DataReference of the CandidatePoint entity.

Type DataReference (p. 1371)

Required Arguments

Name Name of the CandidatePoint to retrieve.

Type string (p. 1438)

Example

The following example shows how to retrieve an existing CandidatePoint entity.

```
system1 = GetSystem(Name="DOP")
container = system1.GetContainer(ComponentName="Optimization")
model = container.GetModel()
candidatePoint = model.GetCandidatePoint(Name="CandidatePoint 1")
```

GetCandidatePoints

Get the DataReferenceSet of all of the existing CandidatePoint entities on the model.

Return The DataReferenceSet of the CandidatePoint entities.

Type DataReferenceSet (p. 1371)

Example

The following example shows how to retrieve the candidate points of an optimization model.

```
system1 = GetSystem(Name="DOP")
container = system1.GetContainer(ComponentName="Optimization")
model = container.GetModel()
candidatePoints = model.GetCandidatePoints()
```

GetChart

Query to return the chart reference for a given model and chart name.

Return The DataReference of the chart.

Type DataReference (p. 1371)

Required Arguments

Name Name of the chart.

Type string (p. 1438)

GetConvergenceChart

Query to return the chart reference for a given model and chart name.

Return The DataReference of the chart.

Type DataReference (p. 1371)

GetCustomCandidatePoint

Get the DataReference of a CustomCandidatePoint entity. An exception is thrown if the entity is not found.

Return The DataReference of the CustomCandidatePoint entity.

Type DataReference (p. 1371)

Required Arguments

Name Name of the CustomCandidatePoint to retrieve.

Type string (p. 1438)

Example

The following example shows how to retrieve an existing CustomCandidatePoint entity.

```
system1 = GetSystem(Name="DOP")
container = system1.GetContainer(ComponentName="Optimization")
model = container.GetModel()
customCandidatePoint = model.GetCustomCandidatePoint(Name="CustomCandidatePoint 1")
```

GetCustomCandidatePoints

Get the DataReferenceSet of all of the existing CustomCandidatePoint entities of the model.

Return The DataReferenceSet of the CustomCandidatePoint entities.

Type DataReferenceSet (p. 1371)

Example

The following example shows how to retrieve the custom candidate points of an optimization model.

```
system1 = GetSystem(Name="DOP")
container = system1.GetContainer(ComponentName="Optimization")
model = container.GetModel()
customCandidatePoints = model.GetCustomCandidatePoints()
```

GetDesignPointReportSetting

Get the DataReference of a DesignPointReportSetting entity associated with a Model. An exception is thrown if the entity is not found.

Return The DataReference of the DesignPointReportSetting.

Type DataReference (p. 1371)

Example

The following example shows how the user can get a DesignPointReportSetting to change one of its properties.

```
system1 = GetSystem(Name="GDO")
designofExperiment1 = system1.GetContainer(ComponentName="Design of Experiment")
dOEModel1 = designofExperiment1.GetModel()
designPointReportSettingEntity1 = dOEModel1.GetDesignPointReportSetting()
designPointReportSettingEntity1.DesignPointReportImage = "FFF-Results:Figure001.png"
```

GetOptimizationCriteria

Get the DataReferenceSet of all of the existing OptimizationCriterion entities of the model.

Return The DataReference of the OptimizationCriterion entities.

Type DataReferenceSet (p. 1371)

Required Arguments

Model Parent Optimization model

Type DataReference (p. 1371)

Example

The following example shows how to retrieve the objectives of an optimization model, as a set of OptimizationCriterion entities.

```
system1 = GetSystem(Name="DOP")
container = system1.GetContainer(ComponentName="Optimization")
model = container.GetModel()
criteria = model.GetOptimizationCriteria()
```

GetParameter

Get the DataReference of a Parameter. An exception is thrown if the entity is not found.

Return The DataReference of the Parameter.

Type DataReference (p. 1371)

Required Arguments

Name Name of the Parameter.

Type string (p. 1438)

Example

The following example shows how the user can get a parameter of a model to change one of its properties.

```
system1 = GetSystem(Name="RSO")
designofExperiment1 = system1.GetContainer(ComponentName="Design of Experiment")
dOEModel1 = designofExperiment1.GetModel()
inputParameter1 = dOEModel1.GetParameter(Name="P1")
inputParameter1.LowerBound = 1
```

GetParameterRelationship

Get the DataReference of a ParameterRelationship entity. An exception is thrown if the entity is not found.

Return The DataReference of the ParameterRelationship entity.

Type DataReference (p. 1371)

Required Arguments

Name Name of the ParameterRelationship to retrieve.

Type string (p. 1438)

Example

The following example shows how the user can retrieve an existing ParameterRelationship entity.

```
container = system1.GetContainer(ComponentName="Optimization")
model = container.GetModel()
parameterRelationship = model.GetParameterRelationship(Name="ParameterRelationShip 1")
```

GetParameterRelationships

Get the Parameter Relationships associated to a Model. If the optional argument Enabled is not specified, the query returns all Parameter Relationships. If it is specified and True, the query returns only enabled Parameter Relationships. If it is False, the query returns only disabled Parameter Relationships.

Return The DataReferenceSet of the ParameterRelationship entities.

Type DataReferenceSet (p. 1371)

Optional Arguments

Enabled If True, the query returns only enabled Parameter Relationships. If False, the query returns only disabled Parameter Relationships. If the argument is ommitted, the query returns all Parameter Relationships.

Type bool (p. 1360)

Example

The following example shows how the user can retrieve all the Parameter Relationships defined on a model.

```
container = system1.GetContainer(ComponentName="Optimization")
model = container.GetModel()
allParameterRelationships = model.GetParameterRelationships()
enabledParameterRelationships = model.GetParameterRelationships(Enabled="True")
disabledParameterRelationships = model.GetParameterRelationships(Enabled="False")
```

GetParameters

Get the DataReferences of the InputParameter and OutputParameter of the model. If the optional argument InputParameters is not specified, the query returns all parameters. If it is specified and True, the query returns only input parameters. If it is False, the query returns only output parameters.

Return The DataReferences of the Parameters

Type DataReferenceSet (p. 1371)

Optional Arguments

InputParameters If True, the query returns only input parameters. If False, the query returns only output parameters. If the argument is ommitted, the query returns all parameters.

Type bool (p. 1360)

Example

The following example shows how the user can get the parameters of a model.

```
system1 = GetSystem(Name="RSO")
designofExperiment1 = system1.GetContainer(ComponentName="Design of Experiment")
dOEModel1 = designofExperiment1.GetModel()
parameters = dOEModel1.GetParameters()
```

GetParametricTable

Get the DataReference of ParametricTable. An exception is thrown if the entity is not found. Names of the tables generated internally are: "DesignPoints", "CorrelationMatrix", "CorrelationScatter", "MinDesignPoints", "MaxDesignPoints", "ResponsePoints", "DeterminationMatrix".

Return The DataReference of the ParametricTable

Type DataReference (p. 1371)

Required Arguments

Name Name of the ParametricTable

Type string (p. 1438)

Example

The following example shows how the user can get a ParametricTable to add a new row and set values.

```
system1 = GetSystem(Name="RSO")
designofExperiment1 = system1.GetContainer(ComponentName="Design of Experiment")
dOEModel1 = designofExperiment1.GetModel()
parametricTable = dOEModel1.GetParametricTable(Name="DesignPoints")
parametricTable.AddRow()
parametricTable.SetCellValue(RowIndex=9,ColumnIndex=0,Value="2.1")
```

ImportCustomCandidatePointsFromDps

Create custom candidate points in an Optimization model by importing points with a query to the DPS.

Return The imported entities.

Type DataReferenceSet (p. 1371)

Required Arguments

QueryString The query.

Type string (p. 1438)

Example

The following example shows how the user can create custom candidate points with a query to the DPS. Up to three design points with the best value for the fitness will be imported as custom candidate points.

```
container = system1.GetContainer(ComponentName="Optimization")
model = container.GetModel()
customCandidatePoints = model.ImportCustomCandidatePointsFromDps( QueryString="?eval_status=evaluated&sort
```

SendStudyToDpsProject

Send optimization study to the Design Point Service.

Return State of the definition.

Type DefinitionState (p. 1373)

Example

The following example shows how the user can send the fitness definition of the optimization model to the single configuration in the connected Design Point Service.

The current fitness definition is overwritten.

```
system1 = GetSystem(Name="DOP")
container = system1.GetContainer(ComponentName="Optimization")
model = container.GetModel()
model.SendStudyToDpsProject()
```

SetVariationReferencePoint

Sets a point as the reference to calculate the variation of the parameters in each of the candidate and custom candidate points of the container. The variation is calculated by the command and can be retrieved for each parameter of each point.

Required Arguments

Point Point entity to set as the reference point.

Type DataReference (p. 1371)

Optional Arguments

Source The source of the output values to set as the reference.

Type OutputSource (p. 1412)

Default Value Simulation

Example

The following example shows how to set a custom candidate point as the reference and how to retrieve the calculated variation on the parameters of a second candidate point.

```
container = system1.GetContainer(ComponentName="Optimization")
model = container.GetModel()
customCandidatePoint = model.GetCustomCandidatePoint(Name="CustomCandidatePoint")
model.SetVariationReferencePoint(Point=customCandidatePoint, Source="Simulation")
bestCandidate = model.GetCandidatePoint(Name="CandidatePoint")
variationOfOutputsAsDecimals = bestCandidate.GetOutputValues(Source="Simulation", ValueType="VariationToRe
```

VerifyPoints

Verify the CandidatePoint entities by performing a design point update to generate Simulation output values, so that they can be compared to the ResponseSurface output values. This command has no effect if the optimization model is used as part of a Direct Optimization system.

Required Arguments

Points List of points to verify.

Type List (p. 1400)<DataReference (p. 1371)>

Optional Arguments

PullFromCacheOnly If True, the command attempts to pull output values from the design point cache but does not trigger any design point update.

Type bool (p. 1360)

Default Value False

Example

The following example shows how the user can verify all the candidate points generated by the optimization model and extract the existing output parameter values from the first candidate point.

```
system1 = GetSystem(Name="RSO")
container = system1.GetContainer(ComponentName="Optimization")
model = container.GetModel()
candidatePoints = model.GetCandidatePoints()
model.VerifyPoints(Points=candidatePoints)
responseSurfaceValues = candidatePoints[0].GetOutputValues(Source="ResponseSurface")
verifiedValues = candidatePoints[0].GetOutputValues(Source="Simulation")
```

OutputParameter

Output parameter entity for DesignXplorer.

Properties

DisplayText

The general property that defines the user-visible name of an entity. This property is defined for all data entities but is used only in those entities that present a label in the user interface.

Type string (p. 1438)

Read Only No

EnableAutoRefinement

Determines whether the Auto Refinement is applicable to the output parameter.

Type bool (p. 1360)

Read Only No

IgnoreForFiltering

Determines whether the parameter filtering is not applicable to the output parameter in the correlation context.

Type bool (p. 1360)

Read Only No

InheritFromModelSettings

Determines whether the Maximum Predicted Relative Error defined at the Model level is applicable to the output parameter.

Type bool (p. 1360)

Read Only No

LowerBound

Minimum value extracted from existing design points and/or sample sets.

Type double (p. 1378)

Read Only Yes

MaximumPredictedError

Maximum Predicted Error for an output parameter with the GARS algorithm. This is the maximum predicted error for the selected output parameter.

Type double (p. 1378)

Read Only Yes

MaxPredictedRelativeError

Maximum Relative Error targeted for an output parameter when refining with the Kriging algorithm. This is the maximum predicted relative error that is acceptable for the selected output parameter.

Type double (p. 1378)

Read Only No

PredictedRelativeError

Current value of the Predicted Relative Error when refining with the Kriging algorithm

Type double (p. 1378)

Read Only Yes

Scaling

Scaling

Type bool (p. 1360)

Read Only No

Tolerance

Maximum Error targeted for an output parameter when refining with the GARS algorithm. This is the maximum predicted error that is acceptable for the selected output parameter.

Type double (p. 1378)

Read Only No

TransformationType

Transformation Type

Type TransformationType (p. 1443)

Read Only No

Units

Units

Type string (p. 1438)

Read Only Yes

UpperBound

Maximum value extracted from existing design points and/or sample sets.

Type double (p. 1378)

Read Only Yes

Methods

CreateOptimizationCriterion

Creates an OptimizationCriterion entity associated to a parameter.

Return The DataReference of the OptimizationCriterion.

Type DataReference (p. 1371)

Required Arguments

Parameter The parameter on which the criterion is created.

Type DataReference (p. 1371)

Example

The following example shows how to create an OptimizationCriterion entity.

```
container = system1.GetContainer(ComponentName="Optimization")
model = container.GetModel()
parameter1 = model.GetParameter(Name="P1")
optimizationCriterion = parameter1.CreateOptimizationCriterion()
```

GetOptimizationCriterion

Get the DataReference of the OptimizationCriterion associated with a Parameter. An exception is thrown if the entity is not found.

Return The DataReference of the OptimizationCriterion.

Type DataReference (p. 1371)

Required Arguments

Parameter Parent Parameter.

Type DataReference (p. 1371)

Example

The following example shows how the user can get a OptimizationCriterion to change one of its properties.

```
system1 = GetSystem(Name="RSO")
optimization1 = system1.GetContainer(ComponentName="Optimization")
optimizationModel1 = optimization1.GetModel()
parameter3 = optimizationModel1.GetParameter(Name="P3")
optimizationCriterion = parameter3.GetOptimizationCriterion()
optimizationCriterion.ObjectiveType = "eGT_MinimumPossible"
```

GetParameterStatistics

Get the DataReference of the ParameterStatistics entity associated with a Parameter.

Return The DataReference of the ParameterStatistics entity.

Type DataReference (p. 1371)

Example

The following example shows how the user can get a ParameterStatistics entity and examine its Mean property.

```
system1 = GetSystem(Name="SSA")
sixSigmaAnalysis1 = system1.GetContainer(ComponentName="Six Sigma Analysis")
sixSigmaModel1 = sixSigmaAnalysis1.GetModel()
parameter4 = sixSigmaModel1.GetParameter(Name="P4")
parameterStatistics1 = parameter4.GetParameterStatistics()
mean = parameterStatistics1.Mean
```

ParameterRelationship

Entity which manages the calculated Statistics of a parameter for a Six-Sigma Component

Properties

Enabled

True if the Parameter is enabled for the current study.

Type bool (p. 1360)

Read Only No

ErrorMessage

Error Message associated to the Parameter Relationship.

Type string (p. 1438)

Read Only Yes

LeftExpression

Left Expression of Parameter Relationship.

Type string (p. 1438)

Read Only No

LeftExpressionQuantityName

Quantity Name of Left Expression of Parameter Relationship.

Type string (p. 1438)

Read Only No

LeftExpressionValue

Value of the left expression (as a quantity string), or an error message.

Type string (p. 1438)

Read Only Yes

RelationshipType

Type of Parameter Relationship.

Type ParameterRelationshipType (p. 1413)

Read Only No

RightExpression

Right Expression of Parameter Relationship.

Type string (p. 1438)

Read Only No

RightExpressionQuantityName

Quantity Name of Right Expression of Parameter Relationship.

Type string (p. 1438)

Read Only No

RightExpressionValue

Value of the right expression (as a quantity string), or an error message.

Type string (p. 1438)

Read Only Yes

Methods

DuplicateParameterRelationship

Duplicates a Parameter Relationship.

Return The created entity.

Type DataReference (p. 1371)

Optional Arguments

TargetModel The model on which the Parameter Relationship is created. If this parameter is not set, the model of the source Parameter Relationship is used.

Type DataReference (p. 1371)

Example

The following example shows how to duplicate a Parameter Relationship.

```
container = system1.GetContainer(ComponentName="Optimization")
model = container.GetModel()
pr1 = model.GetParameterRelationship(Name="ParameterRelationShip")
pr2 = pr1.DuplicateParameterRelationship()
container = system1.GetContainer(ComponentName="Optimization 1")
model2 = container.GetModel()
pr3 = pr1.DuplicateParameterRelationship(TargetModel = model2)
```

GetChart

Get the DataReference of the ParameterRelationshipChart associated with a ParameterRelationship. An exception is thrown if the entity is not found.

Return The DataReference of the ParameterRelationshipChart.

Type DataReference (p. 1371)

Example

The following example shows how the user can get a ParameterRelationshipChart to export its data as .csv file.

```
system1 = GetSystem(Name="RSO")
optimization1 = system1.GetContainer(ComponentName="Optimization")
optimizationModel1 = optimization1.GetModel()
parameterRelationship1 = optimizationModel1.GetParameterRelationship(Name="ParameterRelationship")
```

```
chart1 = parameterRelationship1.GetChart()
Parameters.ExportData(Data=chart1,FileName="E:/Temp/Relationship.csv")
```

ParameterRelationshipChart

The data entity which describes a ParameterRelationship chart. It allows you to visualize how parameter relationship evolves during the optimization process, depending on the optimizer.

Properties

DisplayParameterFullName

If True, the legend of the chart contains the full name of the parameters. Otherwise it contains the short name such as "P1".

Type bool (p. 1360)

Read Only No

DisplayText

The general property that defines the user-visible name of an entity. This property is defined for all data entities but is used only in those entities that present a label in the user interface.

Type string (p. 1438)

Read Only No

IsUpToDate

True if the entity is up-to-date.

Type bool (p. 1360)

Read Only No

ScreeningOptimization

Entity which performs and manages the DesignXplorer Optimization Method Component

Properties

Converged

Convergence state

Type bool (p. 1360)

Read Only Yes

DisplayText

The general property that defines the user-visible name of an entity. This property is defined for all data entities but is used only in those entities that present a label in the user interface.

Type string (p. 1438)

Read Only No

MaximumNumberOfPermutations

Maximum number of permutations for discrete / manufacturable values.

Type int (p. 1394)

Read Only Yes

MaxNumCandidates

Maximum Number of Candidates

Type int (p. 1394)

Read Only No

NumberOfEvaluations

Number of Evaluations

Type int (p. 1394)

Read Only Yes

NumberOfFailures

Number of Failures

Type int (p. 1394)

Read Only Yes

NumberOfSamples

Number of Samples

Type int (p. 1394)

Read Only No

NumCandidates

Number of Candidates

Type int (p. 1394)

Read Only Yes

SampleSetSize

Size of Generated Sample Set

Type int (p. 1394)

Read Only Yes

DX Evaluation Container

This container holds data for an instance of the Evaluation.

Data Entities

CandidatePoint

The CandidatePoint data entity is a candidate point automatically generated by an optimization.

Properties

DisplayText

The general property that defines the user-visible name of an entity. This property is defined for all data entities but is used only in those entities that present a label in the user interface.

Type string (p. 1438)

Read Only No

Methods

CanUpdate

Returns true if the entity can be updated to produce the specific source of output values. As an example, the query will return false for Source=ResponseSurface if the candidate point was generated by a Direct Optimization system.

Return Results of the query telling if the Point entity can be updated to produce the specified source of output values.

Type bool (p. 1360)

Optional Arguments

Source Source of the output values to query for.

Type OutputSource (p. 1412)

Default Value Simulation

Example

The following example shows how to check if a CandidatePoint entity can be updated and how to update it to obtain the Simulation output values.

```
system1 = GetSystem(Name="RSO")
container = system1.GetContainer(ComponentName="Optimization")
model = container.GetModel()
candidatePoint = model.GetCandidatePoint(Name="CandidatePoint 1")
if candidatePoint.CanUpdate(Source="Simulation"):
   candidatePoint.Update(Source="Simulation")
```

GetInputValues

Get the values of the input parameters. Depending on the parameter definition, each value can be a Quantity, a real, a string or an integer.

Return The dictionary of the input parameter values.

Type ReadOnlyDictionary (p. 1423)<DataReference (p. 1371), Object (p. 1409)>

Optional Arguments

ValueType The type of parameter value to return.

Type ParameterValueType (p. 1413)

Default Value ActualValue

Example

The following example shows how to retrieve a candidate point and then extract its input parameter values.

```
system1 = GetSystem(Name="DOP")
container = system1.GetContainer(ComponentName="Optimization")
model = container.GetModel()
candidatePoint = model.GetCandidatePoint(Name="CandidatePoint 1")
dictInputValues = candidatePoint.GetInputValues()
```

GetOutputValues

Get the values of the output parameters for the specified source of output values.

Return The dictionary of the output parameter values.

Type ReadOnlyDictionary (p. 1423)<DataReference (p. 1371), Object (p. 1409)>

Optional Arguments

Source The source of the output values to return.

Type OutputSource (p. 1412)

Default Value Simulation

ValueType The type of parameter value to return.

Type ParameterValueType (p. 1413)

Default Value ActualValue

Example

The following example shows how to retrieve a candidate point and then extract its output parameter values.

```
system1 = GetSystem(Name="DOP")
container = system1.GetContainer(ComponentName="Optimization")
model = container.GetModel()
candidatePoint = model.GetCandidatePoint(Name="CandidatePoint 1")
responseSurfaceOutputValues = candidatePoint.GetOutputValues(Source="ResponseSurface")
simulationOutputValues = candidatePoint.GetOutputValues(Source="Simulation")
```

GetState

Returns the state of the entity for the specified source of output values. A different state is available for each source of output values.

Return State for the specified nature of output values.

Type UpdatableEntityState (p. 1446)

Optional Arguments

Source Source of output values to query for.

Type OutputSource (p. 1412)

Default Value Simulation

Example

The following example shows how to check the state of the Simulation's source of the output values of a CandidatePoint entity.

```
system1 = GetSystem(Name="RSO")
container = system1.GetContainer(ComponentName="Optimization")
model = container.GetModel()
candidatePoint = model.GetCandidatePoint(Name="CandidatePoint 1")
state = candidatePoint.GetState(Source="Simulation")
```

GetValue

Get the value for a given parameter and, if the parameter is an output parameter, a given source of output values. Depending on the parameter definition, each value can be a Quantity, a real, a string or an integer.

Return The retrieve parameter value.

Type Object (p. 1409)

Required Arguments

Parameter The Parameter for which the value is requested.

Type DataReference (p. 1371)

Optional Arguments

Source If Parameter is an output parameter, the source of the output value to return.

Type OutputSource (p. 1412)

Default Value Simulation

ValueType The type of parameter value to return.

Type ParameterValueType (p. 1413)

Default Value ActualValue

Example

The following example shows how to retrieve a candidate point generated by the optimization model, and then extract selected parameter values.

```
system1 = GetSystem(Name="DOP")
container = system1.GetContainer(ComponentName="Optimization")
model = container.GetModel()
candidatePoint = model.GetCandidatePoint(Name="CandidatePoint 1")
inputParam1 = model.GetParameter(Name="P1")
parameterValue1 = candidatePoint.GetValue(Parameter=inputParam1)
outputParam4 = model.GetParameter(Name="P4")
parameterValue2 = candidatePoint.GetValue(Parameter=outputParam4, Source="Simulation")
print parameterValue2.Value
```

GetValues

Get the values of all input parameters and the values of all output parameters for the specified source of output values. Depending on the parameter definition, each value can be a Quantity, a real, a string or an integer.

Return The dictionary of the parameter values.

Type ReadOnlyDictionary (p. 1423)<DataReference (p. 1371), Object (p. 1409)>

Optional Arguments

Source For output parameters, the source of the output values to return.

Type OutputSource (p. 1412)

Default Value Simulation

ValueType The type of parameter value to return.

Type ParameterValueType (p. 1413)

Default Value ActualValue

Example

The following example shows how to retrieve a candidate point and then extract all of its parameter values.

```
system1 = GetSystem(Name="DOP")
container = system1.GetContainer(ComponentName="Optimization")
model = container.GetModel()
candidatePoint = model.GetCandidatePoint(Name="CandidatePoint 1")
parameterValues = candidatePoint.GetValues(Source="Simulation")
```

Update

Updates a point entity to provide the output parameter values of the requested source. If the requested source is ResponseSurface, the output values are generated by evaluating the response surface model, if available. If the requested source is Simulation, the output values are generated by triggering a DesignPoint update. This command applies to CandidatePoint and CustomCandidatePoint entities.

Return The dictionary of parameters with their values.

Type ReadOnlyDictionary (p. 1423)<DataReference (p. 1371), Object (p. 1409)>

Optional Arguments

Source Source of output values to produce. The update method depends on this parameter.

Type OutputSource (p. 1412)

Default Value Simulation

Example

The following example shows how to update an existing candidate point to obtain Simulation output values.

```
system1 = GetSystem(Name="RSO")
container = system1.GetContainer(ComponentName="Optimization")
candidate1 = container.GetCandidatePoint(Name="CandidatePoint 1")
outputValues = candidate1.Update(Source="Simulation")
```

CandidatesChart

The data entity which describes the Candidate Points chart. The Candidate Points chart allows you to view different kinds of information about candidate points. It allows you specify one or more parameters for which you want to display candidates data. Color-coding and a legend make it easy to view and interpret samples, candidate points identified by the optimization, candidates inserted manually, and candidates for which output values have been verified by a design point update. You can specify the chart's properties to control the visibility of each axis, feasible samples, candidates you've inserted manually, and candidates with verified output values.

Properties

CandidatesColoringMethod

Coloring method used to draw the chart.

Type CandidatesColoringMethods (p. 1361)

Read Only No

DisplayParameterFullName

If True, the legend of the chart contains the full name of the parameters. Otherwise it contains the short name such as "P1".

Type bool (p. 1360)

Read Only No

DisplayText

The general property that defines the user-visible name of an entity. This property is defined for all data entities but is used only in those entities that present a label in the user interface.

Type string (p. 1438)

Read Only No

IsUpToDate

True if the entity is up-to-date.

Type bool (p. 1360)

Read Only No

ShowCandidates

If True, the candidates are displayed on the chart.

Type bool (p. 1360)

Read Only No

ShowCustomCandidates

If True, any custom candidates are displayed on the chart.

Type bool (p. 1360)

Read Only No

ShowSamples

If True, the samples are displayed on the chart.

Type bool (p. 1360)

Read Only No

ShowStartingPoint

If True, the starting point is displayed on the chart.

Type bool (p. 1360)

Read Only No

ShowVerifiedCandidates

If True, any verified candidates are displayed on the chart.

Type bool (p. 1360)

Read Only No

Methods

EnableParameters

Enable or disable a list of parameters in a chart.

Required Arguments

IsEnabled False to disable the parameters, or True (default) to enable them.

Type bool (p. 1360)

Optional Arguments

Parameters Parameters to enable or disable, or all parameters if not specified.

Type List (p. 1400)<DataReference (p. 1371)>

Example

The following example shows how to disable two parameters, and then how to enable all parameters by omitting the Parameters optional parameter. This example uses a Correlation Matrix chart. The method also applies for other types of charts like LocalSensitivity, SpiderChart, etc.

```
container = system1.GetContainer(ComponentName="Correlation")
model = container.GetModel()
param1 = model.GetParam(Name="P1")
param4 = model.GetParam(Name="P4")
chart = model.GetChart(Name="Correlation Matrix 1")
chart.EnableParameters( Parameters=[param1;param4], IsEnabled=false )
chart.EnableParameters()
```

ExportData

Export the data of a DesignXplorer entity to a csv file. The entity can be one of the DesignXplorer chart entities, a ParametricTable or an InputParameter entity.

Required Arguments

FileName The exported file name.

Type string (p. 1438)

Optional Arguments

AppendMode True to append to an existing csv file, False to overwrite it.

Type bool (p. 1360)

Default Value False

Example

The following example shows how the user can export the table of Design Points and then the Parameters Parallel chart of a Design of Experiments component.

```
container = system1.GetContainer(ComponentName="Design of Experiment")
model = container.GetModel()
parametricTable = model.GetParametricTable(Name="DesignPoints")
parametricTable.ExportData(FileName="doe.csv")
chart = model.GetChart(Name="DesignPointsParallel")
chart.ExportData(FileName="doe.csv", AppendMode=True)
```

Update

Updates the chart by generating all results or data required to plot it. If the chart is already up-to-date, nothing is done by default.

Example

The following example shows how to update a Tradeoff chart. The same code applies to all other types of charts.

```
container = system1.GetContainer(ComponentName="Optimization")
model = container.GetModel()
chart = model.GetChart(Name="TradeoffChart 1")
chart.Update()
```

ConvergenceCriteriaChart

The data entity which describes an ConvergenceCriteria chart. It allows you to visualize how convergence evolves during the optimization process, depending on the optimizer.

Properties

DisplayParameterFullName

If True, the legend of the chart contains the full name of the parameters. Otherwise it contains the short name such as "P1".

Type bool (p. 1360)

Read Only No

DisplayText

The general property that defines the user-visible name of an entity. This property is defined for all data entities but is used only in those entities that present a label in the user interface.

Type string (p. 1438)

Read Only No

IsUpToDate

True if the entity is up-to-date.

Type bool (p. 1360)

Read Only No

Methods

EnableVariable

Enable or disable a variable in a chart. This command is currently limited to the CorrelationScatter chart, where LinearTrendLine_Variable and QuadraticTrendLine_Variable are the two eligible variables, and the ConvergenceCriteria chart.

Required Arguments

IsEnabled False to disable the variable, or True (default) to enable it.

Type bool (p. 1360)

Optional Arguments

Variable DataReference of the variable to enable or disable.

Type DataReference (p. 1371)

Example

The following example shows how to disable and enable a variable in a CorrelationScatter or ConvergenceCriteria chart.

```
container = system1.GetContainer(ComponentName="Correlation")
model = container.GetModel()
chart = model.GetChart(Name="CorrelationScatter 1")
chart.EnableVariable( Variable=chart.LinearTrendLine_Variable, IsEnabled=false)
chart.EnableParameter( Variable=chart.QuadraticTrendLine_Variable )
```

EnableVariables

Enable or disable a list of variables in a chart.

Required Arguments

IsEnabled False to disable the variable, or True (default) to enable it.

Type bool (p. 1360)

Optional Arguments

Variables DataReference of the variable to enable or disable.

Type List (p. 1400)<DataReference (p. 1371)>

Example

The following example shows how to disable two variables, and then how to enable all variables by omitting the Parameters optional variable. This example uses a CorrelationScatter chart. The method also applies for ConvergenceCriteria Chart.

```
container = system1.GetContainer(ComponentName="Correlation")
model = container.GetModel()
var1 = Graphics.GetVariableXY(Name="MyName")
chart = model.GetChart(Name="CorrelationScatter 1")
chart.EnableVariables( Variables = [var1;var2], IsEnabled=false)
chart.EnableVariable()
```

ConvergenceCurvesChart

The data entity which describes a Convergence Curves chart. It allows you to visualize the convergence of an algorithm: with the steps of the convergence as X-axis and the criteria of the convergence as Y-axis. This chart can display several curves based on the same steps of convergence.

Properties

DisplayParameterFullName

If True, the legend of the chart contains the full name of the parameters. Otherwise it contains the short name such as "P1".

Type bool (p. 1360)

Read Only No

DisplayText

The general property that defines the user-visible name of an entity. This property is defined for all data entities but is used only in those entities that present a label in the user interface.

Type string (p. 1438)

Read Only No

IsUpToDate

True if the entity is up-to-date.

Type bool (p. 1360)

Read Only No

Methods

EnableParameters

Enable or disable a list of parameters in a chart.

Required Arguments

IsEnabled False to disable the parameters, or True (default) to enable them.

Type bool (p. 1360)

Optional Arguments

Parameters Parameters to enable or disable, or all parameters if not specified.

Type List (p. 1400)<DataReference (p. 1371)>

Example

The following example shows how to disable two parameters, and then how to enable all parameters by omitting the Parameters optional parameter. This example uses a Correlation Matrix chart. The method also applies for other types of charts like LocalSensitivity, SpiderChart, etc.

```
container = system1.GetContainer(ComponentName="Correlation")
model = container.GetModel()
param1 = model.GetParam(Name="P1")
param4 = model.GetParam(Name="P4")
chart = model.GetChart(Name="Correlation Matrix 1")
chart.EnableParameters( Parameters=[param1;param4], IsEnabled=false )
chart.EnableParameters()
```

ExportData

Export the data of a DesignXplorer entity to a csv file. The entity can be one of the DesignXplorer chart entities, a ParametricTable or an InputParameter entity.

Required Arguments

FileName The exported file name.

Type string (p. 1438)

Optional Arguments

AppendMode True to append to an existing csv file, False to overwrite it.

Type bool (p. 1360)

Default Value False

Example

The following example shows how the user can export the table of Design Points and then the Parameters Parallel chart of a Design of Experiments component.

```
container = system1.GetContainer(ComponentName="Design of Experiment")
model = container.GetModel()
parametricTable = model.GetParametricTable(Name="DesignPoints")
parametricTable.ExportData(FileName="doe.csv")
chart = model.GetChart(Name="DesignPointsParallel")
chart.ExportData(FileName="doe.csv", AppendMode=True)
```

Update

Updates the chart by generating all results or data required to plot it. If the chart is already up-to-date, nothing is done by default.

Example

The following example shows how to update a Tradeoff chart. The same code applies to all other types of charts.

```
container = system1.GetContainer(ComponentName="Optimization")
model = container.GetModel()
chart = model.GetChart(Name="TradeoffChart 1")
chart.Update()
```

CorrelationMatrixChart

The data entity which describes a Correlation Matrix chart. This matrix allows the user to visualize how the input and output parameters are coupled.

Properties

DisplayParameterFullName

If True, the legend of the chart contains the full name of the parameters. Otherwise it contains the short name such as "P1".

Type bool (p. 1360)

Read Only No

DisplayText

The general property that defines the user-visible name of an entity. This property is defined for all data entities but is used only in those entities that present a label in the user interface.

Type string (p. 1438)

Read Only No

IsUpToDate

True if the entity is up-to-date.

Type bool (p. 1360)

Read Only No

Methods

EnableParameters

Enable or disable a list of parameters in a chart.

Required Arguments

IsEnabled False to disable the parameters, or True (default) to enable them.

Type bool (p. 1360)

Optional Arguments

Parameters Parameters to enable or disable, or all parameters if not specified.

Type List (p. 1400)<DataReference (p. 1371)>

Example

The following example shows how to disable two parameters, and then how to enable all parameters by omitting the Parameters optional parameter. This example uses a Correlation Matrix chart. The method also applies for other types of charts like LocalSensitivity, SpiderChart, etc.

```
container = system1.GetContainer(ComponentName="Correlation")
model = container.GetModel()
param1 = model.GetParam(Name="P1")
```

```
param4 = model.GetParam(Name="P4")
chart = model.GetChart(Name="Correlation Matrix 1")
chart.EnableParameters( Parameters=[param1;param4], IsEnabled=false )
chart.EnableParameters()
```

ExportData

Export the data of a DesignXplorer entity to a csv file. The entity can be one of the DesignXplorer chart entities, a ParametricTable or an InputParameter entity.

Required Arguments

FileName The exported file name.

Type string (p. 1438)

Optional Arguments

AppendMode True to append to an existing csv file, False to overwrite it.

Type bool (p. 1360)

Default Value False

Example

The following example shows how the user can export the table of Design Points and then the Parameters Parallel chart of a Design of Experiments component.

```
container = system1.GetContainer(ComponentName="Design of Experiment")
model = container.GetModel()
parametricTable = model.GetParametricTable(Name="DesignPoints")
parametricTable.ExportData(FileName="doe.csv")
chart = model.GetChart(Name="DesignPointsParallel")
chart.ExportData(FileName="doe.csv", AppendMode=True)
```

Update

Updates the chart by generating all results or data required to plot it. If the chart is already up-to-date, nothing is done by default.

Example

The following example shows how to update a Tradeoff chart. The same code applies to all other types of charts.

```
container = system1.GetContainer(ComponentName="Optimization")
model = container.GetModel()
chart = model.GetChart(Name="TradeoffChart 1")
chart.Update()
```

CorrelationScatterChart

The data entity which describes a Correlation Scatter chart. This scatter chart allows the user to visualize the samples used to compute the correlation for the selected parameter pair, as well as the linear and the quadratic trend lines.

Properties

Axes

Dictionary of the parameters associated to axes.

Type Dictionary (p. 1375)<ChartAxes (p. 1363), DataReference (p. 1371)>

Read Only Yes

DisplayParameterFullName

If True, the legend of the chart contains the full name of the parameters. Otherwise it contains the short name such as "P1".

Type bool (p. 1360)

Read Only No

DisplayText

The general property that defines the user-visible name of an entity. This property is defined for all data entities but is used only in those entities that present a label in the user interface.

Type string (p. 1438)

Read Only No

IsUpToDate

True if the entity is up-to-date.

Type bool (p. 1360)

Read Only No

LinearTrendLine_Variable

The DataReference of the LinearTrendLine variable.

Type DataReference (p. 1371)

Read Only No

QuadraticTrendLine_Variable

The DataReference of the QuadraticTrendLine variable.

Type DataReference (p. 1371)

Read Only No

Methods

AssociateParameterToAxis

Associates a Parameter to the specified axis of the chart. The Parameter argument can be omitted which means that the axis is not set.

Required Arguments

Axis Axis to modify.

Type ChartAxes (p. 1363)

Optional Arguments

Parameter Parameter entity to be assigned to the specified axis.

Type DataReference (p. 1371)

Example

The following example shows how to assign parameters to the axes of a DesignPointsCurves chart.

```
container = system1.GetContainer(ComponentName="Design of Experiment")
model = container.GetModel()
inputParam1 = model.GetParam(Name="P1")
outputParam1 = model.GetParam(Name="P5")
outputParam2 = model.GetParam(Name="P6")
chart = model.GetChart(Name="Design Point vs Parameter 1")
chart.AssociateParameterToAxis(Parameter=inputParam1, Axis="XAxis")
chart.AssociateParameterToAxis(Parameter=outputParam1, Axis="YAxis")
chart.AssociateParameterToAxis(Parameter=outputParam2, Axis="YRightAxis")
```

EnableVariable

Enable or disable a variable in a chart. This command is currently limited to the CorrelationScatter chart, where LinearTrendLine_Variable and QuadraticTrendLine_Variable are the two eligible variables, and the ConvergenceCriteria chart.

Required Arguments

IsEnabled False to disable the variable, or True (default) to enable it.

Type bool (p. 1360)

Optional Arguments

Variable DataReference of the variable to enable or disable.

Type DataReference (p. 1371)

Example

The following example shows how to disable and enable a variable in a CorrelationScatter or ConvergenceCriteria chart.

```
container = system1.GetContainer(ComponentName="Correlation")
model = container.GetModel()
chart = model.GetChart(Name="CorrelationScatter 1")
chart.EnableVariable( Variable=chart.LinearTrendLine_Variable, IsEnabled=false)
chart.EnableParameter( Variable=chart.QuadraticTrendLine_Variable )
```

ExportData

Export the data of a DesignXplorer entity to a csv file. The entity can be one of the DesignXplorer chart entities, a ParametricTable or an InputParameter entity.

Required Arguments

FileName The exported file name.

Type string (p. 1438)

Optional Arguments

AppendMode True to append to an existing csv file, False to overwrite it.

Type bool (p. 1360)

Default Value False

Example

The following example shows how the user can export the table of Design Points and then the Parameters Parallel chart of a Design of Experiments component.

```
container = system1.GetContainer(ComponentName="Design of Experiment")
model = container.GetModel()
parametricTable = model.GetParametricTable(Name="DesignPoints")
parametricTable.ExportData(FileName="doe.csv")
chart = model.GetChart(Name="DesignPointsParallel")
chart.ExportData(FileName="doe.csv", AppendMode=True)
```

Update

Updates the chart by generating all results or data required to plot it. If the chart is already up-to-date, nothing is done by default.

Example

The following example shows how to update a Tradeoff chart. The same code applies to all other types of charts.

```
container = system1.GetContainer(ComponentName="Optimization")
```

```
model = container.GetModel()
chart = model.GetChart(Name="TradeoffChart 1")
chart.Update()
```

CustomCandidatePoint

The CustomCandidatePoint data entity is a candidate point created and edited by the user for comparison with the results of an optimization.

Properties

DisplayText

The general property that defines the user-visible name of an entity. This property is defined for all data entities but is used only in those entities that present a label in the user interface.

Type string (p. 1438)

Read Only No

Methods

CanUpdate

Returns true if the entity can be updated to produce the specific source of output values. As an example, the query will return false for Source=ResponseSurface if the candidate point was generated by a Direct Optimization system.

Return Results of the query telling if the Point entity can be updated to produce the specified source of output values.

Type bool (p. 1360)

Optional Arguments

Source Source of the output values to query for.

Type OutputSource (p. 1412)

Default Value Simulation

Example

The following example shows how to check if a CandidatePoint entity can be updated and how to update it to obtain the Simulation output values.

```
system1 = GetSystem(Name="RSO")
container = system1.GetContainer(ComponentName="Optimization")
model = container.GetModel()
candidatePoint = model.GetCandidatePoint(Name="CandidatePoint 1")
if candidatePoint.CanUpdate(Source="Simulation"):
   candidatePoint.Update(Source="Simulation")
```

GetInputValues

Get the values of the input parameters. Depending on the parameter definition, each value can be a Quantity, a real, a string or an integer.

Return The dictionary of the input parameter values.

Type ReadOnlyDictionary (p. 1423)<DataReference (p. 1371), Object (p. 1409)>

Optional Arguments

ValueType The type of parameter value to return.

Type ParameterValueType (p. 1413)

Default Value ActualValue

Example

The following example shows how to retrieve a candidate point and then extract its input parameter values.

```
system1 = GetSystem(Name="DOP")
container = system1.GetContainer(ComponentName="Optimization")
model = container.GetModel()
candidatePoint = model.GetCandidatePoint(Name="CandidatePoint 1")
dictInputValues = candidatePoint.GetInputValues()
```

GetOutputValues

Get the values of the output parameters for the specified source of output values.

Return The dictionary of the output parameter values.

Type ReadOnlyDictionary (p. 1423)<DataReference (p. 1371), Object (p. 1409)>

Optional Arguments

Source The source of the output values to return.

Type OutputSource (p. 1412)

Default Value Simulation

ValueType The type of parameter value to return.

Type ParameterValueType (p. 1413)

Default Value ActualValue

Example

The following example shows how to retrieve a candidate point and then extract its output parameter values.

```
system1 = GetSystem(Name="DOP")
container = system1.GetContainer(ComponentName="Optimization")
model = container.GetModel()
candidatePoint = model.GetCandidatePoint(Name="CandidatePoint 1")
responseSurfaceOutputValues = candidatePoint.GetOutputValues(Source="ResponseSurface")
simulationOutputValues = candidatePoint.GetOutputValues(Source="Simulation")
```

GetState

Returns the state of the entity for the specified source of output values. A different state is available for each source of output values.

Return State for the specified nature of output values.

Type UpdatableEntityState (p. 1446)

Optional Arguments

Source Source of output values to query for.

Type OutputSource (p. 1412)

Default Value Simulation

Example

The following example shows how to check the state of the Simulation's source of the output values of a CandidatePoint entity.

```
system1 = GetSystem(Name="RSO")
container = system1.GetContainer(ComponentName="Optimization")
model = container.GetModel()
candidatePoint = model.GetCandidatePoint(Name="CandidatePoint 1")
state = candidatePoint.GetState(Source="Simulation")
```

GetValue

Get the value for a given parameter and, if the parameter is an output parameter, a given source of output values. Depending on the parameter definition, each value can be a Quantity, a real, a string or an integer.

Return The retrieve parameter value.

Type Object (p. 1409)

Required Arguments

Parameter The Parameter for which the value is requested.

Type DataReference (p. 1371)

Optional Arguments

Source If Parameter is an output parameter, the source of the output value to return.

Type OutputSource (p. 1412)

Default Value Simulation

ValueType The type of parameter value to return.

Type ParameterValueType (p. 1413)

Default Value ActualValue

Example

The following example shows how to retrieve a candidate point generated by the optimization model, and then extract selected parameter values.

```
system1 = GetSystem(Name="DOP")
container = system1.GetContainer(ComponentName="Optimization")
model = container.GetModel()
candidatePoint = model.GetCandidatePoint(Name="CandidatePoint 1")
inputParam1 = model.GetParameter(Name="P1")
parameterValue1 = candidatePoint.GetValue(Parameter=inputParam1)
outputParam4 = model.GetParameter(Name="P4")
parameterValue2 = candidatePoint.GetValue(Parameter=outputParam4, Source="Simulation")
print parameterValue2.Value
```

GetValues

Get the values of all input parameters and the values of all output parameters for the specified source of output values. Depending on the parameter definition, each value can be a Quantity, a real, a string or an integer.

Return The dictionary of the parameter values.

Type ReadOnlyDictionary (p. 1423)<DataReference (p. 1371), Object (p. 1409)>

Optional Arguments

Source For output parameters, the source of the output values to return.

Type OutputSource (p. 1412)

Default Value Simulation

ValueType The type of parameter value to return.

Type ParameterValueType (p. 1413)

Default Value ActualValue

Example

The following example shows how to retrieve a candidate point and then extract all of its parameter values.

```
system1 = GetSystem(Name="DOP")
container = system1.GetContainer(ComponentName="Optimization")
model = container.GetModel()
candidatePoint = model.GetCandidatePoint(Name="CandidatePoint 1")
parameterValues = candidatePoint.GetValues(Source="Simulation")
```

SetParameter

Sets the expression of a parameter in a CustomCandidatePoint entity. In the context of a response surface based optimization, the output values are updated from the response surface.

Required Arguments

Expression Assigned Expression (value or quantity).

Type string (p. 1438)

Parameter DataReference of the parameter.

Type DataReference (p. 1371)

Example

The following example shows how to set the expression for one parameter in an existing custom candidate point.

```
container = system1.GetContainer(ComponentName="Optimization")
model = container.GetModel()
customCandidatePoint = model.GetCustomCandidatePoint(Name="CustomCandidatePoint 1")
inputParameter1 = model.GetParameter(Name="P1")
customCandidatePoint.SetParameter(Parameter=inputParameter1, Expression="2.01")
inputParameter2 = model.GetParameter(Name="P2")
customCandidatePoint.SetParameter(Parameter=inputParameter2, Expression="2.1 [m]")
```

SetParameters

Sets the expression of several parameters in a CustomCandidatePoint entity. In the context of a response surface based optimization, the output values are updated from the response surface.

Required Arguments

Expressions Dictionary of the parameters and their assigned expressions.

Type IDictionary (p. 1375)<DataReference (p. 1371), string (p. 1438)>

Example

The following example shows how to set the expression for several parameters in an existing custom candidate point.

```
container = system1.GetContainer(ComponentName="Optimization")
model = container.GetModel()
customCandidatePoint = model.GetCustomCandidatePoint(Name="CustomCandidatePoint 1")
inputParameter1 = model.GetParameter(Name="P1")
customCandidatePoint.SetParameter(Parameter=inputParameter1, Expression="2.01")
inputParameter2 = model.GetParameter(Name="P2")
customCandidatePoint.SetParameters(Expressions={inputParameter1: "2.01", inputParameter2: "15.5 [m]"})
```

Update

Updates a point entity to provide the output parameter values of the requested source. If the requested source is ResponseSurface, the output values are generated by evaluating the response surface model, if available. If the requested source is Simulation, the output values are generated by triggering a DesignPoint update. This command applies to CandidatePoint and CustomCandidatePoint entities.

Return The dictionary of parameters with their values.

Type ReadOnlyDictionary (p. 1423)<DataReference (p. 1371), Object (p. 1409)>

Optional Arguments

Source Source of output values to produce. The update method depends on this parameter.

Type OutputSource (p. 1412)

Default Value Simulation

Example

The following example shows how to update an existing candidate point to obtain Simulation output values.

```
system1 = GetSystem(Name="RSO")
container = system1.GetContainer(ComponentName="Optimization")
candidate1 = container.GetCandidatePoint(Name="CandidatePoint 1")
outputValues = candidate1.Update(Source="Simulation")
```

DesignPointsCurvesChart

The data entity which describes a "Design Points vs. Parameters" chart. It provides a 2D chart with two Y axes and two X axes to display Design Points versus Parameter and/or Parameter versus Parameter curves.

Properties

Axes

Dictionary of the parameters associated to axes

Type Dictionary (p. 1375)<ChartAxes (p. 1363), DataReference (p. 1371)>

Read Only Yes

DisplayParameterFullName

If True, the legend of the chart contains the full name of the parameters. Otherwise it contains the short name such as "P1".

Type bool (p. 1360)

Read Only No

DisplayText

The general property that defines the user-visible name of an entity. This property is defined for all data entities but is used only in those entities that present a label in the user interface.

Type string (p. 1438)

Read Only No

IsUpToDate

True if the entity is up-to-date.

Type bool (p. 1360)

Read Only No

Methods

AssociateParameterToAxis

Associates a Parameter to the specified axis of the chart. The Parameter argument can be omitted which means that the axis is not set.

Required Arguments

Axis Axis to modify.

Type ChartAxes (p. 1363)

Optional Arguments

Parameter Parameter entity to be assigned to the specified axis.

Type DataReference (p. 1371)

Example

The following example shows how to assign parameters to the axes of a DesignPointsCurves chart.

```
container = system1.GetContainer(ComponentName="Design of Experiment")
model = container.GetModel()
inputParam1 = model.GetParam(Name="P1")
outputParam1 = model.GetParam(Name="P5")
outputParam2 = model.GetParam(Name="P6")
chart = model.GetChart(Name="Design Point vs Parameter 1")
chart.AssociateParameterToAxis(Parameter=inputParam1, Axis="XAxis")
chart.AssociateParameterToAxis(Parameter=outputParam1, Axis="YAxis")
chart.AssociateParameterToAxis(Parameter=outputParam2, Axis="YRightAxis")
```

ExportData

Export the data of a DesignXplorer entity to a csv file. The entity can be one of the DesignXplorer chart entities, a ParametricTable or an InputParameter entity.

Required Arguments

FileName The exported file name.

Type string (p. 1438)

Optional Arguments

AppendMode True to append to an existing csv file, False to overwrite it.

Type bool (p. 1360)

Default Value False

Example

The following example shows how the user can export the table of Design Points and then the Parameters Parallel chart of a Design of Experiments component.

```
container = system1.GetContainer(ComponentName="Design of Experiment")
model = container.GetModel()
parametricTable = model.GetParametricTable(Name="DesignPoints")
parametricTable.ExportData(FileName="doe.csv")
chart = model.GetChart(Name="DesignPointsParallel")
chart.ExportData(FileName="doe.csv", AppendMode=True)
```

Update

Updates the chart by generating all results or data required to plot it. If the chart is already up-to-date, nothing is done by default.

Example

The following example shows how to update a Tradeoff chart. The same code applies to all other types of charts.

```
container = system1.GetContainer(ComponentName="Optimization")
model = container.GetModel()
chart = model.GetChart(Name="TradeoffChart 1")
chart.Update()
```

DesignPointsParallelChart

The data entity which describes a Parameters Parallel chart. It allows you to visualize the DOE matrix using parallel Y axes to represent all of the input and output parameters on the same 2D representation, whatever the number of parameters.

Properties

DisplayParameterFullName

If True, the legend of the chart contains the full name of the parameters. Otherwise it contains the short name such as "P1".

Type bool (p. 1360)

Read Only No

DisplayText

The general property that defines the user-visible name of an entity. This property is defined for all data entities but is used only in those entities that present a label in the user interface.

Type string (p. 1438)

Read Only No

IsUpToDate

True if the entity is up-to-date.

Type bool (p. 1360)

Read Only No

Methods

EnableParameters

Enable or disable a list of parameters in a chart.

Required Arguments

IsEnabled False to disable the parameters, or True (default) to enable them.

Type bool (p. 1360)

Optional Arguments

Parameters Parameters to enable or disable, or all parameters if not specified.

Type List (p. 1400)<DataReference (p. 1371)>

Example

The following example shows how to disable two parameters, and then how to enable all parameters by omitting the Parameters optional parameter. This example uses a Correlation Matrix chart. The method also applies for other types of charts like LocalSensitivity, SpiderChart, etc.

```
container = system1.GetContainer(ComponentName="Correlation")
model = container.GetModel()
param1 = model.GetParam(Name="P1")
param4 = model.GetParam(Name="P4")
chart = model.GetChart(Name="Correlation Matrix 1")
chart.EnableParameters( Parameters=[param1;param4], IsEnabled=false )
chart.EnableParameters()
```

ExportData

Export the data of a DesignXplorer entity to a csv file. The entity can be one of the DesignXplorer chart entities, a ParametricTable or an InputParameter entity.

Required Arguments

FileName The exported file name.

Type string (p. 1438)

Optional Arguments

AppendMode True to append to an existing csv file, False to overwrite it.

Type bool (p. 1360)

Default Value False

Example

The following example shows how the user can export the table of Design Points and then the Parameters Parallel chart of a Design of Experiments component.

```
container = system1.GetContainer(ComponentName="Design of Experiment")
model = container.GetModel()
parametricTable = model.GetParametricTable(Name="DesignPoints")
parametricTable.ExportData(FileName="doe.csv")
chart = model.GetChart(Name="DesignPointsParallel")
chart.ExportData(FileName="doe.csv", AppendMode=True)
```

Update

Updates the chart by generating all results or data required to plot it. If the chart is already up-to-date, nothing is done by default.

Example

The following example shows how to update a Tradeoff chart. The same code applies to all other types of charts.

```
container = system1.GetContainer(ComponentName="Optimization")
model = container.GetModel()
chart = model.GetChart(Name="TradeoffChart 1")
chart.Update()
```

DeterminationHistogramChart

The data entity which describes the Determination Histogram chart. It allows you to visualize the coefficient of determination (linear or quadratic) of each input for an output parameter.

Properties

Axes

Dictionary of the parameters associated to axes.

Type Dictionary (p. 1375)<ChartAxes (p. 1363), DataReference (p. 1371)>

Read Only Yes

DeterminationType

Determination type, either Linear or Quadratic.

Type DeterminationCoefficientChartModes (p. 1375)

Read Only No

DisplayParameterFullName

If True, the legend of the chart contains the full name of the parameters. Otherwise it contains the short name such as "P1".

Type bool (p. 1360)

Read Only No

DisplayText

The general property that defines the user-visible name of an entity. This property is defined for all data entities but is used only in those entities that present a label in the user interface.

Type string (p. 1438)

Read Only No

FullModelR2

Full Model R2 value.

Type double (p. 1378)

Read Only Yes

IsUpToDate

True if the entity is up-to-date.

Type bool (p. 1360)

Read Only No

ThresholdR2

Threshold value of R2: the chart displays only input parameters with a coefficient of determination is greater than thus threshold.

Type double (p. 1378)

Read Only No

Methods

AssociateParameterToAxis

Associates a Parameter to the specified axis of the chart. The Parameter argument can be omitted which means that the axis is not set.

Required Arguments

Axis Axis to modify.

Type ChartAxes (p. 1363)

Optional Arguments

Parameter Parameter entity to be assigned to the specified axis.

Type DataReference (p. 1371)

Example

The following example shows how to assign parameters to the axes of a DesignPointsCurves chart.

```
container = system1.GetContainer(ComponentName="Design of Experiment")
model = container.GetModel()
inputParam1 = model.GetParam(Name="P1")
outputParam1 = model.GetParam(Name="P5")
outputParam2 = model.GetParam(Name="P6")
chart = model.GetChart(Name="Design Point vs Parameter 1")
```

```
chart.AssociateParameterToAxis(Parameter=inputParam1, Axis="XAxis")
chart.AssociateParameterToAxis(Parameter=outputParam1, Axis="YAxis")
chart.AssociateParameterToAxis(Parameter=outputParam2, Axis="YRightAxis")
```

ExportData

Export the data of a DesignXplorer entity to a csv file. The entity can be one of the DesignXplorer chart entities, a ParametricTable or an InputParameter entity.

Required Arguments

FileName The exported file name.

Type string (p. 1438)

Optional Arguments

AppendMode True to append to an existing csv file, False to overwrite it.

Type bool (p. 1360)

Default Value False

Example

The following example shows how the user can export the table of Design Points and then the Parameters Parallel chart of a Design of Experiments component.

```
container = system1.GetContainer(ComponentName="Design of Experiment")
model = container.GetModel()
parametricTable = model.GetParametricTable(Name="DesignPoints")
parametricTable.ExportData(FileName="doe.csv")
chart = model.GetChart(Name="DesignPointsParallel")
chart.ExportData(FileName="doe.csv", AppendMode=True)
```

Update

Updates the chart by generating all results or data required to plot it. If the chart is already up-to-date, nothing is done by default.

Example

The following example shows how to update a Tradeoff chart. The same code applies to all other types of charts.

```
container = system1.GetContainer(ComponentName="Optimization")
model = container.GetModel()
chart = model.GetChart(Name="TradeoffChart 1")
chart.Update()
```

DeterminationMatrixChart

The data entity which describes a Determination Matrix chart. This matrix allows you to visualize how the input and output parameters are coupled in a quadratic regression.

Properties

DisplayParameterFullName

If True, the legend of the chart contains the full name of the parameters. Otherwise it contains the short name such as "P1".

Type bool (p. 1360)

Read Only No

DisplayText

The general property that defines the user-visible name of an entity. This property is defined for all data entities but is used only in those entities that present a label in the user interface.

Type string (p. 1438)

Read Only No

IsUpToDate

True if the entity is up-to-date.

Type bool (p. 1360)

Read Only No

Methods

EnableParameters

Enable or disable a list of parameters in a chart.

Required Arguments

IsEnabled False to disable the parameters, or True (default) to enable them.

Type bool (p. 1360)

Optional Arguments

Parameters Parameters to enable or disable, or all parameters if not specified.

Type List (p. 1400)<DataReference (p. 1371)>

Example

The following example shows how to disable two parameters, and then how to enable all parameters by omitting the Parameters optional parameter. This example uses a Correlation Matrix chart. The method also applies for other types of charts like LocalSensitivity, SpiderChart, etc.

```
container = system1.GetContainer(ComponentName="Correlation")
model = container.GetModel()
param1 = model.GetParam(Name="P1")
param4 = model.GetParam(Name="P4")
chart = model.GetChart(Name="Correlation Matrix 1")
chart.EnableParameters( Parameters=[param1;param4], IsEnabled=false )
chart.EnableParameters()
```

ExportData

Export the data of a DesignXplorer entity to a csv file. The entity can be one of the DesignXplorer chart entities, a ParametricTable or an InputParameter entity.

Required Arguments

FileName The exported file name.

Type string (p. 1438)

Optional Arguments

AppendMode True to append to an existing csv file, False to overwrite it.

Type bool (p. 1360)

Default Value False

Example

The following example shows how the user can export the table of Design Points and then the Parameters Parallel chart of a Design of Experiments component.

```
container = system1.GetContainer(ComponentName="Design of Experiment")
model = container.GetModel()
parametricTable = model.GetParametricTable(Name="DesignPoints")
parametricTable.ExportData(FileName="doe.csv")
chart = model.GetChart(Name="DesignPointsParallel")
chart.ExportData(FileName="doe.csv", AppendMode=True)
```

Update

Updates the chart by generating all results or data required to plot it. If the chart is already up-to-date, nothing is done by default.

Example

The following example shows how to update a Tradeoff chart. The same code applies to all other types of charts.

```
container = system1.GetContainer(ComponentName="Optimization")
model = container.GetModel()
chart = model.GetChart(Name="TradeoffChart 1")
chart.Update()
```

DistributionChart

The entity which describes a parameter's Distribution chart. This chart is always associated with an uncertainty parameter and allows you to visualize the statistical distribution defined or calculated for this parameter.

Properties

DisplayParameterFullName

If True, the legend of the chart contains the full name of the parameters. Otherwise it contains the short name such as "P1".

Type bool (p. 1360)

Read Only No

DisplayText

The general property that defines the user-visible name of an entity. This property is defined for all data entities but is used only in those entities that present a label in the user interface.

Type string (p. 1438)

Read Only No

IsUpToDate

True if the entity is up-to-date.

Type bool (p. 1360)

Read Only No

Parameter

The parameter entity associated with the chart.

Type DataReference (p. 1371)

Read Only No

Methods

ExportData

Export the data of a DesignXplorer entity to a csv file. The entity can be one of the DesignXplorer chart entities, a ParametricTable or an InputParameter entity.

Required Arguments

FileName The exported file name.

Type string (p. 1438)

Optional Arguments

AppendMode True to append to an existing csv file, False to overwrite it.

Type bool (p. 1360)

Default Value False

Example

The following example shows how the user can export the table of Design Points and then the Parameters Parallel chart of a Design of Experiments component.

```
container = system1.GetContainer(ComponentName="Design of Experiment")
model = container.GetModel()
parametricTable = model.GetParametricTable(Name="DesignPoints")
parametricTable.ExportData(FileName="doe.csv")
chart = model.GetChart(Name="DesignPointsParallel")
chart.ExportData(FileName="doe.csv", AppendMode=True)
```

Update

Updates the chart by generating all results or data required to plot it. If the chart is already up-to-date, nothing is done by default.

Example

The following example shows how to update a Tradeoff chart. The same code applies to all other types of charts.

```
container = system1.GetContainer(ComponentName="Optimization")
model = container.GetModel()
chart = model.GetChart(Name="TradeoffChart 1")
chart.Update()
```

GoodnessOfFit

Entity which manages the Goodness Of Fit Information of a Response Surface for an Output Parameter

Properties

ConfidenceLevel

The measure of the likelihood that a confidence interval contains the quantity or parameter being estimated

Type double (p. 1378)

Read Only No

DiscreteExpressions

A Dictionary holding the values of all discrete input. Note: Discrete parameter values are level values, not indices.

Type Dictionary (p. 1375)<DataReference (p. 1371), Object (p. 1409)>

Read Only No

DisplayText

The general property that defines the user-visible name of an entity. This property is defined for all data entities but is used only in those entities that present a label in the user interface.

Type string (p. 1438)

Read Only No

IsUpToDate

True if the entity is up-to-date.

Type bool (p. 1360)

Read Only No

Methods

CreateAdvancedReport

For the standard response surface only (Full second order Polynomials), creates an Advanced Goodness of Fit report for any direct output parameter.

Return A string which contains the generated Advanced Goodness of Fit report.

Type string (p. 1438)

Required Arguments

Parameter Parent Parameter of the report.

Type DataReference (p. 1371)

Optional Arguments

PlainTextFormat Plain text formatting instead of HTML (default).

Type bool (p. 1360)

Default Value False

Example

The following example shows how to create an Advanced Goodness of Fit report.

```
container = system1.GetContainer(ComponentName="Response Surface")
model = container.GetModel()
gof1 = model.GetGoodnessOfFit(Name="GoodnessOfFit")
outputParameter1 = model.GetParameter(Name="P4")
report1 = gof1.CreateAdvancedReport(Parameter=outputParameter1)
```

ExportData

Export the data of a DesignXplorer entity to a csv file. The entity can be one of the DesignXplorer chart entities, a ParametricTable or an InputParameter entity.

Required Arguments

FileName The exported file name.

Type string (p. 1438)

Optional Arguments

AppendMode True to append to an existing csv file, False to overwrite it.

Type bool (p. 1360)

Default Value False

Example

The following example shows how the user can export the table of Design Points and then the Parameters Parallel chart of a Design of Experiments component.

```
container = system1.GetContainer(ComponentName="Design of Experiment")
model = container.GetModel()
parametricTable = model.GetParametricTable(Name="DesignPoints")
parametricTable.ExportData(FileName="doe.csv")
chart = model.GetChart(Name="DesignPointsParallel")
chart.ExportData(FileName="doe.csv", AppendMode=True)
```

SetDiscreteParameter

Sets the Expression of a discrete input parameter in a GoodnessOfFit and updates the associated output parameter values. The chart entities depending on the GoodnessOfFit are updated as well.

Required Arguments

DiscreteParameter DataReference of the Discrete Input parameter.

Type DataReference (p. 1371)

Expression Assigned Expression (discrete value).

Type string (p. 1438)

Example

The following example shows how to set the expression for one discrete parameter in an existing GoodnessOfFit.

```
container = system1.GetContainer(ComponentName="Response Surface")
model = container.GetModel()
gof = model.GetGoodnessOfFit(Name="Goodness Of Fit")
inputParameter1 = model.GetParameter(Name="P1")
gof.SetDiscreteParameter(DiscreteParameter=inputParameter1, Expression="15")
```

Update

Updates the goodness of fit and the results which depend on it. If the goodness of fit is already uptodate, nothing is done unless the Force flag is set to True.

Optional Arguments

Force Set to true if the update operation of the goodness of fit point is required even if it's already uptodate.

Type bool (p. 1360)

Default Value False

Example

The following example shows how to update an existing GoodnessOfFit.

```
container = system1.GetContainer(ComponentName="Response Surface")
model = container.GetModel()
gof1 = model.GetGoodnessOfFit(Name="GOF 1")
gof1.Update()
```

HistoryChart

The data entity which describes an History chart. It allows you to visualize how a parameter evolves during the optimization process, point by point or iteration by iteration depending on the optimizer. Defined target and constraints values are also represented.

Properties

DisplayParameterFullName

If True, the legend of the chart contains the full name of the parameters. Otherwise it contains the short name such as "P1".

Type bool (p. 1360)

Read Only No

DisplayText

The general property that defines the user-visible name of an entity. This property is defined for all data entities but is used only in those entities that present a label in the user interface.

Type string (p. 1438)

Read Only No

IsUpToDate

True if the entity is up-to-date.

Type bool (p. 1360)

Read Only No

Methods

ExportData

Export the data of a DesignXplorer entity to a csv file. The entity can be one of the DesignXplorer chart entities, a ParametricTable or an InputParameter entity.

Required Arguments

FileName The exported file name.

Type string (p. 1438)

Optional Arguments

AppendMode True to append to an existing csv file, False to overwrite it.

Type bool (p. 1360)

Default Value False

Example

The following example shows how the user can export the table of Design Points and then the Parameters Parallel chart of a Design of Experiments component.

```
container = system1.GetContainer(ComponentName="Design of Experiment")
model = container.GetModel()
parametricTable = model.GetParametricTable(Name="DesignPoints")
parametricTable.ExportData(FileName="doe.csv")
chart = model.GetChart(Name="DesignPointsParallel")
chart.ExportData(FileName="doe.csv", AppendMode=True)
```

Update

Updates the chart by generating all results or data required to plot it. If the chart is already up-to-date, nothing is done by default.

Example

The following example shows how to update a Tradeoff chart. The same code applies to all other types of charts.

```
container = system1.GetContainer(ComponentName="Optimization")
model = container.GetModel()
chart = model.GetChart(Name="TradeoffChart 1")
chart.Update()
```

LocalSensitivityChart

The data entity which describes a Local Sensitivity chart. It allows you to visualize the impact that changing each input parameter independently has on the output parameters. A Local Sensitivity chart accepts two different graphical modes -BarChart and PieChart- and supports enabling/disabling input and/or output parameters. A Local Sensitivity depends on a Response Point which serves as a reference point for sensitivity computation.

Properties

AxesRangeMode

Axes range for output parameters controls if the range is determined from the chart's data or the min-max of outputs.

Type AxesRangeModes (p. 1356)

Read Only No

DisplayParameterFullName

If True, the legend of the chart contains the full name of the parameters. Otherwise it contains the short name such as "P1".

Type bool (p. 1360)

Read Only No

DisplayText

The general property that defines the user-visible name of an entity. This property is defined for all data entities but is used only in those entities that present a label in the user interface.

Type string (p. 1438)

Read Only No

IsUpToDate

True if the entity is up-to-date.

Type bool (p. 1360)

Read Only No

Mode

Specifies the graphical representation used to render local sensitivity data. It can be either BarChart or PieChart.

Type SensitivityChartModes (p. 1430)

Read Only No

Methods

EnableParameters

Enable or disable a list of parameters in a chart.

Required Arguments

IsEnabled False to disable the parameters, or True (default) to enable them.

Type bool (p. 1360)

Optional Arguments

Parameters Parameters to enable or disable, or all parameters if not specified.

Type List (p. 1400)<DataReference (p. 1371)>

Example

The following example shows how to disable two parameters, and then how to enable all parameters by omitting the Parameters optional parameter. This example uses a Correlation Matrix chart. The method also applies for other types of charts like LocalSensitivity, SpiderChart, etc.

```
container = system1.GetContainer(ComponentName="Correlation")
model = container.GetModel()
param1 = model.GetParam(Name="P1")
param4 = model.GetParam(Name="P4")
chart = model.GetChart(Name="Correlation Matrix 1")
chart.EnableParameters( Parameters=[param1;param4], IsEnabled=false )
chart.EnableParameters()
```

ExportData

Export the data of a DesignXplorer entity to a csv file. The entity can be one of the DesignXplorer chart entities, a ParametricTable or an InputParameter entity.

Required Arguments

FileName The exported file name.

Type string (p. 1438)

Optional Arguments

AppendMode True to append to an existing csv file, False to overwrite it.

Type bool (p. 1360)

Default Value False

Example

The following example shows how the user can export the table of Design Points and then the Parameters Parallel chart of a Design of Experiments component.

```
container = system1.GetContainer(ComponentName="Design of Experiment")
model = container.GetModel()
parametricTable = model.GetParametricTable(Name="DesignPoints")
parametricTable.ExportData(FileName="doe.csv")
chart = model.GetChart(Name="DesignPointsParallel")
chart.ExportData(FileName="doe.csv", AppendMode=True)
```

Update

Updates the chart by generating all results or data required to plot it. If the chart is already up-to-date, nothing is done by default.

Example

The following example shows how to update a Tradeoff chart. The same code applies to all other types of charts.

```
container = system1.GetContainer(ComponentName="Optimization")
model = container.GetModel()
chart = model.GetChart(Name="TradeoffChart 1")
chart.Update()
```

LocalSensitivityCurvesChart

The data entity which describes a Local Sensitivity Curves chart. It allows you to visualize the impact that changing each input parameter independently has on the output parameters. It allows you to visualize one or two output parameters at the same time. A Local Sensitivity Curves chart supports enabling and disabling input parameters. A Local Sensitivity Curves depends on a Response Point which serves as a reference point for computation of the sensitivity.

Properties

Axes

Dictionary of the parameters associated to axes.

Type Dictionary (p. 1375)<ChartAxes (p. 1363), DataReference (p. 1371)>

Read Only Yes

AxesRangeMode

Axes range for output parameters controls if the range is determined from the chart's data or the min-max of outputs.

Type AxesRangeModes (p. 1356)

Read Only No

ChartResolution

Chart resolution.

Type int (p. 1394)

Read Only No

DisplayParameterFullName

If True, the legend of the chart contains the full name of the parameters. Otherwise it contains the short name such as "P1".

Type bool (p. 1360)

Read Only No

DisplayText

The general property that defines the user-visible name of an entity. This property is defined for all data entities but is used only in those entities that present a label in the user interface.

Type string (p. 1438)

Read Only No

IsUpToDate

True if the entity is up-to-date.

Type bool (p. 1360)

Read Only No

Methods

AssociateParameterToAxis

Associates a Parameter to the specified axis of the chart. The Parameter argument can be omitted which means that the axis is not set.

Required Arguments

Axis Axis to modify.

Type ChartAxes (p. 1363)

Optional Arguments

Parameter Parameter entity to be assigned to the specified axis.

Type DataReference (p. 1371)

Example

The following example shows how to assign parameters to the axes of a DesignPointsCurves chart.

```
container = system1.GetContainer(ComponentName="Design of Experiment")
model = container.GetModel()
inputParam1 = model.GetParam(Name="P1")
outputParam1 = model.GetParam(Name="P5")
outputParam2 = model.GetParam(Name="P6")
chart = model.GetChart(Name="Design Point vs Parameter 1")
chart.AssociateParameterToAxis(Parameter=inputParam1, Axis="XAxis")
chart.AssociateParameterToAxis(Parameter=outputParam1, Axis="YAxis")
chart.AssociateParameterToAxis(Parameter=outputParam2, Axis="YRightAxis")
```

EnableParameters

Enable or disable a list of parameters in a chart.

Required Arguments

IsEnabled False to disable the parameters, or True (default) to enable them.

Type bool (p. 1360)

Optional Arguments

Parameters Parameters to enable or disable, or all parameters if not specified.

Type List (p. 1400)<DataReference (p. 1371)>

Example

The following example shows how to disable two parameters, and then how to enable all parameters by omitting the Parameters optional parameter. This example uses a Correlation Matrix chart. The method also applies for other types of charts like LocalSensitivity, SpiderChart, etc.

```
container = system1.GetContainer(ComponentName="Correlation")
model = container.GetModel()
param1 = model.GetParam(Name="P1")
param4 = model.GetParam(Name="P4")
chart = model.GetChart(Name="Correlation Matrix 1")
chart.EnableParameters( Parameters=[param1;param4], IsEnabled=false )
chart.EnableParameters()
```

ExportData

Export the data of a DesignXplorer entity to a csv file. The entity can be one of the DesignXplorer chart entities, a ParametricTable or an InputParameter entity.

Required Arguments

FileName The exported file name.

Type string (p. 1438)

Optional Arguments

AppendMode True to append to an existing csv file, False to overwrite it.

Type bool (p. 1360)

Default Value False

Example

The following example shows how the user can export the table of Design Points and then the Parameters Parallel chart of a Design of Experiments component.

```
container = system1.GetContainer(ComponentName="Design of Experiment")
model = container.GetModel()
parametricTable = model.GetParametricTable(Name="DesignPoints")
parametricTable.ExportData(FileName="doe.csv")
chart = model.GetChart(Name="DesignPointsParallel")
chart.ExportData(FileName="doe.csv", AppendMode=True)
```

Update

Updates the chart by generating all results or data required to plot it. If the chart is already up-to-date, nothing is done by default.

Example

The following example shows how to update a Tradeoff chart. The same code applies to all other types of charts.

```
container = system1.GetContainer(ComponentName="Optimization")
model = container.GetModel()
chart = model.GetChart(Name="TradeoffChart 1")
chart.Update()
```

ParameterStatistics

Entity which manages the calculated Statistics of a parameter for a Six-Sigma Component

Properties

DisplayText

The general property that defines the user-visible name of an entity. This property is defined for all data entities but is used only in those entities that present a label in the user interface.

Type string (p. 1438)

Read Only No

Entropy

Shannon Entropy value (Complexity)

Type double (p. 1378)

Read Only No

InvProbabilityTable

DataReference of the Inverse Probability ParametricTable for the associated parameter

Type DataReference (p. 1371)

Read Only No

IsUpToDate

True if the entity is up-to-date.

Type bool (p. 1360)

Read Only No

Kurtosis

Kurtosis value

Type double (p. 1378)

Read Only No

Mean

Mean value

Type double (p. 1378)

Read Only No

Parameter

DataReference of the input or output parameter associated with this statistics entity.

Type DataReference (p. 1371)

Read Only No

ProbabilityMaximum

Probability Maximum value

Type double (p. 1378)

Read Only No

ProbabilityMinimum

Probability Minimum value

Type double (p. 1378)

Read Only No

ProbabilityTable

DataReference of the Probability ParametricTable for the associated parameter

Type DataReference (p. 1371)

Read Only No

QuantileMaximum

Quantile Maximum value

Type double (p. 1378)

Read Only No

QuantileMinimum

Quantile Minimum value

Type double (p. 1378)

Read Only No

SigmaMaximum

Sigma Maximum value

Type double (p. 1378)

Read Only No

SigmaMinimum

Sigma Minimum value

Type double (p. 1378)

Read Only No

SignalNoiseLarge

Signal-Noise Ratio value (Larger is Better)

Type double (p. 1378)

Read Only No

SignalNoiseNominal

Signal-Noise Ratio value (Nominal is Best)

Type double (p. 1378)

Read Only No

SignalNoiseSmall

Signal-Noise Ratio value (Smaller is Better)

Type double (p. 1378)

Read Only No

Skewness

Skewness value

Type double (p. 1378)

Read Only No

StandardDeviation

Standard Deviation value

Type double (p. 1378)

Read Only No

StatisticsChart

DataReference of the statistics chart for the associated parameter

Type DataReference (p. 1371)

Read Only No

TableType

Type of the probability table

Type SixSigmaTableTypes (p. 1432)

Read Only No

ParametricTable

ParametricTable entity used to encapsulate most of the evaluation results in a convenient 2D matrix format.

Properties

DimCol

Number of columns.

Type int (p. 1394)

Read Only Yes

DimRow

Number of rows.

Type int (p. 1394)

Read Only Yes

DisplayText

The general property that defines the user-visible name of an entity. This property is defined for all data entities but is used only in those entities that present a label in the user interface.

Type string (p. 1438)

Read Only No

Methods

AddRow

Adds a row to the bottom of a ParametricTable entity.

Optional Arguments

RowValues New values for the row.

Type List (p. 1400)<string (p. 1438)>

Example

The following example shows how to make a DOE editable, to retrieve the table of design points and add a new row to it.

```
container = system1.GetContainer(ComponentName="Design of Experiment")
model = container.GetModel()
model.DOEType = "eDOETYPE_USER"
DOEMatrix = model.GetParametricTable(Name="DesignPoints")
DOEMatrix.AddRow()
```

DeleteRows

Delete rows from a ParametricTable entity.

Required Arguments

Indices Indices of the rows to delete.

Type List (p. 1400)<int (p. 1394)>

Example

The following example shows how to delete rows from the DOEmatrix in a custom DOE context.

```
container = system1.GetContainer(ComponentName="Design of Experiment")
model = container.GetModel()
DOEMatrix = model.GetParametricTable(Name="DesignPoints")
DOEMatrix.DeleteRows(Indices=[0,7,8,9])
```

ExportData

Export the data of a DesignXplorer entity to a csv file. The entity can be one of the DesignXplorer chart entities, a ParametricTable or an InputParameter entity.

Required Arguments

FileName The exported file name.

Type string (p. 1438)

Optional Arguments

AppendMode True to append to an existing csv file, False to overwrite it.

Type bool (p. 1360)

Default Value False

Example

The following example shows how the user can export the table of Design Points and then the Parameters Parallel chart of a Design of Experiments component.

```
container = system1.GetContainer(ComponentName="Design of Experiment")
model = container.GetModel()
parametricTable = model.GetParametricTable(Name="DesignPoints")
parametricTable.ExportData(FileName="doe.csv")
chart = model.GetChart(Name="DesignPointsParallel")
chart.ExportData(FileName="doe.csv", AppendMode=True)
```

ExportSnapshotsTableAsArchive

Export a table of snapshots as an archive.

Required Arguments

FilePath The exported file name.

Type string (p. 1438)

Example

The following example shows how the user can export a table of snapshots as an archive.

```
system1 = GetSystem(Name="RBU")
designofExperiment1 = system1.GetContainer(ComponentName="Design of Experiment")
dOEModel1 = designofExperiment1.GetModel()
parametricTable1 = dOEModel1.GetParametricTable(Name="DesignPoints")
parametricTable1.ExportSnapshotsTableAsArchive(FilePath="C:/Temp/doesnapshots.snpz")
```

GetCellValue

Get the Value of a ParametricTable's cell. An exception is thrown if the entity is not found.

Return The value of the cell.

Type string (p. 1438)

Required Arguments

ColumnIndex ColumnIndex (zero-based) of the cell.

Type int (p. 1394)

RowIndex RowIndex (zero-based) of the cell.

Type int (p. 1394)

Example

The following example shows how the user can get the value of the ParametricTable's cell [0,3].

```
system1 = GetSystem(Name="RSO")
designofExperiment1 = system1.GetContainer(ComponentName="Design of Experiment")
dOEModel1 = designofExperiment1.GetModel()
parametricTable = dOEModel1.GetParametricTable(Name="DesignPoints")
cellValue = parametricTable.GetCellValue(RowIndex=0,ColumnIndex=3)
```

GetRowRomSnapshotName

Get the Rom snapshot's name value of a row in a ParametricTable. An exception is thrown if the entity is not found.

Return The value of the Rom snapshot's name.

Type string (p. 1438)

Required Arguments

RowIndex RowIndex (zero-based) of the cell.

Type int (p. 1394)

Example

The following example shows how to get the Rom snapshot's name value of row [3] in the ParametricTable.

```
system1 = GetSystem(Name="RBU")
designofExperiment1 = system1.GetContainer(ComponentName="Design of Experiment")
dOEModel1 = designofExperiment1.GetModel()
parametricTable1 = dOEModel1.GetParametricTable(Name="DesignPoints")
snapshotName = parametricTable1.GetRowRomSnapshotName(RowIndex=3)
```

GetRowUpdateOrder

Get the Update Order value of a row in a ParametricTable. An exception is thrown if the entity is not found.

Return The value of the update order.

Type double (p. 1378)

Required Arguments

RowIndex RowIndex (zero-based) of the cell.

Type int (p. 1394)

Example

The following example shows how to get the Update Order value of row [3] in the ParametricTable.

```
system1 = GetSystem(Name="RSO")
designofExperiment1 = system1.GetContainer(ComponentName="Design of Experiment")
dOEModel1 = designofExperiment1.GetModel()
parametricTable = dOEModel1.GetParametricTable(Name="DesignPoints")
updateOrderValue = parametricTable.GetRowUpdateOrder(RowIndex=3)
```

GetRowValues

Get the Values of a ParametricTable's row. An exception is thrown if the entity is not found.

Return List of the values of the specified row.

Type List (p. 1400)<string (p. 1438)>

Required Arguments

RowIndex RowIndex (zero-based) of the row to retrieve.

Type int (p. 1394)

Example

The following example shows how the user can get the values of the 4th ParametricTable's row.

```
system1 = GetSystem(Name="RSO")
designofExperiment1 = system1.GetContainer(ComponentName="Design of Experiment")
dOEModel1 = designofExperiment1.GetModel()
parametricTable = dOEModel1.GetParametricTable(Name="DesignPoints")
rowValues = parametricTable.GetRowValues(RowIndex=3)
```

OptimizeUpdateOrder

Optimizes the Update Order of Design Points to minimize the number of modifications between two consecutive Design Points. This command applies to the "DesignPoints" ParametricTable of a Design of Experiments model or of a Response Surface in a manual refinement context. It also applies to the "VerificationPoints" ParametricTable of a Response Surface model.

Example

The following example shows how to optimize the update order of the DesignPoints table in a Design of Experiments model.

```
container = system1.GetContainer(ComponentName="Design of Experiment")
model = container.GetModel()
model.DOEType = "eDOETYPE_USER"
DOEMatrix = model.GetParametricTable(Name="DesignPoints")
DOEMatrix.OptimizeUpdateOrder()
```

SetCellValue

Sets the value of a ParametricTable cell. If the table is read-only, the command has no effect.

Required Arguments

ColumnIndex Zero-based column index of the cell.

Type int (p. 1394)

RowIndex Zero-based row index of the cell.

Type int (p. 1394)

Value New value of the cell.

Type string (p. 1438)

Example

The following example shows how to set the value of the DesignPoints table in a custom DOE context.

```
container = system1.GetContainer(ComponentName="Design of Experiment")
model = container.GetModel()
model.DOEType = "eDOETYPE_USER"
DOEMatrix = model.GetParametricTable(Name="DesignPoints")
DOEMatrix.AddRow()
DOEMatrix.SetCellValue(RowIndex=0, ColumnIndex=0, Value="12.5 [mm]")
```

SetOutputValuesEditable

Sets the values of the output parameters as Editable (Editable=True) or Calculated (Editable=False) for the complete table, or a set of rows specified by the RowIndices argument. This command is applicable to the "DesignPoints" ParametricTable of a Design of Experiments model in a custom context (DOEType is "eDOETYPE_USER" or "eDOETYPE_CUSTOM_OSFD"), of a Correlation model is a custom context (SamplingType is "eCustom"), or of a Response Surface model in a manual refinement context. It also applies to the "VerificationPoints" ParametricTable of a Response Surface model.

Required Arguments

Editable True to define output values as Editable, or False to mark output values as calculated.

Type bool (p. 1360)

Optional Arguments

RowIndices Optional list of row zero-based indices. If this argument is not specified, the command applies to the complete ParametricTable.

Type List (p. 1400)<int (p. 1394)>

Example

The following example shows how to switch a DOE model to the custom mode and then set the output values as editable for the first three design points of the DOE matrix.

```
container = system1.GetContainer(ComponentName="Design of Experiment")
model = container.GetModel()
model.DOEType = "eDOETYPE_USER"
DOEMatrix = model.GetParametricTable(Name="DesignPoints")
DOEMatrix.SetOutputValuesEditable(Editable=True, RowIndices=[0,1,2])
```

SetRowRomSnapshotName

Sets the Rom Snapshot's name value for a row in a ParametricTable. If the table doesn't support Rom functionality, the command has no effect.

Required Arguments

RowIndex Zero-based row index of the cell.

Type int (p. 1394)

SnapshotName New value of the Rom snapshot's name.

Type string (p. 1438)

Example

The following example shows how to set the Rom Snapshot's name value of the DesignPoints table in a Design of Experiments model.

```
system1 = GetSystem(Name="RBU")
designofExperiment1 = system1.GetContainer(ComponentName="Design of Experiment")
dOEModel1 = designofExperiment1.GetModel()
parametricTable1 = dOEModel1.GetParametricTable(Name="DesignPoints")
parametricTable1.SetOutputValuesEditable(Editable=True,RowIndices=[1])
parametricTable1.SetRowRomSnapshotName(RowIndex=1,SnapshotName="ff66dfaf-c34b-4dff-938e-bf497a377b9f.romsn
```

SetRowUpdateOrder

Sets the Update Order value for a row in a ParametricTable. If the table doesn't support Update Order functionality, the command has no effect.

Required Arguments

RowIndex Zero-based row index of the cell.

Type int (p. 1394)

UpdateOrder New value of the update order.

Type double (p. 1378)

Example

The following example shows how to set the Update Order value of the DesignPoints table in a Design of Experiments model.

```
container = system1.GetContainer(ComponentName="Design of Experiment")
model = container.GetModel()
DOEMatrix = model.GetParametricTable(Name="DesignPoints")
DOEMatrix.SetRowUpdateOrder(RowIndex=0, Value="2.0")
```

SetRowValues

Sets the values of a ParametricTable's row. If the table is read-only, the command has no effect.

Required Arguments

RowIndex Zero-based row index of the cell.

Type int (p. 1394)

RowValues New values for the row.

Type List (p. 1400)<string (p. 1438)>

Example

The following example shows how to set the values of the DesignPoints table in a custom DOE context.

```
container = system1.GetContainer(ComponentName="Design of Experiment")
model = container.GetModel()
model.DOEType = "eDOETYPE_USER"
DOEMatrix = model.GetParametricTable(Name="DesignPoints")
DOEMatrix.AddRow()
DOEMatrix.SetRowValues(RowIndex=0, RowValues=[ "12.5 [mm]", "1" ] )
```

SetUpdateOrderByRow

Sets a value for the UpdateOrder property of all Design Points using sorting settings. This command applies to the "DesignPoints" ParametricTable of a Design of Experiments model or of a Response Surface in a manual refinement context. It also applies to the "VerificationPoints" ParametricTable of a Response Surface model.

Optional Arguments

SortBy Definition of the sort.

Type List (p. 1400)<string (p. 1438)>

Example

The following example shows how to set the update order of the DesignPoints table in a Design of Experiments model. The Design Points are sorted first by their values for the parameter P3 (in ascending order), and then by their values for the parameter P1 (in descending order).

```
container = system1.GetContainer(ComponentName="Design of Experiment")
model = container.GetModel()
DOEMatrix = model.GetParametricTable(Name="DesignPoints")
DOEMatrix.SetUpdateOrderByRow(SortBy=["P3:+", "P1:-"])
```

UpdateRows

Updates the design points held in rows from a ParametricTable entity and the results which depend on it.

Required Arguments

Indices Indices of the rows to update.

Type List (p. 1400)<int (p. 1394)>

Example

The following example shows how to update the design points from the VerificationPoints Table.

```
responseSurface1 = system1.GetContainer(ComponentName="Response Surface")
responseSurfaceModel1 = responseSurface1.GetModel()
```

```
parametricTable1 = responseSurfaceModel1.GetParametricTable(Name="VerificationPoints")
parametricTable1.UpdateRows(Indices=[0,1])
```

PredictedvsObservedScatterChart

The data entity which describes a PredictedvsObserved Scatter chart. This scatter chart allows the user to visualize the predicted output values versus observed output values for all design points used to compute the response surface.

Properties

DisplayParameterFullName

If True, the legend of the chart contains the full name of the parameters. Otherwise it contains the short name such as "P1".

Type bool (p. 1360)

Read Only No

DisplayText

The general property that defines the user-visible name of an entity. This property is defined for all data entities but is used only in those entities that present a label in the user interface.

Type string (p. 1438)

Read Only No

IsUpToDate

True if the entity is up-to-date.

Type bool (p. 1360)

Read Only No

ShowLearningPoints

Option to display the design points.

Type bool (p. 1360)

Read Only No

ShowVerificationPoints

Option to display the verification points.

Type bool (p. 1360)

Read Only No

Methods

EnableParameters

Enable or disable a list of parameters in a chart.

Required Arguments

IsEnabled False to disable the parameters, or True (default) to enable them.

Type bool (p. 1360)

Optional Arguments

Parameters Parameters to enable or disable, or all parameters if not specified.

Type List (p. 1400)<DataReference (p. 1371)>

Example

The following example shows how to disable two parameters, and then how to enable all parameters by omitting the Parameters optional parameter. This example uses a Correlation Matrix chart. The method also applies for other types of charts like LocalSensitivity, SpiderChart, etc.

```
container = system1.GetContainer(ComponentName="Correlation")
model = container.GetModel()
param1 = model.GetParam(Name="P1")
param4 = model.GetParam(Name="P4")
chart = model.GetChart(Name="Correlation Matrix 1")
chart.EnableParameters( Parameters=[param1;param4], IsEnabled=false )
chart.EnableParameters()
```

ExportData

Export the data of a DesignXplorer entity to a csv file. The entity can be one of the DesignXplorer chart entities, a ParametricTable or an InputParameter entity.

Required Arguments

FileName The exported file name.

Type string (p. 1438)

Optional Arguments

AppendMode True to append to an existing csv file, False to overwrite it.

Type bool (p. 1360)

Default Value False

Example

The following example shows how the user can export the table of Design Points and then the Parameters Parallel chart of a Design of Experiments component.

```
container = system1.GetContainer(ComponentName="Design of Experiment")
model = container.GetModel()
parametricTable = model.GetParametricTable(Name="DesignPoints")
parametricTable.ExportData(FileName="doe.csv")
chart = model.GetChart(Name="DesignPointsParallel")
chart.ExportData(FileName="doe.csv", AppendMode=True)
```

Update

Updates the chart by generating all results or data required to plot it. If the chart is already up-to-date, nothing is done by default.

Example

The following example shows how to update a Tradeoff chart. The same code applies to all other types of charts.

```
container = system1.GetContainer(ComponentName="Optimization")
model = container.GetModel()
chart = model.GetChart(Name="TradeoffChart 1")
chart.Update()
```

PredictionErrorScatterChart

The data entity which describes a PredictionError Scatter chart. This scatter chart allows the user to visualize per region or for all regions, the metric errors versus all design points used to compute the ROM.

Properties

DisplayParameterFullName

If True, the legend of the chart contains the full name of the parameters. Otherwise it contains the short name such as "P1".

Type bool (p. 1360)

Read Only No

DisplayText

The general property that defines the user-visible name of an entity. This property is defined for all data entities but is used only in those entities that present a label in the user interface.

Type string (p. 1438)

Read Only No

ErrorType

No details are provided for this entry.

Type MetricErrorScatterType (p. 1405)

Read Only No

FieldToShow

Field to Show

Type string (p. 1438)

Read Only No

IsUpToDate

True if the entity is up-to-date.

Type bool (p. 1360)

Read Only No

RegionToShow

Region to Show

Type string (p. 1438)

Read Only No

ScatterType

Scatter Type

Type PredictionErrorScatterType (p. 1418)

Read Only No

ShowLearningPoints

Option to display the design points.

Type bool (p. 1360)

Read Only No

ShowVerificationPoints

Option to display the verification points.

Type bool (p. 1360)

Read Only No

ResponseChart

The data entity which describes a Response chart. It allows you to visualize the impact that changing each input parameter has on the selected output parameter. It has three Modes - 2D, 3D and 2DSlices - allowing you to vary one or two input parameters at the same time. A Response depends on a Response Point which serves as a reference point for non-varying input parameters.

Properties

Axes

Dictionary of the parameters associated to axes

Type Dictionary (p. 1375)<ChartAxes (p. 1363), DataReference (p. 1371)>

Read Only Yes

ChartResolutionAlongX

Chart resolution along the X axis.

Type int (p. 1394)

Read Only No

ChartResolutionAlongY

Chart resolution along the Y axis.

Type int (p. 1394)

Read Only No

DisplayParameterFullName

If True, the legend of the chart contains the full name of the parameters. Otherwise it contains the short name such as "P1".

Type bool (p. 1360)

Read Only No

DisplayText

The general property that defines the user-visible name of an entity. This property is defined for all data entities but is used only in those entities that present a label in the user interface.

Type string (p. 1438)

Read Only No

IsUpToDate

True if the entity is up-to-date.

Type bool (p. 1360)

Read Only No

Mode

Response chart mode: 2D, 3D or 2D Slices.

Type ResponseChartModes (p. 1425)

Read Only No

NumberOfSlices

Number of slices for the 2D Slices mode.

Type int (p. 1394)

Read Only No

ShowDesignPoints

Option to display the design points of the DOE and the refinement points.

Type bool (p. 1360)

Read Only No

Methods

AssociateParameterToAxis

Associates a Parameter to the specified axis of the chart. The Parameter argument can be omitted which means that the axis is not set.

Required Arguments

Axis Axis to modify.

Type ChartAxes (p. 1363)

Optional Arguments

Parameter Parameter entity to be assigned to the specified axis.

Type DataReference (p. 1371)

Example

The following example shows how to assign parameters to the axes of a DesignPointsCurves chart.

```
container = system1.GetContainer(ComponentName="Design of Experiment")
model = container.GetModel()
inputParam1 = model.GetParam(Name="P1")
outputParam1 = model.GetParam(Name="P5")
outputParam2 = model.GetParam(Name="P6")
```

```
chart = model.GetChart(Name="Design Point vs Parameter 1")
chart.AssociateParameterToAxis(Parameter=inputParam1, Axis="XAxis")
chart.AssociateParameterToAxis(Parameter=outputParam1, Axis="YAxis")
chart.AssociateParameterToAxis(Parameter=outputParam2, Axis="YRightAxis")
```

ExportData

Export the data of a DesignXplorer entity to a csv file. The entity can be one of the DesignXplorer chart entities, a ParametricTable or an InputParameter entity.

Required Arguments

FileName The exported file name.

Type string (p. 1438)

Optional Arguments

AppendMode True to append to an existing csv file, False to overwrite it.

Type bool (p. 1360)

Default Value False

Example

The following example shows how the user can export the table of Design Points and then the Parameters Parallel chart of a Design of Experiments component.

```
container = system1.GetContainer(ComponentName="Design of Experiment")
model = container.GetModel()
parametricTable = model.GetParametricTable(Name="DesignPoints")
parametricTable.ExportData(FileName="doe.csv")
chart = model.GetChart(Name="DesignPointsParallel")
chart.ExportData(FileName="doe.csv", AppendMode=True)
```

Update

Updates the chart by generating all results or data required to plot it. If the chart is already up-to-date, nothing is done by default.

Example

The following example shows how to update a Tradeoff chart. The same code applies to all other types of charts.

```
container = system1.GetContainer(ComponentName="Optimization")
model = container.GetModel()
chart = model.GetChart(Name="TradeoffChart 1")
chart.Update()
```

ResponsePoint

The ResponsePoint data entity: a design point for which output values are computed from a Response Surface.

Properties

DisplayText

The general property that defines the user-visible name of an entity. This property is defined for all data entities but is used only in those entities that present a label in the user interface.

Type string (p. 1438)

Read Only No

Expressions

A Dictionary of the input parameter values defining the response point and the corresponding output parameter values predicted from the response surface. Note that for discrete parameters and continuous parameters using manufacturable values, the Dictionary contains the value of the level rather than its index.

Type Dictionary (p. 1375)<DataReference (p. 1371), Object (p. 1409)>

Read Only No

Note

User notes about the response point.

Type string (p. 1438)

Read Only No

Methods

CreateChart

Creates a Chart entity attached to the Response Point. This chart is updated automatically when the parameter values of the ResponsePoint change. It is deleted automatically when the ResponsePoint is deleted. The chart must be updated once after its creation by invoking its Update method. Then changes to its properties or its parent ResponsePoint will update the chart automatically.

Return The DataReference of the Chart.

Type DataReference (p. 1371)

Required Arguments

ChartType Type of chart to be created. The possible values are "eChartLocalSensitivity", "eChartLocalSensitivityCurves", "eChartResponse" and "eChartSpiderResponses"

Type ChartType (p. 1363)

Optional Arguments

DisplayText DisplayText of the chart. If not specified, a default name is applied, depending on the value of ChartType.

Type string (p. 1438)

Example

The following example shows how to create a ResponsePoint chart.

```
container = system1.GetContainer(ComponentName="Response Surface")
model = container.GetModel()
responsePoint = model.GetResponsePoint(Name="Response Point 1")
spiderChart = responsePoint.CreateChart(ChartType="eChartSpiderResponses")
spiderChart.Update()
```

DuplicateChart

Duplicates a Chart entity attached to a Response Point. The chart must be updated once after its creation by invoking its Update method. Then changes to its properties will update the chart automatically.

Return The DataReference of the Chart.

Type DataReference (p. 1371)

Required Arguments

Chart The source chart to duplicate.

Type DataReference (p. 1371)

Optional Arguments

DisplayText Displayed name of the chart. If not specified, a default name is applied, depending on the value of ChartType.

Type string (p. 1438)

TargetResponsePoint ResponsePoint on which the duplicated chart is created. If this parameter is not set, the response point of the source chart is used.

Type DataReference (p. 1371)

Example

The following example shows how to duplicate a chart.

```
container = system1.GetContainer(ComponentName="Response Surface")
model = container.GetModel()
responsePoint = model.GetResponsePoint(Name="Response Point 1")
spiderChart = responsePoint.GetChart(Name="Spider")
spiderChart2 = responsePoint.DuplicateChart(Chart=spiderChart)
spiderChart2.Update()
```

GetChart

Get the DataReference of a Chart from a ResponsePoint. An exception is thrown if the entity is not found.

Return The DataReference of the Chart entity.

Type DataReference (p. 1371)

Required Arguments

Name Name of the chart

Type string (p. 1438)

Example

The following example shows how to retrieve an existing Chart from a ResponsePoint.

```
container = system1.GetContainer(ComponentName="Response Surface")
model = container.GetModel()
rp = model.GetResponsePoint(Name="Response Point 1")
chart = rp.GetChart(Name="SpiderChart 1")
```

GetCharts

Get the DataReferences of the Charts from a ResponsePoint.

Return The DataReferences of the Chart entities.

Type DataReferenceSet (p. 1371)

Example

The following example shows how to retrieve all the charts defined for a ResponsePoint.

```
container = system1.GetContainer(ComponentName="Response Surface")
model = container.GetModel()
rp = model.GetResponsePoint(Name="Response Point 1")
charts = rp.GetCharts()
```

SetDiscreteParameters

Sets the Expression of several discrete input parameters in a GoodnessOfFit and updates the associated output parameter values. The chart depending on the GoodnessOfFit is updated as well.

Required Arguments

Expressions Dictionary of the discrete input parameters and their assigned expressions.

Type Dictionary (p. 1375)<DataReference (p. 1371), string (p. 1438)>

Example

The following example shows how to set the expression for several discrete input parameters in an existing GoodnessOfFit.

```
container = system1.GetContainer(ComponentName="Response Surface")
model = container.GetModel()
gof = model.GetGoodnessOfFit(Name="Goodness Of Fit 1")
inputParameter1 = model.GetParameter(Name="P1")
inputParameter2 = model.GetParameter(Name="P2")
gof.SetDiscreteParameters(Expressions={inputParameter1: "2", inputParameter2: "15"})
```

SetParameter

Sets the Expression of an input parameter in a ResponsePoint and updates the associated output parameter values. If there are chart entities depending on the ResponsePoint, they are updated as well.

Required Arguments

Expression Assigned Expression (value or quantity).

Type string (p. 1438)

Parameter DataReference of the Input parameter.

Type DataReference (p. 1371)

Example

The following example shows how to set the expression for one parameter in an existing ResponsePoint.

```
container = system1.GetContainer(ComponentName="Response Surface")
model = container.GetModel()
rp = model.GetResponsePoint(Name="Response Point 1")
inputParameter1 = model.GetParameter(Name="P1")
rp.SetParameter(Parameter=inputParameter1, Expression="2.01")
inputParameter2 = model.GetParameter(Name="P2")
rp.SetParameter(Parameter=inputParameter2,Expression="2.1 [m]")
```

SetParameters

Sets the Expression of several input parameters in a ResponsePoint and updates the associated output parameter values. If there are chart entities depending on the ResponsePoint, they are updated as well.

Required Arguments

Expressions Dictionary of the input parameters and their assigned expressions.

Type Dictionary (p. 1375)<DataReference (p. 1371), string (p. 1438)>

Example

The following example shows how to set the expression for several input parameters in an existing ResponsePoint.

```
container = system1.GetContainer(ComponentName="Response Surface")
model = container.GetModel()
rp = model.GetResponsePoint(Name="Response Point 1")
inputParameter1 = model.GetParameter(Name="P1")
inputParameter2 = model.GetParameter(Name="P2")
rp.SetParameters(Expressions={inputParameter1: "2.01", inputParameter2: "15.5 [m]"})
```

Update

Updates the response point and the results which depend on it. If the reponse point is already uptodate, nothing is done unless the Force flag is set to True.

Optional Arguments

Force Set to true if the update operation of the response point is required even if it's already uptodate.

Type bool (p. 1360)

Default Value False

Example

The following example shows how to update an existing ResponsePoint.

```
container = system1.GetContainer(ComponentName="Response Surface")
model = container.GetModel()
rp = model.GetResponsePoint(Name="Response Point 1")
rp.Update()
```

SamplesChart

The data entity which describes the Samples chart. It allows you to explore the samples generated for an Optimization study by using parallel Y axes to represent all of the input and output parameters on the same 2D representation, whatever the number of parameters. It can filter the number of visible Pareto Fronts and supports two Modes: Candidates or Pareto Fronts.

Properties

ColoringMethod

Coloring method used to draw the chart.

Type ChartColoringMethods (p. 1363)

Read Only No

DisplayParameterFullName

If True, the legend of the chart contains the full name of the parameters. Otherwise it contains the short name such as "P1".

Type bool (p. 1360)

Read Only No

DisplayText

The general property that defines the user-visible name of an entity. This property is defined for all data entities but is used only in those entities that present a label in the user interface.

Type string (p. 1438)

Read Only No

IsUpToDate

True if the entity is up-to-date.

Type bool (p. 1360)

Read Only No

Mode

Samples chart mode.

Type SamplesChartModes (p. 1428)

Read Only No

NumberOfParetoFront

Number of Pareto fronts to display. This is used as a filter to display only the most interesting fronts, given an optimization study.

Type uint (p. 1446)

Read Only No

ShowInfeasiblePoints

If True, any infeasible points are displayed on the chart.

Type bool (p. 1360)

Read Only No

Methods

EnableParameters

Enable or disable a list of parameters in a chart.

Required Arguments

IsEnabled False to disable the parameters, or True (default) to enable them.

Type bool (p. 1360)

Optional Arguments

Parameters Parameters to enable or disable, or all parameters if not specified.

Type List (p. 1400)<DataReference (p. 1371)>

Example

The following example shows how to disable two parameters, and then how to enable all parameters by omitting the Parameters optional parameter. This example uses a Correlation Matrix chart. The method also applies for other types of charts like LocalSensitivity, SpiderChart, etc.

```
container = system1.GetContainer(ComponentName="Correlation")
model = container.GetModel()
param1 = model.GetParam(Name="P1")
param4 = model.GetParam(Name="P4")
chart = model.GetChart(Name="Correlation Matrix 1")
chart.EnableParameters( Parameters=[param1;param4], IsEnabled=false )
chart.EnableParameters()
```

ExportData

Export the data of a DesignXplorer entity to a csv file. The entity can be one of the DesignXplorer chart entities, a ParametricTable or an InputParameter entity.

Required Arguments

FileName The exported file name.

Type string (p. 1438)

Optional Arguments

AppendMode True to append to an existing csv file, False to overwrite it.

Type bool (p. 1360)

Default Value False

Example

The following example shows how the user can export the table of Design Points and then the Parameters Parallel chart of a Design of Experiments component.

```
container = system1.GetContainer(ComponentName="Design of Experiment")
model = container.GetModel()
parametricTable = model.GetParametricTable(Name="DesignPoints")
parametricTable.ExportData(FileName="doe.csv")
chart = model.GetChart(Name="DesignPointsParallel")
chart.ExportData(FileName="doe.csv", AppendMode=True)
```

Update

Updates the chart by generating all results or data required to plot it. If the chart is already up-to-date, nothing is done by default.

Example

The following example shows how to update a Tradeoff chart. The same code applies to all other types of charts.

```
container = system1.GetContainer(ComponentName="Optimization")
model = container.GetModel()
chart = model.GetChart(Name="TradeoffChart 1")
chart.Update()
```

SensitivitiesChart

The data entity which describes a Sensitivities chart. It allows you to visualize the global sensitivities of each output with respect to the input parameters. A Sensitivities chart has two different graphical modes -BarChart and PieChart- and supports enabling/disabling input and/or output parameters.

Properties

DisplayParameterFullName

If True, the legend of the chart contains the full name of the parameters. Otherwise it contains the short name such as "P1".

Type bool (p. 1360)

Read Only No

DisplayText

The general property that defines the user-visible name of an entity. This property is defined for all data entities but is used only in those entities that present a label in the user interface.

Type string (p. 1438)

Read Only No

IsUpToDate

True if the entity is up-to-date.

Type bool (p. 1360)

Read Only No

Mode

Specifies the graphical representation used to render local sensitivity data. It can be either BarChart or PieChart.

Type SensitivityChartModes (p. 1430)

Read Only No

Methods

EnableParameters

Enable or disable a list of parameters in a chart.

Required Arguments

IsEnabled False to disable the parameters, or True (default) to enable them.

Type bool (p. 1360)

Optional Arguments

Parameters Parameters to enable or disable, or all parameters if not specified.

Type List (p. 1400)<DataReference (p. 1371)>

Example

The following example shows how to disable two parameters, and then how to enable all parameters by omitting the Parameters optional parameter. This example uses a Correlation Matrix chart. The method also applies for other types of charts like LocalSensitivity, SpiderChart, etc.

```
container = system1.GetContainer(ComponentName="Correlation")
model = container.GetModel()
param1 = model.GetParam(Name="P1")
param4 = model.GetParam(Name="P4")
chart = model.GetChart(Name="Correlation Matrix 1")
chart.EnableParameters( Parameters=[param1;param4], IsEnabled=false )
chart.EnableParameters()
```

ExportData

Export the data of a DesignXplorer entity to a csv file. The entity can be one of the DesignXplorer chart entities, a ParametricTable or an InputParameter entity.

Required Arguments

FileName The exported file name.

Type string (p. 1438)

Optional Arguments

AppendMode True to append to an existing csv file, False to overwrite it.

Type bool (p. 1360)

Default Value False

Example

The following example shows how the user can export the table of Design Points and then the Parameters Parallel chart of a Design of Experiments component.

```
container = system1.GetContainer(ComponentName="Design of Experiment")
model = container.GetModel()
parametricTable = model.GetParametricTable(Name="DesignPoints")
parametricTable.ExportData(FileName="doe.csv")
chart = model.GetChart(Name="DesignPointsParallel")
chart.ExportData(FileName="doe.csv", AppendMode=True)
```

Update

Updates the chart by generating all results or data required to plot it. If the chart is already up-to-date, nothing is done by default.

Example

The following example shows how to update a Tradeoff chart. The same code applies to all other types of charts.

```
container = system1.GetContainer(ComponentName="Optimization")
model = container.GetModel()
chart = model.GetChart(Name="TradeoffChart 1")
chart.Update()
```

SpiderChart

The data entity which describes a Spider chart. It allows you to visualize the impact that changing the input parameters has on all output parameters simultaneously. A Spider chart depends on a Response Point and shows the same values as its parent Response Point.

Properties

DisplayParameterFullName

If True, the legend of the chart contains the full name of the parameters. Otherwise it contains the short name such as "P1".

Type bool (p. 1360)

Read Only No

DisplayText

The general property that defines the user-visible name of an entity. This property is defined for all data entities but is used only in those entities that present a label in the user interface.

Type string (p. 1438)

Read Only No

IsUpToDate

True if the entity is up-to-date.

Type bool (p. 1360)

Read Only No

Methods

EnableParameters

Enable or disable a list of parameters in a chart.

Required Arguments

IsEnabled False to disable the parameters, or True (default) to enable them.

Type bool (p. 1360)

Optional Arguments

Parameters Parameters to enable or disable, or all parameters if not specified.

Type List (p. 1400)<DataReference (p. 1371)>

Example

The following example shows how to disable two parameters, and then how to enable all parameters by omitting the Parameters optional parameter. This example uses a Correlation Matrix chart. The method also applies for other types of charts like LocalSensitivity, SpiderChart, etc.

```
container = system1.GetContainer(ComponentName="Correlation")
model = container.GetModel()
param1 = model.GetParam(Name="P1")
```

```
param4 = model.GetParam(Name="P4")
chart = model.GetChart(Name="Correlation Matrix 1")
chart.EnableParameters( Parameters=[param1;param4], IsEnabled=false )
chart.EnableParameters()
```

ExportData

Export the data of a DesignXplorer entity to a csv file. The entity can be one of the DesignXplorer chart entities, a ParametricTable or an InputParameter entity.

Required Arguments

FileName The exported file name.

Type string (p. 1438)

Optional Arguments

AppendMode True to append to an existing csv file, False to overwrite it.

Type bool (p. 1360)

Default Value False

Example

The following example shows how the user can export the table of Design Points and then the Parameters Parallel chart of a Design of Experiments component.

```
container = system1.GetContainer(ComponentName="Design of Experiment")
model = container.GetModel()
parametricTable = model.GetParametricTable(Name="DesignPoints")
parametricTable.ExportData(FileName="doe.csv")
chart = model.GetChart(Name="DesignPointsParallel")
chart.ExportData(FileName="doe.csv", AppendMode=True)
```

Update

Updates the chart by generating all results or data required to plot it. If the chart is already up-to-date, nothing is done by default.

Example

The following example shows how to update a Tradeoff chart. The same code applies to all other types of charts.

```
container = system1.GetContainer(ComponentName="Optimization")
model = container.GetModel()
chart = model.GetChart(Name="TradeoffChart 1")
chart.Update()
```

StatisticsChart

The StatisticsChart shows the Probability Distribution Function and Cumulative Distribution Function results of a Six Sigma analysis

Properties

DisplayParameterFullName

If True, the legend of the chart contains the full name of the parameters. Otherwise it contains the short name such as "P1".

Type bool (p. 1360)

Read Only No

DisplayText

The general property that defines the user-visible name of an entity. This property is defined for all data entities but is used only in those entities that present a label in the user interface.

Type string (p. 1438)

Read Only No

IsUpToDate

True if the entity is up-to-date.

Type bool (p. 1360)

Read Only No

Methods

ExportData

Export the data of a DesignXplorer entity to a csv file. The entity can be one of the DesignXplorer chart entities, a ParametricTable or an InputParameter entity.

Required Arguments

FileName The exported file name.

Type string (p. 1438)

Optional Arguments

AppendMode True to append to an existing csv file, False to overwrite it.

Type bool (p. 1360)

Default Value False

Example

The following example shows how the user can export the table of Design Points and then the Parameters Parallel chart of a Design of Experiments component.

```
container = system1.GetContainer(ComponentName="Design of Experiment")
model = container.GetModel()
parametricTable = model.GetParametricTable(Name="DesignPoints")
parametricTable.ExportData(FileName="doe.csv")
chart = model.GetChart(Name="DesignPointsParallel")
chart.ExportData(FileName="doe.csv", AppendMode=True)
```

Update

Updates the chart by generating all results or data required to plot it. If the chart is already up-to-date, nothing is done by default.

Example

The following example shows how to update a Tradeoff chart. The same code applies to all other types of charts.

```
container = system1.GetContainer(ComponentName="Optimization")
model = container.GetModel()
chart = model.GetChart(Name="TradeoffChart 1")
chart.Update()
```

TradeoffChart

The data entity which describes the Tradeoff chart. It allows you to visualize the samples used in an optimization study and the Pareto fronts associated with them, if any. It supports the exploration of the generated samples in a 2D or 3D Mode, and can filter the number of visible Pareto Fronts.

Properties

Axes

Dictionary of the parameters associated to axes.

Type Dictionary (p. 1375)<ChartAxes (p. 1363), DataReference (p. 1371)>

Read Only Yes

DisplayParameterFullName

If True, the legend of the chart contains the full name of the parameters. Otherwise it contains the short name such as "P1".

Type bool (p. 1360)

Read Only No

DisplayText

The general property that defines the user-visible name of an entity. This property is defined for all data entities but is used only in those entities that present a label in the user interface.

Type string (p. 1438)

Read Only No

IsUpToDate

True if the entity is up-to-date.

Type bool (p. 1360)

Read Only No

Mode

Chart mode, either 2D or 3D.

Type TradeoffChartModes (p. 1442)

Read Only No

NumberOfParetoFront

Number of Pareto front to display. This is used as a filter to display only the most interesting fronts, given an optimization study.

Type uint (p. 1446)

Read Only No

ShowInfeasiblePoints

If True, any infeasible points are displayed on the chart.

Type bool (p. 1360)

Read Only No

Methods

AssociateParameterToAxis

Associates a Parameter to the specified axis of the chart. The Parameter argument can be omitted which means that the axis is not set.

Required Arguments

Axis Axis to modify.

Type ChartAxes (p. 1363)

Optional Arguments

Parameter Parameter entity to be assigned to the specified axis.

Type DataReference (p. 1371)

Example

The following example shows how to assign parameters to the axes of a DesignPointsCurves chart.

```
container = system1.GetContainer(ComponentName="Design of Experiment")
model = container.GetModel()
inputParam1 = model.GetParam(Name="P1")
outputParam1 = model.GetParam(Name="P5")
outputParam2 = model.GetParam(Name="P6")
chart = model.GetChart(Name="Design Point vs Parameter 1")
chart.AssociateParameterToAxis(Parameter=inputParam1, Axis="XAxis")
chart.AssociateParameterToAxis(Parameter=outputParam1, Axis="YAxis")
chart.AssociateParameterToAxis(Parameter=outputParam2, Axis="YRightAxis")
```

ExportData

Export the data of a DesignXplorer entity to a csv file. The entity can be one of the DesignXplorer chart entities, a ParametricTable or an InputParameter entity.

Required Arguments

FileName The exported file name.

Type string (p. 1438)

Optional Arguments

AppendMode True to append to an existing csv file, False to overwrite it.

Type bool (p. 1360)

Default Value False

Example

The following example shows how the user can export the table of Design Points and then the Parameters Parallel chart of a Design of Experiments component.

```
container = system1.GetContainer(ComponentName="Design of Experiment")
model = container.GetModel()
parametricTable = model.GetParametricTable(Name="DesignPoints")
parametricTable.ExportData(FileName="doe.csv")
chart = model.GetChart(Name="DesignPointsParallel")
chart.ExportData(FileName="doe.csv", AppendMode=True)
```

Update

Updates the chart by generating all results or data required to plot it. If the chart is already up-to-date, nothing is done by default.

Example

The following example shows how to update a Tradeoff chart. The same code applies to all other types of charts.

```
container = system1.GetContainer(ComponentName="Optimization")
model = container.GetModel()
chart = model.GetChart(Name="TradeoffChart 1")
chart.Update()
```

DX GDO Design of Experiment

This container holds Design of Experiment data for a Goal Driven Optimization.

Methods

ApproveGeneratedData

Approve generated data after nonparametric changes by keeping DX results. This command is recursive and applied with any component dependent on the first component.

Example

The following example shows how to approve generated data.

```
system1 = GetSystem(Name="RSO")
designofExperiment1 = system1.GetContainer(ComponentName="Design of Experiment")
designofExperiment1.ApproveGeneratedData()
```

GetModel

Get the DataReference of the Model. An exception is thrown if the entity is not found.

Return The DataReference of the Model.

Type DataReference (p. 1371)

Example

The following example shows how the user can get a Model to change one of its properties.

```
system1 = GetSystem(Name="RSO")
designofExperiment1 = system1.GetContainer(ComponentName="Design of Experiment")
dOEModel1 = designofExperiment1.GetModel()
```

```
dOEModel1.DOEType = "eDOETYPE_OSFD"
```

Data Entities

BBDMethod

Entity which performs and manages the DesignXplorer Sampling Method Component

Properties

DisplayText

The general property that defines the user-visible name of an entity. This property is defined for all data entities but is used only in those entities that present a label in the user interface.

Type string (p. 1438)

Read Only No

CCDMethod

Entity which performs and manages the DesignXplorer Sampling Method Component

Properties

CCDTemplateType

Template Type for CCD algorithm.

Type CCDTemplateType (p. 1362)

Read Only No

CCDType

Design Type for CCD algorithm.

Type CentralCompositeDesignType (p. 1362)

Read Only No

DisplayText

The general property that defines the user-visible name of an entity. This property is defined for all data entities but is used only in those entities that present a label in the user interface.

Type string (p. 1438)

Read Only No

CustomDOEMethod

Entity which performs and manages the DesignXplorer Sampling Method Component

Properties

DisplayText

The general property that defines the user-visible name of an entity. This property is defined for all data entities but is used only in those entities that present a label in the user interface.

Type string (p. 1438)

Read Only No

CustomDOEPlusSampling

Entity which performs and manages the DesignXplorer Sampling Method Component

Properties

DisplayText

The general property that defines the user-visible name of an entity. This property is defined for all data entities but is used only in those entities that present a label in the user interface.

Type string (p. 1438)

Read Only No

DiscreteLevel

The data entity which describes a Discrete Level of an Input Parameter.

Properties

DisplayText

The general property that defines the user-visible name of an entity. This property is defined for all data entities but is used only in those entities that present a label in the user interface.

Type string (p. 1438)

Read Only No

Index

Zero-based Index of the discrete level in the list of levels of the owning parameter.

Type int (p. 1394)

Read Only No

Value

Value of the DiscreteLevel.

Type Object (p. 1409)

Read Only No

Methods

SetValue

Sets the value of a discrete level entity. A discrete level can have an integer value (e.g. a number if holes, a number of turns, etc) or a string value (e.g. a material name or a geometry file name).

Required Arguments

Value Value set to the discrete level entity.

Type Object (p. 1409)

Example

The following example shows how to retrieve a discrete level from an input parameter and then change its value.

```
container = system1.GetContainer(ComponentName="Design of Experiment")
model = container.GetModel()
DiscreteInputParameter = model.GetParameter(Name="P2")
level1 = DiscreteInputParameter.GetDiscreteLevel(Name="Level 1")
level1.SetValue( Value="2500" )
```

DistributionChart

The entity which describes a parameter's Distribution chart. This chart is always associated with an uncertainty parameter and allows you to visualize the statistical distribution defined or calculated for this parameter.

Properties

DisplayParameterFullName

If True, the legend of the chart contains the full name of the parameters. Otherwise it contains the short name such as "P1".

Type bool (p. 1360)

Read Only No

DisplayText

The general property that defines the user-visible name of an entity. This property is defined for all data entities but is used only in those entities that present a label in the user interface.

Type string (p. 1438)

Read Only No

IsUpToDate

True if the entity is up-to-date.

Type bool (p. 1360)

Read Only No

Parameter

The parameter entity associated with the chart.

Type DataReference (p. 1371)

Read Only No

Methods

ExportData

Export the data of a DesignXplorer entity to a csv file. The entity can be one of the DesignXplorer chart entities, a ParametricTable or an InputParameter entity.

Required Arguments

FileName The exported file name.

Type string (p. 1438)

Optional Arguments

AppendMode True to append to an existing csv file, False to overwrite it.

Type bool (p. 1360)

Default Value False

Example

The following example shows how the user can export the table of Design Points and then the Parameters Parallel chart of a Design of Experiments component.

```
container = system1.GetContainer(ComponentName="Design of Experiment")
model = container.GetModel()
parametricTable = model.GetParametricTable(Name="DesignPoints")
parametricTable.ExportData(FileName="doe.csv")
chart = model.GetChart(Name="DesignPointsParallel")
chart.ExportData(FileName="doe.csv", AppendMode=True)
```

Update

Updates the chart by generating all results or data required to plot it. If the chart is already up-to-date, nothing is done by default.

Example

The following example shows how to update a Tradeoff chart. The same code applies to all other types of charts.

```
container = system1.GetContainer(ComponentName="Optimization")
model = container.GetModel()
chart = model.GetChart(Name="TradeoffChart 1")
chart.Update()
```

DOEModel

Entity which performs and manages the DesignXplorer Design Of Experiments Component

Properties

DisplayText

The general property that defines the user-visible name of an entity. This property is defined for all data entities but is used only in those entities that present a label in the user interface.

Type string (p. 1438)

Read Only No

ExportDesignPoints

If True and PreserveDesignPoints is True as well, export project for each preserved Design Point.

Type bool (p. 1360)

Read Only No

Method

Optimization Method

Type DataReference (p. 1371)

Read Only No

MethodName

Type of the Design of Experiments

Type string (p. 1438)

Read Only No

NumberOfRetries

Indicates the number of times DX will try to update the failed design points.

Type int (p. 1394)

Read Only No

PreserveDesignPoints

If True, preserve the Design Points at the project level after the component Update.

Type bool (p. 1360)

Read Only No

RetainDesignPoints

If True and PreserveDesignPoints is True as well, retain data for each preserved Design Point.

Type bool (p. 1360)

Read Only No

RetryDelay

Indicates how much time will elapse between tries. This option is only applicable when NumberOfRetries is greater than 0, otherwise it has no effect.

Type Quantity (p. 1422)

Read Only No

RomMeshFile

ROM mesh file manually selected by the user. Null in a non-ROM context or if the default generated one is used.

Type DataReference (p. 1371)

Read Only No

Methods

CreateChart

Creates a Chart entity. The chart must be updated once after its creation by invoking its Update method. Then changes to its properties will update the chart automatically.

Return The DataReference of the Chart.

Type DataReference (p. 1371)

Required Arguments

ChartType Type of chart to be created. The possible values depend on the type of Model. For instance, a ResponseSurface model accepts Spider, LocalSensitivity and Response chart while a CorrelationModel accepts CorrelationMatrix, DeterminationMatrix and CorrelationScatter charts.

Type ChartType (p. 1363)

Optional Arguments

DisplayText Displayed name of the chart. If not specified, a default name is applied, depending on the value of ChartType.

Type string (p. 1438)

Example

The following example shows how to create a chart.

```
container = system1.GetContainer(ComponentName="Optimization")
model = container.GetModel()
TradeoffChart = model.CreateChart(ChartType="eChartTradeoff")
TradeoffChart.Update()
```

DeleteCharts

Deletes a list of Chart entities.

Required Arguments

Charts List of Chart entities to delete.

Type List (p. 1400)<DataReference (p. 1371)>

Example

The following example shows how to delete existing charts.

```
container = system1.GetContainer(ComponentName="Optimization")
model = container.GetModel()
chart1 = model.GetChart(Name="TradeoffChart 1")
chart2 = model.GetChart(Name="SamplesChart 1")
model.DeleteCharts(Charts=[chart1, chart2])
```

DuplicateChart

Duplicates a Chart entity. The chart must be updated once after its creation by invoking its Update method. Then changes to its properties will update the chart automatically.

Return The DataReference of the Chart.

Type DataReference (p. 1371)

Required Arguments

Chart The source chart to duplicate.

Type DataReference (p. 1371)

Optional Arguments

DisplayText Displayed name of the chart. If not specified, a default name is applied, depending on the value of ChartType.

Type string (p. 1438)

TargetModel The model on which the duplicated chart is created. If this parameter is not set, the model of the source chart is used.

Type DataReference (p. 1371)

TargetResponsePoint Parent TargetResponsePoint of the chart. If this parameter is not set, the response point of the source chart is used if applicable.

Type DataReference (p. 1371)

Example

The following example shows how to duplicate a chart.

```
designofExperiment1 = system1.GetContainer(ComponentName="Design of Experiment")
dOEModel1 = designofExperiment1.GetModel()
designPointsCurvesChart1 = dOEModel1.GetChart(Name="DesignPointsCurves")
chart1 = dOEModel1.DuplicateChart(Chart=designPointsCurvesChart1)
chart1.Update()
```

GetChart

Query to return the chart reference for a given model and chart name.

Return The DataReference of the chart.

Type DataReference (p. 1371)

Required Arguments

Name Name of the chart.

Type string (p. 1438)

GetDesignPointReportSetting

Get the DataReference of a DesignPointReportSetting entity associated with a Model. An exception is thrown if the entity is not found.

Return The DataReference of the DesignPointReportSetting.

Type DataReference (p. 1371)

Example

The following example shows how the user can get a DesignPointReportSetting to change one of its properties.

```
system1 = GetSystem(Name="GDO")
designofExperiment1 = system1.GetContainer(ComponentName="Design of Experiment")
dOEModel1 = designofExperiment1.GetModel()
designPointReportSettingEntity1 = dOEModel1.GetDesignPointReportSetting()
designPointReportSettingEntity1.DesignPointReportImage = "FFF-Results:Figure001.png"
```

GetParameter

Get the DataReference of a Parameter. An exception is thrown if the entity is not found.

Return The DataReference of the Parameter.

Type DataReference (p. 1371)

Required Arguments

Name Name of the Parameter.

Type string (p. 1438)

Example

The following example shows how the user can get a parameter of a model to change one of its properties.

```
system1 = GetSystem(Name="RSO")
designofExperiment1 = system1.GetContainer(ComponentName="Design of Experiment")
dOEModel1 = designofExperiment1.GetModel()
inputParameter1 = dOEModel1.GetParameter(Name="P1")
inputParameter1.LowerBound = 1
```

GetParameters

Get the DataReferences of the InputParameter and OutputParameter of the model. If the optional argument InputParameters is not specified, the query returns all parameters. If it is specified and True, the query returns only input parameters. If it is False, the query returns only output parameters.

Return The DataReferences of the Parameters

Type DataReferenceSet (p. 1371)

Optional Arguments

InputParameters If True, the query returns only input parameters. If False, the query returns only output parameters. If the argument is ommitted, the query returns all parameters.

Type bool (p. 1360)

Example

The following example shows how the user can get the parameters of a model.

```
system1 = GetSystem(Name="RSO")
designofExperiment1 = system1.GetContainer(ComponentName="Design of Experiment")
dOEModel1 = designofExperiment1.GetModel()
parameters = dOEModel1.GetParameters()
```

GetParametricTable

Get the DataReference of ParametricTable. An exception is thrown if the entity is not found. Names of the tables generated internally are: "DesignPoints", "CorrelationMatrix", "CorrelationScatter", "MinDesignPoints", "MaxDesignPoints", "ResponsePoints", "DeterminationMatrix".

Return The DataReference of the ParametricTable

Type DataReference (p. 1371)

Required Arguments

Name Name of the ParametricTable

Type string (p. 1438)

Example

The following example shows how the user can get a ParametricTable to add a new row and set values.

```
system1 = GetSystem(Name="RSO")
designofExperiment1 = system1.GetContainer(ComponentName="Design of Experiment")
dOEModel1 = designofExperiment1.GetModel()
parametricTable = dOEModel1.GetParametricTable(Name="DesignPoints")
parametricTable.AddRow()
parametricTable.SetCellValue(RowIndex=9,ColumnIndex=0,Value="2.1")
```

ImportDesignPoints

Import existing Design Point entities in a custom Design of Experiments or Correlation model. If a Design Point is solved, the output parameter values are also imported if the customized model is not a Correlation model linked to a Response Surface model.

Optional Arguments

DesignPoints List of the existing Design Point entities to import. If not specified, all the Design Point entities found in the Parametric container are imported.

Type List (p. 1400)<DataReference (p. 1371)>

ExpandRanges If true, the command expands the range of variation of the input parameters based on the imported design points. This is done by modifying automatically the upper and lower bounds of the input parameters. If false, the design points which are not contained in the actual variation ranges are not imported. This option is not supported in the context of: - a DOE for a Six Sigma Analysis. - a Correlation based on a Response Surface.

Type bool (p. 1360)

Default Value False

ShrinkRanges If true, the command shrinks the range of variation of the input parameters based on the imported design points. This is done by modifying automatically the upper and lower bounds of the input parameters. If false, the design points which are not contained in the actual variation ranges are not imported. This option is not supported in the context of: - a DOE for a Six Sigma Analysis. - a Correlation based on a Response Surface.

Type bool (p. 1360)

Default Value False

Example

The following example shows how the user can import all design points or a list of design points into a Design of Experiments model.

```
container = system1.GetContainer(ComponentName="Design of Experiment")
model = container.GetModel()
model.DOEType = "eDOETYPE_USER"
model.ImportDesignPoints()
designPoint0 = Parameters.GetDesignPoint(Name="0")
designPoint1 = Parameters.GetDesignPoint(Name="1")
designPoint2 = Parameters.GetDesignPoint(Name="2")
model.ImportDesignPoints(DesignPoints=[designPoint0, designPoint1, designPoint2])
model.ImportDesignPoints(DesignPoints=[designPoint1])
```

ImportDesignPointsFromArchive

Import Design Point values in a custom Design of Experiments model from an archive file.

Required Arguments

ArchivePath The archive path.

Type string (p. 1438)

ReuseSnapshotFiles If true, reuse existing snapshot files of ROM production folder when they correspond to referenced snapshots of the archive. If false, copy all the snapshot files referenced in the archive (and rename them when they exist already in the ROM production folder).

Type bool (p. 1360)

Optional Arguments

ExpandRanges If true, the command expands the range of variation of the input parameters based on the imported design points. This is done by modifying automatically the upper and lower bounds of the input parameters. If false, the design points which are not contained in the actual variation ranges are not imported. This option is not supported in the context of a DOE for a Six Sigma Analysis.

Type bool (p. 1360)

Default Value False

ShrinkRanges If true, the command shrinks the range of variation of the input parameters based on the imported design points. This is done by modifying automatically the upper and lower bounds of the input parameters. If false, the design points which are not contained in the actual variation ranges are not imported. This option is not supported in the context of a DOE for a Six Sigma Analysis.

Type bool (p. 1360)

Default Value False

Example

The following example shows how the user can import design points from a valid archive file.

```
system = GetSystem(Name="RBU")
designofExperiment = system.GetContainer(ComponentName="Design of Experiment")
dOEModel = designofExperiment.GetModel()
dOEModel.MethodName = "ANSYS_USER"
dOEModel.ImportDesignPointsFromArchive(FileName="E:/temp/doesnapshots2.snpz",OverwriteSnapshotFile=False)
```

ImportDesignPointsFromDps

Import Design Point values in a custom Design of Experiments or Correlation model with a query to the DPS.

Required Arguments

QueryString The query.

Type string (p. 1438)

Optional Arguments

ExpandRanges If true, the command expands the range of variation of the input parameters based on the imported design points. This is done by modifying automatically the upper and lower bounds of the input parameters. If false, the design points which are not contained in the actual variation ranges are not imported. This option is not supported in the context of: - a DOE for a Six Sigma Analysis. - a Correlation based on a Response Surface.

Type bool (p. 1360)

Default Value False

ShrinkRanges If true, the command shrinks the range of variation of the input parameters based on the imported design points. This is done by modifying automatically the upper and lower bounds of the input parameters. If false, the design points which are not contained in the actual variation ranges are not imported. This option is not supported in the context of: - a DOE for a Six Sigma Analysis. - a Correlation based on a Response Surface.

Type bool (p. 1360)

Default Value False

Example

The following example shows how the user can import the design point with the ID equal to 1.

```
container = system1.GetContainer(ComponentName="Design of Experiment")
model = container.GetModel()
model.DOEType = "eDOETYPE_USER"
model.ImportDesignPointsFromDps(QueryString="?id=1", ExpandRanges=True)
```

ImportDesignPointsFromFile

Import Design Point values in a custom Design of Experiments model or in a custom Correlation model from a csv file.

Required Arguments

FileName The imported file name.

Type string (p. 1438)

Optional Arguments

ExpandRanges If true, the command expands the range of variation of the input parameters based on the imported design points. This is done by modifying automatically the upper and lower bounds of the input parameters. If false, the design points which are not contained in the actual variation ranges are not imported. This option is not supported in the context of: - a DOE for a Six Sigma Analysis. - a Correlation based on a Response Surface.

Type bool (p. 1360)

Default Value False

ShrinkRanges If true, the command shrinks the range of variation of the input parameters based on the imported design points. This is done by modifying automatically the upper and lower bounds of the input parameters. If false, the design points which are not contained in the actual variation ranges are not imported. This option is not supported in the context of: - a DOE for a Six Sigma Analysis. - a Correlation based on a Response Surface.

Type bool (p. 1360)

Default Value False

Example

The following example shows how the user can import design points from a valid csv file in a custom Correlation model.

```
container = system1.GetContainer(ComponentName="Correlation")
model = container.GetModel()
model.SamplingType = "Manual"
model.ImportDesignPointsFromFile(FilePath="designs.csv", ExpandRanges=True)
```

PreviewDesignPoints

Previews the Design Points of a model, without actually updating them, so that the user can adjust settings before launching a long update operation. This command applies to a Design of Experiments model, or the Refinement Points of a Kriging Response Surface, or a Parameters Correlation model. The command returns the table of the generated points.

Return The DataReference of the ParametricTable containing the generated data.

Type DataReference (p. 1371)

Example

The following example shows how to preview a DOE model.

```
container = system1.GetContainer(ComponentName="Design of Experiment")
model = container.GetModel()
DOEMatrix = model.PreviewDesignPoints()
```

SetCustomRomMeshFile

Choose a custom mesh file for visualizing ROM. This mesh file will be used when opening ROM Viewer, and be included in the exported ROMs. If not already done, a processing step will be done for the mesh file, and the file will be renamed. Any existing file with the same name will be overwritten.

Required Arguments

KeepOriginal If true, then the original file will be kept. If MeshPath is already in the DesignXplorer folder, this parameter is ignored.

Type bool (p. 1360)

MeshPath Path to the mesh file to set. If this file is outside of the DesignXplorer folder, it will be moved into it. Null or empty to clear the custom mesh, and use the one generated automatically from the Workbench project.

Type string (p. 1438)

SetCustomRomSnapshotFile

Manually set a ROM snapshot file generated from an external solve.

Required Arguments

KeepOriginal If true, then the original file will be copied. If false it will be moved. If SnapshotPath is already in the DesignXplorer folder then this parameter is ignored. If a snapshot with the same name is already in the DesignXplorer folder and ReuseSnapshot is set to "true" then this parameter is ignored.

Type bool (p. 1360)

ReuseSnapshot If true and there is already a snapshot with the same name in the DesignXplorer directory, then it will be reused and any copy or move operation will be skipped. If false and there is already a snapshot with the same name in the DesignXplorer directory, then a new name will be automatically assigned to the snapshot. If SnapshotPath is already in the DesignXplorer folder then this parameter is ignored. If there is no snapshot with the same name, this parameter is ignored.

Type bool (p. 1360)

RowIndex Zero-based row index of the cell.

Type int (p. 1394)

SnapshotPath Path to the snapshot file to set. If this file is outside of the DesignXplorer folder, it will be moved into it.

Type string (p. 1438)

SolverSystem System ID that produced the snapshot.

Type string (p. 1438)

Example

The following example shows how to set the ROM Snapshot file for the DesignPoints table in a Design of Experiments model.

```
system1 = GetSystem(Name="RBU")
designofExperiment1 = system1.GetContainer(ComponentName="Design of Experiment")
dOEModel1 = designofExperiment1.GetModel()
parametricTable1 = dOEModel1.GetParametricTable(Name="DesignPoints")
parametricTable1.SetOutputValuesEditable(Editable=True,RowIndices=[1])
Parameters.SetCustomRomSnapshotFile(
    ParametricTable=parametricTable1,
    RowIndex=0,
    SolverSystem="FLU",
    SnapshotPath="D:/ROM/snp_1.romsnp",
    KeepOriginal=True,
    ReuseSnapshot=False)
```

InputParameter

The data entity which describes an Input Parameter in DesignXplorer.

Properties

Attribute1

First editable attribute of the distribution for an uncertainty parameter. The nature of the attribute depends on the distribution type. For instance, the first attribute of a Normal distribution is the Mean value.

Type double (p. 1378)

Read Only No

Attribute2

Second editable attribute of the distribution for an uncertainty parameter. The nature of the attribute depends on the distribution type. For instance, the second attribute of a Normal distribution is the Standard Deviation value. Some distribution type do not have a second attribute.

Type double (p. 1378)

Read Only No

ConstantValue

Constant value of the Parameter when it is disabled.

Type Object (p. 1409)

Read Only No

CustomDefinitionList

Custom Definition List

Type List (p. 1400)<DataReference (p. 1371)>

Read Only No

DiscreteLevels

List of the discrete levels.

Type List (p. 1400)<DataReference (p. 1371)>

Read Only Yes

DisplayText

The general property that defines the user-visible name of an entity. This property is defined for all data entities but is used only in those entities that present a label in the user interface.

Type string (p. 1438)

Read Only No

DistributionLowerBound

Distribution lower bound of the variation range for an uncertainty Parameter.

Type double (p. 1378)

Read Only No

DistributionType

Distribution type for an uncertainty parameter.

Type DistributionType (p. 1377)

Read Only No

DistributionUpperBound

Distribution upper bound of the variation range for an uncertainty Parameter.

Type double (p. 1378)

Read Only No

Enabled

True if the Parameter is enabled for the current study.

Type bool (p. 1360)

Read Only No

Kurtosis

Kurtosis value of the distribution for an uncertainty parameter.

Type double (p. 1378)

Read Only Yes

LowerBound

Lower bound of the variation range for a Continuous Parameter.

Type double (p. 1378)

Read Only No

Mean

Mean value of the distribution for an uncertainty parameter.

Type double (p. 1378)

Read Only Yes

Nature

Nature of the Parameter.

Type ParameterNature (p. 1413)

Read Only No

NumberOfLevels

Number of levels if the parameter nature is Discrete, or the parameter nature is Continuous and the UseManufacturableValues property is set to True.

Type int (p. 1394)

Read Only Yes

Skewness

Skewness value of the distribution for an uncertainty parameter.

Type double (p. 1378)

Read Only Yes

StandardDeviation

Standard deviation value of the distribution for an uncertainty parameter.

Type double (p. 1378)

Read Only Yes

Type

Type of the Parameter, either a DesignVariable in a GDO context, or an UncertaintyVariable in a SixSigma Analysis context.

Type SimulationType (p. 1432)

Read Only Yes

Units

Units

Type string (p. 1438)

Read Only Yes

UpperBound

Upper bound of the variation range for a Continuous Parameter.

Type double (p. 1378)

Read Only No

UseManufacturableValues

True to restrict the variation of the parameter to defined Manufacturable Values.

Type bool (p. 1360)

Read Only No

Methods

AddDiscreteLevel

Adds a Discrete Level entity on a discrete input parameter. A discrete level can have an integer value (e.g. a number of holes, a number of turns, etc) or a string value (e.g. a material name or a geometry file name). The command has optional arguments to specify the Name of the level and its Index in the list of levels of the parameter. By default, the new level is added to the end of the list. The various discrete levels of an input parameter represent independent configurations of the project, processed in the order of their creation.

Return The created entity.

Type DataReference (p. 1371)

Required Arguments

Value The value of the discrete level. Value can be an integer or a string.

Type Object (p. 1409)

Optional Arguments

DisplayText DisplayText of the created entity. If not specified, a default name of the form "Level [#]" is used.

Type string (p. 1438)

Index The position of the new level in the list of discrete levels of the parameter. Index is zero-based. If it is not specified, the new level is appended to the list.

Type int (p. 1394)

Default Value -1

Example

The following example shows how to add new discrete levels on a discrete input parameter. The third level is inserted between the two others.

```
container = system1.GetContainer(ComponentName="Design of Experiment")
model = container.GetModel()
DiscreteInputParameter = model.GetParameter(Name="P2")
DiscreteInputParameter.AddDiscreteLevel(Value=3000, DisplayText="Three thousand turns")
DiscreteInputParameter.AddDiscreteLevel(Value=2000, DisplayText="Two thousand turns")
DiscreteInputParameter.AddDiscreteLevel(Value=5000, Index="1")
```

AddLevels

Adds a list of levels to a continuous input parameter. Each level is a quantity or a real number corresponding to a manufacturable value. The list of levels forms a restriction filter used when post-processing the input parameter. If levels are added outside of the variation range, the lower and upper bounds are adjusted accordingly.

Required Arguments

Levels List of added levels.

Type List (p. 1400)<string (p. 1438)>

Optional Arguments

Overwrite True in order to overwrite the existing levels, False by default.

Type bool (p. 1360)

Example

The following example shows how to overwrite the manufacturable values of an input parameter and how to define an additional value later.

```
container = system1.GetContainer(ComponentName="Design of Experiment")
model = container.GetModel()
InputParameter = model.GetParameter(Name="P1")
InputParameter.UseManufacturableValues = True
InputParameter.AddLevels(Levels=["0.3 [mm]", "0.5 [mm]", "1e-3 [m]"], Overwrite=True)
InputParameter.AddLevels(Levels="7 [mm]")
```

CreateOptimizationCriterion

Creates an OptimizationCriterion entity associated to a parameter.

Return The DataReference of the OptimizationCriterion.

Type DataReference (p. 1371)

Required Arguments

Parameter The parameter on which the criterion is created.

Type DataReference (p. 1371)

Example

The following example shows how to create an OptimizationCriterion entity.

```
container = system1.GetContainer(ComponentName="Optimization")
model = container.GetModel()
parameter1 = model.GetParameter(Name="P1")
optimizationCriterion = parameter1.CreateOptimizationCriterion()
```

DeleteDiscreteLevels

Deletes a list of levels from a discrete input parameter.

Required Arguments

DiscreteLevels List of the DiscreteLevel entities to delete.

Type List (p. 1400)<DataReference (p. 1371)>

Example

The following example shows how to add and then delete one or more levels from a discrete input parameter.

```
container = system1.GetContainer(ComponentName="Design of Experiment")
model = container.GetModel()
DiscreteInputParameter = model.GetParameter(Name="P1")
level1 = DiscreteInputParameter.AddDiscreteLevel(Value=3000, DisplayText="Three thousand turns")
level2 = DiscreteInputParameter.AddDiscreteLevel(Value=2000, DisplayText="Two thousand turns")
level3 = DiscreteInputParameter.AddDiscreteLevel(Value=5000, Index="1")
DiscreteInputParameter.DeleteDiscreteLevels(DiscreteLevels=[level1, level2])
```

DeleteLevels

Deletes a list of levels from a continuous input parameter.

Required Arguments

Indices Indices of the items to remove from the levels list

Type List (p. 1400)<int (p. 1394)>

Example

The following example shows how to add and then delete one or more levels from a continuous input parameter for which the UseManufacturableValues property is set to True.

```
container = system1.GetContainer(ComponentName="Design of Experiment")
model = container.GetModel()
InputParameter = model.GetParameter(Name="P1")
InputParameter.UseManufacturableValues = True
InputParameter.AddLevels(Levels=["0.3 [mmm]", "0.5 [mm]", "1e-3 [m]"], Overwrite=True)
InputParameter.DeleteLevels( Indices=[0, 1] )
```

ExportData

Export the data of a DesignXplorer entity to a csv file. The entity can be one of the DesignXplorer chart entities, a ParametricTable or an InputParameter entity.

Required Arguments

FileName The exported file name.

Type string (p. 1438)

Optional Arguments

AppendMode True to append to an existing csv file, False to overwrite it.

Type bool (p. 1360)

Default Value False

Example

The following example shows how the user can export the table of Design Points and then the Parameters Parallel chart of a Design of Experiments component.

```
container = system1.GetContainer(ComponentName="Design of Experiment")
model = container.GetModel()
parametricTable = model.GetParametricTable(Name="DesignPoints")
parametricTable.ExportData(FileName="doe.csv")
chart = model.GetChart(Name="DesignPointsParallel")
chart.ExportData(FileName="doe.csv", AppendMode=True)
```

GetCustomParameterDefinition

Get the DataReference of a CustomParameterProperties entity associated with an Input Parameter. An exception is thrown if the entity is not found.

Return The DataReference of the CustomParameterProperties.

Type DataReference (p. 1371)

Required Arguments

ExtensionName Extension's Name.

Type string (p. 1438)

Example

The following example shows how the user can get a CustomParameterProperties.

```
system1 = GetSystem(Name="DOP")
optimization1 = system1.GetContainer(ComponentName="Optimization")
optimizationModel1 = optimization1.GetModel()
inputParameter1 = optimizationModel1.GetParameter(Name="P1")
customParameterDefinition1 = inputParameter1.GetCustomParameterDefinition(ExtensionName="UncertaintyParame
```

```
customParameterDefinition1.Distribution = "Triangular"
```

GetDiscreteLevel

Get a discrete level by name from an input parameter. The parameter's full and ordered list of discrete levels is available as its "DiscreteLevels" property.

Return The DataReference of the discrete level entity.

Type DataReference (p. 1371)

Required Arguments

Name The name of the discrete level.

Type string (p. 1438)

Example

The following example shows how the user can retrieve a discrete level of a discrete input parameter by its name.

```
system1 = GetSystem(Name="RSO")
designofExperiment1 = system1.GetContainer(ComponentName="Design of Experiment")
dOEModel1 = designofExperiment1.GetModel()
DiscreteInputParameter1 = dOEModel1.GetParameter(Name="P1")
level = DiscreteInputParameter1.GetDiscreteLevel(Name="Level 1")
```

GetOptimizationCriterion

Get the DataReference of the OptimizationCriterion associated with a Parameter. An exception is thrown if the entity is not found.

Return The DataReference of the OptimizationCriterion.

Type DataReference (p. 1371)

Required Arguments

Parameter Parent Parameter.

Type DataReference (p. 1371)

Example

The following example shows how the user can get a OptimizationCriterion to change one of its properties.

```
system1 = GetSystem(Name="RSO")
optimization1 = system1.GetContainer(ComponentName="Optimization")
optimizationModel1 = optimization1.GetModel()
parameter3 = optimizationModel1.GetParameter(Name="P3")
optimizationCriterion = parameter3.GetOptimizationCriterion()
```

```
optimizationCriterion.ObjectiveType = "eGT_MinimumPossible"
```

GetParameterStatistics

Get the DataReference of the ParameterStatistics entity associated with a Parameter.

Return The DataReference of the ParameterStatistics entity.

Type DataReference (p. 1371)

Example

The following example shows how the user can get a ParameterStatistics entity and examine its Mean property.

```
system1 = GetSystem(Name="SSA")
sixSigmaAnalysis1 = system1.GetContainer(ComponentName="Six Sigma Analysis")
sixSigmaModel1 = sixSigmaAnalysis1.GetModel()
parameter4 = sixSigmaModel1.GetParameter(Name="P4")
parameterStatistics1 = parameter4.GetParameterStatistics()
mean = parameterStatistics1.Mean
```

LHSDMethod

Entity which performs and manages the DesignXplorer Sampling Method Component

Properties

DisplayText

The general property that defines the user-visible name of an entity. This property is defined for all data entities but is used only in those entities that present a label in the user interface.

Type string (p. 1438)

Read Only No

NumSamp

Number of Samples for User-Defined LHSD algorithm.

Type int (p. 1394)

Read Only No

RandomGeneratorSeed

Seed value for LHS and LHSD algorithm.

Type int (p. 1394)

Read Only No

SampType

Samples Type for LHS and LHSD algorithm.

Type NumSampType (p. 1408)

Read Only No

OSFDMethod

Entity which performs and manages the DesignXplorer Optimization Method Component

Properties

DisplayText

The general property that defines the user-visible name of an entity. This property is defined for all data entities but is used only in those entities that present a label in the user interface.

Type string (p. 1438)

Read Only No

MaxNumCycles

Maximum Number Of Cycles for OSFD algorithm.

Type int (p. 1394)

Read Only No

NumSamp

Number of Samples for User-Defined OSFD algorithm.

Type int (p. 1394)

Read Only No

OSFDType

Design Type for OSFD algorithm.

Type OptimalSpaceFillingType (p. 1410)

Read Only No

RandomGeneratorSeed

Seed value for LHS and OSFD algorithm.

Type int (p. 1394)

Read Only No

SampType

Samples Type for LHS and OSFD algorithm.

Type NumSampType (p. 1408)

Read Only No

OutputParameter

Output parameter entity for DesignXplorer.

Properties

DisplayText

The general property that defines the user-visible name of an entity. This property is defined for all data entities but is used only in those entities that present a label in the user interface.

Type string (p. 1438)

Read Only No

EnableAutoRefinement

Determines whether the Auto Refinement is applicable to the output parameter.

Type bool (p. 1360)

Read Only No

IgnoreForFiltering

Determines whether the parameter filtering is not applicable to the output parameter in the correlation context.

Type bool (p. 1360)

Read Only No

InheritFromModelSettings

Determines whether the Maximum Predicted Relative Error defined at the Model level is applicable to the output parameter.

Type bool (p. 1360)

Read Only No

LowerBound

Minimum value extracted from existing design points and/or sample sets.

Type double (p. 1378)

Read Only Yes

MaximumPredictedError

Maximum Predicted Error for an output parameter with the GARS algorithm. This is the maximum predicted error for the selected output parameter.

Type double (p. 1378)

Read Only Yes

MaxPredictedRelativeError

Maximum Relative Error targeted for an output parameter when refining with the Kriging algorithm. This is the maximum predicted relative error that is acceptable for the selected output parameter.

Type double (p. 1378)

Read Only No

PredictedRelativeError

Current value of the Predicted Relative Error when refining with the Kriging algorithm

Type double (p. 1378)

Read Only Yes

Scaling

Scaling

Type bool (p. 1360)

Read Only No

Tolerance

Maximum Error targeted for an output parameter when refining with the GARS algorithm. This is the maximum predicted error that is acceptable for the selected output parameter.

Type double (p. 1378)

Read Only No

TransformationType

Transformation Type

Type TransformationType (p. 1443)

Read Only No

Units

Units

Type string (p. 1438)

Read Only Yes

UpperBound

Maximum value extracted from existing design points and/or sample sets.

Type double (p. 1378)

Read Only Yes

Methods

CreateOptimizationCriterion

Creates an OptimizationCriterion entity associated to a parameter.

Return The DataReference of the OptimizationCriterion.

Type DataReference (p. 1371)

Required Arguments

Parameter The parameter on which the criterion is created.

Type DataReference (p. 1371)

Example

The following example shows how to create an OptimizationCriterion entity.

```
container = system1.GetContainer(ComponentName="Optimization")
model = container.GetModel()
parameter1 = model.GetParameter(Name="P1")
optimizationCriterion = parameter1.CreateOptimizationCriterion()
```

GetOptimizationCriterion

Get the DataReference of the OptimizationCriterion associated with a Parameter. An exception is thrown if the entity is not found.

Return The DataReference of the OptimizationCriterion.

Type DataReference (p. 1371)

Required Arguments

Parameter Parent Parameter.

Type DataReference (p. 1371)

Example

The following example shows how the user can get a OptimizationCriterion to change one of its properties.

```
system1 = GetSystem(Name="RSO")
optimization1 = system1.GetContainer(ComponentName="Optimization")
optimizationModel1 = optimization1.GetModel()
parameter3 = optimizationModel1.GetParameter(Name="P3")
optimizationCriterion = parameter3.GetOptimizationCriterion()
optimizationCriterion.ObjectiveType = "eGT_MinimumPossible"
```

GetParameterStatistics

Get the DataReference of the ParameterStatistics entity associated with a Parameter.

Return The DataReference of the ParameterStatistics entity.

Type DataReference (p. 1371)

Example

The following example shows how the user can get a ParameterStatistics entity and examine its Mean property.

```
system1 = GetSystem(Name="SSA")
sixSigmaAnalysis1 = system1.GetContainer(ComponentName="Six Sigma Analysis")
sixSigmaModel1 = sixSigmaAnalysis1.GetModel()
parameter4 = sixSigmaModel1.GetParameter(Name="P4")
parameterStatistics1 = parameter4.GetParameterStatistics()
mean = parameterStatistics1.Mean
```

SamplingMethod

Entity which wraps and manages the external Sampling Method for the Design of Experiments component

Properties

No Properties.

SPGDMethod

Entity which performs and manages the DesignXplorer Sampling Method Component

Properties

DisplayText

The general property that defines the user-visible name of an entity. This property is defined for all data entities but is used only in those entities that present a label in the user interface.

Type string (p. 1438)

Read Only No

DX GDO Response Surface

This container holds Response Surface data for a Goal Driven Optimization.

Methods

GetModel

Get the DataReference of the Model. An exception is thrown if the entity is not found.

Return The DataReference of the Model.

Type DataReference (p. 1371)

Example

The following example shows how the user can get a Model to change one of its properties.

```
system1 = GetSystem(Name="RSO")
designofExperiment1 = system1.GetContainer(ComponentName="Design of Experiment")
dOEModel1 = designofExperiment1.GetModel()
dOEModel1.DOEType = "eDOETYPE_OSFD"
```

Data Entities

GoodnessOfFit

Entity which manages the Goodness Of Fit Information of a Response Surface for an Output Parameter

Properties

ConfidenceLevel

The measure of the likelihood that a confidence interval contains the quantity or parameter being estimated

Type double (p. 1378)

Read Only No

DiscreteExpressions

A Dictionary holding the values of all discrete input. Note: Discrete parameter values are level values, not indices.

Type Dictionary (p. 1375)<DataReference (p. 1371), Object (p. 1409)>

Read Only No

DisplayText

The general property that defines the user-visible name of an entity. This property is defined for all data entities but is used only in those entities that present a label in the user interface.

Type string (p. 1438)

Read Only No

IsUpToDate

True if the entity is up-to-date.

Type bool (p. 1360)

Read Only No

Methods

CreateAdvancedReport

For the standard response surface only (Full second order Polynomials), creates an Advanced Goodness of Fit report for any direct output parameter.

Return A string which contains the generated Advanced Goodness of Fit report.

Type string (p. 1438)

Required Arguments

Parameter Parent Parameter of the report.

Type DataReference (p. 1371)

Optional Arguments

PlainTextFormat Plain text formatting instead of HTML (default).

Type bool (p. 1360)

Default Value False

Example

The following example shows how to create an Advanced Goodness of Fit report.

```
container = system1.GetContainer(ComponentName="Response Surface")
model = container.GetModel()
gof1 = model.GetGoodnessOfFit(Name="GoodnessOfFit")
outputParameter1 = model.GetParameter(Name="P4")
report1 = gof1.CreateAdvancedReport(Parameter=outputParameter1)
```

ExportData

Export the data of a DesignXplorer entity to a csv file. The entity can be one of the DesignXplorer chart entities, a ParametricTable or an InputParameter entity.

Required Arguments

FileName The exported file name.

Type string (p. 1438)

Optional Arguments

AppendMode True to append to an existing csv file, False to overwrite it.

Type bool (p. 1360)

Default Value False

Example

The following example shows how the user can export the table of Design Points and then the Parameters Parallel chart of a Design of Experiments component.

```
container = system1.GetContainer(ComponentName="Design of Experiment")
model = container.GetModel()
parametricTable = model.GetParametricTable(Name="DesignPoints")
parametricTable.ExportData(FileName="doe.csv")
chart = model.GetChart(Name="DesignPointsParallel")
chart.ExportData(FileName="doe.csv", AppendMode=True)
```

SetDiscreteParameter

Sets the Expression of a discrete input parameter in a GoodnessOfFit and updates the associated output parameter values. The chart entities depending on the GoodnessOfFit are updated as well.

Required Arguments

DiscreteParameter DataReference of the Discrete Input parameter.

Type DataReference (p. 1371)

Expression Assigned Expression (discrete value).

Type string (p. 1438)

Example

The following example shows how to set the expression for one discrete parameter in an existing GoodnessOfFit.

```
container = system1.GetContainer(ComponentName="Response Surface")
model = container.GetModel()
gof = model.GetGoodnessOfFit(Name="Goodness Of Fit")
inputParameter1 = model.GetParameter(Name="P1")
```

```
gof.SetDiscreteParameter(DiscreteParameter=inputParameter1, Expression="15")
```

Update

Updates the goodness of fit and the results which depend on it. If the goodness of fit is already uptodate, nothing is done unless the Force flag is set to True.

Optional Arguments

Force Set to true if the update operation of the goodness of fit point is required even if it's already uptodate.

Type bool (p. 1360)

Default Value False

Example

The following example shows how to update an existing GoodnessOfFit.

```
container = system1.GetContainer(ComponentName="Response Surface")
model = container.GetModel()
gof1 = model.GetGoodnessOfFit(Name="GOF 1")
gof1.Update()
```

InputParameter

The data entity which describes an Input Parameter in DesignXplorer.

Properties

Attribute1

First editable attribute of the distribution for an uncertainty parameter. The nature of the attribute depends on the distribution type. For instance, the first attribute of a Normal distribution is the Mean value.

Type double (p. 1378)

Read Only No

Attribute2

Second editable attribute of the distribution for an uncertainty parameter. The nature of the attribute depends on the distribution type. For instance, the second attribute of a Normal distribution is the Standard Deviation value. Some distribution type do not have a second attribute.

Type double (p. 1378)

Read Only No

ConstantValue

Constant value of the Parameter when it is disabled.

Type Object (p. 1409)

Read Only No

CustomDefinitionList

Custom Definition List

Type List (p. 1400)<DataReference (p. 1371)>

Read Only No

DiscreteLevels

List of the discrete levels.

Type List (p. 1400)<DataReference (p. 1371)>

Read Only Yes

DisplayText

The general property that defines the user-visible name of an entity. This property is defined for all data entities but is used only in those entities that present a label in the user interface.

Type string (p. 1438)

Read Only No

DistributionLowerBound

Distribution lower bound of the variation range for an uncertainty Parameter.

Type double (p. 1378)

Read Only No

DistributionType

Distribution type for an uncertainty parameter.

Type DistributionType (p. 1377)

Read Only No

DistributionUpperBound

Distribution upper bound of the variation range for an uncertainty Parameter.

Type double (p. 1378)

Read Only No

Enabled

True if the Parameter is enabled for the current study.

Type bool (p. 1360)

Read Only No

Kurtosis

Kurtosis value of the distribution for an uncertainty parameter.

Type double (p. 1378)

Read Only Yes

LowerBound

Lower bound of the variation range for a Continuous Parameter.

Type double (p. 1378)

Read Only No

Mean

Mean value of the distribution for an uncertainty parameter.

Type double (p. 1378)

Read Only Yes

Nature

Nature of the Parameter.

Type ParameterNature (p. 1413)

Read Only No

NumberOfLevels

Number of levels if the parameter nature is Discrete, or the parameter nature is Continuous and the UseManufacturableValues property is set to True.

Type int (p. 1394)

Read Only Yes

Skewness

Skewness value of the distribution for an uncertainty parameter.

Type double (p. 1378)

Read Only Yes

StandardDeviation

Standard deviation value of the distribution for an uncertainty parameter.

Type double (p. 1378)

Read Only Yes

Type

Type of the Parameter, either a DesignVariable in a GDO context, or an UncertaintyVariable in a SixSigma Analysis context.

Type SimulationType (p. 1432)

Read Only Yes

Units

Units

Type string (p. 1438)

Read Only Yes

UpperBound

Upper bound of the variation range for a Continuous Parameter.

Type double (p. 1378)

Read Only No

UseManufacturableValues

True to restrict the variation of the parameter to defined Manufacturable Values.

Type bool (p. 1360)

Read Only No

Methods

AddDiscreteLevel

Adds a Discrete Level entity on a discrete input parameter. A discrete level can have an integer value (e.g. a number of holes, a number of turns, etc) or a string value (e.g. a material name or a geometry file name). The command has optional arguments to specify the Name of the level and its Index in the list of levels of the parameter. By default, the new level is added to the end of the list. The various discrete levels of an input parameter represent independent configurations of the project, processed in the order of their creation.

Return The created entity.

Type DataReference (p. 1371)

Required Arguments

Value The value of the discrete level. Value can be an integer or a string.

Type Object (p. 1409)

Optional Arguments

DisplayText DisplayText of the created entity. If not specified, a default name of the form "Level [#]" is used.

Type string (p. 1438)

Index The position of the new level in the list of discrete levels of the parameter. Index is zero-based. If it is not specified, the new level is appended to the list.

Type int (p. 1394)

Default Value -1

Example

The following example shows how to add new discrete levels on a discrete input parameter. The third level is inserted between the two others.

```
container = system1.GetContainer(ComponentName="Design of Experiment")
model = container.GetModel()
DiscreteInputParameter = model.GetParameter(Name="P2")
DiscreteInputParameter.AddDiscreteLevel(Value=3000, DisplayText="Three thousand turns")
DiscreteInputParameter.AddDiscreteLevel(Value=2000, DisplayText="Two thousand turns")
DiscreteInputParameter.AddDiscreteLevel(Value=5000, Index="1")
```

AddLevels

Adds a list of levels to a continuous input parameter. Each level is a quantity or a real number corresponding to a manufacturable value. The list of levels forms a restriction filter used when post-processing the input parameter. If levels are added outside of the variation range, the lower and upper bounds are adjusted accordingly.

Required Arguments

Levels List of added levels.

Type List (p. 1400)<string (p. 1438)>

Optional Arguments

Overwrite True in order to overwrite the existing levels, False by default.

Type bool (p. 1360)

Example

The following example shows how to overwrite the manufacturable values of an input parameter and how to define an additional value later.

```
container = system1.GetContainer(ComponentName="Design of Experiment")
model = container.GetModel()
InputParameter = model.GetParameter(Name="P1")
InputParameter.UseManufacturableValues = True
InputParameter.AddLevels(Levels=["0.3 [mm]", "0.5 [mm]", "1e-3 [m]"], Overwrite=True)
InputParameter.AddLevels(Levels="7 [mm]")
```

CreateOptimizationCriterion

Creates an OptimizationCriterion entity associated to a parameter.

Return The DataReference of the OptimizationCriterion.

Type DataReference (p. 1371)

Required Arguments

Parameter The parameter on which the criterion is created.

Type DataReference (p. 1371)

Example

The following example shows how to create an OptimizationCriterion entity.

```
container = system1.GetContainer(ComponentName="Optimization")
model = container.GetModel()
parameter1 = model.GetParameter(Name="P1")
optimizationCriterion = parameter1.CreateOptimizationCriterion()
```

DeleteDiscreteLevels

Deletes a list of levels from a discrete input parameter.

Required Arguments

DiscreteLevels List of the DiscreteLevel entities to delete.

Type List (p. 1400)<DataReference (p. 1371)>

Example

The following example shows how to add and then delete one or more levels from a discrete input parameter.

```
container = system1.GetContainer(ComponentName="Design of Experiment")
model = container.GetModel()
DiscreteInputParameter = model.GetParameter(Name="P1")
```

```
level1 = DiscreteInputParameter.AddDiscreteLevel(Value=3000, DisplayText="Three thousand turns")
level2 = DiscreteInputParameter.AddDiscreteLevel(Value=2000, DisplayText="Two thousand turns")
level3 = DiscreteInputParameter.AddDiscreteLevel(Value=5000, Index="1")
DiscreteInputParameter.DeleteDiscreteLevels(DiscreteLevels=[level1, level2])
```

DeleteLevels

Deletes a list of levels from a continuous input parameter.

Required Arguments

Indices Indices of the items to remove from the levels list

Type List (p. 1400)<int (p. 1394)>

Example

The following example shows how to add and then delete one or more levels from a continuous input parameter for which the UseManufacturableValues property is set to True.

```
container = system1.GetContainer(ComponentName="Design of Experiment")
model = container.GetModel()
InputParameter = model.GetParameter(Name="P1")
InputParameter.UseManufacturableValues = True
InputParameter.AddLevels(Levels=["0.3 [mmm]", "0.5 [mm]", "1e-3 [m]"], Overwrite=True)
InputParameter.DeleteLevels( Indices=[0, 1] )
```

ExportData

Export the data of a DesignXplorer entity to a csv file. The entity can be one of the DesignXplorer chart entities, a ParametricTable or an InputParameter entity.

Required Arguments

FileName The exported file name.

Type string (p. 1438)

Optional Arguments

AppendMode True to append to an existing csv file, False to overwrite it.

Type bool (p. 1360)

Default Value False

Example

The following example shows how the user can export the table of Design Points and then the Parameters Parallel chart of a Design of Experiments component.

```
container = system1.GetContainer(ComponentName="Design of Experiment")
model = container.GetModel()
parametricTable = model.GetParametricTable(Name="DesignPoints")
```

```
parametricTable.ExportData(FileName="doe.csv")
chart = model.GetChart(Name="DesignPointsParallel")
chart.ExportData(FileName="doe.csv", AppendMode=True)
```

GetCustomParameterDefinition

Get the DataReference of a CustomParameterProperties entity associated with an Input Parameter. An exception is thrown if the entity is not found.

Return The DataReference of the CustomParameterProperties.

Type DataReference (p. 1371)

Required Arguments

ExtensionName Extension's Name.

Type string (p. 1438)

Example

The following example shows how the user can get a CustomParameterProperties.

```
system1 = GetSystem(Name="DOP")
optimization1 = system1.GetContainer(ComponentName="Optimization")
optimizationModel1 = optimization1.GetModel()
inputParameter1 = optimizationModel1.GetParameter(Name="P1")
customParameterDefinition1 = inputParameter1.GetCustomParameterDefinition(ExtensionName="UncertaintyParame
customParameterDefinition1.Distribution = "Triangular"
```

GetDiscreteLevel

Get a discrete level by name from an input parameter. The parameter's full and ordered list of discrete levels is available as its "DiscreteLevels" property.

Return The DataReference of the discrete level entity.

Type DataReference (p. 1371)

Required Arguments

Name The name of the discrete level.

Type string (p. 1438)

Example

The following example shows how the user can retrieve a discrete level of a discrete input parameter by its name.

```
system1 = GetSystem(Name="RSO")
designofExperiment1 = system1.GetContainer(ComponentName="Design of Experiment")
dOEModel1 = designofExperiment1.GetModel()
DiscreteInputParameter1 = dOEModel1.GetParameter(Name="P1")
```

```
level = DiscreteInputParameter1.GetDiscreteLevel(Name="Level 1")
```

GetOptimizationCriterion

Get the DataReference of the OptimizationCriterion associated with a Parameter. An exception is thrown if the entity is not found.

Return The DataReference of the OptimizationCriterion.

Type DataReference (p. 1371)

Required Arguments

Parameter Parent Parameter.

Type DataReference (p. 1371)

Example

The following example shows how the user can get a OptimizationCriterion to change one of its properties.

```
system1 = GetSystem(Name="RSO")
optimization1 = system1.GetContainer(ComponentName="Optimization")
optimizationModel1 = optimization1.GetModel()
parameter3 = optimizationModel1.GetParameter(Name="P3")
optimizationCriterion = parameter3.GetOptimizationCriterion()
optimizationCriterion.ObjectiveType = "eGT_MinimumPossible"
```

GetParameterStatistics

Get the DataReference of the ParameterStatistics entity associated with a Parameter.

Return The DataReference of the ParameterStatistics entity.

Type DataReference (p. 1371)

Example

The following example shows how the user can get a ParameterStatistics entity and examine its Mean property.

```
system1 = GetSystem(Name="SSA")
sixSigmaAnalysis1 = system1.GetContainer(ComponentName="Six Sigma Analysis")
sixSigmaModel1 = sixSigmaAnalysis1.GetModel()
parameter4 = sixSigmaModel1.GetParameter(Name="P4")
parameterStatistics1 = parameter4.GetParameterStatistics()
mean = parameterStatistics1.Mean
```

MinMaxSearch

The data entity which described the MinMax Search option of a Response Surface Component.

Properties

DisplayText

The general property that defines the user-visible name of an entity. This property is defined for all data entities but is used only in those entities that present a label in the user interface.

Type string (p. 1438)

Read Only No

Enabled

If True, performs a min-max search performed when the response surface is built.

Type bool (p. 1360)

Read Only No

IsUpToDate

True if the entity is up-to-date.

Type bool (p. 1360)

Read Only No

NumberInitialPoints

Number of initial samples for the min-max search algorithm.

Type int (p. 1394)

Read Only No

NumberStartPoints

Number of start points for the min-max search algorithm.

Type int (p. 1394)

Read Only No

OutputParameter

Output parameter entity for DesignXplorer.

Properties

DisplayText

The general property that defines the user-visible name of an entity. This property is defined for all data entities but is used only in those entities that present a label in the user interface.

Type string (p. 1438)

Read Only No

EnableAutoRefinement

Determines whether the Auto Refinement is applicable to the output parameter.

Type bool (p. 1360)

Read Only No

IgnoreForFiltering

Determines whether the parameter filtering is not applicable to the output parameter in the correlation context.

Type bool (p. 1360)

Read Only No

InheritFromModelSettings

Determines whether the Maximum Predicted Relative Error defined at the Model level is applicable to the output parameter.

Type bool (p. 1360)

Read Only No

LowerBound

Minimum value extracted from existing design points and/or sample sets.

Type double (p. 1378)

Read Only Yes

MaximumPredictedError

Maximum Predicted Error for an output parameter with the GARS algorithm. This is the maximum predicted error for the selected output parameter.

Type double (p. 1378)

Read Only Yes

MaxPredictedRelativeError

Maximum Relative Error targeted for an output parameter when refining with the Kriging algorithm. This is the maximum predicted relative error that is acceptable for the selected output parameter.

Type double (p. 1378)

Read Only No

PredictedRelativeError

Current value of the Predicted Relative Error when refining with the Kriging algorithm

Type double (p. 1378)

Read Only Yes

Scaling

Scaling

Type bool (p. 1360)

Read Only No

Tolerance

Maximum Error targeted for an output parameter when refining with the GARS algorithm. This is the maximum predicted error that is acceptable for the selected output parameter.

Type double (p. 1378)

Read Only No

TransformationType

Transformation Type

Type TransformationType (p. 1443)

Read Only No

Units

Units

Type string (p. 1438)

Read Only Yes

UpperBound

Maximum value extracted from existing design points and/or sample sets.

Type double (p. 1378)

Read Only Yes

Methods

CreateOptimizationCriterion

Creates an OptimizationCriterion entity associated to a parameter.

Return The DataReference of the OptimizationCriterion.

Type DataReference (p. 1371)

Required Arguments

Parameter The parameter on which the criterion is created.

Type DataReference (p. 1371)

Example

The following example shows how to create an OptimizationCriterion entity.

```
container = system1.GetContainer(ComponentName="Optimization")
model = container.GetModel()
parameter1 = model.GetParameter(Name="P1")
optimizationCriterion = parameter1.CreateOptimizationCriterion()
```

GetOptimizationCriterion

Get the DataReference of the OptimizationCriterion associated with a Parameter. An exception is thrown if the entity is not found.

Return The DataReference of the OptimizationCriterion.

Type DataReference (p. 1371)

Required Arguments

Parameter Parent Parameter.

Type DataReference (p. 1371)

Example

The following example shows how the user can get a OptimizationCriterion to change one of its properties.

```
system1 = GetSystem(Name="RSO")
optimization1 = system1.GetContainer(ComponentName="Optimization")
optimizationModel1 = optimization1.GetModel()
parameter3 = optimizationModel1.GetParameter(Name="P3")
optimizationCriterion = parameter3.GetOptimizationCriterion()
optimizationCriterion.ObjectiveType = "eGT_MinimumPossible"
```

GetParameterStatistics

Get the DataReference of the ParameterStatistics entity associated with a Parameter.

Return The DataReference of the ParameterStatistics entity.

Type DataReference (p. 1371)

Example

The following example shows how the user can get a ParameterStatistics entity and examine its Mean property.

```
system1 = GetSystem(Name="SSA")
sixSigmaAnalysis1 = system1.GetContainer(ComponentName="Six Sigma Analysis")
sixSigmaModel1 = sixSigmaAnalysis1.GetModel()
parameter4 = sixSigmaModel1.GetParameter(Name="P4")
parameterStatistics1 = parameter4.GetParameterStatistics()
mean = parameterStatistics1.Mean
```

ParametricTable

ParametricTable entity used to encapsulate most of the evaluation results in a convenient 2D matrix format.

Properties

DimCol

Number of columns.

Type int (p. 1394)

Read Only Yes

DimRow

Number of rows.

Type int (p. 1394)

Read Only Yes

DisplayText

The general property that defines the user-visible name of an entity. This property is defined for all data entities but is used only in those entities that present a label in the user interface.

Type string (p. 1438)

Read Only No

Methods

AddRow

Adds a row to the bottom of a ParametricTable entity.

Optional Arguments

RowValues New values for the row.

Type List (p. 1400)<string (p. 1438)>

Example

The following example shows how to make a DOE editable, to retrieve the table of design points and add a new row to it.

```
container = system1.GetContainer(ComponentName="Design of Experiment")
model = container.GetModel()
model.DOEType = "eDOETYPE_USER"
DOEMatrix = model.GetParametricTable(Name="DesignPoints")
DOEMatrix.AddRow()
```

DeleteRows

Delete rows from a ParametricTable entity.

Required Arguments

Indices Indices of the rows to delete.

Type List (p. 1400)<int (p. 1394)>

Example

The following example shows how to delete rows from the DOEmatrix in a custom DOE context.

```
container = system1.GetContainer(ComponentName="Design of Experiment")
model = container.GetModel()
DOEMatrix = model.GetParametricTable(Name="DesignPoints")
DOEMatrix.DeleteRows(Indices=[0,7,8,9])
```

ExportData

Export the data of a DesignXplorer entity to a csv file. The entity can be one of the DesignXplorer chart entities, a ParametricTable or an InputParameter entity.

Required Arguments

FileName The exported file name.

Type string (p. 1438)

Optional Arguments

AppendMode True to append to an existing csv file, False to overwrite it.

Type bool (p. 1360)

Default Value False

Example

The following example shows how the user can export the table of Design Points and then the Parameters Parallel chart of a Design of Experiments component.

```
container = system1.GetContainer(ComponentName="Design of Experiment")
model = container.GetModel()
parametricTable = model.GetParametricTable(Name="DesignPoints")
parametricTable.ExportData(FileName="doe.csv")
chart = model.GetChart(Name="DesignPointsParallel")
chart.ExportData(FileName="doe.csv", AppendMode=True)
```

ExportSnapshotsTableAsArchive

Export a table of snapshots as an archive.

Required Arguments

FilePath The exported file name.

Type string (p. 1438)

Example

The following example shows how the user can export a table of snapshots as an archive.

```
system1 = GetSystem(Name="RBU")
designofExperiment1 = system1.GetContainer(ComponentName="Design of Experiment")
dOEModel1 = designofExperiment1.GetModel()
parametricTable1 = dOEModel1.GetParametricTable(Name="DesignPoints")
parametricTable1.ExportSnapshotsTableAsArchive(FilePath="C:/Temp/doesnapshots.snpz")
```

GetCellValue

Get the Value of a ParametricTable's cell. An exception is thrown if the entity is not found.

Return The value of the cell.

Type string (p. 1438)

Required Arguments

ColumnIndex ColumnIndex (zero-based) of the cell.

Type int (p. 1394)

RowIndex RowIndex (zero-based) of the cell.

Type int (p. 1394)

Example

The following example shows how the user can get the value of the ParametricTable's cell [0,3].

```
system1 = GetSystem(Name="RSO")
designofExperiment1 = system1.GetContainer(ComponentName="Design of Experiment")
dOEModel1 = designofExperiment1.GetModel()
parametricTable = dOEModel1.GetParametricTable(Name="DesignPoints")
cellValue = parametricTable.GetCellValue(RowIndex=0,ColumnIndex=3)
```

GetRowRomSnapshotName

Get the Rom snapshot's name value of a row in a ParametricTable. An exception is thrown if the entity is not found.

Return The value of the Rom snapshot's name.

Type string (p. 1438)

Required Arguments

RowIndex RowIndex (zero-based) of the cell.

Type int (p. 1394)

Example

The following example shows how to get the Rom snapshot's name value of row [3] in the ParametricTable.

```
system1 = GetSystem(Name="RBU")
designofExperiment1 = system1.GetContainer(ComponentName="Design of Experiment")
dOEModel1 = designofExperiment1.GetModel()
parametricTable1 = dOEModel1.GetParametricTable(Name="DesignPoints")
snapshotName = parametricTable1.GetRowRomSnapshotName(RowIndex=3)
```

GetRowUpdateOrder

Get the Update Order value of a row in a ParametricTable. An exception is thrown if the entity is not found.

Return The value of the update order.

Type double (p. 1378)

Required Arguments

RowIndex RowIndex (zero-based) of the cell.

Type int (p. 1394)

Example

The following example shows how to get the Update Order value of row [3] in the ParametricTable.

```
system1 = GetSystem(Name="RSO")
designofExperiment1 = system1.GetContainer(ComponentName="Design of Experiment")
```

```
dOEModel1 = designofExperiment1.GetModel()
parametricTable = dOEModel1.GetParametricTable(Name="DesignPoints")
updateOrderValue = parametricTable.GetRowUpdateOrder(RowIndex=3)
```

GetRowValues

Get the Values of a ParametricTable's row. An exception is thrown if the entity is not found.

Return List of the values of the specified row.

Type List (p. 1400)<string (p. 1438)>

Required Arguments

RowIndex RowIndex (zero-based) of the row to retrieve.

Type int (p. 1394)

Example

The following example shows how the user can get the values of the 4th ParametricTable's row.

```
system1 = GetSystem(Name="RSO")
designofExperiment1 = system1.GetContainer(ComponentName="Design of Experiment")
dOEModel1 = designofExperiment1.GetModel()
parametricTable = dOEModel1.GetParametricTable(Name="DesignPoints")
rowValues = parametricTable.GetRowValues(RowIndex=3)
```

OptimizeUpdateOrder

Optimizes the Update Order of Design Points to minimize the number of modifications between two consecutive Design Points. This command applies to the "DesignPoints" ParametricTable of a Design of Experiments model or of a Response Surface in a manual refinement context. It also applies to the "VerificationPoints" ParametricTable of a Response Surface model.

Example

The following example shows how to optimize the update order of the DesignPoints table in a Design of Experiments model.

```
container = system1.GetContainer(ComponentName="Design of Experiment")
model = container.GetModel()
model.DOEType = "eDOETYPE_USER"
DOEMatrix = model.GetParametricTable(Name="DesignPoints")
DOEMatrix.OptimizeUpdateOrder()
```

SetCellValue

Sets the value of a ParametricTable cell. If the table is read-only, the command has no effect.

Required Arguments

ColumnIndex Zero-based column index of the cell.

Type int (p. 1394)

RowIndex Zero-based row index of the cell.

Type int (p. 1394)

Value New value of the cell.

Type string (p. 1438)

Example

The following example shows how to set the value of the DesignPoints table in a custom DOE context.

```
container = system1.GetContainer(ComponentName="Design of Experiment")
model = container.GetModel()
model.DOEType = "eDOETYPE_USER"
DOEMatrix = model.GetParametricTable(Name="DesignPoints")
DOEMatrix.AddRow()
DOEMatrix.SetCellValue(RowIndex=0, ColumnIndex=0, Value="12.5 [mm]")
```

SetOutputValuesEditable

Sets the values of the output parameters as Editable (Editable=True) or Calculated (Editable=False) for the complete table, or a set of rows specified by the RowIndices argument. This command is applicable to the "DesignPoints" ParametricTable of a Design of Experiments model in a custom context (DOEType is "eDOETYPE_USER" or "eDOETYPE_CUSTOM_OSFD"), of a Correlation model is a custom context (SamplingType is "eCustom"), or of a Response Surface model in a manual refinement context. It also applies to the "VerificationPoints" ParametricTable of a Response Surface model.

Required Arguments

Editable True to define output values as Editable, or False to mark output values as calculated.

Type bool (p. 1360)

Optional Arguments

RowIndices Optional list of row zero-based indices. If this argument is not specified, the command applies to the complete ParametricTable.

Type List (p. 1400)<int (p. 1394)>

Example

The following example shows how to switch a DOE model to the custom mode and then set the output values as editable for the first three design points of the DOE matrix.

```
container = system1.GetContainer(ComponentName="Design of Experiment")
```

```
model = container.GetModel()
model.DOEType = "eDOETYPE_USER"
DOEMatrix = model.GetParametricTable(Name="DesignPoints")
DOEMatrix.SetOutputValuesEditable(Editable=True, RowIndices=[0,1,2])
```

SetRowRomSnapshotName

Sets the Rom Snapshot's name value for a row in a ParametricTable. If the table doesn't support Rom functionality, the command has no effect.

Required Arguments

RowIndex Zero-based row index of the cell.

Type int (p. 1394)

SnapshotName New value of the Rom snapshot's name.

Type string (p. 1438)

Example

The following example shows how to set the Rom Snapshot's name value of the DesignPoints table in a Design of Experiments model.

```
system1 = GetSystem(Name="RBU")
designofExperiment1 = system1.GetContainer(ComponentName="Design of Experiment")
dOEModel1 = designofExperiment1.GetModel()
parametricTable1 = dOEModel1.GetParametricTable(Name="DesignPoints")
parametricTable1.SetOutputValuesEditable(Editable=True,RowIndices=[1])
parametricTable1.SetRowRomSnapshotName(RowIndex=1,SnapshotName="ff66dfaf-c34b-4dff-938e-bf497a377b9f.romsn
```

SetRowUpdateOrder

Sets the Update Order value for a row in a ParametricTable. If the table doesn't support Update Order functionality, the command has no effect.

Required Arguments

RowIndex Zero-based row index of the cell.

Type int (p. 1394)

UpdateOrder New value of the update order.

Type double (p. 1378)

Example

The following example shows how to set the Update Order value of the DesignPoints table in a Design of Experiments model.

```
container = system1.GetContainer(ComponentName="Design of Experiment")
model = container.GetModel()
```

```
DOEMatrix = model.GetParametricTable(Name="DesignPoints")
DOEMatrix.SetRowUpdateOrder(RowIndex=0, Value="2.0")
```

SetRowValues

Sets the values of a ParametricTable's row. If the table is read-only, the command has no effect.

Required Arguments

RowIndex Zero-based row index of the cell.

Type int (p. 1394)

RowValues New values for the row.

Type List (p. 1400)<string (p. 1438)>

Example

The following example shows how to set the values of the DesignPoints table in a custom DOE context.

```
container = system1.GetContainer(ComponentName="Design of Experiment")
model = container.GetModel()
model.DOEType = "eDOETYPE_USER"
DOEMatrix = model.GetParametricTable(Name="DesignPoints")
DOEMatrix.AddRow()
DOEMatrix.SetRowValues(RowIndex=0, RowValues=[ "12.5 [mm]", "1" ] )
```

SetUpdateOrderByRow

Sets a value for the UpdateOrder property of all Design Points using sorting settings. This command applies to the "DesignPoints" ParametricTable of a Design of Experiments model or of a Response Surface in a manual refinement context. It also applies to the "VerificationPoints" ParametricTable of a Response Surface model.

Optional Arguments

SortBy Definition of the sort.

Type List (p. 1400)<string (p. 1438)>

Example

The following example shows how to set the update order of the DesignPoints table in a Design of Experiments model. The Design Points are sorted first by their values for the parameter P3 (in ascending order), and then by their values for the parameter P1 (in descending order).

```
container = system1.GetContainer(ComponentName="Design of Experiment")
model = container.GetModel()
DOEMatrix = model.GetParametricTable(Name="DesignPoints")
DOEMatrix.SetUpdateOrderByRow(SortBy=["P3:+", "P1:-"])
```

UpdateRows

Updates the design points held in rows from a ParametricTable entity and the results which depend on it.

Required Arguments

Indices Indices of the rows to update.

Type List (p. 1400)<int (p. 1394)>

Example

The following example shows how to update the design points from the VerificationPoints Table.

```
responseSurface1 = system1.GetContainer(ComponentName="Response Surface")
responseSurfaceModel1 = responseSurface1.GetModel()
parametricTable1 = responseSurfaceModel1.GetParametricTable(Name="VerificationPoints")
parametricTable1.UpdateRows(Indices=[0,1])
```

ResponseSurfaceModel

Entity which performs and manages the DesignXplorer Response Surface Component

Properties

Converged

Convergence state

Type bool (p. 1360)

Read Only Yes

ConvergenceState

Convergence state for GARS Refinement

Type GARefinementStatusType (p. 1387)

Read Only Yes

CrowdingDistSeparationPercentage

Crowding Distance Separation Percentage when refining with the Kriging algorithm

Type double (p. 1378)

Read Only No

CrowdingDistSeparationPercentageGA

Crowding Distance Separation Percentage when refining with the Genetic Aggregation algorithm

Type double (p. 1378)

Read Only No

CurrentRelativeError

Current value of the Relative Error

Type double (p. 1378)

Read Only Yes

DisplayLevel

Level of Display of Log File of Genetic Aggregation Response Surface

Type LevelDisplayType (p. 1399)

Read Only No

DisplayText

The general property that defines the user-visible name of an entity. This property is defined for all data entities but is used only in those entities that present a label in the user interface.

Type string (p. 1438)

Read Only No

ExportDesignPoints

If True and PreserveDesignPoints is True as well, export project for each preserved Design Point.

Type bool (p. 1360)

Read Only No

FittingType

Response Surface Type

Type FittingType (p. 1385)

Read Only No

GenerateVerificationPoints

If True, generate verification points.

Type bool (p. 1360)

Read Only No

InputParametersScaling

Input parameters scaling.

Type bool (p. 1360)

Read Only No

InputParametersTransformationType

Input parameters transformation type.

Type TransformationType (p. 1443)

Read Only No

KernelVariationType

Kernel Variation Type for the Kriging algorithm

Type KernelVariationType (p. 1398)

Read Only No

LogFile

Log File of Genetic Aggregation Response Surface

Type string (p. 1438)

Read Only Yes

MaximumDepth

Maximum Depth limit when refining with the Sparse Grid response surface

Type int (p. 1394)

Read Only No

MaximumNumberOfGenerations

Maximum Number of Generations for Genetic Aggregation Response Surface

Type int (p. 1394)

Read Only No

MaximumNumberRefinementPointsPerIteration

Available when GARSMultiPointsRefinement feature is enabled Maximum number of points per refinement iteration with the GARS algorithm.

Type int (p. 1394)

Read Only No

MaximumRelativeErrorSparseGrid

Maximum Relative Error targeted when refining with the Sparse Grid response surface

Type double (p. 1378)

Read Only No

MaxNumberRefinementPointsGA

Maximum Number of Refinement Points that can be generated for refinement with the Genetic Aggregation algorithm.

Type int (p. 1394)

Read Only No

MaxNumberRefinementPointsKriging

Maximum Number of Refinement Points that can be generated for refinement with the Kriging algorithm.

Type int (p. 1394)

Read Only No

MaxNumberRefinementPointsSparseGrid

Maximum Number of Refinement Points that can be generated for refinement with the Sparse Grid response surface

Type int (p. 1394)

Read Only No

MaxPredictedRelativeError

Maximum Relative Error targeted for all input parameters when refining with the Kriging algorithm. This is the maximum predicted relative error that is acceptable for all parameters.

Type double (p. 1378)

Read Only No

NumberOfCells

Number of Cells for the Neural Network algorithm

Type int (p. 1394)

Read Only No

NumberOfRetries

Indicates the number of times DX will try to update the failed design points.

Type int (p. 1394)

Read Only No

NumberRefinementPoints

Number of existing Refinement Points

Type int (p. 1394)

Read Only Yes

NumberVerificationPoints

Number of verification points to generate.

Type int (p. 1394)

Read Only No

OutputVarCombinations

Output Variable Combinations when refining with the Kriging algorithm. Controls how output variables are considered in terms of Predicated Relative Error and determines the number of refinement points generated per iteration.

Type AdaptKrigOutType (p. 1353)

Read Only No

PredictedRelativeError

Current value of the Predicted Relative Error when refining with the Kriging algorithm

Type double (p. 1378)

Read Only Yes

PreserveDesignPoints

If True, preserve the Design Points at the project level after the component Update.

Type bool (p. 1360)

Read Only No

RandomGeneratorSeed

ProgramControlled Seed

Type int (p. 1394)

Read Only No

RefinementCombinationType

Output Variable Combinations when refining with the GARS algorithm. Controls how output variables are considered in terms of Predicted Error and determines the number of refinement points generated per iteration.

Type GARefinementOutType (p. 1387)

Read Only No

RefinementType

Refinement Type

Type ResponseSurfaceRefinementType (p. 1425)

Read Only No

RetainDesignPoints

If True and PreserveDesignPoints is True as well, retain data for each preserved Design Point.

Type bool (p. 1360)

Read Only No

RetryDelay

Indicates how much time will elapse between tries. This option is only applicable when NumberOfRetries is greater than 0, otherwise it has no effect.

Type Quantity (p. 1422)

Read Only No

SignificanceLevel

Threshold for the selection process for significant polynomial terms.

Type double (p. 1378)

Read Only No

Methods

CreateChart

Creates a Chart entity. The chart must be updated once after its creation by invoking its Update method. Then changes to its properties will update the chart automatically.

Return The DataReference of the Chart.

Type DataReference (p. 1371)

Required Arguments

ChartType Type of chart to be created. The possible values depend on the type of Model. For instance, a ResponseSurface model accepts Spider, LocalSensitivity and Response chart while a CorrelationModel accepts CorrelationMatrix, DeterminationMatrix and CorrelationScatter charts.

Type ChartType (p. 1363)

Optional Arguments

DisplayText Displayed name of the chart. If not specified, a default name is applied, depending on the value of ChartType.

Type string (p. 1438)

Example

The following example shows how to create a chart.

```
container = system1.GetContainer(ComponentName="Optimization")
model = container.GetModel()
TradeoffChart = model.CreateChart(ChartType="eChartTradeoff")
TradeoffChart.Update()
```

CreateGoodnessOfFit

Creates a GoodnessOfFit. The ParameterValues dictionary can be used to specify a value for some or all of the discrete input parameters.

Return The created entity.

Type DataReference (p. 1371)

Optional Arguments

DisplayText DisplayText of the created entity. If not specified, a default name of the form "Goodness Of Fit [#]" is used.

Type string (p. 1438)

ForceUpdate Update the created entity if true.

Type bool (p. 1360)

Default Value True

ParameterValues The values for each discrete input parameter. If not specified, each parameter is initialized to the current level.

Type Dictionary (p. 1375)<DataReference (p. 1371), string (p. 1438)>

Example

The following example shows how to create a GoodnessOfFit from an existing ResponseSurface model. The code retrieves the parameters P1 and P2 and then creates the response point by assigning a value to each of these parameters.

```
container = system1.GetContainer(ComponentName="Response Surface")
model = container.GetModel()
inputParameter1 = model.GetParameter(Name="P1")
inputParameter2 = model.GetParameter(Name="P2")
goodnessOfFit = model.CreateGoodnessOfFit( DisplayText="GOF1",
                  ParameterValues={inputParameter1: "2", inputParameter2: "15"})
```

CreateResponsePoint

Creates a ResponsePoint. The ParameterValues dictionary can be used to specify a value or a quantity for some or all of the input parameters. The output parameter values cannot be specified. They are evaluated automatically from the ResponseSurface model once it is updated. Several types of charts can only be created as children of a ResponsePoint. These charts depend on the ResponsePoint and use the same parameter values.

Return The created entity.

Type DataReference (p. 1371)

Optional Arguments

DisplayText DisplayText of the created entity. If not specified, a default name of the form "Response Point [#]" is used.

Type string (p. 1438)

ParameterValues The values for each input parameter. If not specified, each parameter is initialized to the middle of its variation range.

Type IDictionary (p. 1375)<DataReference (p. 1371), string (p. 1438)>

Example

The following example shows how to create a ResponsePoint from an existing ResponseSurface model. The code retrieves the parameters P1 and P2 and then creates the response point by assigning a value to each of these parameters.

```
container = system1.GetContainer(ComponentName="Response Surface")
model = container.GetModel()
inputParameter1 = model.GetParameter(Name="P1")
inputParameter2 = model.GetParameter(Name="P2")
responsePoint = model.CreateResponsePoint( DisplayText="Improved Design",
                  ParameterValues={inputParameter1: "2.01", inputParameter2: "15.5"})
```

DeleteCharts

Deletes a list of Chart entities.

Required Arguments

Charts List of Chart entities to delete.

Type List (p. 1400)<DataReference (p. 1371)>

Example

The following example shows how to delete existing charts.

```
container = system1.GetContainer(ComponentName="Optimization")
model = container.GetModel()
chart1 = model.GetChart(Name="TradeoffChart 1")
chart2 = model.GetChart(Name="SamplesChart 1")
model.DeleteCharts(Charts=[chart1, chart2])
```

DeleteGoodnessOfFit

Deletes a list of GoodnessOfFit entities and all the depending Chart entities.

Required Arguments

GoodnessOfFit DataReferences of the entities to delete

Type List (p. 1400)<DataReference (p. 1371)>

Example

The following example shows how to delete existing GoodnessOfFit.

```
container = system1.GetContainer(ComponentName="Response Surface")
model = container.GetModel()
gof1 = model.GetGoodnessOfFit(Name="GOF 1")
gof5 = model.GetGoodnessOfFit(Name="GOF 5")
gof6 = model.GetGoodnessOfFit(Name="GOF 6")
model.DeleteGoodnessOfFit(GoodnessOfFit=[gof1, gof5, gof6])
```

DeleteResponsePoints

Deletes a list of ResponsePoint entities and all the depending Chart entities.

Required Arguments

ResponsePoints DataReferences of the entities to delete

Type List (p. 1400)<DataReference (p. 1371)>

Example

The following example shows how to delete existing ResponsePoints.

```
container = system1.GetContainer(ComponentName="Response Surface")
model = container.GetModel()
rp1 = model.GetResponsePoint(Name="Response Point 1")
```

```
rp5 = model.GetResponsePoint(Name="Response Point 5")
rp6 = model.GetResponsePoint(Name="Response Point 6")
model.DeleteResponsePoints(ResponsePoints=[rp1, rp5, rp6])
```

DuplicateChart

Duplicates a Chart entity. The chart must be updated once after its creation by invoking its Update method. Then changes to its properties will update the chart automatically.

Return The DataReference of the Chart.

Type DataReference (p. 1371)

Required Arguments

Chart The source chart to duplicate.

Type DataReference (p. 1371)

Optional Arguments

DisplayText Displayed name of the chart. If not specified, a default name is applied, depending on the value of ChartType.

Type string (p. 1438)

TargetModel The model on which the duplicated chart is created. If this parameter is not set, the model of the source chart is used.

Type DataReference (p. 1371)

TargetResponsePoint Parent TargetResponsePoint of the chart. If this parameter is not set, the response point of the source chart is used if applicable.

Type DataReference (p. 1371)

Example

The following example shows how to duplicate a chart.

```
designofExperiment1 = system1.GetContainer(ComponentName="Design of Experiment")
dOEModel1 = designofExperiment1.GetModel()
designPointsCurvesChart1 = dOEModel1.GetChart(Name="DesignPointsCurves")
chart1 = dOEModel1.DuplicateChart(Chart=designPointsCurvesChart1)
chart1.Update()
```

DuplicateGoodnessOfFit

Duplicates a GoodnessOfFit.

Return The created entity.

Type DataReference (p. 1371)

Required Arguments

GoodnessOfFit DataReference of the GoodnessOfFit.

Type DataReference (p. 1371)

Optional Arguments

DisplayText DisplayText of the created entity. If not specified, a default name of the form "Goodness Of Fit [#]" is used.

Type string (p. 1438)

ForceUpdate Update the created entity if true.

Type bool (p. 1360)

Default Value True

TargetModel The ResponseSurface model on which the goodness of fit is created. If this parameter is not set, the model of the source goodness of fit is used.

Type DataReference (p. 1371)

Example

The following example shows how to duplicate a GoodnessOfFit from an existing ResponseSurface model. The code retrieves the parameters P1 and P2 and then creates the response point by assigning a value to each of these parameters.

```
container = system1.GetContainer(ComponentName="Response Surface")
responseSurfaceModel1 = container.GetModel()
gof = responseSurfaceModel1.GetGoodnessOfFit(Name="GoodnessOfFit")
goodnessOfFit2 = model.DuplicateGoodnessOfFit(GoodnessOfFit=gof, DisplayText="GOF2")
goodnessOfFit2.Update()
```

DuplicateResponsePoint

Duplicates a ReponsePoint entity. The response point must be updated once after its creation by invoking its Update method. Then changes to its properties will update the response point automatically.

Return The created entity.

Type DataReference (p. 1371)

Required Arguments

ResponsePoint The source response point to duplicate.

Type DataReference (p. 1371)

Optional Arguments

DisplayText DisplayText of the created entity. If not specified, a default name of the form "Response Point [#]" is used.

Type string (p. 1438)

DuplicateCharts True if the charts of the response point have to be duplicated.

Type bool (p. 1360)

Default Value False

TargetModel The model on which the duplicated response point is created. If this parameter is not set, the model of the source response point is used.

Type DataReference (p. 1371)

Example

The following example shows how to duplicate a response point and all its charts.

```
responseSurface1 = system1.GetContainer(ComponentName="Response Surface")
responseSurfaceModel1 = responseSurface1.GetModel()
responsePoint1 = responseSurfaceModel1.GetResponsePoint(Name="ResponsePoint")
responsePoint2 = responseSurfaceModel1.DuplicateResponsePoint(ResponsePoint=responsePoint1,DuplicateCharts
responsePoint2.Update()
```

GetChart

Query to return the chart reference for a given model and chart name.

Return The DataReference of the chart.

Type DataReference (p. 1371)

Required Arguments

Name Name of the chart.

Type string (p. 1438)

GetDesignPointReportSetting

Get the DataReference of a DesignPointReportSetting entity associated with a Model. An exception is thrown if the entity is not found.

Return The DataReference of the DesignPointReportSetting.

Type DataReference (p. 1371)

Example

The following example shows how the user can get a DesignPointReportSetting to change one of its properties.

```
system1 = GetSystem(Name="GDO")
designofExperiment1 = system1.GetContainer(ComponentName="Design of Experiment")
dOEModel1 = designofExperiment1.GetModel()
designPointReportSettingEntity1 = dOEModel1.GetDesignPointReportSetting()
designPointReportSettingEntity1.DesignPointReportImage = "FFF-Results:Figure001.png"
```

GetGoodnessOfFit

Get the DataReference of a GoodnessOfFit entity associated with a Model. An exception is thrown if the entity is not found.

Return The DataReference of the GoodnessOfFit.

Type DataReference (p. 1371)

Required Arguments

Name Name of the GoodnessOfFit.

Type string (p. 1438)

Example

The following example shows how the user can get a GoodnessOfFit.

```
system1 = GetSystem(Name="RSO")
responseSurface1 = system1.GetContainer(ComponentName="Response Surface")
responseSurfaceModel1 = responseSurface1.GetModel()
gof = responseSurfaceModel1.GetGoodnessOfFit(Name="GoodnessOfFit")
```

GetGoodnessOfFits

Get the DataReferences of the GoodnessOfFit entities associated with a Model.

Return The DataReferences of the GoodnessOfFits.

Type DataReferenceSet (p. 1371)

Example

The following example shows how to retrieve the GoodnessOfFit entities of a response surface model.

```
system1 = GetSystem(Name="RSO")
responseSurface1 = system1.GetContainer(ComponentName="Response Surface")
responseSurfaceModel1 = responseSurface1.GetModel()
gof = responseSurfaceModel1.GetGoodnessOfFits()
```

GetMinMaxSearch

Get the DataReference of the MinMaxSearch entity associated with a Response Surface model. An exception is thrown if the entity is not found.

Return The DataReference of the MinMaxSearch entity.

Type DataReference (p. 1371)

Example

The following example shows how the user can get the MinMaxSearch of a ResponseSurfaceModel to change one of its properties.

```
responseSurface1 = system1.GetContainer(ComponentName="Response Surface")
responseSurfaceModel1 = responseSurface1.GetModel()
minMaxSearch1 = responseSurfaceModel1.GetMinMaxSearch()
minMaxSearch1.NumberInitialPoints = 200
```

GetParameter

Get the DataReference of a Parameter. An exception is thrown if the entity is not found.

Return The DataReference of the Parameter.

Type DataReference (p. 1371)

Required Arguments

Name Name of the Parameter.

Type string (p. 1438)

Example

The following example shows how the user can get a parameter of a model to change one of its properties.

```
system1 = GetSystem(Name="RSO")
designofExperiment1 = system1.GetContainer(ComponentName="Design of Experiment")
dOEModel1 = designofExperiment1.GetModel()
inputParameter1 = dOEModel1.GetParameter(Name="P1")
inputParameter1.LowerBound = 1
```

GetParameters

Get the DataReferences of the InputParameter and OutputParameter of the model. If the optional argument InputParameters is not specified, the query returns all parameters. If it is specified and True, the query returns only input parameters. If it is False, the query returns only output parameters.

Return The DataReferences of the Parameters

Type DataReferenceSet (p. 1371)

Optional Arguments

InputParameters If True, the query returns only input parameters. If False, the query returns only output parameters. If the argument is ommitted, the query returns all parameters.

Type bool (p. 1360)

Example

The following example shows how the user can get the parameters of a model.

```
system1 = GetSystem(Name="RSO")
designofExperiment1 = system1.GetContainer(ComponentName="Design of Experiment")
dOEModel1 = designofExperiment1.GetModel()
parameters = dOEModel1.GetParameters()
```

GetParametricTable

Get the DataReference of ParametricTable. An exception is thrown if the entity is not found. Names of the tables generated internally are: "DesignPoints", "CorrelationMatrix", "CorrelationScatter", "MinDesignPoints", "MaxDesignPoints", "ResponsePoints", "DeterminationMatrix".

Return The DataReference of the ParametricTable

Type DataReference (p. 1371)

Required Arguments

Name Name of the ParametricTable

Type string (p. 1438)

Example

The following example shows how the user can get a ParametricTable to add a new row and set values.

```
system1 = GetSystem(Name="RSO")
designofExperiment1 = system1.GetContainer(ComponentName="Design of Experiment")
dOEModel1 = designofExperiment1.GetModel()
parametricTable = dOEModel1.GetParametricTable(Name="DesignPoints")
parametricTable.AddRow()
parametricTable.SetCellValue(RowIndex=9,ColumnIndex=0,Value="2.1")
```

GetResponsePoint

Get the DataReference of a ResponsePoint. An exception is thrown if the entity is not found.

Return The DataReference of the ResponsePoint.

Type DataReference (p. 1371)

Required Arguments

Name Name

Type string (p. 1438)

Example

The following example shows how to retrieve an existing ResponsePoint.

```
container = system1.GetContainer(ComponentName="Response Surface")
model = container.GetModel()
rp = model.GetResponsePoint(Name="Response Point 1")
```

GetResponsePoints

Get the DataReferences of the ResponsePoints.

Return The DataReferences of the ResponsePoints

Type DataReferenceSet (p. 1371)

Example

The following example shows how to retrieve the response points of a response surface model.

```
system1 = GetSystem(Name="RSO")
container = system1.GetContainer(ComponentName="Response Surface")
model = container.GetModel()
responsePoints = model.GetResponsePoints()
```

ImportRefinementPointsFromDps

Import Refinement Point values in the manual refinement table of a Response Surface model with a query to the DPS.

Required Arguments

QueryString The query.

Type string (p. 1438)

Example

The following example shows how the user can import the design point with the ID equal to 1.

```
container = system1.GetContainer(ComponentName="Response Surface")
model = container.GetModel()
model.ImportRefinementPointsFromDps(QueryString="?id=1")
```

ImportRefinementPointsFromFile

Import Refinement Point values in the manual refinement table of a Response Surface model from a csv file.

Required Arguments

FileName The imported file name.

Type string (p. 1438)

Example

The following example shows how the user can import refinement points from a valid csv file.

```
container = system1.GetContainer(ComponentName="Response Surface")
model = container.GetModel()
model.ImportRefinementPointsFromFile(FilePath="designs.csv")
```

ImportVerificationPointsFromDps

Import Verification Point values in the verification table of a Response Surface model with a query to the DPS.

Required Arguments

QueryString The query.

Type string (p. 1438)

Example

The following example shows how the user can import verification points with a query to the DPS.

```
container = system1.GetContainer(ComponentName="Response Surface")
model = container.GetModel()
model.ImportVerificationPointsFromDps(QueryString="?id=1")
```

ImportVerificationPointsFromFile

Import Verification Point values in the verification table of a Response Surface model from a csv file.

Required Arguments

FileName The imported file name.

Type string (p. 1438)

Example

The following example shows how the user can import verification points from a valid csv file.

```
container = system1.GetContainer(ComponentName="Response Surface")
model = container.GetModel()
model.ImportVerificationPointsFromFile(FilePath="designs.csv")
```

PreviewDesignPoints

Previews the Design Points of a model, without actually updating them, so that the user can adjust settings before launching a long update operation. This command applies to a Design of Experiments model, or the Refinement Points of a Kriging Response Surface, or a Parameters Correlation model. The command returns the table of the generated points.

Return The DataReference of the ParametricTable containing the generated data.

Type DataReference (p. 1371)

Example

The following example shows how to preview a DOE model.

```
container = system1.GetContainer(ComponentName="Design of Experiment")
model = container.GetModel()
DOEMatrix = model.PreviewDesignPoints()
```

UpdateAllGoodnessOfFit

Updates all goodness of fit and the results which depend on them. If the goodness of fit are already uptodate, nothing is done.

Example

The following example shows how to update all existing GoodnessOfFit.

```
system1 = GetSystem(Name="RBU")
romBuilder1 = system2.GetContainer(ComponentName="Rom Builder")
romBuilderModel1 = romBuilder1.GetModel()
romBuilderModel1.UpdateAllGoodnessOfFit()
```

DX Parameters Correlation

This container holds data for a Parameter Correlation.

Methods

ApproveGeneratedData

Approve generated data after nonparametric changes by keeping DX results. This command is recursive and applied with any component dependent on the first component.

Example

The following example shows how to approve generated data.

```
system1 = GetSystem(Name="RSO")
designofExperiment1 = system1.GetContainer(ComponentName="Design of Experiment")
designofExperiment1.ApproveGeneratedData()
```

GetModel

Get the DataReference of the Model. An exception is thrown if the entity is not found.

Return The DataReference of the Model.

Type DataReference (p. 1371)

Example

The following example shows how the user can get a Model to change one of its properties.

```
system1 = GetSystem(Name="RSO")
designofExperiment1 = system1.GetContainer(ComponentName="Design of Experiment")
dOEModel1 = designofExperiment1.GetModel()
dOEModel1.DOEType = "eDOETYPE_OSFD"
```

Data Entities

CorrelationModel

Entity which performs and manages the DesignXplorer Parameters Correlation Component

Properties

AutoStopType

Auto Stop Type

Type CorrelationAutoStopType (p. 1369)

Read Only No

Converged

Convergence state

Type bool (p. 1360)

Read Only Yes

ConvergenceCheckFrequency

Convergence Check Frequency

Type int (p. 1394)

Read Only No

CorrelationFiltering

True to filter input parameters based on the correlation values

Type bool (p. 1360)

Read Only No

DisplayText

The general property that defines the user-visible name of an entity. This property is defined for all data entities but is used only in those entities that present a label in the user interface.

Type string (p. 1438)

Read Only No

ExportDesignPoints

If True and PreserveDesignPoints is True as well, export project for each preserved Design Point.

Type bool (p. 1360)

Read Only No

LinearCorrelationType

Correlation Type

Type LinearCorrelationType (p. 1399)

Read Only No

MaximumNumberMajorInputs

Maximum Number of Major Input Parameters

Type int (p. 1394)

Read Only No

MeanValueAccuracy

Mean Value Accuracy

Type double (p. 1378)

Read Only No

NumberCustomSamples

Number of Custom Samples

Type int (p. 1394)

Read Only Yes

NumberOfRetries

Indicates the number of times DX will try to update the failed design points.

Type int (p. 1394)

Read Only No

NumberSamples

Number of Samples

Type int (p. 1394)

Read Only No

PreserveDesignPoints

If True, preserve the Design Points at the project level after the component Update.

Type bool (p. 1360)

Read Only No

R2ContributionFiltering

True to filter input parameters based on the gain in prediction

Type bool (p. 1360)

Read Only No

RelevanceThreshold

Relevance threshold for the parameters filtering

Type double (p. 1378)

Read Only No

RestartMode

True to reuse the samples already generated

Type bool (p. 1360)

Read Only No

RetainDesignPoints

If True and PreserveDesignPoints is True as well, retain data for each preserved Design Point.

Type bool (p. 1360)

Read Only No

RetryDelay

Indicates how much time will elapse between tries. This option is only applicable when NumberOfRetries is greater than 0, otherwise it has no effect.

Type Quantity (p. 1422)

Read Only No

SampleSetSize

Size of Generated Sample Set

Type int (p. 1394)

Read Only Yes

SamplingType

Sampling Type

Type SamplingType (p. 1428)

Read Only No

StandardDeviationAccuracy

Standard Deviation Accuracy

Type double (p. 1378)

Read Only No

Methods

CreateChart

Creates a Chart entity. The chart must be updated once after its creation by invoking its Update method. Then changes to its properties will update the chart automatically.

Return The DataReference of the Chart.

Type DataReference (p. 1371)

Required Arguments

ChartType Type of chart to be created. The possible values depend on the type of Model. For instance, a ResponseSurface model accepts Spider, LocalSensitivity and Response chart while a CorrelationModel accepts CorrelationMatrix, DeterminationMatrix and CorrelationScatter charts.

Type ChartType (p. 1363)

Optional Arguments

DisplayText Displayed name of the chart. If not specified, a default name is applied, depending on the value of ChartType.

Type string (p. 1438)

Example

The following example shows how to create a chart.

```
container = system1.GetContainer(ComponentName="Optimization")
model = container.GetModel()
TradeoffChart = model.CreateChart(ChartType="eChartTradeoff")
TradeoffChart.Update()
```

DeleteCharts

Deletes a list of Chart entities.

Required Arguments

Charts List of Chart entities to delete.

Type List (p. 1400)<DataReference (p. 1371)>

Example

The following example shows how to delete existing charts.

```
container = system1.GetContainer(ComponentName="Optimization")
model = container.GetModel()
chart1 = model.GetChart(Name="TradeoffChart 1")
chart2 = model.GetChart(Name="SamplesChart 1")
model.DeleteCharts(Charts=[chart1, chart2])
```

DuplicateChart

Duplicates a Chart entity. The chart must be updated once after its creation by invoking its Update method. Then changes to its properties will update the chart automatically.

Return The DataReference of the Chart.

Type DataReference (p. 1371)

Required Arguments

Chart The source chart to duplicate.

Type DataReference (p. 1371)

Optional Arguments

DisplayText Displayed name of the chart. If not specified, a default name is applied, depending on the value of ChartType.

Type string (p. 1438)

TargetModel The model on which the duplicated chart is created. If this parameter is not set, the model of the source chart is used.

Type DataReference (p. 1371)

TargetResponsePoint Parent TargetResponsePoint of the chart. If this parameter is not set, the response point of the source chart is used if applicable.

Type DataReference (p. 1371)

Example

The following example shows how to duplicate a chart.

```
designofExperiment1 = system1.GetContainer(ComponentName="Design of Experiment")
dOEModel1 = designofExperiment1.GetModel()
designPointsCurvesChart1 = dOEModel1.GetChart(Name="DesignPointsCurves")
chart1 = dOEModel1.DuplicateChart(Chart=designPointsCurvesChart1)
chart1.Update()
```

GetChart

Query to return the chart reference for a given model and chart name.

Return The DataReference of the chart.

Type DataReference (p. 1371)

Required Arguments

Name Name of the chart.

Type string (p. 1438)

GetDesignPointReportSetting

Get the DataReference of a DesignPointReportSetting entity associated with a Model. An exception is thrown if the entity is not found.

Return The DataReference of the DesignPointReportSetting.

Type DataReference (p. 1371)

Example

The following example shows how the user can get a DesignPointReportSetting to change one of its properties.

```
system1 = GetSystem(Name="GDO")
designofExperiment1 = system1.GetContainer(ComponentName="Design of Experiment")
dOEModel1 = designofExperiment1.GetModel()
designPointReportSettingEntity1 = dOEModel1.GetDesignPointReportSetting()
designPointReportSettingEntity1.DesignPointReportImage = "FFF-Results:Figure001.png"
```

GetParameter

Get the DataReference of a Parameter. An exception is thrown if the entity is not found.

Return The DataReference of the Parameter.

Type DataReference (p. 1371)

Required Arguments

Name Name of the Parameter.

Type string (p. 1438)

Example

The following example shows how the user can get a parameter of a model to change one of its properties.

```
system1 = GetSystem(Name="RSO")
designofExperiment1 = system1.GetContainer(ComponentName="Design of Experiment")
dOEModel1 = designofExperiment1.GetModel()
inputParameter1 = dOEModel1.GetParameter(Name="P1")
inputParameter1.LowerBound = 1
```

GetParameters

Get the DataReferences of the InputParameter and OutputParameter of the model. If the optional argument InputParameters is not specified, the query returns all parameters. If it is specified and True, the query returns only input parameters. If it is False, the query returns only output parameters.

Return The DataReferences of the Parameters

Type DataReferenceSet (p. 1371)

Optional Arguments

InputParameters If True, the query returns only input parameters. If False, the query returns only output parameters. If the argument is ommitted, the query returns all parameters.

Type bool (p. 1360)

Example

The following example shows how the user can get the parameters of a model.

```
system1 = GetSystem(Name="RSO")
designofExperiment1 = system1.GetContainer(ComponentName="Design of Experiment")
dOEModel1 = designofExperiment1.GetModel()
parameters = dOEModel1.GetParameters()
```

GetParametricTable

Get the DataReference of ParametricTable. An exception is thrown if the entity is not found. Names of the tables generated internally are: "DesignPoints", "CorrelationMatrix", "CorrelationScatter", "MinDesignPoints", "MaxDesignPoints", "ResponsePoints", "DeterminationMatrix".

Return The DataReference of the ParametricTable

Type DataReference (p. 1371)

Required Arguments

Name Name of the ParametricTable

Type string (p. 1438)

Example

The following example shows how the user can get a ParametricTable to add a new row and set values.

```
system1 = GetSystem(Name="RSO")
designofExperiment1 = system1.GetContainer(ComponentName="Design of Experiment")
dOEModel1 = designofExperiment1.GetModel()
parametricTable = dOEModel1.GetParametricTable(Name="DesignPoints")
parametricTable.AddRow()
parametricTable.SetCellValue(RowIndex=9,ColumnIndex=0,Value="2.1")
```

ImportDesignPoints

Import existing Design Point entities in a custom Design of Experiments or Correlation model. If a Design Point is solved, the output parameter values are also imported if the customized model is not a Correlation model linked to a Response Surface model.

Optional Arguments

DesignPoints List of the existing Design Point entities to import. If not specified, all the Design Point entities found in the Parametric container are imported.

Type List (p. 1400)<DataReference (p. 1371)>

ExpandRanges If true, the command expands the range of variation of the input parameters based on the imported design points. This is done by modifying automatically the upper and lower bounds of the input parameters. If false, the design points which are not contained in the actual variation ranges are not imported. This option is not supported in the context of: - a DOE for a Six Sigma Analysis. - a Correlation based on a Response Surface.

Type bool (p. 1360)

Default Value False

ShrinkRanges If true, the command shrinks the range of variation of the input parameters based on the imported design points. This is done by modifying automatically the upper and lower bounds of the input parameters. If false, the design points which are not contained in the actual variation ranges are not imported. This option is not supported in the context of: - a DOE for a Six Sigma Analysis. - a Correlation based on a Response Surface.

Type bool (p. 1360)

Default Value False

Example

The following example shows how the user can import all design points or a list of design points into a Design of Experiments model.

```
container = system1.GetContainer(ComponentName="Design of Experiment")
model = container.GetModel()
model.DOEType = "eDOETYPE_USER"
model.ImportDesignPoints()
designPoint0 = Parameters.GetDesignPoint(Name="0")
designPoint1 = Parameters.GetDesignPoint(Name="1")
designPoint2 = Parameters.GetDesignPoint(Name="2")
model.ImportDesignPoints(DesignPoints=[designPoint0, designPoint1, designPoint2])
model.ImportDesignPoints(DesignPoints=[designPoint1])
```

ImportDesignPointsFromDps

Import Design Point values in a custom Design of Experiments or Correlation model with a query to the DPS.

Required Arguments

QueryString The query.

Type string (p. 1438)

Optional Arguments

ExpandRanges If true, the command expands the range of variation of the input parameters based on the imported design points. This is done by modifying automatically the upper and lower bounds of the input parameters. If false, the design points which are not contained in the actual variation ranges are not imported. This option is not suppor-

ted in the context of: - a DOE for a Six Sigma Analysis. - a Correlation based on a Response Surface.

Type bool (p. 1360)

Default Value False

ShrinkRanges If true, the command shrinks the range of variation of the input parameters based on the imported design points. This is done by modifying automatically the upper and lower bounds of the input parameters. If false, the design points which are not contained in the actual variation ranges are not imported. This option is not supported in the context of: - a DOE for a Six Sigma Analysis. - a Correlation based on a Response Surface.

Type bool (p. 1360)

Default Value False

Example

The following example shows how the user can import the design point with the ID equal to 1.

```
container = system1.GetContainer(ComponentName="Design of Experiment")
model = container.GetModel()
model.DOEType = "eDOETYPE_USER"
model.ImportDesignPointsFromDps(QueryString="?id=1", ExpandRanges=True)
```

ImportDesignPointsFromFile

Import Design Point values in a custom Design of Experiments model or in a custom Correlation model from a csv file.

Required Arguments

FileName The imported file name.

Type string (p. 1438)

Optional Arguments

ExpandRanges If true, the command expands the range of variation of the input parameters based on the imported design points. This is done by modifying automatically the upper and lower bounds of the input parameters. If false, the design points which are not contained in the actual variation ranges are not imported. This option is not supported in the context of: - a DOE for a Six Sigma Analysis. - a Correlation based on a Response Surface.

Type bool (p. 1360)

Default Value False

ShrinkRanges If true, the command shrinks the range of variation of the input parameters based on the imported design points. This is done by modifying automatically the upper

and lower bounds of the input parameters. If false, the design points which are not contained in the actual variation ranges are not imported. This option is not supported in the context of: - a DOE for a Six Sigma Analysis. - a Correlation based on a Response Surface.

Type bool (p. 1360)

Default Value False

Example

The following example shows how the user can import design points from a valid csv file in a custom Correlation model.

```
container = system1.GetContainer(ComponentName="Correlation")
model = container.GetModel()
model.SamplingType = "Manual"
model.ImportDesignPointsFromFile(FilePath="designs.csv", ExpandRanges=True)
```

PreviewDesignPoints

Previews the Design Points of a model, without actually updating them, so that the user can adjust settings before launching a long update operation. This command applies to a Design of Experiments model, or the Refinement Points of a Kriging Response Surface, or a Parameters Correlation model. The command returns the table of the generated points.

Return The DataReference of the ParametricTable containing the generated data.

Type DataReference (p. 1371)

Example

The following example shows how to preview a DOE model.

```
container = system1.GetContainer(ComponentName="Design of Experiment")
model = container.GetModel()
DOEMatrix = model.PreviewDesignPoints()
```

InputParameter

The data entity which describes an Input Parameter in DesignXplorer.

Properties

Attribute1

First editable attribute of the distribution for an uncertainty parameter. The nature of the attribute depends on the distribution type. For instance, the first attribute of a Normal distribution is the Mean value.

Type double (p. 1378)

Read Only No

Attribute2

Second editable attribute of the distribution for an uncertainty parameter. The nature of the attribute depends on the distribution type. For instance, the second attribute of a Normal distribution is the Standard Deviation value. Some distribution type do not have a second attribute.

Type double (p. 1378)

Read Only No

ConstantValue

Constant value of the Parameter when it is disabled.

Type Object (p. 1409)

Read Only No

CustomDefinitionList

Custom Definition List

Type List (p. 1400)<DataReference (p. 1371)>

Read Only No

DiscreteLevels

List of the discrete levels.

Type List (p. 1400)<DataReference (p. 1371)>

Read Only Yes

DisplayText

The general property that defines the user-visible name of an entity. This property is defined for all data entities but is used only in those entities that present a label in the user interface.

Type string (p. 1438)

Read Only No

DistributionLowerBound

Distribution lower bound of the variation range for an uncertainty Parameter.

Type double (p. 1378)

Read Only No

DistributionType

Distribution type for an uncertainty parameter.

Type DistributionType (p. 1377)

Read Only No

DistributionUpperBound

Distribution upper bound of the variation range for an uncertainty Parameter.

Type double (p. 1378)

Read Only No

Enabled

True if the Parameter is enabled for the current study.

Type bool (p. 1360)

Read Only No

Kurtosis

Kurtosis value of the distribution for an uncertainty parameter.

Type double (p. 1378)

Read Only Yes

LowerBound

Lower bound of the variation range for a Continuous Parameter.

Type double (p. 1378)

Read Only No

Mean

Mean value of the distribution for an uncertainty parameter.

Type double (p. 1378)

Read Only Yes

Nature

Nature of the Parameter.

Type ParameterNature (p. 1413)

Read Only No

NumberOfLevels

Number of levels if the parameter nature is Discrete, or the parameter nature is Continuous and the UseManufacturableValues property is set to True.

Type int (p. 1394)

Read Only Yes

Skewness

Skewness value of the distribution for an uncertainty parameter.

Type double (p. 1378)

Read Only Yes

StandardDeviation

Standard deviation value of the distribution for an uncertainty parameter.

Type double (p. 1378)

Read Only Yes

Type

Type of the Parameter, either a DesignVariable in a GDO context, or an UncertaintyVariable in a SixSigma Analysis context.

Type SimulationType (p. 1432)

Read Only Yes

Units

Units

Type string (p. 1438)

Read Only Yes

UpperBound

Upper bound of the variation range for a Continuous Parameter.

Type double (p. 1378)

Read Only No

UseManufacturableValues

True to restrict the variation of the parameter to defined Manufacturable Values.

Type bool (p. 1360)

Read Only No

Methods

AddDiscreteLevel

Adds a Discrete Level entity on a discrete input parameter. A discrete level can have an integer value (e.g. a number of holes, a number of turns, etc) or a string value (e.g. a material name or a geometry file name). The command has optional arguments to specify the Name of the level and its Index in the list of levels of the parameter. By default, the new level is added to the end of the list. The various discrete levels of an input parameter represent independent configurations of the project, processed in the order of their creation.

Return The created entity.

Type DataReference (p. 1371)

Required Arguments

Value The value of the discrete level. Value can be an integer or a string.

Type Object (p. 1409)

Optional Arguments

DisplayText DisplayText of the created entity. If not specified, a default name of the form "Level [#]" is used.

Type string (p. 1438)

Index The position of the new level in the list of discrete levels of the parameter. Index is zero-based. If it is not specified, the new level is appended to the list.

Type int (p. 1394)

Default Value -1

Example

The following example shows how to add new discrete levels on a discrete input parameter. The third level is inserted between the two others.

```
container = system1.GetContainer(ComponentName="Design of Experiment")
model = container.GetModel()
DiscreteInputParameter = model.GetParameter(Name="P2")
DiscreteInputParameter.AddDiscreteLevel(Value=3000, DisplayText="Three thousand turns")
DiscreteInputParameter.AddDiscreteLevel(Value=2000, DisplayText="Two thousand turns")
DiscreteInputParameter.AddDiscreteLevel(Value=5000, Index="1")
```

AddLevels

Adds a list of levels to a continuous input parameter. Each level is a quantity or a real number corresponding to a manufacturable value. The list of levels forms a restriction filter used when post-processing

the input parameter. If levels are added outside of the variation range, the lower and upper bounds are adjusted accordingly.

Required Arguments

Levels List of added levels.

Type List (p. 1400)<string (p. 1438)>

Optional Arguments

Overwrite True in order to overwrite the existing levels, False by default.

Type bool (p. 1360)

Example

The following example shows how to overwrite the manufacturable values of an input parameter and how to define an additional value later.

```
container = system1.GetContainer(ComponentName="Design of Experiment")
model = container.GetModel()
InputParameter = model.GetParameter(Name="P1")
InputParameter.UseManufacturableValues = True
InputParameter.AddLevels(Levels=["0.3 [mm]", "0.5 [mm]", "1e-3 [m]"], Overwrite=True)
InputParameter.AddLevels(Levels="7 [mm]")
```

CreateOptimizationCriterion

Creates an OptimizationCriterion entity associated to a parameter.

Return The DataReference of the OptimizationCriterion.

Type DataReference (p. 1371)

Required Arguments

Parameter The parameter on which the criterion is created.

Type DataReference (p. 1371)

Example

The following example shows how to create an OptimizationCriterion entity.

```
container = system1.GetContainer(ComponentName="Optimization")
model = container.GetModel()
parameter1 = model.GetParameter(Name="P1")
optimizationCriterion = parameter1.CreateOptimizationCriterion()
```

DeleteDiscreteLevels

Deletes a list of levels from a discrete input parameter.

Required Arguments

DiscreteLevels List of the DiscreteLevel entities to delete.

Type List (p. 1400)<DataReference (p. 1371)>

Example

The following example shows how to add and then delete one or more levels from a discrete input parameter.

```
container = system1.GetContainer(ComponentName="Design of Experiment")
model = container.GetModel()
DiscreteInputParameter = model.GetParameter(Name="P1")
level1 = DiscreteInputParameter.AddDiscreteLevel(Value=3000, DisplayText="Three thousand turns")
level2 = DiscreteInputParameter.AddDiscreteLevel(Value=2000, DisplayText="Two thousand turns")
level3 = DiscreteInputParameter.AddDiscreteLevel(Value=5000, Index="1")
DiscreteInputParameter.DeleteDiscreteLevels(DiscreteLevels=[level1, level2])
```

DeleteLevels

Deletes a list of levels from a continuous input parameter.

Required Arguments

Indices Indices of the items to remove from the levels list

Type List (p. 1400)<int (p. 1394)>

Example

The following example shows how to add and then delete one or more levels from a continuous input parameter for which the UseManufacturableValues property is set to True.

```
container = system1.GetContainer(ComponentName="Design of Experiment")
model = container.GetModel()
InputParameter = model.GetParameter(Name="P1")
InputParameter.UseManufacturableValues = True
InputParameter.AddLevels(Levels=["0.3 [mmm]", "0.5 [mm]", "1e-3 [m]"], Overwrite=True)
InputParameter.DeleteLevels( Indices=[0, 1] )
```

ExportData

Export the data of a DesignXplorer entity to a csv file. The entity can be one of the DesignXplorer chart entities, a ParametricTable or an InputParameter entity.

Required Arguments

FileName The exported file name.

Type string (p. 1438)

Optional Arguments

AppendMode True to append to an existing csv file, False to overwrite it.

Type bool (p. 1360)

Default Value False

Example

The following example shows how the user can export the table of Design Points and then the Parameters Parallel chart of a Design of Experiments component.

```
container = system1.GetContainer(ComponentName="Design of Experiment")
model = container.GetModel()
parametricTable = model.GetParametricTable(Name="DesignPoints")
parametricTable.ExportData(FileName="doe.csv")
chart = model.GetChart(Name="DesignPointsParallel")
chart.ExportData(FileName="doe.csv", AppendMode=True)
```

GetCustomParameterDefinition

Get the DataReference of a CustomParameterProperties entity associated with an Input Parameter. An exception is thrown if the entity is not found.

Return The DataReference of the CustomParameterProperties.

Type DataReference (p. 1371)

Required Arguments

ExtensionName Extension's Name.

Type string (p. 1438)

Example

The following example shows how the user can get a CustomParameterProperties.

```
system1 = GetSystem(Name="DOP")
optimization1 = system1.GetContainer(ComponentName="Optimization")
optimizationModel1 = optimization1.GetModel()
inputParameter1 = optimizationModel1.GetParameter(Name="P1")
customParameterDefinition1 = inputParameter1.GetCustomParameterDefinition(ExtensionName="UncertaintyParame
customParameterDefinition1.Distribution = "Triangular"
```

GetDiscreteLevel

Get a discrete level by name from an input parameter. The parameter's full and ordered list of discrete levels is available as its "DiscreteLevels" property.

Return The DataReference of the discrete level entity.

Type DataReference (p. 1371)

Required Arguments

Name The name of the discrete level.

Type string (p. 1438)

Example

The following example shows how the user can retrieve a discrete level of a discrete input parameter by its name.

```
system1 = GetSystem(Name="RSO")
designofExperiment1 = system1.GetContainer(ComponentName="Design of Experiment")
dOEModel1 = designofExperiment1.GetModel()
DiscreteInputParameter1 = dOEModel1.GetParameter(Name="P1")
level = DiscreteInputParameter1.GetDiscreteLevel(Name="Level 1")
```

GetOptimizationCriterion

Get the DataReference of the OptimizationCriterion associated with a Parameter. An exception is thrown if the entity is not found.

Return The DataReference of the OptimizationCriterion.

Type DataReference (p. 1371)

Required Arguments

Parameter Parent Parameter.

Type DataReference (p. 1371)

Example

The following example shows how the user can get a OptimizationCriterion to change one of its properties.

```
system1 = GetSystem(Name="RSO")
optimization1 = system1.GetContainer(ComponentName="Optimization")
optimizationModel1 = optimization1.GetModel()
parameter3 = optimizationModel1.GetParameter(Name="P3")
optimizationCriterion = parameter3.GetOptimizationCriterion()
optimizationCriterion.ObjectiveType = "eGT_MinimumPossible"
```

GetParameterStatistics

Get the DataReference of the ParameterStatistics entity associated with a Parameter.

Return The DataReference of the ParameterStatistics entity.

Type DataReference (p. 1371)

Example

The following example shows how the user can get a ParameterStatistics entity and examine its Mean property.

```
system1 = GetSystem(Name="SSA")
sixSigmaAnalysis1 = system1.GetContainer(ComponentName="Six Sigma Analysis")
sixSigmaModel1 = sixSigmaAnalysis1.GetModel()
parameter4 = sixSigmaModel1.GetParameter(Name="P4")
parameterStatistics1 = parameter4.GetParameterStatistics()
mean = parameterStatistics1.Mean
```

OutputParameter

Output parameter entity for DesignXplorer.

Properties

DisplayText

The general property that defines the user-visible name of an entity. This property is defined for all data entities but is used only in those entities that present a label in the user interface.

Type string (p. 1438)

Read Only No

EnableAutoRefinement

Determines whether the Auto Refinement is applicable to the output parameter.

Type bool (p. 1360)

Read Only No

IgnoreForFiltering

Determines whether the parameter filtering is not applicable to the output parameter in the correlation context.

Type bool (p. 1360)

Read Only No

InheritFromModelSettings

Determines whether the Maximum Predicted Relative Error defined at the Model level is applicable to the output parameter.

Type bool (p. 1360)

Read Only No

LowerBound

Minimum value extracted from existing design points and/or sample sets.

Type double (p. 1378)

Read Only Yes

MaximumPredictedError

Maximum Predicted Error for an output parameter with the GARS algorithm. This is the maximum predicted error for the selected output parameter.

Type double (p. 1378)

Read Only Yes

MaxPredictedRelativeError

Maximum Relative Error targeted for an output parameter when refining with the Kriging algorithm. This is the maximum predicted relative error that is acceptable for the selected output parameter.

Type double (p. 1378)

Read Only No

PredictedRelativeError

Current value of the Predicted Relative Error when refining with the Kriging algorithm

Type double (p. 1378)

Read Only Yes

Scaling

Scaling

Type bool (p. 1360)

Read Only No

Tolerance

Maximum Error targeted for an output parameter when refining with the GARS algorithm. This is the maximum predicted error that is acceptable for the selected output parameter.

Type double (p. 1378)

Read Only No

TransformationType

Transformation Type

Type TransformationType (p. 1443)

Read Only No

Units

Units

Type string (p. 1438)

Read Only Yes

UpperBound

Maximum value extracted from existing design points and/or sample sets.

Type double (p. 1378)

Read Only Yes

Methods

CreateOptimizationCriterion

Creates an OptimizationCriterion entity associated to a parameter.

Return The DataReference of the OptimizationCriterion.

Type DataReference (p. 1371)

Required Arguments

Parameter The parameter on which the criterion is created.

Type DataReference (p. 1371)

Example

The following example shows how to create an OptimizationCriterion entity.

```
container = system1.GetContainer(ComponentName="Optimization")
model = container.GetModel()
parameter1 = model.GetParameter(Name="P1")
optimizationCriterion = parameter1.CreateOptimizationCriterion()
```

GetOptimizationCriterion

Get the DataReference of the OptimizationCriterion associated with a Parameter. An exception is thrown if the entity is not found.

Return The DataReference of the OptimizationCriterion.

Type DataReference (p. 1371)

Required Arguments

Parameter Parent Parameter.

Type DataReference (p. 1371)

Example

The following example shows how the user can get a OptimizationCriterion to change one of its properties.

```
system1 = GetSystem(Name="RSO")
optimization1 = system1.GetContainer(ComponentName="Optimization")
optimizationModel1 = optimization1.GetModel()
parameter3 = optimizationModel1.GetParameter(Name="P3")
optimizationCriterion = parameter3.GetOptimizationCriterion()
optimizationCriterion.ObjectiveType = "eGT_MinimumPossible"
```

GetParameterStatistics

Get the DataReference of the ParameterStatistics entity associated with a Parameter.

Return The DataReference of the ParameterStatistics entity.

Type DataReference (p. 1371)

Example

The following example shows how the user can get a ParameterStatistics entity and examine its Mean property.

```
system1 = GetSystem(Name="SSA")
sixSigmaAnalysis1 = system1.GetContainer(ComponentName="Six Sigma Analysis")
sixSigmaModel1 = sixSigmaAnalysis1.GetModel()
parameter4 = sixSigmaModel1.GetParameter(Name="P4")
parameterStatistics1 = parameter4.GetParameterStatistics()
mean = parameterStatistics1.Mean
```

DX ROM Builder

This container holds data for the Rom Builder.

Methods

GetModel

Get the DataReference of the Model. An exception is thrown if the entity is not found.

Return The DataReference of the Model.

Type DataReference (p. 1371)

Example

The following example shows how the user can get a Model to change one of its properties.

```
system1 = GetSystem(Name="RSO")
designofExperiment1 = system1.GetContainer(ComponentName="Design of Experiment")
dOEModel1 = designofExperiment1.GetModel()
dOEModel1.DOEType = "eDOETYPE_OSFD"
```

Data Entities

OutputParameter

Output parameter entity for DesignXplorer.

Properties

DisplayText

The general property that defines the user-visible name of an entity. This property is defined for all data entities but is used only in those entities that present a label in the user interface.

Type string (p. 1438)

Read Only No

EnableAutoRefinement

Determines whether the Auto Refinement is applicable to the output parameter.

Type bool (p. 1360)

Read Only No

IgnoreForFiltering

Determines whether the parameter filtering is not applicable to the output parameter in the correlation context.

Type bool (p. 1360)

Read Only No

InheritFromModelSettings

Determines whether the Maximum Predicted Relative Error defined at the Model level is applicable to the output parameter.

Type bool (p. 1360)

Read Only No

LowerBound

Minimum value extracted from existing design points and/or sample sets.

Type double (p. 1378)

Read Only Yes

MaximumPredictedError

Maximum Predicted Error for an output parameter with the GARS algorithm. This is the maximum predicted error for the selected output parameter.

Type double (p. 1378)

Read Only Yes

MaxPredictedRelativeError

Maximum Relative Error targeted for an output parameter when refining with the Kriging algorithm. This is the maximum predicted relative error that is acceptable for the selected output parameter.

Type double (p. 1378)

Read Only No

PredictedRelativeError

Current value of the Predicted Relative Error when refining with the Kriging algorithm

Type double (p. 1378)

Read Only Yes

Scaling

Scaling

Type bool (p. 1360)

Read Only No

Tolerance

Maximum Error targeted for an output parameter when refining with the GARS algorithm. This is the maximum predicted error that is acceptable for the selected output parameter.

Type double (p. 1378)

Read Only No

TransformationType

Transformation Type

Type TransformationType (p. 1443)

Read Only No

Units

Units

Type string (p. 1438)

Read Only Yes

UpperBound

Maximum value extracted from existing design points and/or sample sets.

Type double (p. 1378)

Read Only Yes

Methods

CreateOptimizationCriterion

Creates an OptimizationCriterion entity associated to a parameter.

Return The DataReference of the OptimizationCriterion.

Type DataReference (p. 1371)

Required Arguments

Parameter The parameter on which the criterion is created.

Type DataReference (p. 1371)

Example

The following example shows how to create an OptimizationCriterion entity.

```
container = system1.GetContainer(ComponentName="Optimization")
model = container.GetModel()
parameter1 = model.GetParameter(Name="P1")
optimizationCriterion = parameter1.CreateOptimizationCriterion()
```

GetOptimizationCriterion

Get the DataReference of the OptimizationCriterion associated with a Parameter. An exception is thrown if the entity is not found.

Return The DataReference of the OptimizationCriterion.

Type DataReference (p. 1371)

Required Arguments

Parameter Parent Parameter.

Type DataReference (p. 1371)

Example

The following example shows how the user can get a OptimizationCriterion to change one of its properties.

```
system1 = GetSystem(Name="RSO")
optimization1 = system1.GetContainer(ComponentName="Optimization")
optimizationModel1 = optimization1.GetModel()
parameter3 = optimizationModel1.GetParameter(Name="P3")
optimizationCriterion = parameter3.GetOptimizationCriterion()
optimizationCriterion.ObjectiveType = "eGT_MinimumPossible"
```

GetParameterStatistics

Get the DataReference of the ParameterStatistics entity associated with a Parameter.

Return The DataReference of the ParameterStatistics entity.

Type DataReference (p. 1371)

Example

The following example shows how the user can get a ParameterStatistics entity and examine its Mean property.

```
system1 = GetSystem(Name="SSA")
sixSigmaAnalysis1 = system1.GetContainer(ComponentName="Six Sigma Analysis")
sixSigmaModel1 = sixSigmaAnalysis1.GetModel()
parameter4 = sixSigmaModel1.GetParameter(Name="P4")
parameterStatistics1 = parameter4.GetParameterStatistics()
mean = parameterStatistics1.Mean
```

RomBuilder

Entity that performs and manages the DesignXplorer RomBuilder component

Properties

AssociationType

Region association type

Type AssociationType (p. 1355)

Read Only No

ConstructionType

Construction type

Type ConstructionType (p. 1368)

Read Only No

DisplayLevel

Display Level type of Log File

Type DisplayLevelType (p. 1377)

Read Only No

DisplayText

The general property that defines the user-visible name of an entity. This property is defined for all data entities but is used only in those entities that present a label in the user interface.

Type string (p. 1438)

Read Only No

Engine

Engine type

Type RomEngine (p. 1426)

Read Only No

ExportDesignPoints

If True and PreserveDesignPoints is True as well, export project for each preserved Design Point.

Type bool (p. 1360)

Read Only No

GenerateVerificationPoints

If True, generate verification points.

Type bool (p. 1360)

Read Only No

LogFile

Log File of rom generation process

Type string (p. 1438)

Read Only Yes

MaxRelativeError

Maximum accuracy

Type double (p. 1378)

Read Only No

NumberOfRetries

Indicates the number of times DX will try to update the failed design points.

Type int (p. 1394)

Read Only No

NumberVerificationPoints

Number of verification points to generate.

Type int (p. 1394)

Read Only No

NumMode

Number of modes

Type int (p. 1394)

Read Only No

PreserveDesignPoints

If True, preserve the Design Points at the project level after the component Update.

Type bool (p. 1360)

Read Only No

ProjectFile

Generated project file. Null if not updated yet.

Type DataReference (p. 1371)

Read Only Yes

ProjectionStrategy

Projection strategy

Type ProjectionStrategy (p. 1421)

Read Only No

RetainDesignPoints

If True and PreserveDesignPoints is True as well, retain data for each preserved Design Point.

Type bool (p. 1360)

Read Only No

RetryDelay

Indicates how much time will elapse between tries. This option is only applicable when NumberOfRetries is greater than 0, otherwise it has no effect.

Type Quantity (p. 1422)

Read Only No

SolverSystem

Solver system ID that is used for building a ROM.

Type string (p. 1438)

Read Only No

StoreSnapshots

Activate storage of snapshots in rom file

Type bool (p. 1360)

Read Only No

Methods

CreateChart

Creates a Chart entity. The chart must be updated once after its creation by invoking its Update method. Then changes to its properties will update the chart automatically.

Return The DataReference of the Chart.

Type DataReference (p. 1371)

Required Arguments

ChartType Type of chart to be created. The possible values depend on the type of Model. For instance, a ResponseSurface model accepts Spider, LocalSensitivity and Response chart while a CorrelationModel accepts CorrelationMatrix, DeterminationMatrix and CorrelationScatter charts.

Type ChartType (p. 1363)

Optional Arguments

DisplayText Displayed name of the chart. If not specified, a default name is applied, depending on the value of ChartType.

Type string (p. 1438)

Example

The following example shows how to create a chart.

```
container = system1.GetContainer(ComponentName="Optimization")
model = container.GetModel()
TradeoffChart = model.CreateChart(ChartType="eChartTradeoff")
TradeoffChart.Update()
```

CreateGoodnessOfFit

Creates a GoodnessOfFit. The ParameterValues dictionary can be used to specify a value for some or all of the discrete input parameters.

Return The created entity.

Type DataReference (p. 1371)

Optional Arguments

DisplayText DisplayText of the created entity. If not specified, a default name of the form "Goodness Of Fit [#]" is used.

Type string (p. 1438)

ForceUpdate Update the created entity if true.

Type bool (p. 1360)

Default Value True

ParameterValues The values for each discrete input parameter. If not specified, each parameter is initialized to the current level.

Type Dictionary (p. 1375)<DataReference (p. 1371), string (p. 1438)>

Example

The following example shows how to create a GoodnessOfFit from an existing ResponseSurface model. The code retrieves the parameters P1 and P2 and then creates the response point by assigning a value to each of these parameters.

```
container = system1.GetContainer(ComponentName="Response Surface")
model = container.GetModel()
inputParameter1 = model.GetParameter(Name="P1")
inputParameter2 = model.GetParameter(Name="P2")
goodnessOfFit = model.CreateGoodnessOfFit( DisplayText="GOF1",
```

```
                        ParameterValues={inputParameter1: "2", inputParameter2: "15"})
```

CreateRomErrorSnapshot

Create a Rom Snapshot file containing the prediction errors generated by a ROM Model for a row of a ParametricTable.

Required Arguments

ParametricTable ParametricTable.

Type DataReference (p. 1371)

RowIndex Zero-based row index of the cell.

Type int (p. 1394)

SnapshotPath Destination of the ROM snapshot file.

Type string (p. 1438)

Example

The following example shows how to create a Rom Snapshot containing the prediction errors generated by a ROM Model for a verification point and for a design of experiments point.

```
system1 = GetSystem(Name="RBU")
romBuilder1 = system1.GetContainer(ComponentName="Rom Builder")
romBuilderModel1 = romBuilder1.GetModel()
parametricTable1 = romBuilderModel1.GetParametricTable(Name="VerificationPoints")
romBuilderModel1.CreateRomErrorSnapshot(ParametricTable=parametricTable1,RowIndex=6,SnapshotPath="C:/Temp/
designofExperiment1 = system1.GetContainer(ComponentName="Design of Experiment")
dOEModel1 = designofExperiment1.GetModel()
parametricTable2 = dOEModel1.GetParametricTable(Name="DesignPoints")
romBuilderModel1.CreateRomErrorSnapshot(ParametricTable=parametricTable2,RowIndex=9,SnapshotPath="C:/Temp/
```

CreateRomPredictionSnapshot

Create a Rom Snapshot file containing the values predicted by a ROM Model for a row of a ParametricTable.

Required Arguments

ParametricTable ParametricTable.

Type DataReference (p. 1371)

RowIndex Zero-based row index of the cell.

Type int (p. 1394)

SnapshotPath Destination of the ROM snapshot file.

Type string (p. 1438)

Example

The following example shows how to create a Rom Snapshot containing the predicted values based on ROM Model for a verification point and for a design of experiments point.

```
system1 = GetSystem(Name="RBU")
romBuilder1 = system1.GetContainer(ComponentName="Rom Builder")
romBuilderModel1 = romBuilder1.GetModel()
parametricTable1 = romBuilderModel1.GetParametricTable(Name="VerificationPoints")
romBuilderModel1.CreateRomPredictionSnapshot(ParametricTable=parametricTable1,RowIndex=6,SnapshotPath="C:/
designofExperiment1 = system1.GetContainer(ComponentName="Design of Experiment")
dOEModel1 = designofExperiment1.GetModel()
parametricTable2 = dOEModel1.GetParametricTable(Name="DesignPoints")
romBuilderModel1.CreateRomPredictionSnapshot(ParametricTable=parametricTable2,RowIndex=9,SnapshotPath="C:/
```

DeleteGoodnessOfFit

Deletes a list of GoodnessOfFit entities and all the depending Chart entities.

Required Arguments

GoodnessOfFit DataReferences of the entities to delete

Type List (p. 1400)<DataReference (p. 1371)>

Example

The following example shows how to delete existing GoodnessOfFit.

```
container = system1.GetContainer(ComponentName="Response Surface")
model = container.GetModel()
gof1 = model.GetGoodnessOfFit(Name="GOF 1")
gof5 = model.GetGoodnessOfFit(Name="GOF 5")
gof6 = model.GetGoodnessOfFit(Name="GOF 6")
model.DeleteGoodnessOfFit(GoodnessOfFit=[gof1, gof5, gof6])
```

DuplicateGoodnessOfFit

Duplicates a GoodnessOfFit.

Return The created entity.

Type DataReference (p. 1371)

Required Arguments

GoodnessOfFit DataReference of the GoodnessOfFit.

Type DataReference (p. 1371)

Optional Arguments

DisplayText DisplayText of the created entity. If not specified, a default name of the form "Goodness Of Fit [#]" is used.

Type string (p. 1438)

ForceUpdate Update the created entity if true.

Type bool (p. 1360)

Default Value True

TargetModel The ResponseSurface model on which the goodness of fit is created. If this parameter is not set, the model of the source goodness of fit is used.

Type DataReference (p. 1371)

Example

The following example shows how to duplicate a GoodnessOfFit from an existing ResponseSurface model. The code retrieves the parameters P1 and P2 and then creates the response point by assigning a value to each of these parameters.

```
container = system1.GetContainer(ComponentName="Response Surface")
responseSurfaceModel1 = container.GetModel()
gof = responseSurfaceModel1.GetGoodnessOfFit(Name="GoodnessOfFit")
goodnessOfFit2 = model.DuplicateGoodnessOfFit(GoodnessOfFit=gof, DisplayText="GOF2")
goodnessOfFit2.Update()
```

ExportFmu2

Export a ROM in a .fmu file. The file can then be opened in any FMU consuming application. Applications such as TwinBuilder can also provide visualization of the ROM.

Required Arguments

FilePath Destination of the ROM archive file.

Type string (p. 1438)

ExportRomArchive

Export a ROM in a .romz archive. The file can then be opened in a ROM consuming application such as ROM Viewer.

Required Arguments

FilePath Destination of the ROM archive file.

Type string (p. 1438)

GetChart

Query to return the chart reference for a given model and chart name.

Return The DataReference of the chart.

Type DataReference (p. 1371)

Required Arguments

Name Name of the chart.

Type string (p. 1438)

GetDesignPointReportSetting

Get the DataReference of a DesignPointReportSetting entity associated with a Model. An exception is thrown if the entity is not found.

Return The DataReference of the DesignPointReportSetting.

Type DataReference (p. 1371)

Example

The following example shows how the user can get a DesignPointReportSetting to change one of its properties.

```
system1 = GetSystem(Name="GDO")
designofExperiment1 = system1.GetContainer(ComponentName="Design of Experiment")
dOEModel1 = designofExperiment1.GetModel()
designPointReportSettingEntity1 = dOEModel1.GetDesignPointReportSetting()
designPointReportSettingEntity1.DesignPointReportImage = "FFF-Results:Figure001.png"
```

GetGoodnessOfFit

Get the DataReference of a GoodnessOfFit entity associated with a Model. An exception is thrown if the entity is not found.

Return The DataReference of the GoodnessOfFit.

Type DataReference (p. 1371)

Required Arguments

Name Name of the GoodnessOfFit.

Type string (p. 1438)

Example

The following example shows how the user can get a GoodnessOfFit.

```
system1 = GetSystem(Name="RSO")
responseSurface1 = system1.GetContainer(ComponentName="Response Surface")
responseSurfaceModel1 = responseSurface1.GetModel()
gof = responseSurfaceModel1.GetGoodnessOfFit(Name="GoodnessOfFit")
```

GetGoodnessOfFits

Get the DataReferences of the GoodnessOfFit entities associated with a Model.

Return The DataReferences of the GoodnessOfFits.

Type DataReferenceSet (p. 1371)

Example

The following example shows how to retrieve the GoodnessOfFit entities of a response surface model.

```
system1 = GetSystem(Name="RSO")
responseSurface1 = system1.GetContainer(ComponentName="Response Surface")
responseSurfaceModel1 = responseSurface1.GetModel()
gof = responseSurfaceModel1.GetGoodnessOfFits()
```

GetParameter

Get the DataReference of a Parameter. An exception is thrown if the entity is not found.

Return The DataReference of the Parameter.

Type DataReference (p. 1371)

Required Arguments

Name Name of the Parameter.

Type string (p. 1438)

Example

The following example shows how the user can get a parameter of a model to change one of its properties.

```
system1 = GetSystem(Name="RSO")
designofExperiment1 = system1.GetContainer(ComponentName="Design of Experiment")
dOEModel1 = designofExperiment1.GetModel()
inputParameter1 = dOEModel1.GetParameter(Name="P1")
inputParameter1.LowerBound = 1
```

GetParameters

Get the DataReferences of the InputParameter and OutputParameter of the model. If the optional argument InputParameters is not specified, the query returns all parameters. If it is specified and True, the query returns only input parameters. If it is False, the query returns only output parameters.

Return The DataReferences of the Parameters

Type DataReferenceSet (p. 1371)

Optional Arguments

InputParameters If True, the query returns only input parameters. If False, the query returns only output parameters. If the argument is ommitted, the query returns all parameters.

Type bool (p. 1360)

Example

The following example shows how the user can get the parameters of a model.

```
system1 = GetSystem(Name="RSO")
designofExperiment1 = system1.GetContainer(ComponentName="Design of Experiment")
dOEModel1 = designofExperiment1.GetModel()
parameters = dOEModel1.GetParameters()
```

GetParametricTable

Get the DataReference of ParametricTable. An exception is thrown if the entity is not found. Names of the tables generated internally are: "DesignPoints", "CorrelationMatrix", "CorrelationScatter", "MinDesign-Points", "MaxDesignPoints", "ResponsePoints", "DeterminationMatrix".

Return The DataReference of the ParametricTable

Type DataReference (p. 1371)

Required Arguments

Name Name of the ParametricTable

Type string (p. 1438)

Example

The following example shows how the user can get a ParametricTable to add a new row and set values.

```
system1 = GetSystem(Name="RSO")
designofExperiment1 = system1.GetContainer(ComponentName="Design of Experiment")
dOEModel1 = designofExperiment1.GetModel()
parametricTable = dOEModel1.GetParametricTable(Name="DesignPoints")
parametricTable.AddRow()
parametricTable.SetCellValue(RowIndex=9,ColumnIndex=0,Value="2.1")
```

ImportRefinementPointsFromArchive

Import Refinement Point values in the manual refinement table of a Rom Builder from an archive file.

Required Arguments

ArchivePath The archive path.

Type string (p. 1438)

ReuseSnapshotFiles If true, reuse existing snapshot files of ROM production folder when they correspond to referenced snapshots of the archive. If false, copy all the snapshot files referenced in the archive (and rename them when they exist already in the ROM production folder).

Type bool (p. 1360)

Example

The following example shows how the user can import refinement points from a valid archive file.

```
system = GetSystem(Name="RBU")
romBuilder = system.GetContainer(ComponentName="Rom Builder")
romBuilderModel = romBuilder.GetModel()
romBuilderModel.ImportRefinementPointsFromArchiveCommand(FileName="E:/temp/doesnapshots2.snpz",OverwriteSn
```

ImportRefinementPointsFromDps

Import Refinement Point values in the manual refinement table of a Response Surface model with a query to the DPS.

Required Arguments

QueryString The query.

Type string (p. 1438)

Example

The following example shows how the user can import the design point with the ID equal to 1.

```
container = system1.GetContainer(ComponentName="Response Surface")
model = container.GetModel()
model.ImportRefinementPointsFromDps(QueryString="?id=1")
```

ImportRefinementPointsFromFile

Import Refinement Point values in the manual refinement table of a Response Surface model from a csv file.

Required Arguments

FileName The imported file name.

Type string (p. 1438)

Example

The following example shows how the user can import refinement points from a valid csv file.

```
container = system1.GetContainer(ComponentName="Response Surface")
model = container.GetModel()
model.ImportRefinementPointsFromFile(FilePath="designs.csv")
```

ImportVerificationPointsFromArchive

Import Verification Point values in the verification table of a Rom Builder from an archive file.

Required Arguments

ArchivePath The archive path.

Type string (p. 1438)

ReuseSnapshotFiles If true, reuse existing snapshot files of ROM production folder when they correspond to referenced snapshots of the archive. If false, copy all the snapshot files referenced in the archive (and rename them when they exist already in the ROM production folder).

Type bool (p. 1360)

Example

The following example shows how the user can import verification points from a valid archive file.

```
system = GetSystem(Name="RBU")
romBuilder = system.GetContainer(ComponentName="Rom Builder")
romBuilderModel = romBuilder.GetModel()
romBuilderModel.ImportVerificationPointsFromArchiveCommand(FileName="E:/temp/doesnapshots2.snpz",Overwrite
```

ImportVerificationPointsFromDps

Import Verification Point values in the verification table of a Response Surface model with a query to the DPS.

Required Arguments

QueryString The query.

Type string (p. 1438)

Example

The following example shows how the user can import verification points with a query to the DPS.

```
container = system1.GetContainer(ComponentName="Response Surface")
model = container.GetModel()
model.ImportVerificationPointsFromDps(QueryString="?id=1")
```

ImportVerificationPointsFromFile

Import Verification Point values in the verification table of a Response Surface model from a csv file.

Required Arguments

FileName The imported file name.

Type string (p. 1438)

Example

The following example shows how the user can import verification points from a valid csv file.

```
container = system1.GetContainer(ComponentName="Response Surface")
model = container.GetModel()
model.ImportVerificationPointsFromFile(FilePath="designs.csv")
```

UpdateAllGoodnessOfFit

Updates all goodness of fit and the results which depend on them. If the goodness of fit are already uptodate, nothing is done.

Example

The following example shows how to update all existing GoodnessOfFit.

```
system1 = GetSystem(Name="RBU")
romBuilder1 = system2.GetContainer(ComponentName="Rom Builder")
romBuilderModel1 = romBuilder1.GetModel()
romBuilderModel1.UpdateAllGoodnessOfFit()
```

DX Six Sigma Analysis

This container holds data for a Six Sigma Analysis.

Methods

GetModel

Get the DataReference of the Model. An exception is thrown if the entity is not found.

Return The DataReference of the Model.

Type DataReference (p. 1371)

Example

The following example shows how the user can get a Model to change one of its properties.

```
system1 = GetSystem(Name="RSO")
designofExperiment1 = system1.GetContainer(ComponentName="Design of Experiment")
dOEModel1 = designofExperiment1.GetModel()
dOEModel1.DOEType = "eDOETYPE_OSFD"
```

Data Entities

InputParameter

The data entity which describes an Input Parameter in DesignXplorer.

Properties

Attribute1

First editable attribute of the distribution for an uncertainty parameter. The nature of the attribute depends on the distribution type. For instance, the first attribute of a Normal distribution is the Mean value.

Type double (p. 1378)

Read Only No

Attribute2

Second editable attribute of the distribution for an uncertainty parameter. The nature of the attribute depends on the distribution type. For instance, the second attribute of a Normal distribution is the Standard Deviation value. Some distribution type do not have a second attribute.

Type double (p. 1378)

Read Only No

ConstantValue

Constant value of the Parameter when it is disabled.

Type Object (p. 1409)

Read Only No

CustomDefinitionList

Custom Definition List

Type List (p. 1400)<DataReference (p. 1371)>

Read Only No

DiscreteLevels

List of the discrete levels.

Type List (p. 1400)<DataReference (p. 1371)>

Read Only Yes

DisplayText

The general property that defines the user-visible name of an entity. This property is defined for all data entities but is used only in those entities that present a label in the user interface.

Type string (p. 1438)

Read Only No

DistributionLowerBound

Distribution lower bound of the variation range for an uncertainty Parameter.

Type double (p. 1378)

Read Only No

DistributionType

Distribution type for an uncertainty parameter.

Type DistributionType (p. 1377)

Read Only No

DistributionUpperBound

Distribution upper bound of the variation range for an uncertainty Parameter.

Type double (p. 1378)

Read Only No

Enabled

True if the Parameter is enabled for the current study.

Type bool (p. 1360)

Read Only No

Kurtosis

Kurtosis value of the distribution for an uncertainty parameter.

Type double (p. 1378)

Read Only Yes

LowerBound

Lower bound of the variation range for a Continuous Parameter.

Type double (p. 1378)

Read Only No

Mean

Mean value of the distribution for an uncertainty parameter.

Type double (p. 1378)

Read Only Yes

Nature

Nature of the Parameter.

Type ParameterNature (p. 1413)

Read Only No

NumberOfLevels

Number of levels if the parameter nature is Discrete, or the parameter nature is Continuous and the UseManufacturableValues property is set to True.

Type int (p. 1394)

Read Only Yes

Skewness

Skewness value of the distribution for an uncertainty parameter.

Type double (p. 1378)

Read Only Yes

StandardDeviation

Standard deviation value of the distribution for an uncertainty parameter.

Type double (p. 1378)

Read Only Yes

Type

Type of the Parameter, either a DesignVariable in a GDO context, or an UncertaintyVariable in a SixSigma Analysis context.

Type SimulationType (p. 1432)

Read Only Yes

Units

Units

Type string (p. 1438)

Read Only Yes

UpperBound

Upper bound of the variation range for a Continuous Parameter.

Type double (p. 1378)

Read Only No

UseManufacturableValues

True to restrict the variation of the parameter to defined Manufacturable Values.

Type bool (p. 1360)

Read Only No

Methods

AddDiscreteLevel

Adds a Discrete Level entity on a discrete input parameter. A discrete level can have an integer value (e.g. a number of holes, a number of turns, etc) or a string value (e.g. a material name or a geometry file name). The command has optional arguments to specify the Name of the level and its Index in the list of levels of the parameter. By default, the new level is added to the end of the list. The various discrete levels of an input parameter represent independent configurations of the project, processed in the order of their creation.

Return The created entity.

Type DataReference (p. 1371)

Required Arguments

Value The value of the discrete level. Value can be an integer or a string.

Type Object (p. 1409)

Optional Arguments

DisplayText DisplayText of the created entity. If not specified, a default name of the form "Level [#]" is used.

Type string (p. 1438)

Index The position of the new level in the list of discrete levels of the parameter. Index is zero-based. If it is not specified, the new level is appended to the list.

Type int (p. 1394)

Default Value -1

Example

The following example shows how to add new discrete levels on a discrete input parameter. The third level is inserted between the two others.

```
container = system1.GetContainer(ComponentName="Design of Experiment")
model = container.GetModel()
DiscreteInputParameter = model.GetParameter(Name="P2")
DiscreteInputParameter.AddDiscreteLevel(Value=3000, DisplayText="Three thousand turns")
DiscreteInputParameter.AddDiscreteLevel(Value=2000, DisplayText="Two thousand turns")
DiscreteInputParameter.AddDiscreteLevel(Value=5000, Index="1")
```

AddLevels

Adds a list of levels to a continuous input parameter. Each level is a quantity or a real number corresponding to a manufacturable value. The list of levels forms a restriction filter used when post-processing the input parameter. If levels are added outside of the variation range, the lower and upper bounds are adjusted accordingly.

Required Arguments

Levels List of added levels.

Type List (p. 1400)<string (p. 1438)>

Optional Arguments

Overwrite True in order to overwrite the existing levels, False by default.

Type bool (p. 1360)

Example

The following example shows how to overwrite the manufacturable values of an input parameter and how to define an additional value later.

```
container = system1.GetContainer(ComponentName="Design of Experiment")
model = container.GetModel()
InputParameter = model.GetParameter(Name="P1")
InputParameter.UseManufacturableValues = True
InputParameter.AddLevels(Levels=["0.3 [mm]", "0.5 [mm]", "1e-3 [m]"], Overwrite=True)
InputParameter.AddLevels(Levels="7 [mm]")
```

CreateOptimizationCriterion

Creates an OptimizationCriterion entity associated to a parameter.

Return The DataReference of the OptimizationCriterion.

Type DataReference (p. 1371)

Required Arguments

Parameter The parameter on which the criterion is created.

Type DataReference (p. 1371)

Example

The following example shows how to create an OptimizationCriterion entity.

```
container = system1.GetContainer(ComponentName="Optimization")
model = container.GetModel()
parameter1 = model.GetParameter(Name="P1")
optimizationCriterion = parameter1.CreateOptimizationCriterion()
```

DeleteDiscreteLevels

Deletes a list of levels from a discrete input parameter.

Required Arguments

DiscreteLevels List of the DiscreteLevel entities to delete.

Type List (p. 1400)<DataReference (p. 1371)>

Example

The following example shows how to add and then delete one or more levels from a discrete input parameter.

```
container = system1.GetContainer(ComponentName="Design of Experiment")
model = container.GetModel()
DiscreteInputParameter = model.GetParameter(Name="P1")
level1 = DiscreteInputParameter.AddDiscreteLevel(Value=3000, DisplayText="Three thousand turns")
level2 = DiscreteInputParameter.AddDiscreteLevel(Value=2000, DisplayText="Two thousand turns")
level3 = DiscreteInputParameter.AddDiscreteLevel(Value=5000, Index="1")
DiscreteInputParameter.DeleteDiscreteLevels(DiscreteLevels=[level1, level2])
```

DeleteLevels

Deletes a list of levels from a continuous input parameter.

Required Arguments

Indices Indices of the items to remove from the levels list

Type List (p. 1400)<int (p. 1394)>

Example

The following example shows how to add and then delete one or more levels from a continuous input parameter for which the UseManufacturableValues property is set to True.

```
container = system1.GetContainer(ComponentName="Design of Experiment")
model = container.GetModel()
InputParameter = model.GetParameter(Name="P1")
InputParameter.UseManufacturableValues = True
InputParameter.AddLevels(Levels=["0.3 [mmm]", "0.5 [mm]", "1e-3 [m]"], Overwrite=True)
InputParameter.DeleteLevels( Indices=[0, 1] )
```

ExportData

Export the data of a DesignXplorer entity to a csv file. The entity can be one of the DesignXplorer chart entities, a ParametricTable or an InputParameter entity.

Required Arguments

FileName The exported file name.

Type string (p. 1438)

Optional Arguments

AppendMode True to append to an existing csv file, False to overwrite it.

Type bool (p. 1360)

Default Value False

Example

The following example shows how the user can export the table of Design Points and then the Parameters Parallel chart of a Design of Experiments component.

```
container = system1.GetContainer(ComponentName="Design of Experiment")
model = container.GetModel()
parametricTable = model.GetParametricTable(Name="DesignPoints")
parametricTable.ExportData(FileName="doe.csv")
chart = model.GetChart(Name="DesignPointsParallel")
chart.ExportData(FileName="doe.csv", AppendMode=True)
```

GetCustomParameterDefinition

Get the DataReference of a CustomParameterProperties entity associated with an Input Parameter. An exception is thrown if the entity is not found.

Return The DataReference of the CustomParameterProperties.

Type DataReference (p. 1371)

Required Arguments

ExtensionName Extension's Name.

Type string (p. 1438)

Example

The following example shows how the user can get a CustomParameterProperties.

```
system1 = GetSystem(Name="DOP")
optimization1 = system1.GetContainer(ComponentName="Optimization")
optimizationModel1 = optimization1.GetModel()
inputParameter1 = optimizationModel1.GetParameter(Name="P1")
customParameterDefinition1 = inputParameter1.GetCustomParameterDefinition(ExtensionName="UncertaintyParame
customParameterDefinition1.Distribution = "Triangular"
```

GetDiscreteLevel

Get a discrete level by name from an input parameter. The parameter's full and ordered list of discrete levels is available as its "DiscreteLevels" property.

Return The DataReference of the discrete level entity.

Type DataReference (p. 1371)

Required Arguments

Name The name of the discrete level.

Type string (p. 1438)

Example

The following example shows how the user can retrieve a discrete level of a discrete input parameter by its name.

```
system1 = GetSystem(Name="RSO")
designofExperiment1 = system1.GetContainer(ComponentName="Design of Experiment")
dOEModel1 = designofExperiment1.GetModel()
DiscreteInputParameter1 = dOEModel1.GetParameter(Name="P1")
level = DiscreteInputParameter1.GetDiscreteLevel(Name="Level 1")
```

GetOptimizationCriterion

Get the DataReference of the OptimizationCriterion associated with a Parameter. An exception is thrown if the entity is not found.

Return The DataReference of the OptimizationCriterion.

Type DataReference (p. 1371)

Required Arguments

Parameter Parent Parameter.

Type DataReference (p. 1371)

Example

The following example shows how the user can get a OptimizationCriterion to change one of its properties.

```
system1 = GetSystem(Name="RSO")
optimization1 = system1.GetContainer(ComponentName="Optimization")
optimizationModel1 = optimization1.GetModel()
parameter3 = optimizationModel1.GetParameter(Name="P3")
optimizationCriterion = parameter3.GetOptimizationCriterion()
optimizationCriterion.ObjectiveType = "eGT_MinimumPossible"
```

GetParameterStatistics

Get the DataReference of the ParameterStatistics entity associated with a Parameter.

Return The DataReference of the ParameterStatistics entity.

Type DataReference (p. 1371)

Example

The following example shows how the user can get a ParameterStatistics entity and examine its Mean property.

```
system1 = GetSystem(Name="SSA")
sixSigmaAnalysis1 = system1.GetContainer(ComponentName="Six Sigma Analysis")
sixSigmaModel1 = sixSigmaAnalysis1.GetModel()
parameter4 = sixSigmaModel1.GetParameter(Name="P4")
parameterStatistics1 = parameter4.GetParameterStatistics()
mean = parameterStatistics1.Mean
```

OutputParameter

Output parameter entity for DesignXplorer.

Properties

DisplayText

The general property that defines the user-visible name of an entity. This property is defined for all data entities but is used only in those entities that present a label in the user interface.

Type string (p. 1438)

Read Only No

EnableAutoRefinement

Determines whether the Auto Refinement is applicable to the output parameter.

Type bool (p. 1360)

Read Only No

IgnoreForFiltering

Determines whether the parameter filtering is not applicable to the output parameter in the correlation context.

Type bool (p. 1360)

Read Only No

InheritFromModelSettings

Determines whether the Maximum Predicted Relative Error defined at the Model level is applicable to the output parameter.

Type bool (p. 1360)

Read Only No

LowerBound

Minimum value extracted from existing design points and/or sample sets.

Type double (p. 1378)

Read Only Yes

MaximumPredictedError

Maximum Predicted Error for an output parameter with the GARS algorithm. This is the maximum predicted error for the selected output parameter.

Type double (p. 1378)

Read Only Yes

MaxPredictedRelativeError

Maximum Relative Error targeted for an output parameter when refining with the Kriging algorithm. This is the maximum predicted relative error that is acceptable for the selected output parameter.

Type double (p. 1378)

Read Only No

PredictedRelativeError

Current value of the Predicted Relative Error when refining with the Kriging algorithm

Type double (p. 1378)

Read Only Yes

Scaling

Scaling

Type bool (p. 1360)

Read Only No

Tolerance

Maximum Error targeted for an output parameter when refining with the GARS algorithm. This is the maximum predicted error that is acceptable for the selected output parameter.

Type double (p. 1378)

Read Only No

TransformationType

Transformation Type

Type TransformationType (p. 1443)

Read Only No

Units

Units

Type string (p. 1438)

Read Only Yes

UpperBound

Maximum value extracted from existing design points and/or sample sets.

Type double (p. 1378)

Read Only Yes

Methods

CreateOptimizationCriterion

Creates an OptimizationCriterion entity associated to a parameter.

Return The DataReference of the OptimizationCriterion.

Type DataReference (p. 1371)

Required Arguments

Parameter The parameter on which the criterion is created.

Type DataReference (p. 1371)

Example

The following example shows how to create an OptimizationCriterion entity.

```
container = system1.GetContainer(ComponentName="Optimization")
model = container.GetModel()
parameter1 = model.GetParameter(Name="P1")
optimizationCriterion = parameter1.CreateOptimizationCriterion()
```

GetOptimizationCriterion

Get the DataReference of the OptimizationCriterion associated with a Parameter. An exception is thrown if the entity is not found.

Return The DataReference of the OptimizationCriterion.

Type DataReference (p. 1371)

Required Arguments

Parameter Parent Parameter.

Type DataReference (p. 1371)

Example

The following example shows how the user can get a OptimizationCriterion to change one of its properties.

```
system1 = GetSystem(Name="RSO")
optimization1 = system1.GetContainer(ComponentName="Optimization")
optimizationModel1 = optimization1.GetModel()
parameter3 = optimizationModel1.GetParameter(Name="P3")
optimizationCriterion = parameter3.GetOptimizationCriterion()
optimizationCriterion.ObjectiveType = "eGT_MinimumPossible"
```

GetParameterStatistics

Get the DataReference of the ParameterStatistics entity associated with a Parameter.

Return The DataReference of the ParameterStatistics entity.

Type DataReference (p. 1371)

Example

The following example shows how the user can get a ParameterStatistics entity and examine its Mean property.

```
system1 = GetSystem(Name="SSA")
sixSigmaAnalysis1 = system1.GetContainer(ComponentName="Six Sigma Analysis")
```

```
sixSigmaModel1 = sixSigmaAnalysis1.GetModel()
parameter4 = sixSigmaModel1.GetParameter(Name="P4")
parameterStatistics1 = parameter4.GetParameterStatistics()
mean = parameterStatistics1.Mean
```

SixSigmaModel

Entity which performs and manages the Six Sigma Analysis Component

Properties

DisplayText

The general property that defines the user-visible name of an entity. This property is defined for all data entities but is used only in those entities that present a label in the user interface.

Type string (p. 1438)

Read Only No

ExportDesignPoints

If True and PreserveDesignPoints is True as well, export project for each preserved Design Point.

Type bool (p. 1360)

Read Only No

NumberOfRetries

Indicates the number of times DX will try to update the failed design points.

Type int (p. 1394)

Read Only No

NumSamp

Number of Samples

Type int (p. 1394)

Read Only No

PreserveDesignPoints

If True, preserve the Design Points at the project level after the component Update.

Type bool (p. 1360)

Read Only No

RetainDesignPoints

If True and PreserveDesignPoints is True as well, retain data for each preserved Design Point.

Type bool (p. 1360)

Read Only No

RetryDelay

Indicates how much time will elapse between tries. This option is only applicable when NumberOfRetries is greater than 0, otherwise it has no effect.

Type Quantity (p. 1422)

Read Only No

SampleGenType

Sampling Type

Type SampleGenType (p. 1428)

Read Only No

Methods

CreateChart

Creates a Chart entity. The chart must be updated once after its creation by invoking its Update method. Then changes to its properties will update the chart automatically.

Return The DataReference of the Chart.

Type DataReference (p. 1371)

Required Arguments

ChartType Type of chart to be created. The possible values depend on the type of Model. For instance, a ResponseSurface model accepts Spider, LocalSensitivity and Response chart while a CorrelationModel accepts CorrelationMatrix, DeterminationMatrix and CorrelationScatter charts.

Type ChartType (p. 1363)

Optional Arguments

DisplayText Displayed name of the chart. If not specified, a default name is applied, depending on the value of ChartType.

Type string (p. 1438)

Example

The following example shows how to create a chart.

```
container = system1.GetContainer(ComponentName="Optimization")
model = container.GetModel()
TradeoffChart = model.CreateChart(ChartType="eChartTradeoff")
TradeoffChart.Update()
```

DeleteCharts

Deletes a list of Chart entities.

Required Arguments

Charts List of Chart entities to delete.

Type List (p. 1400)<DataReference (p. 1371)>

Example

The following example shows how to delete existing charts.

```
container = system1.GetContainer(ComponentName="Optimization")
model = container.GetModel()
chart1 = model.GetChart(Name="TradeoffChart 1")
chart2 = model.GetChart(Name="SamplesChart 1")
model.DeleteCharts(Charts=[chart1, chart2])
```

DuplicateChart

Duplicates a Chart entity. The chart must be updated once after its creation by invoking its Update method. Then changes to its properties will update the chart automatically.

Return The DataReference of the Chart.

Type DataReference (p. 1371)

Required Arguments

Chart The source chart to duplicate.

Type DataReference (p. 1371)

Optional Arguments

DisplayText Displayed name of the chart. If not specified, a default name is applied, depending on the value of ChartType.

Type string (p. 1438)

TargetModel The model on which the duplicated chart is created. If this parameter is not set, the model of the source chart is used.

Type DataReference (p. 1371)

TargetResponsePoint Parent TargetResponsePoint of the chart. If this parameter is not set, the response point of the source chart is used if applicable.

Type DataReference (p. 1371)

Example

The following example shows how to duplicate a chart.

```
designofExperiment1 = system1.GetContainer(ComponentName="Design of Experiment")
dOEModel1 = designofExperiment1.GetModel()
designPointsCurvesChart1 = dOEModel1.GetChart(Name="DesignPointsCurves")
chart1 = dOEModel1.DuplicateChart(Chart=designPointsCurvesChart1)
chart1.Update()
```

GetChart

Query to return the chart reference for a given model and chart name.

Return The DataReference of the chart.

Type DataReference (p. 1371)

Required Arguments

Name Name of the chart.

Type string (p. 1438)

GetParameter

Get the DataReference of a Parameter. An exception is thrown if the entity is not found.

Return The DataReference of the Parameter.

Type DataReference (p. 1371)

Required Arguments

Name Name of the Parameter.

Type string (p. 1438)

Example

The following example shows how the user can get a parameter of a model to change one of its properties.

```
system1 = GetSystem(Name="RSO")
designofExperiment1 = system1.GetContainer(ComponentName="Design of Experiment")
dOEModel1 = designofExperiment1.GetModel()
inputParameter1 = dOEModel1.GetParameter(Name="P1")
inputParameter1.LowerBound = 1
```

GetParameters

Get the DataReferences of the InputParameter and OutputParameter of the model. If the optional argument InputParameters is not specified, the query returns all parameters. If it is specified and True, the query returns only input parameters. If it is False, the query returns only output parameters.

Return The DataReferences of the Parameters

Type DataReferenceSet (p. 1371)

Optional Arguments

InputParameters If True, the query returns only input parameters. If False, the query returns only output parameters. If the argument is ommitted, the query returns all parameters.

Type bool (p. 1360)

Example

The following example shows how the user can get the parameters of a model.

```
system1 = GetSystem(Name="RSO")
designofExperiment1 = system1.GetContainer(ComponentName="Design of Experiment")
dOEModel1 = designofExperiment1.GetModel()
parameters = dOEModel1.GetParameters()
```

GetParametricTable

Get the DataReference of ParametricTable. An exception is thrown if the entity is not found. Names of the tables generated internally are: "DesignPoints", "CorrelationMatrix", "CorrelationScatter", "MinDesignPoints", "MaxDesignPoints", "ResponsePoints", "DeterminationMatrix".

Return The DataReference of the ParametricTable

Type DataReference (p. 1371)

Required Arguments

Name Name of the ParametricTable

Type string (p. 1438)

Example

The following example shows how the user can get a ParametricTable to add a new row and set values.

```
system1 = GetSystem(Name="RSO")
designofExperiment1 = system1.GetContainer(ComponentName="Design of Experiment")
dOEModel1 = designofExperiment1.GetModel()
parametricTable = dOEModel1.GetParametricTable(Name="DesignPoints")
parametricTable.AddRow()
parametricTable.SetCellValue(RowIndex=9,ColumnIndex=0,Value="2.1")
```

Engineering Data

Engineering Data

The container used by an Engineering Data component to maintain project data.

Methods

AddToProjectDefaults

Adds an item, to the list of items, that will be added to new projects by default.

The item must be a favorite before it can be added to project defaults.

Required Arguments

Source The source of material to be added to the project defaults.

Type string (p. 1438)

Optional Arguments

ItemType The EngineeringDataType of the item. This can be Material, Load, or BeamSection.

Type string (p. 1438)

Default Value Material

Example

The following example shows how to add a material named Concrete to the project defaults.

```
installDir = r"C:\Program Files\ANSYS Inc\v121"
EngData.AddToProjectDefaults(Name="Concrete",
    Source=installDir+r"\Addins\EngineeringData\Samples\General_Materials.xml"
    ItemType="Material")
```

ConsolidateMaterials

Deletes the material.

CreateMaterial

Adds a new material to the container.

Return The material data reference of the new material.

Type DataReference (p. 1371)

Required Arguments

Name The name to be used for this material.

Type string (p. 1438)

Export

Exports engineering data to the specified file.

The following type of file format is supported for export:

MatML 3.1 schema for Material(s)

Required Arguments

FilePath The target path for the Engineering Data file ("*.xml").

Type string (p. 1438)

Format The file format in which engineering data will be exported.

Type string (p. 1438)

UnitSystem The unit system in which engineering data will be exported.

Type string (p. 1438)

Optional Arguments

ApplyScaleOffset The flag to specify if the scale factor and offset value will be applied during export.

Type bool (p. 1360)

Default Value False

IgnoreSuppressed The flag to specify if suppressed engineering data will be ignored during export.

Type bool (p. 1360)

Default Value False

OverwriteTarget The flag to specify if the target file will be overwritten.

Type bool (p. 1360)

Default Value False

ReplaceMaterial The flag to specify if the earlier material data will be replaced in case of similar material names.

Type bool (p. 1360)

Default Value False

Example

template = GetTemplate(TemplateName="EngData") system = template.CreateSystem(Position="Default") container = system.GetContainer(ComponentName="Engineering Data") materials = container.GetMaterials() EngData.WriteMaterials(MaterialList=materials, FilePath=r"d:\data\OutputMaterials.xml", Format="MatML31", UnitSystem="MKS_STANDARD", ReplaceMaterial=true)

GetMaterial

Returns a material of a given name from a container.

The name matching is case insensitive. If a material is not found an exception is thrown.

Return The material that matches the specified name.

Type DataReference (p. 1371)

Required Arguments

Name The name of the material. This argument is case insensitive.

Type string (p. 1438)

Example

The following example creates a new Engineering Data system and queries the default material, Structural Steel.

```
template = GetTemplate(TemplateName="EngData")
system = template.CreateSystem(Position="Default")
container = system.GetContainer(ComponentName="Engineering Data")
structuralSteel = container.GetMaterial(Name="Structural Steel")
```

GetMaterials

Returns the list of materials in a container. If no materials are in the container, the list is empty.

Return A DataReferenceSet of the materials in the container.

Type DataReferenceSet (p. 1371)

Import

Imports engineering data into an existing source from a specified source.

The following types of files are supported for import:

Engineering Data libraries exported from Workbench 9.0 to 11.0 SP1

Material(s) file following the MatML 3.1 schema
Material(s) file generated by AUTODYN

Required Arguments

Source The source which contains the materials.

Type string (p. 1438)

ImportMaterial

Reads the data for a single material into the requested container.

Return A DataReference for the material that was read.

Type DataReference (p. 1371)

Required Arguments

Name The name of the material to read from the source.

Type string (p. 1438)

Source The source which contains the requested material.

Type string (p. 1438)

Example

This code shows how to read the Copper Alloy material from the provided samples, into Engineering Data to use in an analysis.

```
installDir = r"C:\Program Files\ANSYS Inc\v121"
matl1 = engineeringData1.ReadMaterial(
   Name="Copper Alloy",
   Source=installDir+r"\Addins\EngineeringData\Samples\General_Materials.xml")
```

RemoveFromProjectDefaults

Removes an item, from the list of items, that are added to new projects by default.

Required Arguments

Source The path to the file containing the item to remove.

Type string (p. 1438)

Optional Arguments

ItemType The EngineeringDataType of the item. This can be Material, Load, or BeamSection.

Type string (p. 1438)

Default Value Material

Example

The following example shows how to remove Concrete from project defaults.

```
installDir = r"C:\Program Files\ANSYS Inc\v121"
EngData.RemoveFromProjectDefaults(Name="Concrete",
    Source=installDir+r"\Addins\EngineeringData\Samples\General_Materials.xml"
    ItemType="Material")
```

Data Entities

CurveFit

The entity to store curve fitting information.

Properties

CurveFitData

The coefficients of material model that approximates the experimental test data.

Type DataReference (p. 1371)

Read Only No

DependentPropertyColl

The collection of experimental test data, e.g., Uniaxial Test data (Engineering Strain Vs Engineering Stress).

Type DataReferenceSet (p. 1371)

Read Only No

DisplayText

The general property that defines the user-visible name of an entity. This property is defined for all data entities but is used only in those entities that present a label in the user interface.

Type string (p. 1438)

Read Only No

UnitSystemWhenSolved

The unit system of the coefficients of material model.

Type string (p. 1438)

Read Only No

Methods

AddTestData

Adds test data from a given curve fitting.

Required Arguments

TestData The test data property to add.

Type DataReference (p. 1371)

Example

The following example loads a material with experimental test data and a Neo-Hookean hyperelastic property.

```
template = GetTemplate(TemplateName="EngData")
system = template.CreateSystem()
engineeringData = system.GetContainer(ComponentName="Engineering Data")
neopreneRubber = engineeringData.ReadMaterial(
    Name="Neoprene Rubber",
    Source="Hyperelastic_Materials.xml")
neoHookeanProperty = neopreneRubber.GetProperty(Name="Neo-Hookean")
neoHookeanPropertyData = neoHookeanProperty.GetPropertyData(Name="Neo-Hookean")
curveFit = neoHookeanPropertyData.CreateCurveFitting(
    Type="Neo-Hookean",
    Definition="")
uniaxialProperty = neopreneRubber.GetProperty(Name="Uniaxial Test Data")
curveFit.AddTestData(TestData=uniaxialProperty)
biaxialProperty = neopreneRubber.GetProperty(Name="Biaxial Test Data")
curveFit.AddTestData(TestData=biaxialProperty)
shearProperty = neopreneRubber.GetProperty(Name="Shear Test Data")
curveFit.AddTestData(TestData=shearProperty)
volumetricProperty = neopreneRubber.GetProperty(Name="Volumetric Test Data")
curveFit.AddTestData(TestData=volumetricProperty)
curveFit.RemoveTestData(TestData=uniaxialProperty)
curveFit.AddTestData(TestData=uniaxialProperty)
curveFit.Solve()
curveFit.CopyCoefficients(Destination=neoHookeanPropertyData)
curveFit.Delete()
```

CopyCoefficients

Copies the fitted coefficients from the curve fitting to the property data provided.

Required Arguments

Destination The property data that will receive the coefficients.

Type DataReference (p. 1371)

Example

The following example loads a material with experimental test data and a Neo-Hookean hyperelastic property.

```
template = GetTemplate(TemplateName="EngData")
```

```
system = template.CreateSystem()
engineeringData = system.GetContainer(ComponentName="Engineering Data")
neopreneRubber = engineeringData.ReadMaterial(
    Name="Neoprene Rubber",
    Source="Hyperelastic_Materials.xml")
neoHookeanProperty = neopreneRubber.GetProperty(Name="Neo-Hookean")
neoHookeanPropertyData = neoHookeanProperty.GetPropertyData(Name="Neo-Hookean")
curveFit = neoHookeanPropertyData.CreateCurveFitting(
    Type="Neo-Hookean",
    Definition="")
uniaxialProperty = neopreneRubber.GetProperty(Name="Uniaxial Test Data")
curveFit.AddTestData(TestData=uniaxialProperty)
biaxialProperty = neopreneRubber.GetProperty(Name="Biaxial Test Data")
curveFit.AddTestData(TestData=biaxialProperty)
shearProperty = neopreneRubber.GetProperty(Name="Shear Test Data")
curveFit.AddTestData(TestData=shearProperty)
volumetricProperty = neopreneRubber.GetProperty(Name="Volumetric Test Data")
curveFit.AddTestData(TestData=volumetricProperty)
curveFit.RemoveTestData(TestData=uniaxialProperty)
curveFit.AddTestData(TestData=uniaxialProperty)
curveFit.Solve()
curveFit.CopyCoefficients(Destination=neoHookeanPropertyData)
curveFit.Delete()
```

Delete

Deletes a curve fitting.

Example

The following example loads a material with experimental test data and a Neo-Hookean hyperelastic property.

```
template = GetTemplate(TemplateName="EngData")
system = template.CreateSystem()
engineeringData = system.GetContainer(ComponentName="Engineering Data")
neopreneRubber = engineeringData.ReadMaterial(
    Name="Neoprene Rubber",
    Source="Hyperelastic_Materials.xml")
neoHookeanProperty = neopreneRubber.GetProperty(Name="Neo-Hookean")
neoHookeanPropertyData = neoHookeanProperty.GetPropertyData(Name="Neo-Hookean")
curveFit = neoHookeanPropertyData.CreateCurveFitting(
    Type="Neo-Hookean",
    Definition="")
uniaxialProperty = neopreneRubber.GetProperty(Name="Uniaxial Test Data")
curveFit.AddTestData(TestData=uniaxialProperty)
biaxialProperty = neopreneRubber.GetProperty(Name="Biaxial Test Data")
curveFit.AddTestData(TestData=biaxialProperty)
shearProperty = neopreneRubber.GetProperty(Name="Shear Test Data")
curveFit.AddTestData(TestData=shearProperty)
volumetricProperty = neopreneRubber.GetProperty(Name="Volumetric Test Data")
curveFit.AddTestData(TestData=volumetricProperty)
curveFit.RemoveTestData(TestData=uniaxialProperty)
curveFit.AddTestData(TestData=uniaxialProperty)
curveFit.Solve()
curveFit.CopyCoefficients(Destination=neoHookeanPropertyData)
curveFit.Delete()
```

GetChartData

Returns a generated graph data for the specified source data.

Valid source data are:

Material Property
Material Property Data

Return The graph data for the specified source data.

Type Dictionary (p. 1375)<string (p. 1438), Object (p. 1409)>

RemoveTestData

Removes test data from a given curve fitting.

Required Arguments

TestData The test data property to add.

Type DataReference (p. 1371)

Example

The following example loads a material with experimental test data and a Neo-Hookean hyperelastic property.

```
template = GetTemplate(TemplateName="EngData")
system = template.CreateSystem()
engineeringData = system.GetContainer(ComponentName="Engineering Data")
neopreneRubber = engineeringData.ReadMaterial(
    Name="Neoprene Rubber",
    Source="Hyperelastic_Materials.xml")
neoHookeanProperty = neopreneRubber.GetProperty(Name="Neo-Hookean")
neoHookeanPropertyData = neoHookeanProperty.GetPropertyData(Name="Neo-Hookean")
curveFit = neoHookeanPropertyData.CreateCurveFitting(
    Type="Neo-Hookean",
    Definition="")
uniaxialProperty = neopreneRubber.GetProperty(Name="Uniaxial Test Data")
curveFit.AddTestData(TestData=uniaxialProperty)
biaxialProperty = neopreneRubber.GetProperty(Name="Biaxial Test Data")
curveFit.AddTestData(TestData=biaxialProperty)
shearProperty = neopreneRubber.GetProperty(Name="Shear Test Data")
curveFit.AddTestData(TestData=shearProperty)
volumetricProperty = neopreneRubber.GetProperty(Name="Volumetric Test Data")
curveFit.AddTestData(TestData=volumetricProperty)
curveFit.RemoveTestData(TestData=uniaxialProperty)
curveFit.AddTestData(TestData=uniaxialProperty)
curveFit.Solve()
curveFit.CopyCoefficients(Destination=neoHookeanPropertyData)
curveFit.Delete()
```

Solve

The curve fitting module will solve for the coefficients which most nearly approximate the experimental test data.

CurveFitData

The entity which contains data relevant to the curve fitting solution. It is possible to call GetData on this

entity after the curve fitting solution to determine coefficients. Also for nonlinear fits the seed values can be set via this entity.

Properties

DisplayText

The general property that defines the user-visible name of an entity. This property is defined for all data entities but is used only in those entities that present a label in the user interface.

Type string (p. 1438)

Read Only No

Methods

DeleteData

Delete a row from a tabular data sheet.

Required Arguments

Index Index of the row to delete.

Type int (p. 1394)

Optional Arguments

SheetName Name of the sheet to access.

Type string (p. 1438)

SheetQualifiers SheetQualifiers is used to pass in the qualifiers to select between multiple sheets witht the same name. This is a dictionary of the Qualifier and its Value.

Type Dictionary (p. 1375)<string (p. 1438), string (p. 1438)>

Example

The following example illustrates the deletion of a row from a tabular data sheet.

```
#
# Create a new Engineering Data System and access Structural Steel
#
template1 = GetTemplate(TemplateName="EngData")
system1 = template1.CreateSystem()
engineeringData1 = system1.GetContainer(ComponentName="Engineering Data")
matl1 = engineeringData1.GetMaterial(Name="Structural Steel")
#
# Delete the first row in the Density property
#
matlProp1 = matl1.GetProperty(Name="Density")
DeleteTabularDataRow(matlProp1, Index = 0)
#
# Delete the first row in the Coefficient of Thermal Expansion property with
# optional SheeetName and SheetQualifiers
#
```

```
matlProp2 = matl1.GetProperty(Name="Coefficient of Thermal Expansion")
DeleteTabularDataRow(matlProp2,
                  SheetName="Coefficient of Thermal Expansion",
                  SheetQualifiers={"Definition Method": "Secant", "Behavior": "Isotropic"},
                  Index = 0)
```

GetData

Returns the tabular data associated with the data entity.

Return The returned data in scalar, list, or dictionary format.

Type Object (p. 1409)

Optional Arguments

AsDictionary If set to true, the data will be returned as a dictionary where the keys are variable names and the values are the data for each variable. If set to false, the data will be returned in scalar or list format without the variable names.

Type bool (p. 1360)

Default Value False

ColumnMajor If set to true, the data will be returned in column-major order. If set to false, the data will be returned in row-major order.

Type bool (p. 1360)

Default Value True

EndIndex The end index for requesting a subset of the data (zero-based).

Type int (p. 1394)

Default Value -2147483647

SheetName Specifies the sheet name when the data contains multiple sheets.

Type string (p. 1438)

SheetQualifiers Used to pass in the qualifiers to select between multiple sheets with the same name. This is a dictionary of qualifiers and values.

Type Dictionary (p. 1375)<string (p. 1438), string (p. 1438)>

StartIndex The start index for requesting a subset of the data (zero-based).

Type int (p. 1394)

Default Value 0

Variables Names of the variables for which data is requested (string or list of strings).

Type Object (p. 1409)

Example

In this example, all data is requested for the given tabular data entity.

```
tabData1.GetData()
```

In this example, all data is requested in row-major order.

```
tabData1.GetData(ColumnMajor=False)
```

In this example, all data is requested in dictionary format.

```
tabData1.GetData(AsDictionary=True)
```

In this example, data for variables Density and Temperature is requested in dictionary format.

```
tabData1.GetData(Variables=["Density", "Temperature"], AsDictionary=True)
```

SetData

Set tabular data associated with the data entity.

Optional Arguments

Data Sets the data using a dictionary form. The keys are the variable names and the values are the data. The use of this argument is mutually exclusive with "Values" and "Variables".

Type Dictionary (p. 1375)<string (p. 1438), List (p. 1400)<Object (p. 1409)>>

Index Specifies the starting location used to set the data (zero-based). A value of -1 indicates that the data should be appended to the existing data.

Type int (p. 1394)

Default Value 0

SheetName Specifies the sheet name when the data contains multiple sheets.

Type string (p. 1438)

SheetQualifiers Used to pass in the qualifiers to select between multiple sheets with the same name. This is a dictionary of qualifiers and values.

Type Dictionary (p. 1375)<string (p. 1438), string (p. 1438)>

Values List of data values set in conjunction with the "Variables" parameter. This parameter and the "Data" parameter are mutually exclusive.

Type List (p. 1400)<List (p. 1400)<Object (p. 1409)>>

Variables Names of the variables for which data is being set. This parameter and and the "Data" parameter are mutually exclusive.

Type List (p. 1400)<string (p. 1438)>

Example

```
#
# Create a new Engineering Data System and access Structural Steel
#
template1 = GetTemplate(TemplateName="EngData")
system1 = template1.CreateSystem()
engineeringData1 = system1.GetContainer(ComponentName="Engineering Data")
matl1 = engineeringData1.GetMaterial(Name="Structural Steel")
#
# Change the value of a simple single-valued property
#
matlProp1 = matl1.GetProperty(Name="Density")
SetTabularData(matlProp1,
                  Variables="Density",
                  Values="8500 [kg m^-3]")
#
# Set Temperature-dependent data for Elasticity based
# on lists of variables and values.
matlProp2 = matl1.GetProperty(Name="Elasticity")
temperature = ["400 [K]", "600 [K]", "800 [K]"]
E = ["2e5 [MPa]", "1.9e5 [MPa]", "1.6e5 [MPa]"]
SetTabularData(matlProp2,
                  Variables = ["Temperature","Young's Modulus"],
                  Values = [temperature, E])
#
# Change the Temperature for the second table entry.
#
SetTabularData(matlProp2,
                  Index = 1,
                  Variables = "Temperature",
                  Values = "625 [K]")
#
# Set a list for Poisson's Ratio starting at the second table entry.
#
SetTabularData(matlProp2,
                  Index = 1,
                  Variables = "Poisson's Ratio",
                  Values = [0.3, 0.3])
#
# Set Temperature-dependent property data for the Coefficient of Thermal Expansion
# using a dictionary. The dictionary key is the Variable name,
# followed by the list of values for the variable.
#
matlProp3 = matl1.GetProperty(Name="Coefficient of Thermal Expansion")
newData = {"Temperature": ["200 [F]","400 [F]","600 [F]","800 [F]","1000 [F]"],
"Coefficient of Thermal Expansion" : ["6.3e-6 [F^-1]", "7.0e-6 [F^-1]",
                                      "7.46e-6 [F^-1]", "7.8e-6 [F^-1]",
                                      "8.04e-6 [F^-1]"]}
SetTabularData(matlProp3,
                  SheetName="Coefficient of Thermal Expansion",
                  SheetQualifiers={"Definition Method": "Secant", "Behavior": "Isotropic"},
                  Data = newData)
```

SetQualifier

Changes the values of a specifiec qualifier in a data table.

Required Arguments

Qualifier The Qualifier to Set.

Type string (p. 1438)

Value The new value.

Type string (p. 1438)

Optional Arguments

SheetName The name of the tabular data sheet that contains the qualifier.

Type string (p. 1438)

SheetQualifiers SheetQualifiers can be used to pass in the qualifiers to select between multiple sheets witht the same name. This is a dictionary of the Qualifier and its Value.

Type Dictionary (p. 1375)<string (p. 1438), string (p. 1438)>

VariableName The name of the Variable that contains the qualifier to be changed.

Type string (p. 1438)

VariableQualifiers VariableQualifiers can used to pass in the qualifiers to select between multiple variables witht the same name. This is a dictionary of the Qualifier and its Value.

Type Dictionary (p. 1375)<string (p. 1438), string (p. 1438)>

Example

The following example changes the 'Derive From' setting within an Isotropic Elasticity material property to be "Bulk Modulus and Poisson's Ratio".

```
matl1 = engineeringData1.GetMaterial(Name="Structural Steel")
matlProp1 = matl1.GetProperty(Name="Elasticity")
SetTabularDataQualifier(matlProp1,
   SheetName="Elasticity",
   Qualifier="Derive from",
   Value="Bulk Modulus and Poisson's Ratio")
```

DelimitedDataObject

Entity to store the necessary information for importing delimited data.

Properties

Columns

List of columns to read in. Leave this unset to import all columns.

```
Columns=[3,5]
```

Type List (p. 1400)<int (p. 1394)>

Read Only No

Delimiter

Delimiter type.

```
Delimiter=","
```

Type string (p. 1438)

Read Only No

DisplayText

The general property that defines the user-visible name of an entity. This property is defined for all data entities but is used only in those entities that present a label in the user interface.

Type string (p. 1438)

Read Only No

FileName

Name of file to import. File can have one of two formats:

Format 1 - File contains variable names and units # This information is the experimental data collected from a uniaxial test on a elastomer sample # and was lifted from Hyperelastic Materials sample Elastomer Sample (Mooney-Rivlin) # # Set "Start Import at Line" to 6 Strain, Stress # Variable Names (these correspond to the names shown in the UI) m m^-1, Pa # Unit (these correspond to the unit strings shown in the UI) 0.1338, 494.1474492 # Data point 1 0.2675, 912.7972764 # Data point 2 0.3567, 1172.453938 # Data point 3

Format 2 - File does not contain variable names and units, these are passed in by using the # VariableNames and VariableUnits parameters # This information is the experimental data collected from a uniaxial test on a elastomer sample # and was lifted from Hyperelastic Materials sample Elastomer Sample (Mooney-Rivlin) # # Set "Start Import at Line" to 7 0.1338, 494.1474492 # Data point 1 0.2675, 912.7972764 # Data point 2 0.3567, 1172.453938 # Data point 3

Type string (p. 1438)

Read Only No

ReadLine

Line to start importing. Use this if your file has a header you wish to ignore. The line number you set here must correspond to the row in your file with the variable names.

```
BeginAtLine = 1 -- if the variable names first line of your file.
```

Type int (p. 1394)

Read Only No

VariableNames

List of variable names, the list must have the same number of names as columns in the data. If the list is null, it is assumed that the variable names are located in the file at line "BeginAtLine"

Type List (p. 1400)<string (p. 1438)>

Read Only No

VariableUnits

List of variable units, the list must have the same number of units as columns in the data. If the list is null, it is assumed that the variable units are located in the file at line "BeginAtLine" + 1

Type List (p. 1400)<string (p. 1438)>

Read Only No

Methods

Import

Execute import, and appends delimited data to the MaterialProperty Data. Call this method after setting the following properties of the DelimitedDataObject: FileName, Delimiter, Readline (optional), Columns (optional).

FieldVariableDataObject

The FieldVariableDataObject is used to get/set the properties of one field variable, or a group of field variables (all with the same name and quantity type).

Properties

DataString

Get or set the default data to be used in the interpolation algorithm if this field variable isn't defined.

Type string (p. 1438)

Read Only No

DisplayText

The general property that defines the user-visible name of an entity. This property is defined for all data entities but is used only in those entities that present a label in the user interface.

Type string (p. 1438)

Read Only No

LowerLimitString

Get or set the lower limits of the field variable, and is used to validate the field variable data. This value is also used by the interpolation algorithm during normalization.

Type string (p. 1438)

Read Only No

QuantityType

Get the quantity type of the field variable.

Type string (p. 1438)

Read Only Yes

Unit

Get or set the units.

Type string (p. 1438)

Read Only No

UpperLimitString

Get or set the upper limits of the field variable, and is used to validate the field variable data. This value is also used by the interpolation algorithm during normalization.

Type string (p. 1438)

Read Only No

VariableName

Get the name of the Field Variable.

Type string (p. 1438)

Read Only Yes

Material

The entity to store material information.

Properties

Description

The description of the material.

Type string (p. 1438)

Read Only No

DisplayName

The display name of the material.

Type string (p. 1438)

Read Only No

DisplayText

The general property that defines the user-visible name of an entity. This property is defined for all data entities but is used only in those entities that present a label in the user interface.

Type string (p. 1438)

Read Only No

Source

The path of the material's source file ("*.engd" or "*.xml").

Type string (p. 1438)

Read Only Yes

SourceFileReference

The reference for the material's source file ("*.engd" or "*.xml").

Type DataReference (p. 1371)

Read Only Yes

Methods

AddToFavorites

Adds an item to the Favorites item list.

CreateProperty

Includes a physical quantity or the constitutive relation for the physical response of a material. A property can be visualized as one or more tables of data made up of one or more dependent and independent variables.

The material property is created based on the specified optional parameters "Definition" and "Behavior".

Return Created material properly.

Type DataReference (p. 1371)

Required Arguments

Name The new material property name.

Type string (p. 1438)

Optional Arguments

Behavior The optional string to identify the way in which a new material property will behave.

Behavior of some material properties can be specified in different ways, e.g., Elasticity can be specified as Isotropic, Orthotropic or Anisotropic.

Type string (p. 1438)

CustomData The optional dictionary of custom data. A non-null dictionary designates the material property as custom and not located in the Engineering Data Metadata.

Type Dictionary (p. 1375)<string (p. 1438), string (p. 1438)>

Definition The optional string to identify the way in which new material property will be defined.

Some material properties are defined in different ways, e.g., Thermal Expansion can be defined as Secant Coefficient of Thermal Expansion and Instantaneous Coefficient of Thermal Expansion.

Type string (p. 1438)

Qualifiers The optional dictionary of a qualifier name and it's corresponding value.

Type Dictionary (p. 1375)<string (p. 1438), string (p. 1438)>

Example

The following example creates a new Engineering Data system and adds Coefficient of Thermal Expansion and Elasticity material properties to the default material Structural Steel.

Coefficient of Thermal Expansion is created using Secant definition and Orthotropic behavior. Elasticity is created using Orthotropic behavior.

```
ENGDTemplate = GetTemplate(TemplateName="EngData")
ENGDSystem = ENGDTemplate.CreateSystem(Position="Default")

ENGDContainer = ENGDSystem.GetContainer(ComponentName="Engineering Data")
StructSteel = ENGDContainer.GetMaterial(Name="Structural Steel")

# Create material properties
CTEProperty = StructSteel.CreateProperty(
   Name="Coefficient of Thermal Expansion",
   Definition="Secant",
   Behavior="Orthotropic")
OrthoElasProperty = StructSteel.CreateProperty(
   Name="Elasticity",
   Behavior="Orthotropic")
```

Delete

Deletes the material.

Duplicate

Duplicates the data in this material and returns a new material. The name of the new material will be appended with a numerical value to make it unique.

Return The material data reference of the duplicated material.

Type DataReference (p. 1371)

Required Arguments

TargetContainer The duplicated material will be added to this container.

Type DataContainerReference (p. 1371)

GetFieldVariableDataObject

Gets a FieldVariableDataObject from the specified parent. The parent determines which field variables are tied to the data object

Return DataReference of the field variable.

Type DataReference (p. 1371)

Required Arguments

Name Name of the field variable.

Type string (p. 1438)

Example

```
template1 = GetTemplate(TemplateName="EngData")
system1 = template1.CreateSystem()
engineeringData1 = system1.GetContainer(ComponentName="Engineering Data")
structuralSteel = engineeringData1.GetMaterial(Name="Structural Steel")
temperature = structuralSteel.GetFieldVariableDataObject(Name="Temperature")
```

GetProperty

Returns a material property of a given name from the specified material.

Return The material property that matches the specified name.

Type DataReference (p. 1371)

Required Arguments

Name The name of the material property.

Type string (p. 1438)

Example

The following example creates new Engineering Data system and queries Coefficient of Thermal Expansion property of the default material Structural Steel.

```
ENGDTemplate = GetTemplate(TemplateName="EngData")
ENGDSystem = ENGDTemplate.CreateSystem(Position="Default")

ENGDContainer = ENGDSystem.GetContainer(ComponentName="Engineering Data")
StructSteel = ENGDContainer.GetMaterial(Name="Structural Steel")

# Get material property
CTEProperty = StructSteel.GetProperty(Name="Coefficient of Thermal Expansion")
```

IsSuppressed

Checks if an entity is suppressed.

Valid entities are:

Material
Material Property
Material Property Data

Return Returns true if the entity is suppressed, false otherwise.

Type bool (p. 1360)

IsValid

Validates a material data and provides a message in case of invalid data.

Return The flag that indicates if the material is valid.

Type bool (p. 1360)

Optional Arguments

Message The validation failure message.

Type Output (p. 1411)<string (p. 1438)>

Refresh

This will repopulate the contents of a material with data from a given source. The NameInSource property of a Material will be used to find a match in the source to pull data from. In the event that NameInSource is not set, the DisplayName property will be used instead.

Note: This operation will cause destruction of any data currently in the material.

```
template = GetTemplate(TemplateName="EngData")
system = template.CreateSystem(Position="Default")
container = system.GetContainer(ComponentName="Engineering Data")
```

```
structSteel = container.GetMaterial(Name="Structural Steel")
density = structSteel.GetProperty(Name="Density")
density.Delete()
structSteel.UpdateMaterial(Source="General_Materials.xml")
restoredDensity = structSteel.GetProperty(Name="Density")
restoredDensity.SetData(SheetName="", Index=0, Variables=["Density"], Values=[["8000 [kg m^-3]"]])
```

Required Arguments

Source The path of the source file.

Type string (p. 1438)

RemoveFromFavorites

Removes an item from the Favorite items list.

Optional Arguments

Format The format of the file that contains the item.

Type string (p. 1438)

Material A DataReference to the Material to delete.

Type DataReference (p. 1371)

Name The name of the item to delete.

Type string (p. 1438)

Source The location of the file that contains the item on disk.

Type string (p. 1438)

Type The type of the object to delete. This is either a Material or a Mixture.

Type EngineeringDataType (p. 1381)

Example

The following example gets the list of favorites from Engineering Data. It then selects and deletes the "Gray Cast Iron" material from the list.

```
favorites = EngData.LoadFavoriteItems()
matl = favorites.GetMaterial(Name="Gray Cast Iron")
EngData.DeleteFromFavorites(
    Material=matl,
    Type="Material")
```

SetAsDefaultFluidForModel

This will specify the material to use (or not use) for parts in the model which are marked as a fluid.

If the material is in the Engineering Data component it will set or unset the material to be used in the model component of the system(s) that contain this Engineering Data component.

If the material is contained in Favorites it will set or unset the material to use as the default on fluids in the Engineering Data component when a new system is added to the project.

The material is set as the default in Favorites. It will automatically be added to the list of project defaults.

Optional Arguments

Default This Boolean is used to set or unset the material as the default.

Type bool (p. 1360)

Default Value True

SetAsDefaultSolidForModel

This will specify the material to use (or not use) for parts in the model which are marked as a solid.

If the material is in the Engineering Data component it will set or unset the material to be used in the model component of the system(s) that contain this Engineering Data component.

If the material is contained in Favorites it will set or unset the material to use as the default on solids in the Engineering Data component when a new system is added to the project.

The material is set as the default in Favorites. It will automatically be added to the list of project defaults.

Optional Arguments

Default This Boolean is used to set or unset the material as the default.

Type bool (p. 1360)

Default Value True

SetColor

Add a new 'Color' property in a material.

Required Arguments

Blue Blue component of the color assigned to material.

Type int (p. 1394)

Green Green component of the color assigned to material.

Type int (p. 1394)

Red Red component of the color assigned to material.

Type int (p. 1394)

Example

```
template1 = GetTemplate(TemplateName="EngData")
system1 = template1.CreateSystem()
engineeringData1 = system1.GetContainer(ComponentName="Engineering Data")
structuralSteel = engineeringData1.GetMaterial(Name="Structural Steel")
structuralSteel.SetColor(Material="strucutralSteel", Red="255", Green="0", Blue="0")
```

SetOption

Set option for an "Option" variable

Required Arguments

OptionName Option to set

Type string (p. 1438)

OptionValue Value for option

Type string (p. 1438)

SetSuppression

Suppresses or Unsuppresses item.

Item can be suppressed to prevent it from being sent to a downstream cell in the system.

Following items can be suppressed:

Material

Material property

Material property data

Required Arguments

Suppressed The flag to specify if the item should be suppressed or unsuppressed.

Type bool (p. 1360)

Unlink

Unlinks a material from its underlying source. Once this occurs, the material can no longer be restored to its original state and the material will no longer have a source.

MaterialProperty

The entity to store material property information.

A material property is the identifier for the singular information (for example, Density) that together with other properties defines or models the behavior of the material. A property is always defined by

at least one table (tabular data), which could be singular. Some properties can contain a collection of tabular data (for example, Isotropic Elasticity).

Properties

Behavior

The string that defines the way in which the material property behaves, e.g., Elasticity has Isotropic, Orthotropic or Anisotropic behavior.

Type string (p. 1438)

Read Only No

Definition

The definition of the material property. Some material properties are defined in different ways, e.g., Thermal Expansion can be defined as Secant Coefficient of Thermal Expansion and Instantaneous Coefficient of Thermal Expansion.

Type string (p. 1438)

Read Only No

Description

The description of the material property.

Type string (p. 1438)

Read Only No

DisplayText

The general property that defines the user-visible name of an entity. This property is defined for all data entities but is used only in those entities that present a label in the user interface.

Type string (p. 1438)

Read Only No

PropertyDataColl

The collection of tabular data that defines the material property.

Type DataReferenceSet (p. 1371)

Read Only Yes

TypeName

The name of the material property.

Type string (p. 1438)

Read Only No

Methods

AddTestData

Includes experimental test data for response function calculations.

Required Arguments

TestData The test data property to add.

Type DataReference (p. 1371)

BeginBatchUpdate

Marks the start of a series of data modifications to a table of data, to improve performance.

CreatePropertyData

Include an additional property data for a material property. Some properties may have more than one property data to describe the material property.

The preferred method of adding a material property data is to use CreateMaterialProperty.

Return The new material properly data that was created.

Type DataReference (p. 1371)

Required Arguments

Name The new material property data name.

Type string (p. 1438)

Optional Arguments

Behavior A string to identify how the new material property data will behave.

The behavior of some material properties can be specified in different ways, e.g., Elasticity can be specified as Isotropic, Orthotropic or Anisotropic.

Type string (p. 1438)

CustomData The optional dictionary of custom data. A non-null dictionary designates the material property as custom and not located in the Engineering Data Metadata.

Type Dictionary (p. 1375)<string (p. 1438), string (p. 1438)>

Definition A string to identify how the new material property data will be defined.

Some material properties are defined in different ways, e.g., Thermal Expansion can be defined as Secant Coefficient of Thermal Expansion and Instantaneous Coefficient of Thermal Expansion.

Type string (p. 1438)

Qualifiers The optional dictionary of a qualifier name and it's corresponding value.

Type Dictionary (p. 1375)<string (p. 1438), string (p. 1438)>

Example

The following example creates the material property data Orthotropic Secant Coefficient of Thermal Expansion on the material property Coefficient of Thermal Expansion. This example assumes the material Structural Steel has been obtained from the General Materials library.

Get the material property we are going to create a new material property data on.

```
thermExpansionMatProp = structuralSteel.GetMaterialProperty(Name="Coefficient of Thermal Expansion")
```

Create the new material property data.

```
orthoSecantThermExpansionMatPropData = thermExpansionMatProp.CreatePropertyData(
    Name="Coefficient of Thermal Expansion",
    Definition="Secant",
    Behavior="Orthotropic")
```

Delete

Deletes the material property.

Optional Arguments

Behavior The optional string to specify the material property behavior.

Behavior of some material properties can be specified in different ways e.g. Elasticity can be specified as Isotropic, Orthotropic or Anisotropic.

Type string (p. 1438)

Definition The optional string to specify the material property definition.

Some material properties are defined in different ways e.g. Thermal Expansion can be defined as Secant Coefficient of Thermal Expansion and Instantaneous Coefficient of Thermal Expansion.

Type string (p. 1438)

DeleteData

Delete a row from a tabular data sheet.

Required Arguments

Index Index of the row to delete.

Type int (p. 1394)

Optional Arguments

SheetName Name of the sheet to access.

Type string (p. 1438)

SheetQualifiers SheetQualifiers is used to pass in the qualifiers to select between multiple sheets witht the same name. This is a dictionary of the Qualifier and its Value.

Type Dictionary (p. 1375)<string (p. 1438), string (p. 1438)>

Example

The following example illustrates the deletion of a row from a tabular data sheet.

```
#
# Create a new Engineering Data System and access Structural Steel
#
template1 = GetTemplate(TemplateName="EngData")
system1 = template1.CreateSystem()
engineeringData1 = system1.GetContainer(ComponentName="Engineering Data")
matl1 = engineeringData1.GetMaterial(Name="Structural Steel")
#
# Delete the first row in the Density property
#
matlProp1 = matl1.GetProperty(Name="Density")
DeleteTabularDataRow(matlProp1, Index = 0)
#
# Delete the first row in the Coefficient of Thermal Expansion property with
# optional SheeetName and SheetQualifiers
#
matlProp2 = matl1.GetProperty(Name="Coefficient of Thermal Expansion")
DeleteTabularDataRow(matlProp2,
                  SheetName="Coefficient of Thermal Expansion",
                  SheetQualifiers={"Definition Method": "Secant", "Behavior": "Isotropic"},
                  Index = 0)
```

EndBatchUpdate

Marks the completion of a series of data modifications.

GetChartData

Returns a generated graph data for the specified source data.

Valid source data are:

Material Property
Material Property Data

Return The graph data for the specified source data.

Type Dictionary (p. 1375)<string (p. 1438), Object (p. 1409)>

GetData

Returns the tabular data associated with the data entity.

Return The returned data in scalar, list, or dictionary format.

Type Object (p. 1409)

Optional Arguments

AsDictionary If set to true, the data will be returned as a dictionary where the keys are variable names and the values are the data for each variable. If set to false, the data will be returned in scalar or list format without the variable names.

Type bool (p. 1360)

Default Value False

ColumnMajor If set to true, the data will be returned in column-major order. If set to false, the data will be returned in row-major order.

Type bool (p. 1360)

Default Value True

EndIndex The end index for requesting a subset of the data (zero-based).

Type int (p. 1394)

Default Value -2147483647

SheetName Specifies the sheet name when the data contains multiple sheets.

Type string (p. 1438)

SheetQualifiers Used to pass in the qualifiers to select between multiple sheets with the same name. This is a dictionary of qualifiers and values.

Type Dictionary (p. 1375)<string (p. 1438), string (p. 1438)>

StartIndex The start index for requesting a subset of the data (zero-based).

Type int (p. 1394)

Default Value 0

Variables Names of the variables for which data is requested (string or list of strings).

Type Object (p. 1409)

Example

In this example, all data is requested for the given tabular data entity.

```
tabData1.GetData()
```

In this example, all data is requested in row-major order.

```
tabData1.GetData(ColumnMajor=False)
```

In this example, all data is requested in dictionary format.

```
tabData1.GetData(AsDictionary=True)
```

In this example, data for variables Density and Temperature is requested in dictionary format.

```
tabData1.GetData(Variables=["Density", "Temperature"], AsDictionary=True)
```

GetPropertyData

Returns a property data of the specified material property.

The property data returned is based on specified optional parameters "Definition" and "Behavior".

Return The material property data that matches specified type name, definition and behavior.

Type DataReference (p. 1371)

Required Arguments

Name The material property data type name.

Type string (p. 1438)

Optional Arguments

Behavior The optional string to specify the material property data behavior.

Behavior of some material properties can be specified in different ways, e.g., Elasticity can be specified as Isotropic, Orthotropic or Anisotropic.

Type string (p. 1438)

Definition The optional string to specify the material property data definition.

Some material properties are defined in different ways, e.g., Thermal Expansion can be defined as Secant Coefficient of Thermal Expansion and Instantaneous Coefficient of Thermal Expansion.

Type string (p. 1438)

Qualifiers The optional dictionary of a qualifier name and it's corresponding value.

Type Dictionary (p. 1375)<string (p. 1438), string (p. 1438)>

Example

The following example creates new Engineering Data system and queries Coefficient of Thermal Expansion property data of the default material Structural Steel.

```
ENGDTemplate = GetTemplate(TemplateName="EngData")
ENGDSystem = ENGDTemplate.CreateSystem(Position="Default")

ENGDContainer = ENGDSystem.GetContainer(ComponentName="Engineering Data")
StructSteel = ENGDContainer.GetMaterial(Name="Structural Steel")

#Get material property
CTEProperty = StructSteel.GetProperty(Name="Coefficient of Thermal Expansion")
# Get material property data
CTEPropData = CTEProperty.GetPropertyData(
   Name="Coefficient of Thermal Expansion",
   Definition="Secant",
   Behavior="Isotropic")
RefTempPropData = CTEProperty.GetPropertyData(
   Name="Reference Temperature",
   Definition="Secant",
   Behavior="Isotropic")
```

IsSuppressed

Checks if an entity is suppressed.

Valid entities are:

Material
Material Property
Material Property Data

Return Returns true if the entity is suppressed, false otherwise.

Type bool (p. 1360)

IsValid

Validates a material property and provides a message in case of invalid data.

Return The flag that indicates if the material property is valid.

Type bool (p. 1360)

Required Arguments

Material The parent material of the property.

Type DataReference (p. 1371)

Optional Arguments

Message The validation failure message.

Type Output (p. 1411)<string (p. 1438)>

RemoveTestData

Excludes experimental test data for response function calculations.

Required Arguments

TestData The test data property to add.

Type DataReference (p. 1371)

SetData

Set tabular data associated with the data entity.

Optional Arguments

Data Sets the data using a dictionary form. The keys are the variable names and the values are the data. The use of this argument is mutually exclusive with "Values" and "Variables".

Type Dictionary (p. 1375)<string (p. 1438), List (p. 1400)<Object (p. 1409)>>

Index Specifies the starting location used to set the data (zero-based). A value of -1 indicates that the data should be appended to the existing data.

Type int (p. 1394)

Default Value 0

SheetName Specifies the sheet name when the data contains multiple sheets.

Type string (p. 1438)

SheetQualifiers Used to pass in the qualifiers to select between multiple sheets with the same name. This is a dictionary of qualifiers and values.

Type Dictionary (p. 1375)<string (p. 1438), string (p. 1438)>

Values List of data values set in conjunction with the "Variables" parameter. This parameter and the "Data" parameter are mutually exclusive.

Type List (p. 1400)<List (p. 1400)<Object (p. 1409)>>

Variables Names of the variables for which data is being set. This parameter and and the "Data" parameter are mutually exclusive.

Type List (p. 1400)<string (p. 1438)>

Example

```
#
# Create a new Engineering Data System and access Structural Steel
#
```

```
template1 = GetTemplate(TemplateName="EngData")
system1 = template1.CreateSystem()
engineeringData1 = system1.GetContainer(ComponentName="Engineering Data")
matl1 = engineeringData1.GetMaterial(Name="Structural Steel")
#
# Change the value of a simple single-valued property
#
matlProp1 = matl1.GetProperty(Name="Density")
SetTabularData(matlProp1,
                  Variables="Density",
                  Values="8500 [kg m^-3]")
#
# Set Temperature-dependent data for Elasticity based
# on lists of variables and values.
matlProp2 = matl1.GetProperty(Name="Elasticity")
temperature = ["400 [K]", "600 [K]", "800 [K]"]
E = ["2e5 [MPa]", "1.9e5 [MPa]", "1.6e5 [MPa]"]
SetTabularData(matlProp2,
                  Variables = ["Temperature","Young's Modulus"],
                  Values = [temperature, E])
#
# Change the Temperature for the second table entry.
#
SetTabularData(matlProp2,
                  Index = 1,
                  Variables = "Temperature",
                  Values = "625 [K]")
#
# Set a list for Poisson's Ratio starting at the second table entry.
#
SetTabularData(matlProp2,
                  Index = 1,
                  Variables = "Poisson's Ratio",
                  Values = [0.3, 0.3])
#
# Set Temperature-dependent property data for the Coefficient of Thermal Expansion
# using a dictionary. The dictionary key is the Variable name,
# followed by the list of values for the variable.
#
matlProp3 = matl1.GetProperty(Name="Coefficient of Thermal Expansion")
newData = {"Temperature": ["200 [F]","400 [F]","600 [F]","800 [F]","1000 [F]"],
"Coefficient of Thermal Expansion" : ["6.3e-6 [F^-1]", "7.0e-6 [F^-1]",
                                      "7.46e-6 [F^-1]", "7.8e-6 [F^-1]",
                                      "8.04e-6 [F^-1]"]}
SetTabularData(matlProp3,
                  SheetName="Coefficient of Thermal Expansion",
                  SheetQualifiers={"Definition Method": "Secant", "Behavior": "Isotropic"},
                  Data = newData)
```

SetQualifier

Changes the values of a specifiec qualifier in a data table.

Required Arguments

Qualifier The Qualifier to Set.

Type string (p. 1438)

Value The new value.

Type string (p. 1438)

Optional Arguments

SheetName The name of the tabular data sheet that contains the qualifier.

Type string (p. 1438)

SheetQualifiers SheetQualifiers can be used to pass in the qualifiers to select between multiple sheets witht the same name. This is a dictionary of the Qualifier and its Value.

Type Dictionary (p. 1375)<string (p. 1438), string (p. 1438)>

VariableName The name of the Variable that contains the qualifier to be changed.

Type string (p. 1438)

VariableQualifiers VariableQualifiers can used to pass in the qualifiers to select between multiple variables witht the same name. This is a dictionary of the Qualifier and its Value.

Type Dictionary (p. 1375)<string (p. 1438), string (p. 1438)>

Example

The following example changes the 'Derive From' setting within an Isotropic Elasticity material property to be "Bulk Modulus and Poisson's Ratio".

```
matl1 = engineeringData1.GetMaterial(Name="Structural Steel")
matlProp1 = matl1.GetProperty(Name="Elasticity")
SetTabularDataQualifier(matlProp1,
   SheetName="Elasticity",
   Qualifier="Derive from",
   Value="Bulk Modulus and Poisson's Ratio")
```

SetSuppression

Suppresses or Unsuppresses item.

Item can be suppressed to prevent it from being sent to a downstream cell in the system.

Following items can be suppressed:

Material

Material property

Material property data

Required Arguments

Suppressed The flag to specify if the item should be suppressed or unsuppressed.

Type bool (p. 1360)

MaterialPropertyData

The entity to store material property (tabular) data information. The material property data is a collection of material variable data.

Properties

Behavior

The behavior of the material variable tabular data. Some material properties can have different behavior, e.g., Elasticity has Isotropic, Orthotropic or Anisotropic behavior.

Type string (p. 1438)

Read Only No

Definition

The definition of the material variable tabular data. Some material properties are defined in different ways, e.g., Thermal Expansion can be defined as Secant Coefficient of Thermal Expansion and Instantaneous Coefficient of Thermal Expansion.

Type string (p. 1438)

Read Only No

DependentColl

The collection of dependent variables in the material variable tabular data, e.g., Density.

Type DataReferenceSet (p. 1371)

Read Only Yes

Description

The description of the material variable tabular data.

Type string (p. 1438)

Read Only No

DisplayText

The general property that defines the user-visible name of an entity. This property is defined for all data entities but is used only in those entities that present a label in the user interface.

Type string (p. 1438)

Read Only No

IndependentColl

The collection of independent variables in the material variable tabular data, e.g., Temperature.

Type DataReferenceSet (p. 1371)

Read Only Yes

PrimaryIndependent

The primary independent variable in the material variable tabular data, e.g., Temperature.

Type DataReference (p. 1371)

Read Only Yes

RowsCount

The number of data values for a variable in the material variable tabular data.

Type int (p. 1394)

Read Only Yes

TypeName

The name of the material variable tabular data.

Type string (p. 1438)

Read Only No

VariableColl

The collection of variables in the material variable tabular data.

Type DataReferenceSet (p. 1371)

Read Only Yes

Methods

BeginBatchUpdate

Marks the start of a series of data modifications to a table of data, to improve performance.

CreateCurveFitting

Creates a curve fitting for a given property data.

Return The curve fitting.

Type DataReference (p. 1371)

Required Arguments

Definition The definition of curve fitting to create. This must be a definition that is supported by an engineering data curve fitting.

i.e. 1 Parameter, 2 Parameter, 1st Order, 2nd Order

Type string (p. 1438)

Type The type of curve fitting to create. This must be a type that is supported by an engineering data curve fitting.

i.e. Neo-Hookean, Mooney-Rivlin, Ogden, Yeoh, Polynomial

Type string (p. 1438)

Example

The following example loads a material with experimental test data and a Neo-Hookean hyperelastic property.

```
template = GetTemplate(TemplateName="EngData")
system = template.CreateSystem()
engineeringData = system.GetContainer(ComponentName="Engineering Data")
neopreneRubber = engineeringData.ReadMaterial(
    Name="Neoprene Rubber",
    Source="Hyperelastic_Materials.xml")
neoHookeanProperty = neopreneRubber.GetProperty(Name="Neo-Hookean")
neoHookeanPropertyData = neoHookeanProperty.GetPropertyData(Name="Neo-Hookean")
curveFit = neoHookeanPropertyData.CreateCurveFitting(
    Type="Neo-Hookean",
    Definition="")
uniaxialProperty = neopreneRubber.GetProperty(Name="Uniaxial Test Data")
curveFit.AddTestData(TestData=uniaxialProperty)
biaxialProperty = neopreneRubber.GetProperty(Name="Biaxial Test Data")
curveFit.AddTestData(TestData=biaxialProperty)
shearProperty = neopreneRubber.GetProperty(Name="Shear Test Data")
curveFit.AddTestData(TestData=shearProperty)
volumetricProperty = neopreneRubber.GetProperty(Name="Volumetric Test Data")
curveFit.AddTestData(TestData=volumetricProperty)
curveFit.RemoveTestData(TestData=uniaxialProperty)
curveFit.AddTestData(TestData=uniaxialProperty)
curveFit.Solve()
curveFit.CopyCoefficients(Destination=neoHookeanPropertyData)
curveFit.Delete()
```

CreateDataProvider

Create a data provider for a give format. The only data provider currently supported is for Delimited Text (used to import tabular data from a delimited data file).

Return DataReference of the DelimitedDataObject

Type DataReference (p. 1371)

Required Arguments

Format Format of provider (only supported option is "Delimited Text")

Type string (p. 1438)

Example

```
material1 = engineeringData1.GetMaterial(Name="Structural Steel")
matlProp1 = material1.GetProperty(Name="Coefficient of Thermal Expansion")
materialPropertyData1 = matlProp1.GetPropertyData(
Name="Coefficient of Thermal Expansion",
Qualifiers={"Definition": "Secant", "Behavior": "Isotropic"})
dataProvider1 = materialPropertyData1.CreateDataProvider(Format="Delimited Text")
dataProvider1.FileName = r"C:\Coefficient_of_thermal_expansion.csv"
dataProvider1.ReadLine = 2
dataProvider1.Columns = [1, 2]
dataProvider1.Delimiter = ","
dataProvider1.VariableNames = ["Temperature", "Coefficient of Thermal Expansion"]
dataProvider1.VariableUnits = ["C", "C^-1"]
dataProvider1.Import()
```

CreateVariable

Include an additional variable in the property data for a material property.

Return The new variable data that was created.

Type DataReference (p. 1371)

Required Arguments

Name The new variable data name.

Type string (p. 1438)

Optional Arguments

CustomData The optional dictionary of custom data. A non-null dictionary designates the material property as custom and not located in the Engineering Data Metadata.

Type Dictionary (p. 1375)<string (p. 1438), string (p. 1438)>

Qualifiers The optional dictionary of a qualifier name and it's corresponding value.

Type Dictionary (p. 1375)<string (p. 1438), string (p. 1438)>

Example

```
material1 = engineeringData1.GetMaterial(Name="Structural Steel")
matlProp1 = material1.CreateProperty(
    Name="Custom Model",
    Qualifiers={"UserMat": "USER"},
    CustomData={})
matlPropData1 = matlProp2.CreatePropertyData(
    Name="Model State Variables",
    Qualifiers={"UserMat": "STATE"},
    CustomData={})
PropertyDataVariable1 = matlPropData1.CreateVariable(
    Name="User Variable",
    Qualifiers={"Display": "True", "UserMat Constant": "1"},
    CustomData={"Quantity Type": "Dimensionless", "Data": "0", "Independent": "False"})
```

Delete

Deletes the material property data.

EndBatchUpdate

Marks the completion of a series of data modifications.

GetChartData

Returns a generated graph data for the specified source data.

Valid source data are:

Material Property
Material Property Data

Return The graph data for the specified source data.

Type Dictionary (p. 1375)<string (p. 1438), Object (p. 1409)>

GetCurveFitting

Query to return the DataReference to the CurveFit used by some PropertyData.

Return CurveFit DataReference

Type DataReference (p. 1371)

GetFieldVariableDataObject

Gets a FieldVariableDataObject from the specified parent. The parent determines which field variables are tied to the data object

Return DataReference of the field variable.

Type DataReference (p. 1371)

Required Arguments

Name Name of the field variable.

Type string (p. 1438)

Example

```
template1 = GetTemplate(TemplateName="EngData")
system1 = template1.CreateSystem()
engineeringData1 = system1.GetContainer(ComponentName="Engineering Data")
structuralSteel = engineeringData1.GetMaterial(Name="Structural Steel")
temperature = structuralSteel.GetFieldVariableDataObject(Name="Temperature")
```

GetVariable

Returns a material variable of a given name from a material property data.

Return The requested variable.

Type DataReference (p. 1371)

Required Arguments

Name The name of the variable.

Type string (p. 1438)

Example

The following example creates new Engineering Data system and queries Coefficient of Thermal Expansion property data of the default material Structural Steel.

```
ENGDTemplate = GetTemplate(TemplateName="EngData")
ENGDSystem = ENGDTemplate.CreateSystem(Position="Default")

ENGDContainer = ENGDSystem.GetContainer(ComponentName="Engineering Data")
StructSteel = ENGDContainer.GetMaterial(Name="Structural Steel")

# Get material
CTEProperty = StructSteel.GetProperty(Name="Coefficient of Thermal Expansion")
# Get material property data
CTEPropData = CTEProperty.GetPropertyData(
   Name="Coefficient of Thermal Expansion",
   Definition="Secant",
   Behavior="Isotropic")
# Get material variable
CTEVariable = CTEPropData.GetVariable(
   Name="Coefficient of Thermal Expansion")
```

IsSuppressed

Checks if an entity is suppressed.

Valid entities are:

Material
Material Property
Material Property Data

Return Returns true if the entity is suppressed, false otherwise.

Type bool (p. 1360)

IsValid

Validates a material property data and provides a message in case of invalid data.

Return The flag that indicates if the material property data is valid.

Type bool (p. 1360)

Required Arguments

Material The parent material of the material property data.

Type DataReference (p. 1371)

Optional Arguments

Message The validation failure message.

Type Output (p. 1411)<string (p. 1438)>

SetSuppression

Suppresses or Unsuppresses item.

Item can be suppressed to prevent it from being sent to a downstream cell in the system.

Following items can be suppressed:

Material

Material property

Material property data

Required Arguments

Suppressed The flag to specify if the item should be suppressed or unsuppressed.

Type bool (p. 1360)

MaterialVariable

The entity to store material variable information.

Properties

DatumColl

The collection of the material variable data.

Type DataReferenceSet (p. 1371)

Read Only No

IsDependent

The flag that indicates if the material variable is dependent, e.g., Density is temperature dependent.

Type bool (p. 1360)

Read Only No

LowerBoundUnit

The unit of the lower bound data value in the material variable data.

Type string (p. 1438)

Read Only No

LowerBoundValue

The lower bound data value in the material variable data.

Type double (p. 1378)

Read Only No

MaxValue

The maximum value limit of the material variable data.

Type double (p. 1378)

Read Only No

MinValue

The minimum value limit of the material variable data.

Type double (p. 1378)

Read Only No

Offset

The offset value of the material variable data.

Type Quantity (p. 1422)

Read Only No

Scale

The scale factor of the material variable data.

Type double (p. 1378)

Read Only No

TypeName

The name of the material variable.

Type string (p. 1438)

Read Only No

UniqueData

The collection of unique data values in the material variable data.

Type DataReferenceSet (p. 1371)

Read Only No

UpperBoundUnit

The unit of the upper bound data value in the material variable data.

Type string (p. 1438)

Read Only No

UpperBoundValue

The upper bound data value in the material variable data.

Type double (p. 1378)

Read Only No

Methods

Delete

Deletes an additional variable in the property data from a material property.

IsValid

Validates a material variable data and provides a message in case of invalid data.

Return The flag that indicates if the material variable is valid.

Type bool (p. 1360)

Required Arguments

Material The parent material of the variable.

Type DataReference (p. 1371)

MaterialPropertyData The parent material property data of the variable.

Type DataReference (p. 1371)

Optional Arguments

Message The validation failure message.

Type Output (p. 1411)<string (p. 1438)>

Example

```
template = GetTemplate(TemplateName="EngData") system = template.CreateSystem(Position="Default") container = system.GetContainer(ComponentName="Engineering Data") structSteel = container.GetMaterial(Name="Structural Steel") CTEProp = structSteel.GetProperty(Name="Coefficient of Thermal Expansion") CTEPropData = CTEProperty.GetPropertyData( Name="Coefficient of Thermal Expansion", Definition="Secant", Behavior="Isotropic") CTEVariable = CTEPropData.GetVariable( Name="Coefficient of Thermal Expansion") valid = EngData.ValidateMaterialVariable(MaterialVariable=CTEVariable, Material=structSteel, MaterialPropertyData=CTEPropdata)
```

Engineering Data Curve Fit

The container used by Engineering Data to maintain data for curve fitting operations.

Data Entities

CurveFit

The entity to store curve fitting information.

Properties

CurveFitData

The coefficients of material model that approximates the experimental test data.

Type DataReference (p. 1371)

Read Only No

DependentPropertyColl

The collection of experimental test data, e.g., Uniaxial Test data (Engineering Strain Vs Engineering Stress).

Type DataReferenceSet (p. 1371)

Read Only No

DisplayText

The general property that defines the user-visible name of an entity. This property is defined for all data entities but is used only in those entities that present a label in the user interface.

Type string (p. 1438)

Read Only No

UnitSystemWhenSolved

The unit system of the coefficients of material model.

Type string (p. 1438)

Read Only No

Methods

AddTestData

Adds test data from a given curve fitting.

Required Arguments

TestData The test data property to add.

Type DataReference (p. 1371)

Example

The following example loads a material with experimental test data and a Neo-Hookean hyperelastic property.

```
template = GetTemplate(TemplateName="EngData")
system = template.CreateSystem()
engineeringData = system.GetContainer(ComponentName="Engineering Data")
neopreneRubber = engineeringData.ReadMaterial(
    Name="Neoprene Rubber",
    Source="Hyperelastic_Materials.xml")
neoHookeanProperty = neopreneRubber.GetProperty(Name="Neo-Hookean")
neoHookeanPropertyData = neoHookeanProperty.GetPropertyData(Name="Neo-Hookean")
curveFit = neoHookeanPropertyData.CreateCurveFitting(
    Type="Neo-Hookean",
    Definition="")
uniaxialProperty = neopreneRubber.GetProperty(Name="Uniaxial Test Data")
curveFit.AddTestData(TestData=uniaxialProperty)
biaxialProperty = neopreneRubber.GetProperty(Name="Biaxial Test Data")
curveFit.AddTestData(TestData=biaxialProperty)
shearProperty = neopreneRubber.GetProperty(Name="Shear Test Data")
curveFit.AddTestData(TestData=shearProperty)
volumetricProperty = neopreneRubber.GetProperty(Name="Volumetric Test Data")
curveFit.AddTestData(TestData=volumetricProperty)
curveFit.RemoveTestData(TestData=uniaxialProperty)
curveFit.AddTestData(TestData=uniaxialProperty)
curveFit.Solve()
curveFit.CopyCoefficients(Destination=neoHookeanPropertyData)
curveFit.Delete()
```

CopyCoefficients

Copies the fitted coefficients from the curve fitting to the property data provided.

Required Arguments

Destination The property data that will receive the coefficients.

Type DataReference (p. 1371)

Example

The following example loads a material with experimental test data and a Neo-Hookean hyperelastic property.

```
template = GetTemplate(TemplateName="EngData")
system = template.CreateSystem()
engineeringData = system.GetContainer(ComponentName="Engineering Data")
neopreneRubber = engineeringData.ReadMaterial(
    Name="Neoprene Rubber",
    Source="Hyperelastic_Materials.xml")
neoHookeanProperty = neopreneRubber.GetProperty(Name="Neo-Hookean")
neoHookeanPropertyData = neoHookeanProperty.GetPropertyData(Name="Neo-Hookean")
curveFit = neoHookeanPropertyData.CreateCurveFitting(
    Type="Neo-Hookean",
    Definition="")
uniaxialProperty = neopreneRubber.GetProperty(Name="Uniaxial Test Data")
curveFit.AddTestData(TestData=uniaxialProperty)
biaxialProperty = neopreneRubber.GetProperty(Name="Biaxial Test Data")
curveFit.AddTestData(TestData=biaxialProperty)
shearProperty = neopreneRubber.GetProperty(Name="Shear Test Data")
curveFit.AddTestData(TestData=shearProperty)
volumetricProperty = neopreneRubber.GetProperty(Name="Volumetric Test Data")
curveFit.AddTestData(TestData=volumetricProperty)
curveFit.RemoveTestData(TestData=uniaxialProperty)
curveFit.AddTestData(TestData=uniaxialProperty)
curveFit.Solve()
curveFit.CopyCoefficients(Destination=neoHookeanPropertyData)
curveFit.Delete()
```

Delete

Deletes a curve fitting.

Example

The following example loads a material with experimental test data and a Neo-Hookean hyperelastic property.

```
template = GetTemplate(TemplateName="EngData")
system = template.CreateSystem()
engineeringData = system.GetContainer(ComponentName="Engineering Data")
neopreneRubber = engineeringData.ReadMaterial(
    Name="Neoprene Rubber",
    Source="Hyperelastic_Materials.xml")
neoHookeanProperty = neopreneRubber.GetProperty(Name="Neo-Hookean")
neoHookeanPropertyData = neoHookeanProperty.GetPropertyData(Name="Neo-Hookean")
curveFit = neoHookeanPropertyData.CreateCurveFitting(
    Type="Neo-Hookean",
    Definition="")
uniaxialProperty = neopreneRubber.GetProperty(Name="Uniaxial Test Data")
curveFit.AddTestData(TestData=uniaxialProperty)
biaxialProperty = neopreneRubber.GetProperty(Name="Biaxial Test Data")
curveFit.AddTestData(TestData=biaxialProperty)
shearProperty = neopreneRubber.GetProperty(Name="Shear Test Data")
curveFit.AddTestData(TestData=shearProperty)
volumetricProperty = neopreneRubber.GetProperty(Name="Volumetric Test Data")
curveFit.AddTestData(TestData=volumetricProperty)
curveFit.RemoveTestData(TestData=uniaxialProperty)
curveFit.AddTestData(TestData=uniaxialProperty)
curveFit.Solve()
curveFit.CopyCoefficients(Destination=neoHookeanPropertyData)
```

```
curveFit.Delete()
```

GetChartData

Returns a generated graph data for the specified source data.

Valid source data are:

Material Property
Material Property Data

Return The graph data for the specified source data.

Type Dictionary (p. 1375)<string (p. 1438), Object (p. 1409)>

RemoveTestData

Removes test data from a given curve fitting.

Required Arguments

TestData The test data property to add.

Type DataReference (p. 1371)

Example

The following example loads a material with experimental test data and a Neo-Hookean hyperelastic property.

```
template = GetTemplate(TemplateName="EngData")
system = template.CreateSystem()
engineeringData = system.GetContainer(ComponentName="Engineering Data")
neopreneRubber = engineeringData.ReadMaterial(
    Name="Neoprene Rubber",
    Source="Hyperelastic_Materials.xml")
neoHookeanProperty = neopreneRubber.GetProperty(Name="Neo-Hookean")
neoHookeanPropertyData = neoHookeanProperty.GetPropertyData(Name="Neo-Hookean")
curveFit = neoHookeanPropertyData.CreateCurveFitting(
    Type="Neo-Hookean",
    Definition="")
uniaxialProperty = neopreneRubber.GetProperty(Name="Uniaxial Test Data")
curveFit.AddTestData(TestData=uniaxialProperty)
biaxialProperty = neopreneRubber.GetProperty(Name="Biaxial Test Data")
curveFit.AddTestData(TestData=biaxialProperty)
shearProperty = neopreneRubber.GetProperty(Name="Shear Test Data")
curveFit.AddTestData(TestData=shearProperty)
volumetricProperty = neopreneRubber.GetProperty(Name="Volumetric Test Data")
curveFit.AddTestData(TestData=volumetricProperty)
curveFit.RemoveTestData(TestData=uniaxialProperty)
curveFit.AddTestData(TestData=uniaxialProperty)
curveFit.Solve()
curveFit.CopyCoefficients(Destination=neoHookeanPropertyData)
curveFit.Delete()
```

Solve

The curve fitting module will solve for the coefficients which most nearly approximate the experimental test data.

Engineering Data Favorite Items

The container used by Engineering Data to maintain favorite items.

Methods

GetMaterial

Returns a material of a given name from a container.

The name matching is case insensitive. If a material is not found an exception is thrown.

Return The material that matches the specified name.

Type DataReference (p. 1371)

Required Arguments

Name The name of the material. This argument is case insensitive.

Type string (p. 1438)

Example

The following example creates a new Engineering Data system and queries the default material, Structural Steel.

```
template = GetTemplate(TemplateName="EngData")
system = template.CreateSystem(Position="Default")
container = system.GetContainer(ComponentName="Engineering Data")
structuralSteel = container.GetMaterial(Name="Structural Steel")
```

GetMaterials

Returns the list of materials in a container. If no materials are in the container, the list is empty.

Return A DataReferenceSet of the materials in the container.

Type DataReferenceSet (p. 1371)

Load

Populates the favorite items container with the user's favorite items.

Returns the favorite items container.

Return The favorite items container.

Type DataContainerReference (p. 1371)

Example

The following example creates a new Engineering Data system and queries the default material, Structural Steel.

```
template = GetTemplate(TemplateName="EngData")
system = template.CreateSystem(Position="Default")
favoritesContainer = EngData.LoadFavoriteItems()
structuralSteel = favoritesContainer.GetMaterial(Name="Structural Steel")
```

Data Entities

Material

The entity to store material information.

Properties

Description

The description of the material.

Type string (p. 1438)

Read Only No

DisplayName

The display name of the material.

Type string (p. 1438)

Read Only No

DisplayText

The general property that defines the user-visible name of an entity. This property is defined for all data entities but is used only in those entities that present a label in the user interface.

Type string (p. 1438)

Read Only No

Source

The path of the material's source file ("*.engd" or "*.xml").

Type string (p. 1438)

Read Only Yes

SourceFileReference

The reference for the material's source file ("*.engd" or "*.xml").

Type DataReference (p. 1371)

Read Only Yes

Methods

AddToFavorites

Adds an item to the Favorites item list.

CreateProperty

Includes a physical quantity or the constitutive relation for the physical response of a material. A property can be visualized as one or more tables of data made up of one or more dependent and independent variables.

The material property is created based on the specified optional parameters "Definition" and "Behavior".

Return Created material properly.

Type DataReference (p. 1371)

Required Arguments

Name The new material property name.

Type string (p. 1438)

Optional Arguments

Behavior The optional string to identify the way in which a new material property will behave.

Behavior of some material properties can be specified in different ways, e.g., Elasticity can be specified as Isotropic, Orthotropic or Anisotropic.

Type string (p. 1438)

CustomData The optional dictionary of custom data. A non-null dictionary designates the material property as custom and not located in the Engineering Data Metadata.

Type Dictionary (p. 1375)<string (p. 1438), string (p. 1438)>

Definition The optional string to identify the way in which new material property will be defined.

Some material properties are defined in different ways, e.g., Thermal Expansion can be defined as Secant Coefficient of Thermal Expansion and Instantaneous Coefficient of Thermal Expansion.

Type string (p. 1438)

Qualifiers The optional dictionary of a qualifier name and it's corresponding value.

Type Dictionary (p. 1375)<string (p. 1438), string (p. 1438)>

Example

The following example creates a new Engineering Data system and adds Coefficient of Thermal Expansion and Elasticity material properties to the default material Structural Steel.

Coefficient of Thermal Expansion is created using Secant definition and Orthotropic behavior. Elasticity is created using Orthotropic behavior.

```
ENGDTemplate = GetTemplate(TemplateName="EngData")
ENGDSystem = ENGDTemplate.CreateSystem(Position="Default")

ENGDContainer = ENGDSystem.GetContainer(ComponentName="Engineering Data")
StructSteel = ENGDContainer.GetMaterial(Name="Structural Steel")

# Create material properties
CTEProperty = StructSteel.CreateProperty(
   Name="Coefficient of Thermal Expansion",
   Definition="Secant",
   Behavior="Orthotropic")
OrthoElasProperty = StructSteel.CreateProperty(
   Name="Elasticity",
   Behavior="Orthotropic")
```

Delete

Deletes the material.

Duplicate

Duplicates the data in this material and returns a new material. The name of the new material will be appended with a numerical value to make it unique.

Return The material data reference of the duplicated material.

Type DataReference (p. 1371)

Required Arguments

TargetContainer The duplicated material will be added to this container.

Type DataContainerReference (p. 1371)

GetFieldVariableDataObject

Gets a FieldVariableDataObject from the specified parent. The parent determines which field variables are tied to the data object

Return DataReference of the field variable.

Type DataReference (p. 1371)

Required Arguments

Name Name of the field variable.

Type string (p. 1438)

Example

```
template1 = GetTemplate(TemplateName="EngData")
system1 = template1.CreateSystem()
engineeringData1 = system1.GetContainer(ComponentName="Engineering Data")
structuralSteel = engineeringData1.GetMaterial(Name="Structural Steel")
temperature = structuralSteel.GetFieldVariableDataObject(Name="Temperature")
```

GetProperty

Returns a material property of a given name from the specified material.

Return The material property that matches the specified name.

Type DataReference (p. 1371)

Required Arguments

Name The name of the material property.

Type string (p. 1438)

Example

The following example creates new Engineering Data system and queries Coefficient of Thermal Expansion property of the default material Structural Steel.

```
ENGDTemplate = GetTemplate(TemplateName="EngData")
ENGDSystem = ENGDTemplate.CreateSystem(Position="Default")

ENGDContainer = ENGDSystem.GetContainer(ComponentName="Engineering Data")
StructSteel = ENGDContainer.GetMaterial(Name="Structural Steel")

# Get material property
CTEProperty = StructSteel.GetProperty(Name="Coefficient of Thermal Expansion")
```

IsSuppressed

Checks if an entity is suppressed.

Valid entities are:

Material
Material Property
Material Property Data

Return Returns true if the entity is suppressed, false otherwise.

Type bool (p. 1360)

IsValid

Validates a material data and provides a message in case of invalid data.

Return The flag that indicates if the material is valid.

Type bool (p. 1360)

Optional Arguments

Message The validation failure message.

Type Output (p. 1411)<string (p. 1438)>

Refresh

This will repopulate the contents of a material with data from a given source. The NameInSource property of a Material will be used to find a match in the source to pull data from. In the event that NameInSource is not set, the DisplayName property will be used instead.

Note: This operation will cause destruction of any data currently in the material.

```
template = GetTemplate(TemplateName="EngData")
system = template.CreateSystem(Position="Default")
container = system.GetContainer(ComponentName="Engineering Data")
structSteel = container.GetMaterial(Name="Structural Steel")
density = structSteel.GetProperty(Name="Density")
density.Delete()
structSteel.UpdateMaterial(Source="General_Materials.xml")
restoredDensity = structSteel.GetProperty(Name="Density")
restoredDensity.SetData(SheetName="", Index=0, Variables=["Density"], Values=[["8000 [kg m^-3]"]])
```

Required Arguments

Source The path of the source file.

Type string (p. 1438)

RemoveFromFavorites

Removes an item from the Favorite items list.

Optional Arguments

Format The format of the file that contains the item.

Type string (p. 1438)

Material A DataReference to the Material to delete.

Type DataReference (p. 1371)

Name The name of the item to delete.

Type string (p. 1438)

Source The location of the file that contains the item on disk.

Type string (p. 1438)

Type The type of the object to delete. This is either a Material or a Mixture.

Type EngineeringDataType (p. 1381)

Example

The following example gets the list of favorites from Engineering Data. It then selects and deletes the "Gray Cast Iron" material from the list.

```
favorites = EngData.LoadFavoriteItems()
matl = favorites.GetMaterial(Name="Gray Cast Iron")
EngData.DeleteFromFavorites(
    Material=matl,
    Type="Material")
```

SetAsDefaultFluidForModel

This will specify the material to use (or not use) for parts in the model which are marked as a fluid.

If the material is in the Engineering Data component it will set or unset the material to be used in the model component of the system(s) that contain this Engineering Data component.

If the material is contained in Favorites it will set or unset the material to use as the default on fluids in the Engineering Data component when a new system is added to the project.

The material is set as the default in Favorites. It will automatically be added to the list of project defaults.

Optional Arguments

Default This Boolean is used to set or unset the material as the default.

Type bool (p. 1360)

Default Value True

SetAsDefaultSolidForModel

This will specify the material to use (or not use) for parts in the model which are marked as a solid.

If the material is in the Engineering Data component it will set or unset the material to be used in the model component of the system(s) that contain this Engineering Data component.

If the material is contained in Favorites it will set or unset the material to use as the default on solids in the Engineering Data component when a new system is added to the project.

The material is set as the default in Favorites. It will automatically be added to the list of project defaults.

Optional Arguments

Default This Boolean is used to set or unset the material as the default.

Type bool (p. 1360)

Default Value True

SetColor

Add a new 'Color' property in a material.

Required Arguments

Blue Blue component of the color assigned to material.

Type int (p. 1394)

Green Green component of the color assigned to material.

Type int (p. 1394)

Red Red component of the color assigned to material.

Type int (p. 1394)

Example

```
template1 = GetTemplate(TemplateName="EngData")
system1 = template1.CreateSystem()
engineeringData1 = system1.GetContainer(ComponentName="Engineering Data")
structuralSteel = engineeringData1.GetMaterial(Name="Structural Steel")
structuralSteel.SetColor(Material="strucutralSteel", Red="255", Green="0", Blue="0")
```

SetOption

Set option for an "Option" variable

Required Arguments

OptionName Option to set

Type string (p. 1438)

OptionValue Value for option

Type string (p. 1438)

SetSuppression

Suppresses or Unsuppresses item.

Item can be suppressed to prevent it from being sent to a downstream cell in the system.

Following items can be suppressed:

Material

Material property

Material property data

Required Arguments

Suppressed The flag to specify if the item should be suppressed or unsuppressed.

Type bool (p. 1360)

Unlink

Unlinks a material from its underlying source. Once this occurs, the material can no longer be restored to its original state and the material will no longer have a source.

MaterialProperty

The entity to store material property information.

A material property is the identifier for the singular information (for example, Density) that together with other properties defines or models the behavior of the material. A property is always defined by at least one table (tabular data), which could be singular. Some properties can contain a collection of tabular data (for example, Isotropic Elasticity).

Properties

Behavior

The string that defines the way in which the material property behaves, e.g., Elasticity has Isotropic, Orthotropic or Anisotropic behavior.

Type string (p. 1438)

Read Only No

Definition

The definition of the material property. Some material properties are defined in different ways, e.g., Thermal Expansion can be defined as Secant Coefficient of Thermal Expansion and Instantaneous Coefficient of Thermal Expansion.

Type string (p. 1438)

Read Only No

Description

The description of the material property.

Type string (p. 1438)

Read Only No

DisplayText

The general property that defines the user-visible name of an entity. This property is defined for all data entities but is used only in those entities that present a label in the user interface.

Type string (p. 1438)

Read Only No

PropertyDataColl

The collection of tabular data that defines the material property.

Type DataReferenceSet (p. 1371)

Read Only Yes

TypeName

The name of the material property.

Type string (p. 1438)

Read Only No

Methods

AddTestData

Includes experimental test data for response function calculations.

Required Arguments

TestData The test data property to add.

Type DataReference (p. 1371)

BeginBatchUpdate

Marks the start of a series of data modifications to a table of data, to improve performance.

CreatePropertyData

Include an additional property data for a material property. Some properties may have more than one property data to describe the material property.

The preferred method of adding a material property data is to use CreateMaterialProperty.

Return The new material properly data that was created.

Type DataReference (p. 1371)

Required Arguments

Name The new material property data name.

Type string (p. 1438)

Optional Arguments

Behavior A string to identify how the new material property data will behave.

The behavior of some material properties can be specified in different ways, e.g., Elasticity can be specified as Isotropic, Orthotropic or Anisotropic.

Type string (p. 1438)

CustomData The optional dictionary of custom data. A non-null dictionary designates the material property as custom and not located in the Engineering Data Metadata.

Type Dictionary (p. 1375)<string (p. 1438), string (p. 1438)>

Definition A string to identify how the new material property data will be defined.

Some material properties are defined in different ways, e.g., Thermal Expansion can be defined as Secant Coefficient of Thermal Expansion and Instantaneous Coefficient of Thermal Expansion.

Type string (p. 1438)

Qualifiers The optional dictionary of a qualifier name and it's corresponding value.

Type Dictionary (p. 1375)<string (p. 1438), string (p. 1438)>

Example

The following example creates the material property data Orthotropic Secant Coefficient of Thermal Expansion on the material property Coefficient of Thermal Expansion. This example assumes the material Structural Steel has been obtained from the General Materials library.

Get the material property we are going to create a new material property data on.

```
thermExpansionMatProp = structuralSteel.GetMaterialProperty(Name="Coefficient of Thermal Expansion")
```

Create the new material property data.

```
orthoSecantThermExpansionMatPropData = thermExpansionMatProp.CreatePropertyData(
    Name="Coefficient of Thermal Expansion",
    Definition="Secant",
    Behavior="Orthotropic")
```

Delete

Deletes the material property.

Optional Arguments

Behavior The optional string to specify the material property behavior.

Behavior of some material properties can be specified in different ways e.g. Elasticity can be specified as Isotropic, Orthotropic or Anisotropic.

Type string (p. 1438)

Definition The optional string to specify the material property definition.

Some material properties are defined in different ways e.g. Thermal Expansion can be defined as Secant Coefficient of Thermal Expansion and Instantaneous Coefficient of Thermal Expansion.

Type string (p. 1438)

DeleteData

Delete a row from a tabular data sheet.

Required Arguments

Index Index of the row to delete.

Type int (p. 1394)

Optional Arguments

SheetName Name of the sheet to access.

Type string (p. 1438)

SheetQualifiers SheetQualifiers is used to pass in the qualifiers to select between multiple sheets witht the same name. This is a dictionary of the Qualifier and its Value.

Type Dictionary (p. 1375)<string (p. 1438), string (p. 1438)>

Example

The following example illustrates the deletion of a row from a tabular data sheet.

```
#
# Create a new Engineering Data System and access Structural Steel
#
template1 = GetTemplate(TemplateName="EngData")
system1 = template1.CreateSystem()
engineeringData1 = system1.GetContainer(ComponentName="Engineering Data")
matl1 = engineeringData1.GetMaterial(Name="Structural Steel")
#
# Delete the first row in the Density property
#
matlProp1 = matl1.GetProperty(Name="Density")
DeleteTabularDataRow(matlProp1, Index = 0)
#
# Delete the first row in the Coefficient of Thermal Expansion property with
# optional SheeetName and SheetQualifiers
#
matlProp2 = matl1.GetProperty(Name="Coefficient of Thermal Expansion")
DeleteTabularDataRow(matlProp2,
                  SheetName="Coefficient of Thermal Expansion",
                  SheetQualifiers={"Definition Method": "Secant", "Behavior": "Isotropic"},
```

```
                        Index = 0)
```

EndBatchUpdate

Marks the completion of a series of data modifications.

GetChartData

Returns a generated graph data for the specified source data.

Valid source data are:

Material Property
Material Property Data

Return The graph data for the specified source data.

Type Dictionary (p. 1375)<string (p. 1438), Object (p. 1409)>

GetData

Returns the tabular data associated with the data entity.

Return The returned data in scalar, list, or dictionary format.

Type Object (p. 1409)

Optional Arguments

AsDictionary If set to true, the data will be returned as a dictionary where the keys are variable names and the values are the data for each variable. If set to false, the data will be returned in scalar or list format without the variable names.

Type bool (p. 1360)

Default Value False

ColumnMajor If set to true, the data will be returned in column-major order. If set to false, the data will be returned in row-major order.

Type bool (p. 1360)

Default Value True

EndIndex The end index for requesting a subset of the data (zero-based).

Type int (p. 1394)

Default Value -2147483647

SheetName Specifies the sheet name when the data contains multiple sheets.

Type string (p. 1438)

SheetQualifiers Used to pass in the qualifiers to select between multiple sheets with the same name. This is a dictionary of qualifiers and values.

Type Dictionary (p. 1375)<string (p. 1438), string (p. 1438)>

StartIndex The start index for requesting a subset of the data (zero-based).

Type int (p. 1394)

Default Value 0

Variables Names of the variables for which data is requested (string or list of strings).

Type Object (p. 1409)

Example

In this example, all data is requested for the given tabular data entity.

```
tabData1.GetData()
```

In this example, all data is requested in row-major order.

```
tabData1.GetData(ColumnMajor=False)
```

In this example, all data is requested in dictionary format.

```
tabData1.GetData(AsDictionary=True)
```

In this example, data for variables Density and Temperature is requested in dictionary format.

```
tabData1.GetData(Variables=["Density", "Temperature"], AsDictionary=True)
```

GetPropertyData

Returns a property data of the specified material property.

The property data returned is based on specified optional parameters "Definition" and "Behavior".

Return The material property data that matches specified type name, definition and behavior.

Type DataReference (p. 1371)

Required Arguments

Name The material property data type name.

Type string (p. 1438)

Optional Arguments

Behavior The optional string to specify the material property data behavior.

Behavior of some material properties can be specified in different ways, e.g., Elasticity can be specified as Isotropic, Orthotropic or Anisotropic.

Type string (p. 1438)

Definition The optional string to specify the material property data definition.

Some material properties are defined in different ways, e.g., Thermal Expansion can be defined as Secant Coefficient of Thermal Expansion and Instantaneous Coefficient of Thermal Expansion.

Type string (p. 1438)

Qualifiers The optional dictionary of a qualifier name and it's corresponding value.

Type Dictionary (p. 1375)<string (p. 1438), string (p. 1438)>

Example

The following example creates new Engineering Data system and queries Coefficient of Thermal Expansion property data of the default material Structural Steel.

```
ENGDTemplate = GetTemplate(TemplateName="EngData")
ENGDSystem = ENGDTemplate.CreateSystem(Position="Default")

ENGDContainer = ENGDSystem.GetContainer(ComponentName="Engineering Data")
StructSteel = ENGDContainer.GetMaterial(Name="Structural Steel")

#Get material property
CTEProperty = StructSteel.GetProperty(Name="Coefficient of Thermal Expansion")
# Get material property data
CTEPropData = CTEProperty.GetPropertyData(
   Name="Coefficient of Thermal Expansion",
   Definition="Secant",
   Behavior="Isotropic")
RefTempPropData = CTEProperty.GetPropertyData(
   Name="Reference Temperature",
   Definition="Secant",
   Behavior="Isotropic")
```

IsSuppressed

Checks if an entity is suppressed.

Valid entities are:

Material
Material Property
Material Property Data

Return Returns true if the entity is suppressed, false otherwise.

Type bool (p. 1360)

IsValid

Validates a material property and provides a message in case of invalid data.

Return The flag that indicates if the material property is valid.

Type bool (p. 1360)

Required Arguments

Material The parent material of the property.

Type DataReference (p. 1371)

Optional Arguments

Message The validation failure message.

Type Output (p. 1411)<string (p. 1438)>

RemoveTestData

Excludes experimental test data for response function calculations.

Required Arguments

TestData The test data property to add.

Type DataReference (p. 1371)

SetData

Set tabular data associated with the data entity.

Optional Arguments

Data Sets the data using a dictionary form. The keys are the variable names and the values are the data. The use of this argument is mutually exclusive with "Values" and "Variables".

Type Dictionary (p. 1375)<string (p. 1438), List (p. 1400)<Object (p. 1409)>>

Index Specifies the starting location used to set the data (zero-based). A value of -1 indicates that the data should be appended to the existing data.

Type int (p. 1394)

Default Value 0

SheetName Specifies the sheet name when the data contains multiple sheets.

Type string (p. 1438)

SheetQualifiers Used to pass in the qualifiers to select between multiple sheets with the same name. This is a dictionary of qualifiers and values.

Type Dictionary (p. 1375)<string (p. 1438), string (p. 1438)>

Values List of data values set in conjunction with the "Variables" parameter. This parameter and the "Data" parameter are mutually exclusive.

Type List (p. 1400)<List (p. 1400)<Object (p. 1409)>>

Variables Names of the variables for which data is being set. This parameter and and the "Data" parameter are mutually exclusive.

Type List (p. 1400)<string (p. 1438)>

Example

```
#
# Create a new Engineering Data System and access Structural Steel
#
template1 = GetTemplate(TemplateName="EngData")
system1 = template1.CreateSystem()
engineeringData1 = system1.GetContainer(ComponentName="Engineering Data")
matl1 = engineeringData1.GetMaterial(Name="Structural Steel")
#
# Change the value of a simple single-valued property
#
matlProp1 = matl1.GetProperty(Name="Density")
SetTabularData(matlProp1,
                  Variables="Density",
                  Values="8500 [kg m^-3]")
#
# Set Temperature-dependent data for Elasticity based
# on lists of variables and values.
matlProp2 = matl1.GetProperty(Name="Elasticity")
temperature = ["400 [K]", "600 [K]", "800 [K]"]
E = ["2e5 [MPa]", "1.9e5 [MPa]", "1.6e5 [MPa]"]
SetTabularData(matlProp2,
                  Variables = ["Temperature","Young's Modulus"],
                  Values = [temperature, E])
#
# Change the Temperature for the second table entry.
#
SetTabularData(matlProp2,
                  Index = 1,
                  Variables = "Temperature",
                  Values = "625 [K]")
#
# Set a list for Poisson's Ratio starting at the second table entry.
#
SetTabularData(matlProp2,
                  Index = 1,
                  Variables = "Poisson's Ratio",
                  Values = [0.3, 0.3])
#
# Set Temperature-dependent property data for the Coefficient of Thermal Expansion
# using a dictionary. The dictionary key is the Variable name,
# followed by the list of values for the variable.
#
matlProp3 = matl1.GetProperty(Name="Coefficient of Thermal Expansion")
newData = {"Temperature": ["200 [F]","400 [F]","600 [F]","800 [F]","1000 [F]"],
"Coefficient of Thermal Expansion" : ["6.3e-6 [F^-1]", "7.0e-6 [F^-1]",
                                      "7.46e-6 [F^-1]", "7.8e-6 [F^-1]",
                                      "8.04e-6 [F^-1]"]}
SetTabularData(matlProp3,
                  SheetName="Coefficient of Thermal Expansion",
                  SheetQualifiers={"Definition Method": "Secant", "Behavior": "Isotropic"},
                  Data = newData)
```

SetQualifier

Changes the values of a specifiec qualifier in a data table.

Required Arguments

Qualifier The Qualifier to Set.

Type string (p. 1438)

Value The new value.

Type string (p. 1438)

Optional Arguments

SheetName The name of the tabular data sheet that contains the qualifier.

Type string (p. 1438)

SheetQualifiers SheetQualifiers can be used to pass in the qualifiers to select between multiple sheets witht the same name. This is a dictionary of the Qualifier and its Value.

Type Dictionary (p. 1375)<string (p. 1438), string (p. 1438)>

VariableName The name of the Variable that contains the qualifier to be changed.

Type string (p. 1438)

VariableQualifiers VariableQualifiers can used to pass in the qualifiers to select between multiple variables witht the same name. This is a dictionary of the Qualifier and its Value.

Type Dictionary (p. 1375)<string (p. 1438), string (p. 1438)>

Example

The following example changes the 'Derive From' setting within an Isotropic Elasticity material property to be "Bulk Modulus and Poisson's Ratio".

```
matl1 = engineeringData1.GetMaterial(Name="Structural Steel")
matlProp1 = matl1.GetProperty(Name="Elasticity")
SetTabularDataQualifier(matlProp1,
   SheetName="Elasticity",
   Qualifier="Derive from",
   Value="Bulk Modulus and Poisson's Ratio")
```

SetSuppression

Suppresses or Unsuppresses item.

Item can be suppressed to prevent it from being sent to a downstream cell in the system.

Following items can be suppressed:

Material

Material property

Material property data

Required Arguments

Suppressed The flag to specify if the item should be suppressed or unsuppressed.

Type bool (p. 1360)

MaterialPropertyData

The entity to store material property (tabular) data information. The material property data is a collection of material variable data.

Properties

Behavior

The behavior of the material variable tabular data. Some material properties can have different behavior, e.g., Elasticity has Isotropic, Orthotropic or Anisotropic behavior.

Type string (p. 1438)

Read Only No

Definition

The definition of the material variable tabular data. Some material properties are defined in different ways, e.g., Thermal Expansion can be defined as Secant Coefficient of Thermal Expansion and Instantaneous Coefficient of Thermal Expansion.

Type string (p. 1438)

Read Only No

DependentColl

The collection of dependent variables in the material variable tabular data, e.g., Density.

Type DataReferenceSet (p. 1371)

Read Only Yes

Description

The description of the material variable tabular data.

Type string (p. 1438)

Read Only No

DisplayText

The general property that defines the user-visible name of an entity. This property is defined for all data entities but is used only in those entities that present a label in the user interface.

Type string (p. 1438)

Read Only No

IndependentColl

The collection of independent variables in the material variable tabular data, e.g., Temperature.

Type DataReferenceSet (p. 1371)

Read Only Yes

PrimaryIndependent

The primary independent variable in the material variable tabular data, e.g., Temperature.

Type DataReference (p. 1371)

Read Only Yes

RowsCount

The number of data values for a variable in the material variable tabular data.

Type int (p. 1394)

Read Only Yes

TypeName

The name of the material variable tabular data.

Type string (p. 1438)

Read Only No

VariableColl

The collection of variables in the material variable tabular data.

Type DataReferenceSet (p. 1371)

Read Only Yes

Methods

BeginBatchUpdate

Marks the start of a series of data modifications to a table of data, to improve performance.

CreateCurveFitting

Creates a curve fitting for a given property data.

Return The curve fitting.

Type DataReference (p. 1371)

Required Arguments

Definition The definition of curve fitting to create. This must be a definition that is supported by an engineering data curve fitting.

i.e. 1 Parameter, 2 Parameter, 1st Order, 2nd Order

Type string (p. 1438)

Type The type of curve fitting to create. This must be a type that is supported by an engineering data curve fitting.

i.e. Neo-Hookean, Mooney-Rivlin, Ogden, Yeoh, Polynomial

Type string (p. 1438)

Example

The following example loads a material with experimental test data and a Neo-Hookean hyperelastic property.

```
template = GetTemplate(TemplateName="EngData")
system = template.CreateSystem()
engineeringData = system.GetContainer(ComponentName="Engineering Data")
neopreneRubber = engineeringData.ReadMaterial(
    Name="Neoprene Rubber",
    Source="Hyperelastic_Materials.xml")
neoHookeanProperty = neopreneRubber.GetProperty(Name="Neo-Hookean")
neoHookeanPropertyData = neoHookeanProperty.GetPropertyData(Name="Neo-Hookean")
curveFit = neoHookeanPropertyData.CreateCurveFitting(
    Type="Neo-Hookean",
    Definition="")
uniaxialProperty = neopreneRubber.GetProperty(Name="Uniaxial Test Data")
curveFit.AddTestData(TestData=uniaxialProperty)
biaxialProperty = neopreneRubber.GetProperty(Name="Biaxial Test Data")
curveFit.AddTestData(TestData=biaxialProperty)
shearProperty = neopreneRubber.GetProperty(Name="Shear Test Data")
curveFit.AddTestData(TestData=shearProperty)
volumetricProperty = neopreneRubber.GetProperty(Name="Volumetric Test Data")
curveFit.AddTestData(TestData=volumetricProperty)
curveFit.RemoveTestData(TestData=uniaxialProperty)
curveFit.AddTestData(TestData=uniaxialProperty)
curveFit.Solve()
curveFit.CopyCoefficients(Destination=neoHookeanPropertyData)
curveFit.Delete()
```

CreateDataProvider

Create a data provider for a give format. The only data provider currently supported is for Delimited Text (used to import tabular data from a delimited data file).

Return DataReference of the DelimitedDataObject

Type DataReference (p. 1371)

Required Arguments

Format Format of provider (only supported option is "Delimited Text")

Type string (p. 1438)

Example

```
material1 = engineeringData1.GetMaterial(Name="Structural Steel")
matlProp1 = material1.GetProperty(Name="Coefficient of Thermal Expansion")
materialPropertyData1 = matlProp1.GetPropertyData(
Name="Coefficient of Thermal Expansion",
Qualifiers={"Definition": "Secant", "Behavior": "Isotropic"})
dataProvider1 = materialPropertyData1.CreateDataProvider(Format="Delimited Text")
dataProvider1.FileName = r"C:\Coefficient_of_thermal_expansion.csv"
dataProvider1.ReadLine = 2
dataProvider1.Columns = [1, 2]
dataProvider1.Delimiter = ","
dataProvider1.VariableNames = ["Temperature", "Coefficient of Thermal Expansion"]
dataProvider1.VariableUnits = ["C", "C^-1"]
dataProvider1.Import()
```

CreateVariable

Include an additional variable in the property data for a material property.

Return The new variable data that was created.

Type DataReference (p. 1371)

Required Arguments

Name The new variable data name.

Type string (p. 1438)

Optional Arguments

CustomData The optional dictionary of custom data. A non-null dictionary designates the material property as custom and not located in the Engineering Data Metadata.

Type Dictionary (p. 1375)<string (p. 1438), string (p. 1438)>

Qualifiers The optional dictionary of a qualifier name and it's corresponding value.

Type Dictionary (p. 1375)<string (p. 1438), string (p. 1438)>

Example

```
material1 = engineeringData1.GetMaterial(Name="Structural Steel")
matlProp1 = material1.CreateProperty(
    Name="Custom Model",
    Qualifiers={"UserMat": "USER"},
    CustomData={})
matlPropData1 = matlProp2.CreatePropertyData(
    Name="Model State Variables",
    Qualifiers={"UserMat": "STATE"},
    CustomData={})
PropertyDataVariable1 = matlPropData1.CreateVariable(
    Name="User Variable",
    Qualifiers={"Display": "True", "UserMat Constant": "1"},
    CustomData={"Quantity Type": "Dimensionless", "Data": "0", "Independent": "False"})
```

Delete

Deletes the material property data.

EndBatchUpdate

Marks the completion of a series of data modifications.

GetChartData

Returns a generated graph data for the specified source data.

Valid source data are:

Material Property
Material Property Data

Return The graph data for the specified source data.

Type Dictionary (p. 1375)<string (p. 1438), Object (p. 1409)>

GetCurveFitting

Query to return the DataReference to the CurveFit used by some PropertyData.

Return CurveFit DataReference

Type DataReference (p. 1371)

GetFieldVariableDataObject

Gets a FieldVariableDataObject from the specified parent. The parent determines which field variables are tied to the data object

Return DataReference of the field variable.

Type DataReference (p. 1371)

Required Arguments

Name Name of the field variable.

Type string (p. 1438)

Example

```
template1 = GetTemplate(TemplateName="EngData")
system1 = template1.CreateSystem()
engineeringData1 = system1.GetContainer(ComponentName="Engineering Data")
structuralSteel = engineeringData1.GetMaterial(Name="Structural Steel")
temperature = structuralSteel.GetFieldVariableDataObject(Name="Temperature")
```

GetVariable

Returns a material variable of a given name from a material property data.

Return The requested variable.

Type DataReference (p. 1371)

Required Arguments

Name The name of the variable.

Type string (p. 1438)

Example

The following example creates new Engineering Data system and queries Coefficient of Thermal Expansion property data of the default material Structural Steel.

```
ENGDTemplate = GetTemplate(TemplateName="EngData")
ENGDSystem = ENGDTemplate.CreateSystem(Position="Default")

ENGDContainer = ENGDSystem.GetContainer(ComponentName="Engineering Data")
StructSteel = ENGDContainer.GetMaterial(Name="Structural Steel")

# Get material
CTEProperty = StructSteel.GetProperty(Name="Coefficient of Thermal Expansion")
# Get material property data
CTEPropData = CTEProperty.GetPropertyData(
   Name="Coefficient of Thermal Expansion",
   Definition="Secant",
   Behavior="Isotropic")
# Get material variable
CTEVariable = CTEPropData.GetVariable(
   Name="Coefficient of Thermal Expansion")
```

IsSuppressed

Checks if an entity is suppressed.

Valid entities are:

Material
Material Property
Material Property Data

Return Returns true if the entity is suppressed, false otherwise.

Type bool (p. 1360)

IsValid

Validates a material property data and provides a message in case of invalid data.

Return The flag that indicates if the material property data is valid.

Type bool (p. 1360)

Required Arguments

Material The parent material of the material property data.

Type DataReference (p. 1371)

Optional Arguments

Message The validation failure message.

Type Output (p. 1411)<string (p. 1438)>

SetSuppression

Suppresses or Unsuppresses item.

Item can be suppressed to prevent it from being sent to a downstream cell in the system.

Following items can be suppressed:

Material

Material property

Material property data

Required Arguments

Suppressed The flag to specify if the item should be suppressed or unsuppressed.

Type bool (p. 1360)

Engineering Data Favorite Library

The container used by Engineering Data to allow working with a library.

Methods

Close

Closes an Engineering Data library.

ConsolidateMaterials

Deletes the material.

CreateMaterial

Adds a new material to the container.

Return The material data reference of the new material.

Type DataReference (p. 1371)

Required Arguments

Name The name to be used for this material.

Type string (p. 1438)

Export

Exports engineering data to the specified file.

The following type of file format is supported for export:

MatML 3.1 schema for Material(s)

Required Arguments

FilePath The target path for the Engineering Data file ("*.xml").

Type string (p. 1438)

Format The file format in which engineering data will be exported.

Type string (p. 1438)

UnitSystem The unit system in which engineering data will be exported.

Type string (p. 1438)

Optional Arguments

ApplyScaleOffset The flag to specify if the scale factor and offset value will be applied during export.

Type bool (p. 1360)

Default Value False

IgnoreSuppressed The flag to specify if suppressed engineering data will be ignored during export.

Type bool (p. 1360)

Default Value False

OverwriteTarget The flag to specify if the target file will be overwritten.

Type bool (p. 1360)

Default Value False

ReplaceMaterial The flag to specify if the earlier material data will be replaced in case of similar material names.

Type bool (p. 1360)

Default Value False

Example

```
template = GetTemplate(TemplateName="EngData") system = template.CreateSystem(Position="Default") container = system.GetContainer(ComponentName="Engineering Data") materials = container.GetMaterials() EngData.WriteMaterials(MaterialList=materials, FilePath=r"d:\data\OutputMaterials.xml", Format="MatML31", UnitSystem="MKS_STANDARD", ReplaceMaterial=true)
```

GetMaterial

Returns a material of a given name from a container.

The name matching is case insensitive. If a material is not found an exception is thrown.

Return The material that matches the specified name.

Type DataReference (p. 1371)

Required Arguments

Name The name of the material. This argument is case insensitive.

Type string (p. 1438)

Example

The following example creates a new Engineering Data system and queries the default material, Structural Steel.

```
template = GetTemplate(TemplateName="EngData")
system = template.CreateSystem(Position="Default")
container = system.GetContainer(ComponentName="Engineering Data")
structuralSteel = container.GetMaterial(Name="Structural Steel")
```

GetMaterials

Returns the list of materials in a container. If no materials are in the container, the list is empty.

Return A DataReferenceSet of the materials in the container.

Type DataReferenceSet (p. 1371)

Import

Imports engineering data into an existing source from a specified source.

The following types of files are supported for import:

Engineering Data libraries exported from Workbench 9.0 to 11.0 SP1
Material(s) file following the MatML 3.1 schema
Material(s) file generated by AUTODYN

Required Arguments

Source The source which contains the materials.

Type string (p. 1438)

ImportMaterial

Reads the data for a single material into the requested container.

Return A DataReference for the material that was read.

Type DataReference (p. 1371)

Required Arguments

Name The name of the material to read from the source.

Type string (p. 1438)

Source The source which contains the requested material.

Type string (p. 1438)

Example

This code shows how to read the Copper Alloy material from the provided samples, into Engineering Data to use in an analysis.

```
installDir = r"C:\Program Files\ANSYS Inc\v121"
matl1 = engineeringData1.ReadMaterial(
   Name="Copper Alloy",
   Source=installDir+r"\Addins\EngineeringData\Samples\General_Materials.xml")
```

Refresh

This will repopulate the contents of an Engineering Data library with data from its source.

Save

Saves an Engineering Data library.

Optional Arguments

FilePath The optional string to specify the path if the Engineering Data library should be saved at different location.

Type string (p. 1438)

Data Entities

DelimitedDataObject

Entity to store the necessary information for importing delimited data.

Properties

Columns

List of columns to read in. Leave this unset to import all columns.

```
Columns=[3,5]
```

Type List (p. 1400)<int (p. 1394)>

Read Only No

Delimiter

Delimiter type.

```
Delimiter=","
```

Type string (p. 1438)

Read Only No

DisplayText

The general property that defines the user-visible name of an entity. This property is defined for all data entities but is used only in those entities that present a label in the user interface.

Type string (p. 1438)

Read Only No

FileName

Name of file to import. File can have one of two formats:

Format 1 - File contains variable names and units # This information is the experimental data collected from a uniaxial test on a elastomer sample # and was lifted from Hyperelastic Materials sample Elastomer Sample (Mooney-Rivlin) # # Set "Start Import at Line" to 6 Strain, Stress # Variable Names (these correspond to the names shown in the UI) m m^-1, Pa # Unit (these correspond to the unit strings shown in the UI) 0.1338, 494.1474492 # Data point 1 0.2675, 912.7972764 # Data point 2 0.3567, 1172.453938 # Data point 3

Format 2 - File does not contain variable names and units, these are passed in by using the # VariableNames and VariableUnits parameters # This information is the experimental data collected from a uniaxial test on a elastomer sample # and was lifted from Hyperelastic Materials sample Elastomer Sample (Mooney-Rivlin) # # Set "Start Import at Line" to 7 0.1338, 494.1474492 # Data point 1 0.2675, 912.7972764 # Data point 2 0.3567, 1172.453938 # Data point 3

Type string (p. 1438)

Read Only No

ReadLine

Line to start importing. Use this if your file has a header you wish to ignore. The line number you set here must correspond to the row in your file with the variable names.

```
    BeginAtLine = 1 -- if the variable names first line of your file.
```

Type int (p. 1394)

Read Only No

VariableNames

List of variable names, the list must have the same number of names as columns in the data. If the list is null, it is assumed that the variable names are located in the file at line "BeginAtLine"

Type List (p. 1400)<string (p. 1438)>

Read Only No

VariableUnits

List of variable units, the list must have the same number of units as columns in the data. If the list is null, it is assumed that the variable units are located in the file at line "BeginAtLine" + 1

Type List (p. 1400)<string (p. 1438)>

Read Only No

Methods

Import

Execute import, and appends delimited data to the MaterialProperty Data. Call this method after setting the following properties of the DelimitedDataObject: FileName, Delimiter, Readline (optional), Columns (optional).

Material

The entity to store material information.

Properties

Description

The description of the material.

Type string (p. 1438)

Read Only No

DisplayName

The display name of the material.

Type string (p. 1438)

Read Only No

DisplayText

The general property that defines the user-visible name of an entity. This property is defined for all data entities but is used only in those entities that present a label in the user interface.

Type string (p. 1438)

Read Only No

Source

The path of the material's source file ("*.engd" or "*.xml").

Type string (p. 1438)

Read Only Yes

SourceFileReference

The reference for the material's source file ("*.engd" or "*.xml").

Type DataReference (p. 1371)

Read Only Yes

Methods

AddToFavorites

Adds an item to the Favorites item list.

CreateProperty

Includes a physical quantity or the constitutive relation for the physical response of a material. A property can be visualized as one or more tables of data made up of one or more dependent and independent variables.

The material property is created based on the specified optional parameters "Definition" and "Behavior".

Return Created material properly.

Type DataReference (p. 1371)

Required Arguments

Name The new material property name.

Type string (p. 1438)

Optional Arguments

Behavior The optional string to identify the way in which a new material property will behave.

Behavior of some material properties can be specified in different ways, e.g., Elasticity can be specified as Isotropic, Orthotropic or Anisotropic.

Type string (p. 1438)

CustomData The optional dictionary of custom data. A non-null dictionary designates the material property as custom and not located in the Engineering Data Metadata.

Type Dictionary (p. 1375)<string (p. 1438), string (p. 1438)>

Definition The optional string to identify the way in which new material property will be defined.

Some material properties are defined in different ways, e.g., Thermal Expansion can be defined as Secant Coefficient of Thermal Expansion and Instantaneous Coefficient of Thermal Expansion.

Type string (p. 1438)

Qualifiers The optional dictionary of a qualifier name and it's corresponding value.

Type Dictionary (p. 1375)<string (p. 1438), string (p. 1438)>

Example

The following example creates a new Engineering Data system and adds Coefficient of Thermal Expansion and Elasticity material properties to the default material Structural Steel.

Coefficient of Thermal Expansion is created using Secant definition and Orthotropic behavior. Elasticity is created using Orthotropic behavior.

```
ENGDTemplate = GetTemplate(TemplateName="EngData")
ENGDSystem = ENGDTemplate.CreateSystem(Position="Default")

ENGDContainer = ENGDSystem.GetContainer(ComponentName="Engineering Data")
StructSteel = ENGDContainer.GetMaterial(Name="Structural Steel")

# Create material properties
CTEProperty = StructSteel.CreateProperty(
   Name="Coefficient of Thermal Expansion",
   Definition="Secant",
   Behavior="Orthotropic")
OrthoElasProperty = StructSteel.CreateProperty(
   Name="Elasticity",
   Behavior="Orthotropic")
```

Delete

Deletes the material.

Duplicate

Duplicates the data in this material and returns a new material. The name of the new material will be appended with a numerical value to make it unique.

Return The material data reference of the duplicated material.

Type DataReference (p. 1371)

Required Arguments

TargetContainer The duplicated material will be added to this container.

Type DataContainerReference (p. 1371)

GetFieldVariableDataObject

Gets a FieldVariableDataObject from the specified parent. The parent determines which field variables are tied to the data object

Return DataReference of the field variable.

Type DataReference (p. 1371)

Required Arguments

Name Name of the field variable.

Type string (p. 1438)

Example

```
template1 = GetTemplate(TemplateName="EngData")
system1 = template1.CreateSystem()
engineeringData1 = system1.GetContainer(ComponentName="Engineering Data")
structuralSteel = engineeringData1.GetMaterial(Name="Structural Steel")
temperature = structuralSteel.GetFieldVariableDataObject(Name="Temperature")
```

GetProperty

Returns a material property of a given name from the specified material.

Return The material property that matches the specified name.

Type DataReference (p. 1371)

Required Arguments

Name The name of the material property.

Type string (p. 1438)

Example

The following example creates new Engineering Data system and queries Coefficient of Thermal Expansion property of the default material Structural Steel.

```
ENGDTemplate = GetTemplate(TemplateName="EngData")
ENGDSystem = ENGDTemplate.CreateSystem(Position="Default")

ENGDContainer = ENGDSystem.GetContainer(ComponentName="Engineering Data")
StructSteel = ENGDContainer.GetMaterial(Name="Structural Steel")

# Get material property
CTEProperty = StructSteel.GetProperty(Name="Coefficient of Thermal Expansion")
```

IsSuppressed

Checks if an entity is suppressed.

Valid entities are:

Material
Material Property
Material Property Data

Return Returns true if the entity is suppressed, false otherwise.

Type bool (p. 1360)

IsValid

Validates a material data and provides a message in case of invalid data.

Return The flag that indicates if the material is valid.

Type bool (p. 1360)

Optional Arguments

Message The validation failure message.

Type Output (p. 1411)<string (p. 1438)>

Refresh

This will repopulate the contents of a material with data from a given source. The NameInSource property of a Material will be used to find a match in the source to pull data from. In the event that NameInSource is not set, the DisplayName property will be used instead.

Note: This operation will cause destruction of any data currently in the material.

```
template = GetTemplate(TemplateName="EngData")
system = template.CreateSystem(Position="Default")
container = system.GetContainer(ComponentName="Engineering Data")
structSteel = container.GetMaterial(Name="Structural Steel")
density = structSteel.GetProperty(Name="Density")
density.Delete()
structSteel.UpdateMaterial(Source="General_Materials.xml")
restoredDensity = structSteel.GetProperty(Name="Density")
restoredDensity.SetData(SheetName="", Index=0, Variables=["Density"], Values=[["8000 [kg m^-3]"]])
```

Required Arguments

Source The path of the source file.

Type string (p. 1438)

RemoveFromFavorites

Removes an item from the Favorite items list.

Optional Arguments

Format The format of the file that contains the item.

Type string (p. 1438)

Material A DataReference to the Material to delete.

Type DataReference (p. 1371)

Name The name of the item to delete.

Type string (p. 1438)

Source The location of the file that contains the item on disk.

Type string (p. 1438)

Type The type of the object to delete. This is either a Material or a Mixture.

Type EngineeringDataType (p. 1381)

Example

The following example gets the list of favorites from Engineering Data. It then selects and deletes the "Gray Cast Iron" material from the list.

```
favorites = EngData.LoadFavoriteItems()
matl = favorites.GetMaterial(Name="Gray Cast Iron")
EngData.DeleteFromFavorites(
    Material=matl,
    Type="Material")
```

SetAsDefaultFluidForModel

This will specify the material to use (or not use) for parts in the model which are marked as a fluid.

If the material is in the Engineering Data component it will set or unset the material to be used in the model component of the system(s) that contain this Engineering Data component.

If the material is contained in Favorites it will set or unset the material to use as the default on fluids in the Engineering Data component when a new system is added to the project.

The material is set as the default in Favorites. It will automatically be added to the list of project defaults.

Optional Arguments

Default This Boolean is used to set or unset the material as the default.

Type bool (p. 1360)

Default Value True

SetAsDefaultSolidForModel

This will specify the material to use (or not use) for parts in the model which are marked as a solid.

If the material is in the Engineering Data component it will set or unset the material to be used in the model component of the system(s) that contain this Engineering Data component.

If the material is contained in Favorites it will set or unset the material to use as the default on solids in the Engineering Data component when a new system is added to the project.

The material is set as the default in Favorites. It will automatically be added to the list of project defaults.

Optional Arguments

Default This Boolean is used to set or unset the material as the default.

Type bool (p. 1360)

Default Value True

SetColor

Add a new 'Color' property in a material.

Required Arguments

Blue Blue component of the color assigned to material.

Type int (p. 1394)

Green Green component of the color assigned to material.

Type int (p. 1394)

Red Red component of the color assigned to material.

Type int (p. 1394)

Example

```
template1 = GetTemplate(TemplateName="EngData")
system1 = template1.CreateSystem()
engineeringData1 = system1.GetContainer(ComponentName="Engineering Data")
structuralSteel = engineeringData1.GetMaterial(Name="Structural Steel")
structuralSteel.SetColor(Material="strucutralSteel", Red="255", Green="0", Blue="0")
```

SetOption

Set option for an "Option" variable

Required Arguments

OptionName Option to set

Type string (p. 1438)

OptionValue Value for option

Type string (p. 1438)

SetSuppression

Suppresses or Unsuppresses item.

Item can be suppressed to prevent it from being sent to a downstream cell in the system.

Following items can be suppressed:

Material

Material property

Material property data

Required Arguments

Suppressed The flag to specify if the item should be suppressed or unsuppressed.

Type bool (p. 1360)

Unlink

Unlinks a material from its underlying source. Once this occurs, the material can no longer be restored to its original state and the material will no longer have a source.

MaterialProperty

The entity to store material property information.

A material property is the identifier for the singular information (for example, Density) that together with other properties defines or models the behavior of the material. A property is always defined by at least one table (tabular data), which could be singular. Some properties can contain a collection of tabular data (for example, Isotropic Elasticity).

Properties

Behavior

The string that defines the way in which the material property behaves, e.g., Elasticity has Isotropic, Orthotropic or Anisotropic behavior.

Type string (p. 1438)

Read Only No

Definition

The definition of the material property. Some material properties are defined in different ways, e.g., Thermal Expansion can be defined as Secant Coefficient of Thermal Expansion and Instantaneous Coefficient of Thermal Expansion.

Type string (p. 1438)

Read Only No

Description

The description of the material property.

Type string (p. 1438)

Read Only No

DisplayText

The general property that defines the user-visible name of an entity. This property is defined for all data entities but is used only in those entities that present a label in the user interface.

Type string (p. 1438)

Read Only No

PropertyDataColl

The collection of tabular data that defines the material property.

Type DataReferenceSet (p. 1371)

Read Only Yes

TypeName

The name of the material property.

Type string (p. 1438)

Read Only No

Methods

AddTestData

Includes experimental test data for response function calculations.

Required Arguments

TestData The test data property to add.

Type DataReference (p. 1371)

BeginBatchUpdate

Marks the start of a series of data modifications to a table of data, to improve performance.

CreatePropertyData

Include an additional property data for a material property. Some properties may have more than one property data to describe the material property.

The preferred method of adding a material property data is to use CreateMaterialProperty.

Return The new material properly data that was created.

Type DataReference (p. 1371)

Required Arguments

Name The new material property data name.

Type string (p. 1438)

Optional Arguments

Behavior A string to identify how the new material property data will behave.

The behavior of some material properties can be specified in different ways, e.g., Elasticity can be specified as Isotropic, Orthotropic or Anisotropic.

Type string (p. 1438)

CustomData The optional dictionary of custom data. A non-null dictionary designates the material property as custom and not located in the Engineering Data Metadata.

Type Dictionary (p. 1375)<string (p. 1438), string (p. 1438)>

Definition A string to identify how the new material property data will be defined.

Some material properties are defined in different ways, e.g., Thermal Expansion can be defined as Secant Coefficient of Thermal Expansion and Instantaneous Coefficient of Thermal Expansion.

Type string (p. 1438)

Qualifiers The optional dictionary of a qualifier name and it's corresponding value.

Type Dictionary (p. 1375)<string (p. 1438), string (p. 1438)>

Example

The following example creates the material property data Orthotropic Secant Coefficient of Thermal Expansion on the material property Coefficient of Thermal Expansion. This example assumes the material Structural Steel has been obtained from the General Materials library.

Get the material property we are going to create a new material property data on.

```
thermExpansionMatProp = structuralSteel.GetMaterialProperty(Name="Coefficient of Thermal Expansion")
```

Create the new material property data.

```
orthoSecantThermExpansionMatPropData = thermExpansionMatProp.CreatePropertyData(
    Name="Coefficient of Thermal Expansion",
    Definition="Secant",
    Behavior="Orthotropic")
```

Delete

Deletes the material property.

Optional Arguments

Behavior The optional string to specify the material property behavior.

Behavior of some material properties can be specified in different ways e.g. Elasticity can be specified as Isotropic, Orthotropic or Anisotropic.

Type string (p. 1438)

Definition The optional string to specify the material property definition.

Some material properties are defined in different ways e.g. Thermal Expansion can be defined as Secant Coefficient of Thermal Expansion and Instantaneous Coefficient of Thermal Expansion.

Type string (p. 1438)

DeleteData

Delete a row from a tabular data sheet.

Required Arguments

Index Index of the row to delete.

Type int (p. 1394)

Optional Arguments

SheetName Name of the sheet to access.

Type string (p. 1438)

SheetQualifiers SheetQualifiers is used to pass in the qualifiers to select between multiple sheets witht the same name. This is a dictionary of the Qualifier and its Value.

Type Dictionary (p. 1375)<string (p. 1438), string (p. 1438)>

Example

The following example illustrates the deletion of a row from a tabular data sheet.

```
#
# Create a new Engineering Data System and access Structural Steel
#
template1 = GetTemplate(TemplateName="EngData")
system1 = template1.CreateSystem()
engineeringData1 = system1.GetContainer(ComponentName="Engineering Data")
matl1 = engineeringData1.GetMaterial(Name="Structural Steel")
#
# Delete the first row in the Density property
#
matlProp1 = matl1.GetProperty(Name="Density")
DeleteTabularDataRow(matlProp1, Index = 0)
#
# Delete the first row in the Coefficient of Thermal Expansion property with
# optional SheeetName and SheetQualifiers
#
matlProp2 = matl1.GetProperty(Name="Coefficient of Thermal Expansion")
DeleteTabularDataRow(matlProp2,
                  SheetName="Coefficient of Thermal Expansion",
                  SheetQualifiers={"Definition Method": "Secant", "Behavior": "Isotropic"},
```

```
                        Index = 0)
```

EndBatchUpdate

Marks the completion of a series of data modifications.

GetChartData

Returns a generated graph data for the specified source data.

Valid source data are:

Material Property
Material Property Data

Return The graph data for the specified source data.

Type Dictionary (p. 1375)<string (p. 1438), Object (p. 1409)>

GetData

Returns the tabular data associated with the data entity.

Return The returned data in scalar, list, or dictionary format.

Type Object (p. 1409)

Optional Arguments

AsDictionary If set to true, the data will be returned as a dictionary where the keys are variable names and the values are the data for each variable. If set to false, the data will be returned in scalar or list format without the variable names.

Type bool (p. 1360)

Default Value False

ColumnMajor If set to true, the data will be returned in column-major order. If set to false, the data will be returned in row-major order.

Type bool (p. 1360)

Default Value True

EndIndex The end index for requesting a subset of the data (zero-based).

Type int (p. 1394)

Default Value -2147483647

SheetName Specifies the sheet name when the data contains multiple sheets.

Type string (p. 1438)

SheetQualifiers Used to pass in the qualifiers to select between multiple sheets with the same name. This is a dictionary of qualifiers and values.

Type Dictionary (p. 1375)<string (p. 1438), string (p. 1438)>

StartIndex The start index for requesting a subset of the data (zero-based).

Type int (p. 1394)

Default Value 0

Variables Names of the variables for which data is requested (string or list of strings).

Type Object (p. 1409)

Example

In this example, all data is requested for the given tabular data entity.

```
tabData1.GetData()
```

In this example, all data is requested in row-major order.

```
tabData1.GetData(ColumnMajor=False)
```

In this example, all data is requested in dictionary format.

```
tabData1.GetData(AsDictionary=True)
```

In this example, data for variables Density and Temperature is requested in dictionary format.

```
tabData1.GetData(Variables=["Density", "Temperature"], AsDictionary=True)
```

GetPropertyData

Returns a property data of the specified material property.

The property data returned is based on specified optional parameters "Definition" and "Behavior".

Return The material property data that matches specified type name, definition and behavior.

Type DataReference (p. 1371)

Required Arguments

Name The material property data type name.

Type string (p. 1438)

Optional Arguments

Behavior The optional string to specify the material property data behavior.

Behavior of some material properties can be specified in different ways, e.g., Elasticity can be specified as Isotropic, Orthotropic or Anisotropic.

Type string (p. 1438)

Definition The optional string to specify the material property data definition.

Some material properties are defined in different ways, e.g., Thermal Expansion can be defined as Secant Coefficient of Thermal Expansion and Instantaneous Coefficient of Thermal Expansion.

Type string (p. 1438)

Qualifiers The optional dictionary of a qualifier name and it's corresponding value.

Type Dictionary (p. 1375)<string (p. 1438), string (p. 1438)>

Example

The following example creates new Engineering Data system and queries Coefficient of Thermal Expansion property data of the default material Structural Steel.

```
ENGDTemplate = GetTemplate(TemplateName="EngData")
ENGDSystem = ENGDTemplate.CreateSystem(Position="Default")

ENGDContainer = ENGDSystem.GetContainer(ComponentName="Engineering Data")
StructSteel = ENGDContainer.GetMaterial(Name="Structural Steel")

#Get material property
CTEProperty = StructSteel.GetProperty(Name="Coefficient of Thermal Expansion")
# Get material property data
CTEPropData = CTEProperty.GetPropertyData(
   Name="Coefficient of Thermal Expansion",
   Definition="Secant",
   Behavior="Isotropic")
RefTempPropData = CTEProperty.GetPropertyData(
   Name="Reference Temperature",
   Definition="Secant",
   Behavior="Isotropic")
```

IsSuppressed

Checks if an entity is suppressed.

Valid entities are:

Material
Material Property
Material Property Data

Return Returns true if the entity is suppressed, false otherwise.

Type bool (p. 1360)

IsValid

Validates a material property and provides a message in case of invalid data.

Return The flag that indicates if the material property is valid.

Type bool (p. 1360)

Required Arguments

Material The parent material of the property.

Type DataReference (p. 1371)

Optional Arguments

Message The validation failure message.

Type Output (p. 1411)<string (p. 1438)>

RemoveTestData

Excludes experimental test data for response function calculations.

Required Arguments

TestData The test data property to add.

Type DataReference (p. 1371)

SetData

Set tabular data associated with the data entity.

Optional Arguments

Data Sets the data using a dictionary form. The keys are the variable names and the values are the data. The use of this argument is mutually exclusive with "Values" and "Variables".

Type Dictionary (p. 1375)<string (p. 1438), List (p. 1400)<Object (p. 1409)>>

Index Specifies the starting location used to set the data (zero-based). A value of -1 indicates that the data should be appended to the existing data.

Type int (p. 1394)

Default Value 0

SheetName Specifies the sheet name when the data contains multiple sheets.

Type string (p. 1438)

SheetQualifiers Used to pass in the qualifiers to select between multiple sheets with the same name. This is a dictionary of qualifiers and values.

Type Dictionary (p. 1375)<string (p. 1438), string (p. 1438)>

Values List of data values set in conjunction with the "Variables" parameter. This parameter and the "Data" parameter are mutually exclusive.

Type List (p. 1400)<List (p. 1400)<Object (p. 1409)>>

Variables Names of the variables for which data is being set. This parameter and and the "Data" parameter are mutually exclusive.

Type List (p. 1400)<string (p. 1438)>

Example

```
#
# Create a new Engineering Data System and access Structural Steel
#
template1 = GetTemplate(TemplateName="EngData")
system1 = template1.CreateSystem()
engineeringData1 = system1.GetContainer(ComponentName="Engineering Data")
matl1 = engineeringData1.GetMaterial(Name="Structural Steel")
#
# Change the value of a simple single-valued property
#
matlProp1 = matl1.GetProperty(Name="Density")
SetTabularData(matlProp1,
                  Variables="Density",
                  Values="8500 [kg m^-3]")
#
# Set Temperature-dependent data for Elasticity based
# on lists of variables and values.
matlProp2 = matl1.GetProperty(Name="Elasticity")
temperature = ["400 [K]", "600 [K]", "800 [K]"]
E = ["2e5 [MPa]", "1.9e5 [MPa]", "1.6e5 [MPa]"]
SetTabularData(matlProp2,
                  Variables = ["Temperature","Young's Modulus"],
                  Values = [temperature, E])
#
# Change the Temperature for the second table entry.
#
SetTabularData(matlProp2,
                  Index = 1,
                  Variables = "Temperature",
                  Values = "625 [K]")
#
# Set a list for Poisson's Ratio starting at the second table entry.
#
SetTabularData(matlProp2,
                  Index = 1,
                  Variables = "Poisson's Ratio",
                  Values = [0.3, 0.3])
#
# Set Temperature-dependent property data for the Coefficient of Thermal Expansion
# using a dictionary. The dictionary key is the Variable name,
# followed by the list of values for the variable.
#
matlProp3 = matl1.GetProperty(Name="Coefficient of Thermal Expansion")
newData = {"Temperature": ["200 [F]","400 [F]","600 [F]","800 [F]","1000 [F]"],
"Coefficient of Thermal Expansion" : ["6.3e-6 [F^-1]", "7.0e-6 [F^-1]",
                                      "7.46e-6 [F^-1]", "7.8e-6 [F^-1]",
                                      "8.04e-6 [F^-1]"]}
SetTabularData(matlProp3,
                  SheetName="Coefficient of Thermal Expansion",
                  SheetQualifiers={"Definition Method": "Secant", "Behavior": "Isotropic"},
                  Data = newData)
```

SetQualifier

Changes the values of a specifiec qualifier in a data table.

Required Arguments

Qualifier The Qualifier to Set.

Type string (p. 1438)

Value The new value.

Type string (p. 1438)

Optional Arguments

SheetName The name of the tabular data sheet that contains the qualifier.

Type string (p. 1438)

SheetQualifiers SheetQualifiers can be used to pass in the qualifiers to select between multiple sheets witht the same name. This is a dictionary of the Qualifier and its Value.

Type Dictionary (p. 1375)<string (p. 1438), string (p. 1438)>

VariableName The name of the Variable that contains the qualifier to be changed.

Type string (p. 1438)

VariableQualifiers VariableQualifiers can used to pass in the qualifiers to select between multiple variables witht the same name. This is a dictionary of the Qualifier and its Value.

Type Dictionary (p. 1375)<string (p. 1438), string (p. 1438)>

Example

The following example changes the 'Derive From' setting within an Isotropic Elasticity material property to be "Bulk Modulus and Poisson's Ratio".

```
matl1 = engineeringData1.GetMaterial(Name="Structural Steel")
matlProp1 = matl1.GetProperty(Name="Elasticity")
SetTabularDataQualifier(matlProp1,
   SheetName="Elasticity",
   Qualifier="Derive from",
   Value="Bulk Modulus and Poisson's Ratio")
```

SetSuppression

Suppresses or Unsuppresses item.

Item can be suppressed to prevent it from being sent to a downstream cell in the system.

Following items can be suppressed:

Material

Material property

Material property data

Required Arguments

Suppressed The flag to specify if the item should be suppressed or unsuppressed.

Type bool (p. 1360)

MaterialPropertyData

The entity to store material property (tabular) data information. The material property data is a collection of material variable data.

Properties

Behavior

The behavior of the material variable tabular data. Some material properties can have different behavior, e.g., Elasticity has Isotropic, Orthotropic or Anisotropic behavior.

Type string (p. 1438)

Read Only No

Definition

The definition of the material variable tabular data. Some material properties are defined in different ways, e.g., Thermal Expansion can be defined as Secant Coefficient of Thermal Expansion and Instantaneous Coefficient of Thermal Expansion.

Type string (p. 1438)

Read Only No

DependentColl

The collection of dependent variables in the material variable tabular data, e.g., Density.

Type DataReferenceSet (p. 1371)

Read Only Yes

Description

The description of the material variable tabular data.

Type string (p. 1438)

Read Only No

DisplayText

The general property that defines the user-visible name of an entity. This property is defined for all data entities but is used only in those entities that present a label in the user interface.

Type string (p. 1438)

Read Only No

IndependentColl

The collection of independent variables in the material variable tabular data, e.g., Temperature.

Type DataReferenceSet (p. 1371)

Read Only Yes

PrimaryIndependent

The primary independent variable in the material variable tabular data, e.g., Temperature.

Type DataReference (p. 1371)

Read Only Yes

RowsCount

The number of data values for a variable in the material variable tabular data.

Type int (p. 1394)

Read Only Yes

TypeName

The name of the material variable tabular data.

Type string (p. 1438)

Read Only No

VariableColl

The collection of variables in the material variable tabular data.

Type DataReferenceSet (p. 1371)

Read Only Yes

Methods

BeginBatchUpdate

Marks the start of a series of data modifications to a table of data, to improve performance.

CreateCurveFitting

Creates a curve fitting for a given property data.

Return The curve fitting.

Type DataReference (p. 1371)

Required Arguments

Definition The definition of curve fitting to create. This must be a definition that is supported by an engineering data curve fitting.

i.e. 1 Parameter, 2 Parameter, 1st Order, 2nd Order

Type string (p. 1438)

Type The type of curve fitting to create. This must be a type that is supported by an engineering data curve fitting.

i.e. Neo-Hookean, Mooney-Rivlin, Ogden, Yeoh, Polynomial

Type string (p. 1438)

Example

The following example loads a material with experimental test data and a Neo-Hookean hyperelastic property.

```
template = GetTemplate(TemplateName="EngData")
system = template.CreateSystem()
engineeringData = system.GetContainer(ComponentName="Engineering Data")
neopreneRubber = engineeringData.ReadMaterial(
    Name="Neoprene Rubber",
    Source="Hyperelastic_Materials.xml")
neoHookeanProperty = neopreneRubber.GetProperty(Name="Neo-Hookean")
neoHookeanPropertyData = neoHookeanProperty.GetPropertyData(Name="Neo-Hookean")
curveFit = neoHookeanPropertyData.CreateCurveFitting(
    Type="Neo-Hookean",
    Definition="")
uniaxialProperty = neopreneRubber.GetProperty(Name="Uniaxial Test Data")
curveFit.AddTestData(TestData=uniaxialProperty)
biaxialProperty = neopreneRubber.GetProperty(Name="Biaxial Test Data")
curveFit.AddTestData(TestData=biaxialProperty)
shearProperty = neopreneRubber.GetProperty(Name="Shear Test Data")
curveFit.AddTestData(TestData=shearProperty)
volumetricProperty = neopreneRubber.GetProperty(Name="Volumetric Test Data")
curveFit.AddTestData(TestData=volumetricProperty)
curveFit.RemoveTestData(TestData=uniaxialProperty)
curveFit.AddTestData(TestData=uniaxialProperty)
curveFit.Solve()
curveFit.CopyCoefficients(Destination=neoHookeanPropertyData)
curveFit.Delete()
```

CreateDataProvider

Create a data provider for a give format. The only data provider currently supported is for Delimited Text (used to import tabular data from a delimited data file).

Return DataReference of the DelimitedDataObject

Type DataReference (p. 1371)

Required Arguments

Format Format of provider (only supported option is "Delimited Text")

Type string (p. 1438)

Example

```
material1 = engineeringData1.GetMaterial(Name="Structural Steel")
matlProp1 = material1.GetProperty(Name="Coefficient of Thermal Expansion")
materialPropertyData1 = matlProp1.GetPropertyData(
Name="Coefficient of Thermal Expansion",
Qualifiers={"Definition": "Secant", "Behavior": "Isotropic"})
dataProvider1 = materialPropertyData1.CreateDataProvider(Format="Delimited Text")
dataProvider1.FileName = r"C:\Coefficient_of_thermal_expansion.csv"
dataProvider1.ReadLine = 2
dataProvider1.Columns = [1, 2]
dataProvider1.Delimiter = ","
dataProvider1.VariableNames = ["Temperature", "Coefficient of Thermal Expansion"]
dataProvider1.VariableUnits = ["C", "C^-1"]
dataProvider1.Import()
```

CreateVariable

Include an additional variable in the property data for a material property.

Return The new variable data that was created.

Type DataReference (p. 1371)

Required Arguments

Name The new variable data name.

Type string (p. 1438)

Optional Arguments

CustomData The optional dictionary of custom data. A non-null dictionary designates the material property as custom and not located in the Engineering Data Metadata.

Type Dictionary (p. 1375)<string (p. 1438), string (p. 1438)>

Qualifiers The optional dictionary of a qualifier name and it's corresponding value.

Type Dictionary (p. 1375)<string (p. 1438), string (p. 1438)>

Example

```
material1 = engineeringData1.GetMaterial(Name="Structural Steel")
matlProp1 = material1.CreateProperty(
    Name="Custom Model",
    Qualifiers={"UserMat": "USER"},
    CustomData={})
matlPropData1 = matlProp2.CreatePropertyData(
    Name="Model State Variables",
    Qualifiers={"UserMat": "STATE"},
    CustomData={})
PropertyDataVariable1 = matlPropData1.CreateVariable(
    Name="User Variable",
    Qualifiers={"Display": "True", "UserMat Constant": "1"},
    CustomData={"Quantity Type": "Dimensionless", "Data": "0", "Independent": "False"})
```

Delete

Deletes the material property data.

EndBatchUpdate

Marks the completion of a series of data modifications.

GetChartData

Returns a generated graph data for the specified source data.

Valid source data are:

Material Property
Material Property Data

Return The graph data for the specified source data.

Type Dictionary (p. 1375)<string (p. 1438), Object (p. 1409)>

GetCurveFitting

Query to return the DataReference to the CurveFit used by some PropertyData.

Return CurveFit DataReference

Type DataReference (p. 1371)

GetFieldVariableDataObject

Gets a FieldVariableDataObject from the specified parent. The parent determines which field variables are tied to the data object

Return DataReference of the field variable.

Type DataReference (p. 1371)

Required Arguments

Name Name of the field variable.

Type string (p. 1438)

Example

```
template1 = GetTemplate(TemplateName="EngData")
system1 = template1.CreateSystem()
engineeringData1 = system1.GetContainer(ComponentName="Engineering Data")
structuralSteel = engineeringData1.GetMaterial(Name="Structural Steel")
temperature = structuralSteel.GetFieldVariableDataObject(Name="Temperature")
```

GetVariable

Returns a material variable of a given name from a material property data.

Return The requested variable.

Type DataReference (p. 1371)

Required Arguments

Name The name of the variable.

Type string (p. 1438)

Example

The following example creates new Engineering Data system and queries Coefficient of Thermal Expansion property data of the default material Structural Steel.

```
ENGDTemplate = GetTemplate(TemplateName="EngData")
ENGDSystem = ENGDTemplate.CreateSystem(Position="Default")

ENGDContainer = ENGDSystem.GetContainer(ComponentName="Engineering Data")
StructSteel = ENGDContainer.GetMaterial(Name="Structural Steel")

# Get material
CTEProperty = StructSteel.GetProperty(Name="Coefficient of Thermal Expansion")
# Get material property data
CTEPropData = CTEProperty.GetPropertyData(
   Name="Coefficient of Thermal Expansion",
   Definition="Secant",
   Behavior="Isotropic")
# Get material variable
CTEVariable = CTEPropData.GetVariable(
   Name="Coefficient of Thermal Expansion")
```

IsSuppressed

Checks if an entity is suppressed.

Valid entities are:

Material
Material Property
Material Property Data

Return Returns true if the entity is suppressed, false otherwise.

Type bool (p. 1360)

IsValid

Validates a material property data and provides a message in case of invalid data.

Return The flag that indicates if the material property data is valid.

Type bool (p. 1360)

Required Arguments

Material The parent material of the material property data.

Type DataReference (p. 1371)

Optional Arguments

Message The validation failure message.

Type Output (p. 1411)<string (p. 1438)>

SetSuppression

Suppresses or Unsuppresses item.

Item can be suppressed to prevent it from being sent to a downstream cell in the system.

Following items can be suppressed:

Material

Material property

Material property data

Required Arguments

Suppressed The flag to specify if the item should be suppressed or unsuppressed.

Type bool (p. 1360)

MaterialVariable

The entity to store material variable information.

Properties

DatumColl

The collection of the material variable data.

Type DataReferenceSet (p. 1371)

Read Only No

IsDependent

The flag that indicates if the material variable is dependent, e.g., Density is temperature dependent.

Type bool (p. 1360)

Read Only No

LowerBoundUnit

The unit of the lower bound data value in the material variable data.

Type string (p. 1438)

Read Only No

LowerBoundValue

The lower bound data value in the material variable data.

Type double (p. 1378)

Read Only No

MaxValue

The maximum value limit of the material variable data.

Type double (p. 1378)

Read Only No

MinValue

The minimum value limit of the material variable data.

Type double (p. 1378)

Read Only No

Offset

The offset value of the material variable data.

Type Quantity (p. 1422)

Read Only No

Scale

The scale factor of the material variable data.

Type double (p. 1378)

Read Only No

TypeName

The name of the material variable.

Type string (p. 1438)

Read Only No

UniqueData

The collection of unique data values in the material variable data.

Type DataReferenceSet (p. 1371)

Read Only No

UpperBoundUnit

The unit of the upper bound data value in the material variable data.

Type string (p. 1438)

Read Only No

UpperBoundValue

The upper bound data value in the material variable data.

Type double (p. 1378)

Read Only No

Methods

Delete

Deletes an additional variable in the property data from a material property.

IsValid

Validates a material variable data and provides a message in case of invalid data.

Return The flag that indicates if the material variable is valid.

Type bool (p. 1360)

Required Arguments

Material The parent material of the variable.

Type DataReference (p. 1371)

MaterialPropertyData The parent material property data of the variable.

Type DataReference (p. 1371)

Optional Arguments

Message The validation failure message.

Type Output (p. 1411)<string (p. 1438)>

Example

```
template = GetTemplate(TemplateName="EngData") system = template.CreateSystem(Position="Default") container = system.GetContainer(ComponentName="Engineering Data") structSteel = container.GetMaterial(Name="Structural Steel") CTEProp = structSteel.GetProperty(Name="Coefficient of Thermal Expansion") CTEPropData = CTEProperty.GetPropertyData( Name="Coefficient of Thermal Expansion", Definition="Secant", Behavior="Isotropic") CTEVariable = CTEPropData.GetVariable( Name="Coefficient of Thermal Expansion") valid = EngData.ValidateMaterialVariable(MaterialVariable=CTEVariable, Material=structSteel, MaterialPropertyData=CTEPropdata)
```

External Connection

External Connection

This container hold data for an instance of the External Process Connector.

Methods

ExecuteOperation

Command that wraps the script invoked by a custom GUI Operation

Required Arguments

Operation The name of the operation to execute

Type string (p. 1438)

GetExternalConnectionProperties

A Query to return a reference to the ExternalConnectionProperties entity

Return A reference to the requested data entity.

Type DataReference (p. 1371)

ReadConfiguration

Reads a External Connection configuration file.

Required Arguments

FilePath The full path to the configuration file.

Type string (p. 1438)

ReadParameterValues

Populates all the values of the parameters of a given type within a container.

Required Arguments

ParameterType Type of parameter. Should be "input" or "output".

Type string (p. 1438)

Data Entities

ExternalConnectionProperties

This entity holds the properties used to connect a Workbench project to an external process or application.

Properties

ConfigFile

The path to the configuration file defining application settings.

Type string (p. 1438)

Read Only No

DisplayText

The general property that defines the user-visible name of an entity. This property is defined for all data entities but is used only in those entities that present a label in the user interface.

Type string (p. 1438)

Read Only No

WorkingDirectory

The path to the working directory. This is optional. If this value is not set, it defaults to the add-in directory corresponding to the system ("project_files/dpN/sysDir/addinDir")

Type string (p. 1438)

Read Only No

External Data

External Data

This container holds data to expose external results or data files within the Workbench project.

Methods

AddDataFile

Adds a data file to the outline

Return Returns the reference to the FileData object

Type DataReference (p. 1371)

Required Arguments

FilePath The data file path

Type string (p. 1438)

GetExternalLoadData

Query to return the reference to the container's ExternalLoadData data entity.

Return A reference to the requested ExternalLoadData data entity.

Type DataReference (p. 1371)

GetExternalLoadOutput

Query to return the reference to the container's ExternalDataOutput data entity.

Return A reference to the requested ExternalLoadData data entity.

Type DataReference (p. 1371)

InsertDataFile

Adds a data file to the outline

Return Returns the reference to the FileData object

Type DataReference (p. 1371)

Required Arguments

FilePath The data file path

Type string (p. 1438)

Index OBSOLETE PARAMETER; kept to provide compatability with older scripts Inserts the data files in the specified location in the internal list. It is 0-based.

Type int (p. 1394)

InsertRegisteredFile

Adds a data file to the outline

Required Arguments

AllowDuplicates No details are provided for this entry.

Type bool (p. 1360)

File The file

Type DataReference (p. 1371)

InsertSupportFile

Inserts a support file to the associated FileData

Required Arguments

FileData ExternalLoadFileData reference to add the support file

Type DataReference (p. 1371)

FilePath The support file path

Type string (p. 1438)

ModifySupportFile

Updates a support file to the associated FileData

Required Arguments

FileData ExternalLoadFileData reference to add the support file

Type DataReference (p. 1371)

FilePath The support file path

Type string (p. 1438)

RereadDataFiles

When you modify an External Data system's data file outside of the Workbench and you need to cause the Workbench to re-read the data file.

ScanForFileChanges

This command is useful, when you modify an External Data system's data file outside of the Workbench and you need to cause the Workbench to re-read the data file.

Data Entities

ExternalDataSetup

This is the DataObject which represents the data transfer provider for the External Data setup container.

Properties

DisplayText

The general property that defines the user-visible name of an entity. This property is defined for all data entities but is used only in those entities that present a label in the user interface.

Type string (p. 1438)

Read Only No

ImportType

option that we copy the files or reference the files

Type string (p. 1438)

Read Only No

TransferDataObject

The data transfer object provided by this provider

Type DataReference (p. 1371)

Read Only No

ExternalLoadColumnData

This entity contains information about the column data

Properties

DataType

DataType of this column data

Type string (p. 1438)

Read Only No

DisplayText

The general property that defines the user-visible name of an entity. This property is defined for all data entities but is used only in those entities that present a label in the user interface.

Type string (p. 1438)

Read Only No

Identifier

Identifier of this column data. This will be used by the down stream applications

Type string (p. 1438)

Read Only No

QuantityType

QuantityType of this column data

Type string (p. 1438)

Read Only No

Unit

Unit of this column data

Type string (p. 1438)

Read Only No

ExternalLoadData

This is the root level entity for the external data add-in

Properties

DisplayText

The general property that defines the user-visible name of an entity. This property is defined for all data entities but is used only in those entities that present a label in the user interface.

Type string (p. 1438)

Read Only No

FilesData

Contains the List of FileData references

Type List (p. 1400)<DataReference (p. 1371)>

Read Only No

Methods

DeleteFileData

Removes a data file from the outline

Required Arguments

FileData Reference to the FileData

Type DataReference (p. 1371)

DuplicateFileData

Removes a data file from the outline

Required Arguments

FileData Reference to the FileData

Type DataReference (p. 1371)

GetExternalLoadFileData

Query to return the reference to the container's ExternalLoadFileData data entity.

Return A reference to the requested ExternalLoadFileData data entity.

Type DataReference (p. 1371)

Required Arguments

Name Name of the ExternalLoadFileData entity

Type string (p. 1438)

ExternalLoadFileData

This entity contains information for the DataFile added to the outline

Properties

Description

Contains the description about the Data file

Type string (p. 1438)

Read Only No

DisplayText

The general property that defines the user-visible name of an entity. This property is defined for all data entities but is used only in those entities that present a label in the user interface.

Type string (p. 1438)

Read Only No

File

FileReference maintained by the project

Type DataReference (p. 1371)

Read Only No

FileDataProperty

Contains the reference to the FileDataProperty

Type DataReference (p. 1371)

Read Only No

Master

Makes the Data file as a Master or not. Only one Data file can be a Master.

Type bool (p. 1360)

Read Only No

Output

Contains the reference to the Translated FileData output object

Type DataReference (p. 1371)

Read Only No

Methods

GetDataProperty

Query to return the reference to the container's ExternalLoadFileDataProperty data entity.

Return A reference to the requested ExternalLoadFileData data entity.

Type DataReference (p. 1371)

ModifyFileData

Changes the file path of the existing FileData

Required Arguments

FilePath FilePath to be modified in the FileData

Type string (p. 1438)

SetAsMaster

Changes the Master property on the FileData

Required Arguments

Master controls the whether to make it as Master or not

Type bool (p. 1360)

SetDelimiterCharacter

Changes the Delimited Character on the FileDataProperty

Required Arguments

Delimiter New Delimiter for the FileDataProperty

Type string (p. 1438)

FileDataProperty FileDataProperty that needs to be modified

Type DataReference (p. 1371)

SetDelimiterType

Changes the DelimiterType on the FileDataProperty

Required Arguments

Delimiter New DelimiterType for the FileDataProperty

Type DelimiterType (p. 1373)

DelimiterString New Delimiter string for the FileDataProperty

Type string (p. 1438)

FileDataProperty FileDataProperty that needs to be modified

Type DataReference (p. 1371)

Example

```
solution1.SetDelimiterType(
```

```
    FileData=fileData,
    FileDataProperty=fileDataProperty,
    Delimiter=DelimiterType.Comma,
    DelimiterString=",")
```

SetDummyNetData

Sets the Material Field Data property

Required Arguments

FileDataProperty FileDataProperty that needs to be modified

Type DataReference (p. 1371)

ImportDummyNetData New DelimiterType for the FileDataProperty

Type bool (p. 1360)

SetFormatType

Changes the FormatType on the FileDataProperty

Required Arguments

FileDataProperty FileDataProperty that needs to be modified

Type DataReference (p. 1371)

Type New Format Type for the FileDataProperty

Type FormatType (p. 1386)

SetMaterialFieldData

Sets the Material Field Data property

Required Arguments

MaterialData New DelimiterType for the FileDataProperty

Type bool (p. 1360)

SetStartImportAtLine

Changes the Start Import at Line on the FileDataProperty

Required Arguments

FileDataProperty FileDataProperty that needs to be modified

Type DataReference (p. 1371)

LineNumber New LineNumber for the FileDataProperty

Type int (p. 1394)

ExternalLoadFileDataOutput

This entity contains information about each supporting file

Properties

Dirty

Contains the reference to the FileDataProperty

Type bool (p. 1360)

Read Only No

DisplayText

The general property that defines the user-visible name of an entity. This property is defined for all data entities but is used only in those entities that present a label in the user interface.

Type string (p. 1438)

Read Only No

File

FileReference maintained by the project

Type DataReference (p. 1371)

Read Only No

FileDataProperty

Contains the reference to the FileDataProperty

Type DataReference (p. 1371)

Read Only No

TranslationIdentifier

Contains the reference to the unique translation identifier

Type string (p. 1438)

Read Only No

ExternalLoadFileDataProperty

Contains information displayed on the Properties pane

Properties

ColumnsData

Contains the list references to the Column Data

Type List (p. 1400)<DataReference (p. 1371)>

Read Only No

ConversionOption

Contains the option on how to convert the data

Type VariableConversionOption (p. 1448)

Read Only No

CoordinateSystemType

Specifies whether to use the Cartesian or Cylindrical coordinate system. The default value is Cartesian

Type CoordinateSystemType (p. 1369)

Read Only No

DelimiterCharacter

Contains the delimiter character. Based on this value, number of columns are calculated

Type string (p. 1438)

Read Only No

DelimiterType

Contains either the predefined delimiter or the user defined delimiter type

Type DelimiterType (p. 1373)

Read Only No

Dimensions

You can choose to either import data from 2D or 3D models. If the 2D option is selected, you will be able to import data only at the X and Y coordinates. The Z coordinate is not supported for the 2D option.

Type DimensionsType (p. 1376)

Read Only No

DisplayText

The general property that defines the user-visible name of an entity. This property is defined for all data entities but is used only in those entities that present a label in the user interface.

Type string (p. 1438)

Read Only No

DummyNetData

DummyNetData flag for trace mapping

Type bool (p. 1360)

Read Only No

FileIdentifier

A string that can be used to identify the file in the downstream Mechanical application. The data identifiers, from the Table pane, are appended to this string so that you can pick the correct source data in the downstream Mechanical application.

Type string (p. 1438)

Read Only No

FormatString

Contains the FormatString. Based on this value, number of columns are calculated

Type string (p. 1438)

Read Only No

FormatType

This can be either Delimited or User-Defined. If the value is Delimited - Delimiter Character field is valid If the value is User-Defined - FormatString field is valid

Type FormatType (p. 1386)

Read Only No

LengthUnit

The units of measurement to be used.

Type string (p. 1438)

Read Only No

MaterialFieldData

MaterialsData flag for EngineeringData

Type bool (p. 1360)

Read Only No

OriginX

Contains the OriginX value

Type double (p. 1378)

Read Only No

OriginXUnit

Contains the unit of the OriginX

Type string (p. 1438)

Read Only No

OriginY

Contains the OriginY value

Type double (p. 1378)

Read Only No

OriginYUnit

Contains the unit of the OriginY

Type string (p. 1438)

Read Only No

OriginZ

Contains the OriginZ value

Type double (p. 1378)

Read Only No

OriginZUnit

Contains the unit of the OriginZ

Type string (p. 1438)

Read Only No

ScaleX

Contains the Scale X string

Type string (p. 1438)

Read Only No

ScaleXToolTip

Contains the tool tip of the Scale X

Type string (p. 1438)

Read Only No

ScaleXValid

Contains the whether the Scale X string is valid or not

Type bool (p. 1360)

Read Only No

ScaleY

Contains the Scale Y string

Type string (p. 1438)

Read Only No

ScaleYToolTip

Contains the tool tip of the Scale Y

Type string (p. 1438)

Read Only No

ScaleYValid

Contains the whether the Scale Y string is valid or not

Type bool (p. 1360)

Read Only No

ScaleZ

Contains the Scale Z string

Type string (p. 1438)

Read Only No

ScaleZToolTip

Contains the tool tip of the Scale Z

Type string (p. 1438)

Read Only No

ScaleZValid

Contains the whether the Scale Z string is valid or not

Type bool (p. 1360)

Read Only No

Sliver

Sliver Size for ECAD Files translation

Type double (p. 1378)

Read Only No

StartImportAtLine

The line number at which you want the data import to start. Line numbers start at 1.

Type int (p. 1394)

Read Only No

SupportingFiles

Contains the list to all additional Supporting files

Type List (p. 1400)<DataReference (p. 1371)>

Read Only No

ThetaXY

Contains the ThetaXY value

Type double (p. 1378)

Read Only No

ThetaXYUnit

Contains the unit of the ThetaXY

Type string (p. 1438)

Read Only No

ThetaYZ

Contains the ThetaYZ value

Type double (p. 1378)

Read Only No

ThetaYZUnit

Contains the unit of the ThetaYZ

Type string (p. 1438)

Read Only No

ThetaZX

Contains the ThetaZX value

Type double (p. 1378)

Read Only No

ThetaZXUnit

Contains the unit of the ThetaZX

Type string (p. 1438)

Read Only No

Methods

GetColumnData

Query to return the reference to the container's ExternalLoaColumndData data entity.

Return A reference to the requested ExternalLoadFileData data entity.

Type DataReference (p. 1371)

Required Arguments

Name Name of the ExternalLoadColumnData entity

Type string (p. 1438)

SetColumnDataType

Changes the column data type of the given ColumnData

Required Arguments

ColumnData ColumnData to be mofied

Type DataReference (p. 1371)

DataType New DataType for the ColumnData

Type string (p. 1438)

SetCoordinateSystemType

Changes the CoordinateSystem type in the FileDataProperty

Required Arguments

Type New Coordinate System Type for the FileDataProperty

Type CoordinateSystemType (p. 1369)

SetDimensionLengthUnit

Changes the Data Type on the specified X,Y,Z coordinate

Required Arguments

Coordinate Coordinate X,Y,Z

Type string (p. 1438)

Unit New Length Unit for the FileDataProperty

Type string (p. 1438)

SetDimensionType

Changes the DimensionsType on the FileDataProperty

Required Arguments

Dimensions New DimensionsType for the FileDataProperty

Type DimensionsType (p. 1376)

SetFormatString

Changes the FormatString on the FileDataProperty

Required Arguments

Format New Format string for the FileDataProperty

Type string (p. 1438)

SetLengthUnit

Changes the Length Unit on the FileDataProperty

Required Arguments

Unit New Length Unit for the FileDataProperty

Type string (p. 1438)

ExternalLoadSupportingFileData

This entity contains information about each supporting file

Properties

Description

Supporting file Description

Type string (p. 1438)

Read Only No

DisplayText

The general property that defines the user-visible name of an entity. This property is defined for all data entities but is used only in those entities that present a label in the user interface.

Type string (p. 1438)

Read Only No

File

Supporting file reference

Type DataReference (p. 1371)

Read Only No

FileIdentifier

Supporting file identifier

Type string (p. 1438)

Read Only No

FileType

Supporting file type

Type SupportFileFormatType (p. 1438)

Read Only No

External Model Setup

External Model Setup

This container holds data to expose external model setup data within the Workbench project.

Methods

AddDataFile

Adds a data file to the outline

Return Returns the reference to the FileData object

Type DataReference (p. 1371)

Required Arguments

FilePath The data file path

Type string (p. 1438)

GetExternalModelData

Query to return the reference to the container's ExternalLoadData data entity.

Return A reference to the requested ExternalLoadData data entity.

Type DataReference (p. 1371)

GetExternalModelOutput

Query to return the reference to the container's ExternalDataOutput data entity.

Return A reference to the requested ExternalLoadData data entity.

Type DataReference (p. 1371)

InsertDataFile

Adds a data file to the outline

Return Returns the reference to the FileData object

Type DataReference (p. 1371)

Required Arguments

FilePath The data file path

Type string (p. 1438)

Index Inserts the data files in the specified location in the internal list. It is 0-based.

Type int (p. 1394)

InsertSupportFile

Inserts a support file to the associated FileData

Return Returns the reference to the File reference object

Type DataReference (p. 1371)

Required Arguments

FileData ExternalLoadFileData reference to add the support file

Type DataReference (p. 1371)

FilePath The support file path

Type string (p. 1438)

ModifySupportFile

Updates a support file to the associated FileData

Required Arguments

FileData ExternalLoadFileData reference to add the support file

Type DataReference (p. 1371)

FileIdentifier The support file to replace

Type string (p. 1438)

FilePath The support file path

Type string (p. 1438)

Example

Required example does not exist.

RereadDataFiles

When you modify an External Model system's data file outside of the Workbench and you need to cause the Workbench to re-read the data file.

ScanForFileChanges

This command is useful, when you modify an External Data system's data file outside of the Workbench and you need to cause the Workbench to re-read the data file.

Data Entities

ExternalModelData

This is the root level entity for the external data add-in

Properties

DisplayText

The general property that defines the user-visible name of an entity. This property is defined for all data entities but is used only in those entities that present a label in the user interface.

Type string (p. 1438)

Read Only No

FilesData

Cotnains the List of FileData references

Type List (p. 1400)<DataReference (p. 1371)>

Read Only No

Methods

DeleteFileData

Removes a data file from the outline

Required Arguments

FileData Reference to the FileData

Type DataReference (p. 1371)

DuplicateFileData

Removes a data file from the outline

Required Arguments

FileData Reference to the FileData

Type DataReference (p. 1371)

GetExternalModelFileData

Query to return the reference to the container's ExternalLoadFileData data entity.

Return A reference to the requested ExternalLoadFileData data entity.

Type DataReference (p. 1371)

Required Arguments

Name Name of the ExternalModelFileData entity

Type string (p. 1438)

ExternalModelFileData

This entity contains information for the DataFile added to the outline

Properties

Description

Contains the description about the Data file

Type string (p. 1438)

Read Only No

DisplayText

The general property that defines the user-visible name of an entity. This property is defined for all data entities but is used only in those entities that present a label in the user interface.

Type string (p. 1438)

Read Only No

File

FileReference maintained by the project

Type DataReference (p. 1371)

Read Only No

FileDataProperty

Contains the reference to the FileDataProperty

Type DataReference (p. 1371)

Read Only No

Output

Contains the reference to the Translated FileData output object

Type DataReference (p. 1371)

Read Only No

Methods

GetDataProperty

Query to return the reference to the container's ExternalLoadFileDataProperty data entity.

Return A reference to the requested ExternalLoadFileData data entity.

Type DataReference (p. 1371)

ModifyFileData

Changes the file path of the existing FileData

Required Arguments

FilePath FilePath to be modified in the FileData

Type string (p. 1438)

ExternalModelFileDataOutput

Output data object for parsed/translated files

Properties

Dirty

Contains the reference to the FileDataProperty

Type bool (p. 1360)

Read Only No

DisplayText

The general property that defines the user-visible name of an entity. This property is defined for all data entities but is used only in those entities that present a label in the user interface.

Type string (p. 1438)

Read Only No

File

FileReference maintained by the project

Type DataReference (p. 1371)

Read Only No

FileDataProperty

Contains the reference to the FileDataProperty

Type DataReference (p. 1371)

Read Only No

TranslationIdentifier

Contains the reference to the unique translation identifier

Type string (p. 1438)

Read Only No

ExternalModelFileDataProperty

Contains information displayed on the Properties pane

Properties

CheckValidBlockedCdbFile

Check wether the given file is a valid CDB File. Default is true

Type bool (p. 1360)

Read Only No

DisplayText

The general property that defines the user-visible name of an entity. This property is defined for all data entities but is used only in those entities that present a label in the user interface.

Type string (p. 1438)

Read Only No

ElementComponentKey

Filter element components based on the component key

Type string (p. 1438)

Read Only No

ElementOffset

Number to offset mesh elements by

Type int (p. 1394)

Read Only No

FaceComponentKey

Filter element components based on the component key

Type string (p. 1438)

Read Only No

FileIdentifier

A string that can be used to identify the file in the downstream Mechanical application. The data identifiers, from the Table pane, are appended to this string so that you can pick the correct source data in the downstream Mechanical application.

Type string (p. 1438)

Read Only No

NodalComponentKey

Filter nodal components based on the component key

Type string (p. 1438)

Read Only No

NodeAndElementRenumberingMethod

By default this value is Automatic, specifies whether the user would like us to automatically renumber his mesh elements to prevent unique id conflicts during assembly

Type NodeAndElementRenumberingMethodType (p. 1408)

Read Only No

NodeOffset

Number to offset mesh nodes by

Type int (p. 1394)

Read Only No

NumberOfCopies

Number of copies to be created by downstream consumer

Type int (p. 1394)

Read Only No

OriginX

Contains the OriginX value

Type double (p. 1378)

Read Only No

OriginXUnit

Contains the unit of the OriginX

Type string (p. 1438)

Read Only No

OriginY

Contains the OriginY value

Type double (p. 1378)

Read Only No

OriginYUnit

Contains the unit of the OriginY

Type string (p. 1438)

Read Only No

OriginZ

Contains the OriginZ value

Type double (p. 1378)

Read Only No

OriginZUnit

Contains the unit of the OriginZ

Type string (p. 1438)

Read Only No

PlaneNormalX

Contains the ThetaXY value

Type double (p. 1378)

Read Only No

PlaneNormalY

Contains the ThetaYZ value

Type double (p. 1378)

Read Only No

PlaneNormalZ

Contains the ThetaZX value

Type double (p. 1378)

Read Only No

PlanePointX

Contains the PlanePointX value

Type double (p. 1378)

Read Only No

PlanePointXUnit

Contains the unit of the PlanePointX

Type string (p. 1438)

Read Only No

PlanePointY

Contains the PlanePointY value

Type double (p. 1378)

Read Only No

PlanePointYUnit

Contains the unit of the PlanePointY

Type string (p. 1438)

Read Only No

PlanePointZ

Contains the PlanePointZ value

Type double (p. 1378)

Read Only No

PlanePointZUnit

Contains the unit of the PlanePointZ

Type string (p. 1438)

Read Only No

ProcessElementComponents

Process Element Components. On by default

Type bool (p. 1360)

Read Only No

ProcessFaceComponents

Process Element Components. On by default

Type bool (p. 1360)

Read Only No

ProcessMesh200FromCdb

Process Mesh200 elements from CDB File. Default is false

Type bool (p. 1360)

Read Only No

ProcessModelData

Process Model Data like Coordinate Systems, Element Orientations etc. On by default

Type bool (p. 1360)

Read Only No

ProcessNodalComponents

Process nodal Components. On by default

Type bool (p. 1360)

Read Only No

SupportingFiles

Contains the list to all additional Supporting files

Type List (p. 1400)<DataReference (p. 1371)>

Read Only No

ThetaXY

Contains the ThetaXY value

Type double (p. 1378)

Read Only No

ThetaXYUnit

Contains the unit of the ThetaXY

Type string (p. 1438)

Read Only No

ThetaYZ

Contains the ThetaYZ value

Type double (p. 1378)

Read Only No

ThetaYZUnit

Contains the unit of the ThetaYZ

Type string (p. 1438)

Read Only No

ThetaZX

Contains the ThetaZX value

Type double (p. 1378)

Read Only No

ThetaZXUnit

Contains the unit of the ThetaZX

Type string (p. 1438)

Read Only No

TransformationType

Which type of transform we should display to the user

Type TransformType (p. 1444)

Read Only No

TransformOriginal

Whether or not we are going to transform the first instance of the model

Type bool (p. 1360)

Read Only No

UnitSystem

The unit system the user is using; the length unit is set based on this property

Type string (p. 1438)

Read Only No

Methods

SetApplicationSource

Set the application source for a FileData

Required Arguments

ApplicationSource ApplicationSource for the FileDataProperty

Type string (p. 1438)

SetUnitSystem

Changes the Unit System on the FileDataProperty

Required Arguments

System New Unit System for the FileDataProperty

Type string (p. 1438)

ExternalModelSupportingFileData

Entity to for any additionally linked files

Properties

Description

Supporting file Description

Type string (p. 1438)

Read Only No

DisplayText

The general property that defines the user-visible name of an entity. This property is defined for all data entities but is used only in those entities that present a label in the user interface.

Type string (p. 1438)

Read Only No

File

Supporting file reference

Type DataReference (p. 1371)

Read Only No

FileIdentifier

Supporting file identifier

Type string (p. 1438)

Read Only No

FileType

Supporting file type

Type SupportFileFormatType (p. 1438)

Read Only No

RelativePath

RelativePath for ABAQUS, NASTRAN

Type string (p. 1438)

Read Only No

FLUENT

FLUENT Setup

This container holds Setup data for an instance of FLUENT.

Methods

CopyLauncherSettings

Copies the FLUENT Launcher Settings from one FLUENT Setup or Solution container to another FLUENT Setup or Solution container.

Edit

Opens the FLUENT editor to allow modification of FLUENT data.

This command will open the editor only if one is not already open on this system. If this system's editor is already open and in interactive mode, then it will be raised to the front.

Exit

Exits the FLUENT editor.

If no editor is open on the component in question, this command will have no effect.

GetFluentLauncherSettings

Returns the Data Entity which contains the Setup container's settings for the FLUENT Launcher.

Import

Imports the FLUENT mesh and FLUENT case settings into the FLUENT editor from an existing FLUENT .msh or .cas file; replacing the current FLUENT mesh and FLUENT case settings. If the imported file contains only a mesh, then the FLUENT case settings will be set to FLUENT's default values.

SendCommand

Execute a scheme, FLUENT GUI, or FLUENT TUI command(s) in the currently open FLUENT session. FLUENT GUI commands cannot be run if the currently open FLUENT session is running in no GUI mode.

If the FLUENT editor is not open, it will throw an error message suggesting to start editor first.

Example

The following code shows how to execute the commands in a FLUENT journal file:

```
>> setup1.SendCommand(Command="/file/read-journal \"E:\\WB Projects\\Fluent Case-Data Files\elbow.jou\" "
```

Data Entities

ChartVariable

Entity representing a variable in Convergence Chart

Properties

Color

Color of the curve represnting this variable entity.

Type Color (p. 1365)

Read Only No

DisplayText

The general property that defines the user-visible name of an entity. This property is defined for all data entities but is used only in those entities that present a label in the user interface.

Type string (p. 1438)

Read Only No

LineWidth

Line width of the curve represnting this variable entity.

Type float (p. 1432)

Read Only No

QuantityName

The variable quantity to display.

Type string (p. 1438)

Read Only No

SymbolSize

Symbol size of the points on the curve.

Type uint (p. 1446)

Read Only No

Methods

DeleteChartVariable

Deletes a specified chart variable.

ConvergenceChart

Entity to store a convergence chart information.

Properties

AxisX

Associated X Axis

Type DataReference (p. 1371)

Read Only No

AxisY

Associated Y Axis

Type DataReference (p. 1371)

Read Only No

ChartType

MonitorChartType: Residual or UserDefined

Type MonitorChartType (p. 1406)

Read Only No

Variables

Collection of variables to be plotted.

Type DataReferenceSet (p. 1371)

Read Only No

XAxis

Label for the x-axis/Horizontal Axis.

Type string (p. 1438)

Read Only No

Methods

GetAxis

Returns the axis for a specified convergence chart

Return The axis

Type DataReference (p. 1371)

Required Arguments

Name Name of the Axis

Type string (p. 1438)

GetChartVariable

Returns the chart variable of a given name from a convergence chart

Return The chart variable that matches the specified name.

Type DataReference (p. 1371)

Required Arguments

Name The name of the chart variable.

Type string (p. 1438)

GetChartVariables

Returns the collection of chart variables for a given convergence chart

Return A collection of the variables in the chart.

Type DataReferenceSet (p. 1371)

FluentLauncherSettings

Allows you to specify FLUENT Launcher settings for Fluent Setup/Solution cells.

Properties

CachePassword

Specify whether or not you want to save the HP-MPI password for later use.

Type bool (p. 1360)

Read Only No

ClusterHeadNode

Specify the name of the compute cluster head node.

This property is only available on Windows.

Type string (p. 1438)

Read Only No

ClusterJobTemplate

A custom submission policy created by an IT administrator to define the job parameters for an application and employed by the cluster users to submit jobs. Relevant for Microsoft Job Scheduler.

Type string (p. 1438)

Read Only No

ClusterNodeGroup

Used to set specific node group(s) on which to run the job. Relevant for Microsoft Job Scheduler.

Type string (p. 1438)

Read Only No

ClusterProcessorUnit

Select the unit type (node/socket/core) on which the job would be running. Relevant for Microsoft Job Scheduler.

Type ProcessorUnit (p. 1421)

Read Only No

ConvertUNCPath

Specify whether or not to convert a local path to a UNC path if any matching shared directory is found.

Type bool (p. 1360)

Read Only No

CreateJobSubmissionXML

Specify whether or not to create the job submission XML file.

This property is only available on Windows.

Type bool (p. 1360)

Read Only No

DisplayMesh

Specify whether or not to show the mesh after the mesh file or the case/data file has been read into FLUENT.

Type bool (p. 1360)

Read Only No

DisplayText

The general property that defines the user-visible name of an entity. This property is defined for all data entities but is used only in those entities that present a label in the user interface.

Type string (p. 1438)

Read Only No

EmbedGraphicsWindows

Specify whether or not to embed the graphics windows in the FLUENT application window, or to have the graphics windows free-standing.

Type bool (p. 1360)

Read Only No

EnvPath

Specify the list of environment variables to set before starting FLUENT.

Type Dictionary (p. 1375)<string (p. 1438), string (p. 1438)>

Read Only No

Initialization

Specify which initialization method to use.

Type InitializationMethods (p. 1393)

Read Only No

InitSolutionDataFile

Spepcify the solution data file to be used for initializing the solution.

Type string (p. 1438)

Read Only No

Interconnect

Specify the interconnect that you wish to use for parallel calculations

(e.g., ethernet, infiniband, etc.).

Type string (p. 1438)

Read Only No

JobScheduler

Use available Resource Manager (LSF, SGE, PBSPro) to launch FLUENT job

This property is only available when using compute nodes on Linux.

Type SchedulerSpecification (p. 1429)

Read Only No

JobSubmissionXmlFile

Specify the name of the job submission XML file.

This property is only available on Windows.

Type string (p. 1438)

Read Only No

LoadACTStartPage

Specify whether or not to show ACt Start Page.

Type bool (p. 1360)

Read Only No

LSFCheckpointPeriod

Specify the interval of automatic checkpointing for LSF.

This property is only available when using compute nodes on Linux.

Type int (p. 1394)

Read Only No

LSFQueue

Specify the name of the LSF queue.

This property is only available when using compute nodes on Linux.

Type string (p. 1438)

Read Only No

LSFUseAutomaticCheckpointing

Specify whether or not you want to use automatic checkpointing with LSF. The specific interval for checkpointing is determined by the LSFCheckpointPeriod property.

This property is only available when using compute nodes on Linux.

Type bool (p. 1360)

Read Only No

MachineFileName

Specify the name of the file that contains a list of machine names to run the parallel job.

Type string (p. 1438)

Read Only No

MachineList

Specify a list of machine names to run the parallel job.

Type string (p. 1438)

Read Only No

MachineSpec

Specify a list of machine names, or a file that contains machine names.

Type MachineSpecification (p. 1401)

Read Only No

MpiType

Specify the MPI type that you wish to use for the parallel calculations

(e.g., ibmmpi, intel, etc.).

Type string (p. 1438)

Read Only No

NumberOfProcessors

Specify the number of processors you wish to use for the parallel calculations (e.g., 2, 4, etc.).

Type int (p. 1394)

Read Only No

NumberOfProcessorsMeshing

Specify the number of processors you wish to use for the meshing parallel calculations (e.g., 2, 4, etc.).

Type int (p. 1394)

Read Only No

Precision

Specify whether to use the single-precision or the double-precision solver.

Type CasePrecision (p. 1362)

Read Only No

PrePostOnly

Specify whether or not you want to run FLUENT in PrePost mode, which only allows you to set up or postprocess a problem (i.e., no calculations can be performed)

Type bool (p. 1360)

Read Only No

RemoteFluentRootPath

Specify the root path of the remote FLUENT Linux installation.

This property is only available on Windows.

Type string (p. 1438)

Read Only No

RemoteHostName

Specify the name of the head machine on the remote Linux cluster.

This property is only available on Windows.

Type string (p. 1438)

Read Only No

RemoteRshCommand

Specify the command to connect to the remote node (the default is SSH).

This property is only available on Windows.

Type RshSpecification (p. 1427)

Read Only No

RemoteRshOtherCommand

Specify the custom SSH spawn command used to connect to the remote Linux machine.

This property is only available on Windows.

Type string (p. 1438)

Read Only No

RemoteRunOnLinux

Specify whether or not you want to run your FLUENT simulation on 64-bit Linux machines.

This property is only available on Windows.

Type bool (p. 1360)

Read Only No

RemoteUseHost

Specify whether or not to use the remote cluster head node that FLUENT will connect to for spawning (e.g., via rsh or ssh).

Use the RemoteHostName property to specify the name of the remote cluster head node.

This property is only available on Windows.

Type bool (p. 1360)

Read Only No

RemoteUseWorkingDirectory

Specify whether or not to use a working directory for remote Linux nodes.

Use the RemoteWorkingDirectory property to specify the name of the working directory.

This property is only available on Windows.

Type bool (p. 1360)

Read Only No

RemoteWorkingDirectory

Specify the name of the working directory for remote Linux nodes.

This property is only available on Windows.

Type string (p. 1438)

Read Only No

RunParallel

Specify whether or not you want to run the parallel version of FLUENT.

Type bool (p. 1360)

Read Only No

SetupCompilationEnvironment

Specify whether or not you want to define settings for compiling user-defined functions (UDFs) with FLUENT.

This property is only available on Windows.

Type bool (p. 1360)

Read Only No

SGEPE

Specify the SGE parallel environment where you want to submit your FLUENT jobs.

This property is only available when using compute nodes on Linux.

Type string (p. 1438)

Read Only No

SGEQMaster

Specify the name of the Qmaster host.

This property is only available when using compute nodes on Linux.

Type string (p. 1438)

Read Only No

SGEQueue

Specify the name of the queue where you want to submit your FLUENT jobs.

This property is only available when using compute nodes on Linux.

Type string (p. 1438)

Read Only No

SGESettings

Specify SGE Settings to be used

This property is only available when using compute nodes on Linux.

Type string (p. 1438)

Read Only No

SGEUseSettings

Specify whether or not to use the specified SGE Settings.

Use the SGESettings property to specify the SGE Settings.

This property is only available when using compute nodes on Linux.

Type bool (p. 1360)

Read Only No

ShowLauncher

Specify whether or not to show FLUENT Launcher when FLUENT starts.

Type bool (p. 1360)

Read Only No

SLURMAccount

Specify the name of the SLURM Account.

This property is only available when using compute nodes on Linux.

Type string (p. 1438)

Read Only No

SLURMPartition

Specify the name of the SLURM Partition.

This property is only available when using compute nodes on Linux.

Type string (p. 1438)

Read Only No

SLURMSubmissionHost

Specify the name of the SLURM Submission host.

This property is only available when using compute nodes on Linux.

Type string (p. 1438)

Read Only No

StartWhenResourcesAvailable

Specify whether or not to start the FLUENT job when resources are available.

This property is only available on Windows.

Type bool (p. 1360)

Read Only No

UDFPath

Specify the path to the UDF compilation script (available when SetupCompilationEnvironment is TRUE).

This property is only available on Windows.

Type string (p. 1438)

Read Only No

UseJobScheduler

Specify whether or not to use a job scheduler to run FLUENT jobs.

Type bool (p. 1360)

Read Only No

UseLSFCheckpoint

Specify whether or not to use LSF checkpointing.

This property is only available when using compute nodes on Linux.

Type bool (p. 1360)

Read Only No

UseLSFQueue

Specify whether or not to use the LSF queue.

This property is only available when using compute nodes on Linux.

Type bool (p. 1360)

Read Only No

UseSharedMemory

Specify whether or not to use shared memory on the local machine or to use distributed memory on a cluster.

Type bool (p. 1360)

Read Only No

UseUpstreamLauncherSettings

Specify whether or not the current system's Solution cell should use FLUENT Launcher's property settings from the current system's Setup cell.

Type bool (p. 1360)

Read Only No

WorkbenchColorScheme

Specify whether or not to use the Workbench color scheme in the graphics window, or the classic black background color.

Type bool (p. 1360)

Read Only No

ROMSetup

Entity which manages the setup of the ROM feature. This DataObject must be held by a solver addin container. The solver addin container will then be used as an identifier for the solver addin system.

Properties

DisplayText

The general property that defines the user-visible name of an entity. This property is defined for all data entities but is used only in those entities that present a label in the user interface.

Type string (p. 1438)

Read Only No

Setup

Setup object that contains all the solver setup information.

Type Setup (p. 1431)

Read Only No

Solver

Identification of the solver. The import of an evaluation archive will only be allowed if the solver that wrote the archive has the same name as the one consuming it.

Type string (p. 1438)

Read Only Yes

SetupData

Data entity of Setup cell. Allows you to change attributes of Setup cell.

Properties

DisplayText

The general property that defines the user-visible name of an entity. This property is defined for all data entities but is used only in those entities that present a label in the user interface.

Type string (p. 1438)

Read Only No

GenerateSetupOutput

Allows you to generate case file for Setup cell if mesh file is input or mesh operations are defined

Type bool (p. 1360)

Read Only No

SolverSystemHandler

This entity handles the output produced by a solver addin. There is one instance of SolverSystemHandlerDataObject per registered solver system. It holds all the outputs coming from a solver system (snapshot files, mesh files, solver specific files).

Properties

ActionMode

No details are provided for this entry.

Type ActionMode (p. 1353)

Read Only No

ContainedApprovedGeneratedData

Indicate that the data for this system have been outdated by a non parametric change then approved as still valid by the user.

Type bool (p. 1360)

Read Only No

DisplayText

The general property that defines the user-visible name of an entity. This property is defined for all data entities but is used only in those entities that present a label in the user interface.

Type string (p. 1438)

Read Only No

MeshFile

No details are provided for this entry.

Type DataReference (p. 1371)

Read Only No

OutdatedDueToNonParametricChange

Indicate that the data of this system are outdated and not usable.

Type bool (p. 1360)

Read Only No

ProcessedMeshFile

No details are provided for this entry.

Type DataReference (p. 1371)

Read Only No

ProjectHandler

No details are provided for this entry.

Type ProjectHandler (p. 1421)

Read Only No

Setup

No details are provided for this entry.

Type DataReference (p. 1371)

Read Only Yes

SnapshotCache

Handle the snapshot cache of this solver system. Will be assigned only in Production, and null overwise.

Type SnapshotCacheHandler (p. 1433)

Read Only No

FLUENT Solution

This container holds Solution data for an instance of FLUENT.

Methods

Continue

Continue solving from current solution. This command should only be used when the Solution component is either Interrupted or Up-to-Date.

CopyLauncherSettings

Copies the FLUENT Launcher Settings from one FLUENT Setup or Solution container to another FLUENT Setup or Solution container.

Edit

Opens the FLUENT editor to allow modification of FLUENT data.

This command will open the editor only if one is not already open on this system. If this system's editor is already open and in interactive mode, then it will be raised to the front.

Exit

Exits the FLUENT editor.

If no editor is open on the component in question, this command will have no effect.

GenerateFluentReportContent

Generate the report fluent solution container.

Required Arguments

Container Container reference

Type DataContainerReference (p. 1371)

GetComponentSettingsForRsmDpUpdate

This query is used to obtain the ComponentSettingsForRsmDpUpdate object for Journaling and Scripting

GetConvergenceChart

Returns the convergence chart of a given name in a container.

Return The convergence chart that matches the specified name.

Type DataReference (p. 1371)

Required Arguments

Name The name of the convergence chart.

Type string (p. 1438)

GetConvergenceCharts

Returns the collection of convergence charts in a container. If no convergence charts are in the container, the collection is empty.

Return Collection of the convergence charts in the container.

Type DataReferenceSet (p. 1371)

GetFluentLauncherSettings

Returns the Data Entity which contains the Setup container's settings for the FLUENT Launcher.

GetFluentSolutionProperties

Returns the Data Entity which manages settings and data for the FLUENT Solution component.

GetSolutionSettings

This query is used to obtain the component solve settings object for Journaling and Scripting

ImportInitialData

Imports an existing FLUENT .dat file as initial conditions for the FLUENT editor.

Discards the currently available Solution Data (and all stored previous solution points).

MarkUpdateRequired

Accepts an interrupted Solution as Update Required.

As a result of this command, the Solution will be marked Update Required.

MarkUpToDate

Accepts an interrupted Solution as Up-to-Date.

The specified Solution component should be in Interrupted state. As a result of this command, the Solution will be marked Up-to-date.

SendCommand

Execute a scheme, FLUENT GUI, or FLUENT TUI command(s) in the currently open FLUENT session. FLUENT GUI commands cannot be run if the currently open FLUENT session is running in no GUI mode.

If the FLUENT editor is not open, it will throw an error message suggesting to start editor first.

Example

The following code shows how to execute the commands in a FLUENT journal file:

```
>> setup1.SendCommand(Command="/file/read-journal \"E:\\WB Projects\\Fluent Case-Data Files\elbow.jou\" "
```

Data Entities

AxisContinuous

Data entity for Scenegraph axis.

Properties

AutomaticRange

Boolean var for Automatic Range

Type bool (p. 1360)

Read Only No

DisplayText

The general property that defines the user-visible name of an entity. This property is defined for all data entities but is used only in those entities that present a label in the user interface.

Type string (p. 1438)

Read Only No

QuantityName

Name of the quantity being plotted at this axis

Type XAxisQuantity (p. 1451)

Read Only No

RangeMaximum

Max value of quantity at this axis

Type double (p. 1378)

Read Only No

RangeMinimum

Min value of quantity at this axis

Type double (p. 1378)

Read Only No

Scale

Scale of this axis

Type Scale (p. 1428)

Read Only No

Title

Title of the axis

Type string (p. 1438)

Read Only No

ChartVariable

Entity representing a variable in Convergence Chart

Properties

Color

Color of the curve represnting this variable entity.

Type Color (p. 1365)

Read Only No

DisplayText

The general property that defines the user-visible name of an entity. This property is defined for all data entities but is used only in those entities that present a label in the user interface.

Type string (p. 1438)

Read Only No

LineWidth

Line width of the curve represnting this variable entity.

Type float (p. 1432)

Read Only No

QuantityName

The variable quantity to display.

Type string (p. 1438)

Read Only No

SymbolSize

Symbol size of the points on the curve.

Type uint (p. 1446)

Read Only No

Methods

DeleteChartVariable

Deletes a specified chart variable.

ChartVariableData

Entity representing a variable in Convergence Chart

Properties

DisplayText

The general property that defines the user-visible name of an entity. This property is defined for all data entities but is used only in those entities that present a label in the user interface.

Type string (p. 1438)

Read Only No

VariableDimension

Dimentions in string format

Type string (p. 1438)

Read Only No

VariableIndex

A integer: variable index of given MonitorChartType

Type int (p. 1394)

Read Only No

VariableType

MonitorChartType: Residual or UserDefined, an enum

Type MonitorChartType (p. 1406)

Read Only No

ComponentSolveSettingsForAddin

This class contains the solve settings for Addin solution components to use

Properties

ConfiguredQueue

A configured queue used for new RSM architecture

Type string (p. 1438)

Read Only No

DisplayText

The general property that defines the user-visible name of an entity. This property is defined for all data entities but is used only in those entities that present a label in the user interface.

Type string (p. 1438)

Read Only No

EnablePolling

whether enable polling during remote solve

Type bool (p. 1360)

Read Only No

ExecutionMode

execution mode

Type string (p. 1438)

Read Only No

NumberOfProcesses

number of processes for parallel solve

Type int (p. 1394)

Read Only No

PollingInterval

polling internal during remote solve

Type Quantity (p. 1422)

Read Only No

RsmJobName

RSM job name

Type string (p. 1438)

Read Only No

RsmQueueDetails

Configured queue details

Type RsmQueueDetails (p. 1427)

Read Only No

UpdateOption

Update mode

Type JobRunMode (p. 1397)

Read Only No

ConvergenceChart

Entity to store a convergence chart information.

Properties

AxisX

Associated X Axis

Type DataReference (p. 1371)

Read Only No

AxisY

Associated Y Axis

Type DataReference (p. 1371)

Read Only No

ChartType

MonitorChartType: Residual or UserDefined

Type MonitorChartType (p. 1406)

Read Only No

Variables

Collection of variables to be plotted.

Type DataReferenceSet (p. 1371)

Read Only No

XAxis

Label for the x-axis/Horizontal Axis.

Type string (p. 1438)

Read Only No

Methods

GetAxis

Returns the axis for a specified convergence chart

Return The axis

Type DataReference (p. 1371)

Required Arguments

Name Name of the Axis

Type string (p. 1438)

GetChartVariable

Returns the chart variable of a given name from a convergence chart

Return The chart variable that matches the specified name.

Type DataReference (p. 1371)

Required Arguments

Name The name of the chart variable.

Type string (p. 1438)

GetChartVariables

Returns the collection of chart variables for a given convergence chart

Return A collection of the variables in the chart.

Type DataReferenceSet (p. 1371)

FluentLauncherSettings

Allows you to specify FLUENT Launcher settings for Fluent Setup/Solution cells.

Properties

CachePassword

Specify whether or not you want to save the HP-MPI password for later use.

Type bool (p. 1360)

Read Only No

ClusterHeadNode

Specify the name of the compute cluster head node.

This property is only available on Windows.

Type string (p. 1438)

Read Only No

ClusterJobTemplate

A custom submission policy created by an IT administrator to define the job parameters for an application and employed by the cluster users to submit jobs. Relevant for Microsoft Job Scheduler.

Type string (p. 1438)

Read Only No

ClusterNodeGroup

Used to set specific node group(s) on which to run the job. Relevant for Microsoft Job Scheduler.

Type string (p. 1438)

Read Only No

ClusterProcessorUnit

Select the unit type (node/socket/core) on which the job would be running. Relevant for Microsoft Job Scheduler.

Type ProcessorUnit (p. 1421)

Read Only No

ConvertUNCPath

Specify whether or not to convert a local path to a UNC path if any matching shared directory is found.

Type bool (p. 1360)

Read Only No

CreateJobSubmissionXML

Specify whether or not to create the job submission XML file.

This property is only available on Windows.

Type bool (p. 1360)

Read Only No

DisplayMesh

Specify whether or not to show the mesh after the mesh file or the case/data file has been read into FLUENT.

Type bool (p. 1360)

Read Only No

DisplayText

The general property that defines the user-visible name of an entity. This property is defined for all data entities but is used only in those entities that present a label in the user interface.

Type string (p. 1438)

Read Only No

EmbedGraphicsWindows

Specify whether or not to embed the graphics windows in the FLUENT application window, or to have the graphics windows free-standing.

Type bool (p. 1360)

Read Only No

EnvPath

Specify the list of environment variables to set before starting FLUENT.

Type Dictionary (p. 1375)<string (p. 1438), string (p. 1438)>

Read Only No

Initialization

Specify which initialization method to use.

Type InitializationMethods (p. 1393)

Read Only No

InitSolutionDataFile

Spepcify the solution data file to be used for initializing the solution.

Type string (p. 1438)

Read Only No

Interconnect

Specify the interconnect that you wish to use for parallel calculations

(e.g., ethernet, infiniband, etc.).

Type string (p. 1438)

Read Only No

JobScheduler

Use available Resource Manager (LSF, SGE, PBSPro) to launch FLUENT job

This property is only available when using compute nodes on Linux.

Type SchedulerSpecification (p. 1429)

Read Only No

JobSubmissionXmlFile

Specify the name of the job submission XML file.

This property is only available on Windows.

Type string (p. 1438)

Read Only No

LoadACTStartPage

Specify whether or not to show ACt Start Page.

Type bool (p. 1360)

Read Only No

LSFCheckpointPeriod

Specify the interval of automatic checkpointing for LSF.

This property is only available when using compute nodes on Linux.

Type int (p. 1394)

Read Only No

LSFQueue

Specify the name of the LSF queue.

This property is only available when using compute nodes on Linux.

Type string (p. 1438)

Read Only No

LSFUseAutomaticCheckpointing

Specify whether or not you want to use automatic checkpointing with LSF. The specific interval for checkpointing is determined by the LSFCheckpointPeriod property.

This property is only available when using compute nodes on Linux.

Type bool (p. 1360)

Read Only No

MachineFileName

Specify the name of the file that contains a list of machine names to run the parallel job.

Type string (p. 1438)

Read Only No

MachineList

Specify a list of machine names to run the parallel job.

Type string (p. 1438)

Read Only No

MachineSpec

Specify a list of machine names, or a file that contains machine names.

Type MachineSpecification (p. 1401)

Read Only No

MpiType

Specify the MPI type that you wish to use for the parallel calculations

(e.g., ibmmpi, intel, etc.).

Type string (p. 1438)

Read Only No

NumberOfProcessors

Specify the number of processors you wish to use for the parallel calculations (e.g., 2, 4, etc.).

Type int (p. 1394)

Read Only No

NumberOfProcessorsMeshing

Specify the number of processors you wish to use for the meshing parallel calculations (e.g., 2, 4, etc.).

Type int (p. 1394)

Read Only No

Precision

Specify whether to use the single-precision or the double-precision solver.

Type CasePrecision (p. 1362)

Read Only No

PrePostOnly

Specify whether or not you want to run FLUENT in PrePost mode, which only allows you to set up or postprocess a problem (i.e., no calculations can be performed)

Type bool (p. 1360)

Read Only No

RemoteFluentRootPath

Specify the root path of the remote FLUENT Linux installation.

This property is only available on Windows.

Type string (p. 1438)

Read Only No

RemoteHostName

Specify the name of the head machine on the remote Linux cluster.

This property is only available on Windows.

Type string (p. 1438)

Read Only No

RemoteRshCommand

Specify the command to connect to the remote node (the default is SSH).

This property is only available on Windows.

Type RshSpecification (p. 1427)

Read Only No

RemoteRshOtherCommand

Specify the custom SSH spawn command used to connect to the remote Linux machine.

This property is only available on Windows.

Type string (p. 1438)

Read Only No

RemoteRunOnLinux

Specify whether or not you want to run your FLUENT simulation on 64-bit Linux machines.

This property is only available on Windows.

Type bool (p. 1360)

Read Only No

RemoteUseHost

Specify whether or not to use the remote cluster head node that FLUENT will connect to for spawning (e.g., via rsh or ssh).

Use the RemoteHostName property to specify the name of the remote cluster head node.

This property is only available on Windows.

Type bool (p. 1360)

Read Only No

RemoteUseWorkingDirectory

Specify whether or not to use a working directory for remote Linux nodes.

Use the RemoteWorkingDirectory property to specify the name of the working directory.

This property is only available on Windows.

Type bool (p. 1360)

Read Only No

RemoteWorkingDirectory

Specify the name of the working directory for remote Linux nodes.

This property is only available on Windows.

Type string (p. 1438)

Read Only No

RunParallel

Specify whether or not you want to run the parallel version of FLUENT.

Type bool (p. 1360)

Read Only No

SetupCompilationEnvironment

Specify whether or not you want to define settings for compiling user-defined functions (UDFs) with FLUENT.

This property is only available on Windows.

Type bool (p. 1360)

Read Only No

SGEPE

Specify the SGE parallel environment where you want to submit your FLUENT jobs.

This property is only available when using compute nodes on Linux.

Type string (p. 1438)

Read Only No

SGEQMaster

Specify the name of the Qmaster host.

This property is only available when using compute nodes on Linux.

Type string (p. 1438)

Read Only No

SGEQueue

Specify the name of the queue where you want to submit your FLUENT jobs.

This property is only available when using compute nodes on Linux.

Type string (p. 1438)

Read Only No

SGESettings

Specify SGE Settings to be used

This property is only available when using compute nodes on Linux.

Type string (p. 1438)

Read Only No

SGEUseSettings

Specify whether or not to use the specified SGE Settings.

Use the SGESettings property to specify the SGE Settings.

This property is only available when using compute nodes on Linux.

Type bool (p. 1360)

Read Only No

ShowLauncher

Specify whether or not to show FLUENT Launcher when FLUENT starts.

Type bool (p. 1360)

Read Only No

SLURMAccount

Specify the name of the SLURM Account.

This property is only available when using compute nodes on Linux.

Type string (p. 1438)

Read Only No

SLURMPartition

Specify the name of the SLURM Partition.

This property is only available when using compute nodes on Linux.

Type string (p. 1438)

Read Only No

SLURMSubmissionHost

Specify the name of the SLURM Submission host.

This property is only available when using compute nodes on Linux.

Type string (p. 1438)

Read Only No

StartWhenResourcesAvailable

Specify whether or not to start the FLUENT job when resources are available.

This property is only available on Windows.

Type bool (p. 1360)

Read Only No

UDFPath

Specify the path to the UDF compilation script (available when SetupCompilationEnvironment is TRUE).

This property is only available on Windows.

Type string (p. 1438)

Read Only No

UseJobScheduler

Specify whether or not to use a job scheduler to run FLUENT jobs.

Type bool (p. 1360)

Read Only No

UseLSFCheckpoint

Specify whether or not to use LSF checkpointing.

This property is only available when using compute nodes on Linux.

Type bool (p. 1360)

Read Only No

UseLSFQueue

Specify whether or not to use the LSF queue.

This property is only available when using compute nodes on Linux.

Type bool (p. 1360)

Read Only No

UseSharedMemory

Specify whether or not to use shared memory on the local machine or to use distributed memory on a cluster.

Type bool (p. 1360)

Read Only No

UseUpstreamLauncherSettings

Specify whether or not the current system's Solution cell should use FLUENT Launcher's property settings from the current system's Setup cell.

Type bool (p. 1360)

Read Only No

WorkbenchColorScheme

Specify whether or not to use the Workbench color scheme in the graphics window, or the classic black background color.

Type bool (p. 1360)

Read Only No

FluentSolutionProperties

Entity that manages settings and data for the FLUENT Solution component.

Properties

DisplayText

The general property that defines the user-visible name of an entity. This property is defined for all data entities but is used only in those entities that present a label in the user interface.

Type string (p. 1438)

Read Only No

EnableDataInterpolation

It allows to generate the interpolation file at the end of calculation. It can be used as inial data for next calculation even if inout mesh data has changed.

Type bool (p. 1360)

Read Only No

EnableSolutionMonitoring

It allows to generate the solution monitoring data which can be viewed using commands at Solution cell.

Type bool (p. 1360)

Read Only No

GeneratePlotImages

It allows to generate the solution monitoring plot images.

Type bool (p. 1360)

Read Only No

GeneratePostProcessingImages

It allows to generate the image files at the end of calculation for fluent post processing objects.

Type bool (p. 1360)

Read Only No

FLUENT TGridData

This container holds TGrid data for an instance of FLUENT.

Data Entities

TGridCADImportOptions

Entity represents CAD import properties from upstream geometry to Fluent Meshing cell

Properties

DisplayText

The general property that defines the user-visible name of an entity. This property is defined for all data entities but is used only in those entities that present a label in the user interface.

Type string (p. 1438)

Read Only No

FeatureAngle

Allows user to specify the feature angle to determine the features to be imported. The default value is 40.

Type double (p. 1378)

Read Only No

TessellationOption

Options that allow user to control the tessellation (faceting)

Type string (p. 1438)

Read Only No

Units

Specifies the length unit to scale the mesh on import. Models created in other units will be scaled accordingly. The default is meters (m).

Type string (p. 1438)

Read Only No

UseWorkflow

Option that allow user to choose whether to add or not the Import Geometry task in a workflow

Type bool (p. 1360)

Read Only No

TGridData

Data entity of Fluent Meshing cell. Allows you to change attributes of Fluent Meshing cell.

Properties

DisplayText

The general property that defines the user-visible name of an entity. This property is defined for all data entities but is used only in those entities that present a label in the user interface.

Type string (p. 1438)

Read Only No

RunParallel

Specify whether or not you want to run the parallel version of FLUENT.

Type bool (p. 1360)

Read Only No

SaveCheckpointFiles

Specify whether or not save the checkpoint files inside WB project.

Type bool (p. 1360)

Read Only No

Forte

Forte Solution

This container holds Solution data for an instance of Forte.

Data Entities

ComponentFile

No details are provided for this entry.

Properties

DisplayText

The general property that defines the user-visible name of an entity. This property is defined for all data entities but is used only in those entities that present a label in the user interface.

Type string (p. 1438)

Read Only No

File

No details are provided for this entry.

Type DataReference (p. 1371)

Read Only No

Properties

No details are provided for this entry.

Type Dictionary (p. 1375)<string (p. 1438), string (p. 1438)>

Read Only No

Geometry

Geometry

This container holds imported or generated geometry from an instance of DesignModeler.

Methods

Edit

The Edit command starts the geometry editor, which is either DesignModeler or SpaceClaim, if it is not already running. If the geometry editor is already running, this command brings its window into focus.

If a CAD file is assigned to a geometry component, the file is loaded in the geometry editor.

Optional Arguments

Interactive Whether the geometry editor is to be started in interactive mode or batch mode.

Type bool (p. 1360)

Default Value True

IsDiscoveryGeometry Whether to edit the geometry in Discovery. If the user wants to use Discovery as the geometry editor, then IsDiscoveryGeometry should be set to true. If the user wants to use DesignModeler as the geometry editor, then IsDiscoveryGeometry should be set to false.

Type bool (p. 1360)

Default Value False

IsSpaceClaimGeometry Whether to edit the geometry in SpaceClaim. If the user wants to use SpaceClaim as the geometry editor, then IsSpaceClaimGeometry should be set to true. If the user wants to use DesignModeler as the geometry editor, then IsSpaceClaimGeometry should be set to false.

Type bool (p. 1360)

Default Value False

StartupArguments Additional arguments to send to the modeler. These are to be supplied as command line options when the geometry editor is launched. Applies only to SpaceClaim.

Type string (p. 1438)

Example

The following example creates a geometry system from the Geometry template, then launches the SpaceClaim editor in interactive mode.

```
template1 = GetTemplate(TemplateName="Geometry")
system1 = template1.CreateSystem()
geometry1 = system1.GetContainer(ComponentName="Geometry")
geometry1.Edit(IsSpaceClaimGeometry=True)
```

Exit

The Exit command shuts down the running session of the geometry editor. Before shutting down, the geometry is brought up to date and the editor saves its database.

The editor session cannot be closed if it is busy in generating features or seeking the user's input. In such situations, this command throws an ApplicationBusyException exception.

Export

The export command exports geometry data in the running geometry editor to the location specified in its FilePath argument. The export file CAD format is deduced from the extension of FilePath.

Available options:

FilePath	This argument points to the location where the exported file should be saved. The file-name extension specifies the export file format.

The command throws following exceptions:

CommandFailedException	Export file format is not supported. Export file location doesn't exists. File name contains in-valid characters.
ApplicationBusyException	When the editor is busy.

Required Arguments

FilePath Output file path

Type string (p. 1438)

Example

Suppose the geometry in the container "Geometry" needs to be exported to IGES and STEP formats. This can be achieved by the following example.

```
geometry1 = GetDataContainer("Geometry")
geometry1.Export(FilePath="C:/Models/geometry1.iges")
```

```
geometry1.Export(FilePath=AbsUserPathName("Models/geometry1.step"))
```

GetGeometryProperties

Return a reference to DataEntity managing property settings of Geometry Container.

Return Reference to DataEntity managing geometry properties

Type DataReference (p. 1371)

Refresh

The Refresh command refreshes the input data in a geometry component by consuming all changed data from upstream (source) components. This command also updates the modified parameters in the geometry editor.

After successful execution of this command, the geometry component goes into the "update required" state.

RunScript

Executes a script in the assigned geometry editor, which must be running. SpaceClaim will accept Python (.py) and SpaceClaim Script (.scscript) files. DesignModeler will accept JavaScript (.js) or Python (.py) files.

Required Parameters:

ScriptFile	Path of the script file to be run.

Required Arguments

ScriptFile Path of the script file

Type string (p. 1438)

Optional Arguments

useAsMacro For internal use only.

Type bool (p. 1360)

Default Value True

SendCommand

SendCommand sends a javascript or python command string to the geometry editor for execution. If the editor is not open, it will be launched to execute the commands and then closed. If the editor is already running, it will remain running after command execution.

If the editor is busy and not available for executing the instructions, SendCommand throws an ApplicationBusyException exception.

Available options:

Command	Javascript or Python command string containing scripting commands for the geometry editor
Language	Language of the command. Allowed values are: "Javascript" and "Python". The default language is "Javascript"

The following command will add a sketch with an elliptical curve in the DesignModeler editor.

Required Arguments

Command Command string

Type string (p. 1438)

Optional Arguments

Language Language of the command. The default value of Language is "Javascript".

Type string (p. 1438)

Default Value Javascript

Example

```
system1 = GetSystem(Name="Geom")
geometry1 = system1.GetContainer(ComponentName="Geometry")
geometry1.SendCommand( Command = """var ps1 = new Object();
  ps1.Plane  = agb.GetActivePlane();
 ps1.Origin = ps1.Plane.GetOrigin();
 ps1.XAxis  = ps1.Plane.GetXAxis();
 ps1.YAxis  = ps1.Plane.GetYAxis();
 ps1.Sk1 = ps1.Plane.NewSketch();
 ps1.Sk1.Name = "Sketch1";
 with (ps1.Sk1) { ps1.El7 = Ellipse( 8.0, 10.0, 9.0, 6.0, 5.0, 12.0); }
 agb.Regen();""")
```

SetFile

Adds a Geometry file to the Geometry System. The file processed by the geometry editor.

Available options:

FilePath	Path of the geometry file
PlugInName	PlugIn Name, in case of PlugIn mode geometry transfer

Required Arguments

FilePath Path of the Geometry File

Type string (p. 1438)

Optional Arguments

CreatedFileRef Created File reference

Type Output (p. 1411)<DataReference (p. 1371)>

PlugInName PlugIn Name, in case of PlugIn mode Geometry transfer

Type string (p. 1438)

Example

```
geometry1 = GetDataContainer("Geometry")
geometry1.SetFile("C:/Models/block.iges")
geometry2 = GetDataContainer("Geometry 1")
geometry2.SetFile(FilePath=AbsUserPathName("Models/block.prt.1"), PlugInName="ProEngineer[1]")
```

Stop

The Stop command shuts down the running session of of the Geometry Editor immediately, without saving its unsaved data.

The editor session can not be stopped if it is busy in importing or exporting CAD files. In such situations, this command throws an ApplicationBusyException exception.

UpdateCAD

The UpdateCAD command updates the geometry component using parameter values from the Geometry Editor. If the parameters are coming from external CAD systems through attach features in the editor, then the attach feature is refreshed to update the parameters. The editor model is re-generated using the updated parameter values.

UpdateICManagerParam

Update IC Manager Param updates any of the exposed ICManager parameters

Required Arguments

QuantityName Name of the Quantity To Update

Type string (p. 1438)

QuantityValue Double value of the Quantity to update

Type double (p. 1378)

WriteNDFFile

Writes a Blade Neutral Data Format File for each flowpath present in the DM model. In the case of a single FlowPath a single NDF file is written. For multiple flowpaths the specified filename is appended with the _[flowpath name] as multiple files are exported.

Required Parameters:

FilePath	Path of the NDF file to be exported.

Required Arguments

FilePath Path of the NDF file

Type string (p. 1438)

Data Entities

Geometry

Geometry data object

Properties

DisplayText

The general property that defines the user-visible name of an entity. This property is defined for all data entities but is used only in those entities that present a label in the user interface.

Type string (p. 1438)

Read Only No

GeometryFilePath

The location of file currently assigned to geometry component. DesignModeler process this file when started through Edit command.

Type string (p. 1438)

Read Only No

GeometryImportAnalysisType

Analysis Type preference. Import 3D objects or 2D objects (objects must be in the x-y plane)

Type AnalysisType (p. 1354)

Read Only No

GeometryImportCadAssociativity

Associativity preference. Indicates if action should be taken to allow associativity. This option is present because some CAD systems take too long to compute associativity.

Type bool (p. 1360)

Read Only No

GeometryImportCadAttributes

Import Attributes preference. Allows import of CAD system attributes into the Mechanical application models. Enable this option to import Motion Loads.

Type bool (p. 1360)

Read Only No

GeometryImportCadAttributesFilter

Import Attributes filter Key. (Displayed only when Attributes is selected.) This field can have any number of prefixes with each prefix delimited by a semicolon. If the filter is set to an empty string all applicable entities will be imported as Attributes.

Type string (p. 1438)

Read Only No

GeometryImportCleanGeometryOnImport

Clean

Type bool (p. 1360)

Read Only No

GeometryImportComparePartsOnUpdate

Compare Parts On Update preference. Runs a post update comparison of parts from the original model and the new one. Marking those which have no topological or geometric changes as unmodified. This marking saves the time for remeshing.

Type ComparePartsOnUpdateMethod (p. 1365)

Read Only No

GeometryImportComparePartsTolerance

Compare Parts Tolerance Preference. Sets one of three tolerance values for comparison when running Compare Parts On Update

Type ComparePartsTolerance (p. 1366)

Read Only No

GeometryImportCoordinateSystems

Import Coordinate Systems preference. Specifies whether coordinate systems created in the CAD application should be imported into the Mechanical application.

Type bool (p. 1360)

Read Only No

GeometryImportCoordinateSystemsFilter

Import Coordinate Systems filter Key. (Displayed only when Coordinate Systems is selected.) This field can have any number of prefixes with each prefix delimited by a semicolon. If the filter is set to an empty string all applicable entities will be imported asCoordinate Systems.

Type string (p. 1438)

Read Only No

GeometryImportDecomposeDisjointFaces

Decompose Disjoint Face preference. Use to turn on/off the breaking of disjoint faces into multiple faces.

Type bool (p. 1360)

Read Only No

GeometryImportFacetQuality

Import Facet Quality option

Type ImportFacetQuality (p. 1391)

Read Only No

GeometryImportFlattenAssembly

Import Flattened Assembly Preference // hidden

Type bool (p. 1360)

Read Only No

GeometryImportImportUsingInstances

Import Using Instances preference. Processes a CAD model by honoring its part instances to produce faster attach times and smaller database sizes.

Type bool (p. 1360)

Read Only No

GeometryImportLineBodies

Import Line Bodies preference. (If mixed dimension parts, Mixed Import Resolution preference is used.)

Type bool (p. 1360)

Read Only No

GeometryImportMaterialProperties

Import Material Properties preference. Allows import of material data defined in the CAD system. Only a subset of material data will be imported. This will include Young's Modulus, Poisson Ratio, Mass Density, Specific Heat, Thermal Conductivity and Thermal Expansion Coefficient. Limited additional data may be imported depending on CAD support.

Type bool (p. 1360)

Read Only No

GeometryImportMixedResolutionOption

Mixed Import Resolution preference. Allows parts of mixed dimension to be imported as components of assemblies which have parts of different dimension.

Type MixedImportPref (p. 1406)

Read Only No

GeometryImportNamedSelections

Import Named Selections preference. Creates a named selection based on data generated in the CAD system or in the DesignModeler application.

Type bool (p. 1360)

Read Only No

GeometryImportNamedSelectionsFilter

Import Named Selections filter Key. (Displayed only when Named Selections is selected.) This field can have any number of prefixes with each prefix delimited by a semicolon. If the filter is set to an empty string all applicable entities will be imported as Named Selections.

Type string (p. 1438)

Read Only No

GeometryImportParameters

Import Parameters preference. Allows user to turn on or off parameter processing.

Type CADImportParameterType (p. 1360)

Read Only No

GeometryImportParametersFilter

Import Parameter filter Key. (Displayed only when Parameters is selected.) Allows user to specify a key that must appear at the beginning or end of a CAD parameter name to be imported. If the filter is set to an empty string all CAD parameters will be imported.

Type string (p. 1438)

Read Only No

GeometryImportProcessEnclosures

Enclosure and Symmetry preference. Use to turn on/off the processing of enclosure and symmetry named selections.

Type bool (p. 1360)

Read Only No

GeometryImportSavePartFile

Reader Mode Saves Updated File preference. When set to Yes, the interface will save the part file of a model at the end of an update process using the same file name in the same directory.

Type bool (p. 1360)

Read Only No

GeometryImportSmartUpdate

Smart CAD Update preference. Speeds up refresh of models that have unmodified components. If set to Yes and changes are made to other preferences, these will not be respected if the component is smart updated.

Type bool (p. 1360)

Read Only No

GeometryImportSolidBodies

Import Solid Bodies preference. (If mixed dimension parts, Mixed Import Resolution preference is used.)

Type bool (p. 1360)

Read Only No

GeometryImportSpaceClaimExplodeUnsharedComponents

GeometryImportSpaceClaimExplodeUnsharedComponents

Type bool (p. 1360)

Read Only No

GeometryImportStitchSurfacesOnImport

Attempt to stitch surfaces together

Type ImportStitchPreference (p. 1392)

Read Only No

GeometryImportStitchTolerance

Defines the tolerance value to be used when attempting to stitch the surfaces together the value is in meter units

Type double (p. 1378)

Read Only No

GeometryImportSurfaceBodies

Import Surface Bodies preference. (If mixed dimension parts, Mixed Import Resolution preference is used.)

Type bool (p. 1360)

Read Only No

GeometryImportWeightclass

Import Weightclass option

Type ImportWeightclass (p. 1392)

Read Only No

GeometryImportWorkPoints

Import Work Points preference. Specifies whether work points created in the CAD application should be imported into the Mechanical application.

Type bool (p. 1360)

Read Only No

PlugInName

Current PlugInName - Returns TempPlugin if active DM session is editing. Otherwise return the real plugin name. This is also not persisted.

Type string (p. 1438)

Read Only No

TeamcenterConnection

The source string obtained from the Teamcenter, pointing to the NX geometry.

Type string (p. 1438)

Read Only No

Graphics

Graphics

This container holds charts and graphics objects in the project.

Data Entities

AxisContinuous

A chart axis that spans a set of continuous values. An example is an axis of an XY plot.

Properties

AutoScale

This property will define whether or not automatic scaling should be applied to the axis, or whether the RangeMin and RangeMax should be used.

Type bool (p. 1360)

Read Only No

Label

The label of the axis.

Type string (p. 1438)

Read Only No

Logarithmic

This property controls whether the axis scaling is to be logarithmic or linear. The default is linear scaling.

Type bool (p. 1360)

Read Only No

RangeMax

The maximum range of the values in this axis.

Type double (p. 1378)

Read Only No

RangeMin

The minimum range of the values in this axis.

Type double (p. 1378)

Read Only No

ShowGrid

Defines whether or not to show a grid for this chart axis.

Type bool (p. 1360)

Read Only No

TitleBackgroundColor

This defines the background color of an axis title. This is particularly useful when you want to be able to identify which variable is associated with which axis. The default is transparent.

Type Color (p. 1365)

Read Only No

Usability

Determine whether this axis represents a usability axis. A usability axis presents discrete allowable values rather than continuous values.

Type bool (p. 1360)

Read Only No

AxisDiscrete

A chart axis that represents a set of discrete values. An example is an axis of a bar chart.

Properties

AutoScale

This property will define whether or not automatic scaling should be applied to the axis, or whether the RangeMin and RangeMax should be used.

Type bool (p. 1360)

Read Only No

Label

The label of the axis.

Type string (p. 1438)

Read Only No

RangeMax

The index of the last division of the discrete data to be used. If this is -1 then it is undefined and will be determined dependent on the data.

Type uint (p. 1446)

Read Only No

RangeMin

The index of the first division of the discrete data to be used. If this is -1 then it is undefined and will be determined dependent on the data.

Type uint (p. 1446)

Read Only No

ShowGrid

Defines whether or not to show a grid for this chart axis.

Type bool (p. 1360)

Read Only No

TitleBackgroundColor

This defines the background color of an axis title. This is particularly useful when you want to be able to identify which variable is associated with which axis. The default is transparent.

Type Color (p. 1365)

Read Only No

ChartXY

This entity provides general properties for an XY (i.e. two dimensional) chart. Plotting details are determined from the associated variables and axes.

Properties

Legend

Reference to the Legend entity that is applied to the chart.

Type DataReference (p. 1371)

Read Only No

Style

Reference to the RenderStyle entity that is applied to the chart.

Type DataReference (p. 1371)

Read Only No

Title

The title of the chart.

Type string (p. 1438)

Read Only No

Variables

The variables to be displayed in this chart. This can be a list of Variable, VariableXY or VariableXYZ data entities.

Type List (p. 1400)<DataReference (p. 1371)>

Read Only No

XAxis

The data reference to the x-axis.

Type DataReference (p. 1371)

Read Only No

XAxisSecondary

The data reference to the secondary x-axis.

Type DataReference (p. 1371)

Read Only No

YAxis

The data reference to the y-axis.

Type DataReference (p. 1371)

Read Only No

YAxisSecondary

The data reference to the secondary y-axis.

Type DataReference (p. 1371)

Read Only No

ChartXYZ

This chart is a canvas for an XYZ plot, the manor of plotting will be determined by the specified variables.

Properties

Legend

Reference to the Legend entity that is applied to the chart.

Type DataReference (p. 1371)

Read Only No

Style

Reference to the RenderStyle entity that is applied to the chart.

Type DataReference (p. 1371)

Read Only No

Title

The title of the chart.

Type string (p. 1438)

Read Only No

Variables

The variables to be displayed in this chart. This can be a list of Variable, VariableXY or VariableXYZ data entities.

Type List (p. 1400)<DataReference (p. 1371)>

Read Only No

XAxis

The data reference to the X-Axis.

Type DataReference (p. 1371)

Read Only No

YAxis

The data reference to the Y-Axis.

Type DataReference (p. 1371)

Read Only No

ZAxis

The data reference to the Y-Axis.

Type DataReference (p. 1371)

Read Only No

CorrelationMatrix

A Correlation Matrix uses a tabular graphic to display the strength of the relationships between multiple parameters in a study.

Properties

CorrelationRange

This range defines the values and the distribution of correlation values to be applied to the color range. This defaults from -1 to 1.

Type List (p. 1400)<float (p. 1432)>

Read Only No

Legend

Reference to the Legend entity that is applied to the chart.

Type DataReference (p. 1371)

Read Only No

Style

Reference to the RenderStyle entity that is applied to the chart.

Type DataReference (p. 1371)

Read Only No

Title

The title of the chart.

Type string (p. 1438)

Read Only No

Variables

The variables to be displayed in this chart. This can be a list of Variable, VariableXY or VariableXYZ data entities.

Type List (p. 1400)<DataReference (p. 1371)>

Read Only No

Legend

This entity provides a legend for chart data.

Properties

BackgroundColor

The background color of the legend.

Type Color (p. 1365)

Read Only No

DisplayText

The general property that defines the user-visible name of an entity. This property is defined for all data entities but is used only in those entities that present a label in the user interface.

Type string (p. 1438)

Read Only No

Enabled

This property will enable or disable the legend.

Type bool (p. 1360)

Read Only No

ForegroundColor

The foreground (border) color of the legend.

Type Color (p. 1365)

Read Only No

Orientation

This property defines the orientation of the legend.

Type OrientationStyle (p. 1411)

Read Only No

MultiAxisChart

Specialization of a chart to represent a parallel coordinate plot or a spider chart. Multi-axis charts use an independent axis for each supplied variable.

Properties

ChartType

Sets the type of rendering (e.g. Parallel Coordinate Plot, Spider Plot) for this multi-axis chart.

Type ChartStyle (p. 1363)

Read Only No

Legend

Reference to the Legend entity that is applied to the chart.

Type DataReference (p. 1371)

Read Only No

Style

Reference to the RenderStyle entity that is applied to the chart.

Type DataReference (p. 1371)

Read Only No

Title

The title of the chart.

Type string (p. 1438)

Read Only No

Variables

The variables to be displayed in this chart. This can be a list of Variable, VariableXY or VariableXYZ data entities.

Type List (p. 1400)<DataReference (p. 1371)>

Read Only No

PieChart

Pie chart data object that allows us to represent a displayable pie chart.

Properties

DivisionLabels

In a multi-axis chart we are plotting each variable as an axis, but what we plot are actually displaying are the rows of each variable, as such we need labels for each row.

Type DataReference (p. 1371)

Read Only No

Legend

Reference to the Legend entity that is applied to the chart.

Type DataReference (p. 1371)

Read Only No

ShowPercentages

Should the percentages for the slices be shown.

Type bool (p. 1360)

Read Only No

Style

Reference to the RenderStyle entity that is applied to the chart.

Type DataReference (p. 1371)

Read Only No

Title

The title of the chart.

Type string (p. 1438)

Read Only No

Variables

The variables to be displayed in this chart. This can be a list of Variable, VariableXY or VariableXYZ data entities.

Type List (p. 1400)<DataReference (p. 1371)>

Read Only No

RenderStyle

This entity supplies the render properties for any graphics object.

Properties

BarOffset

This property controls the amount of space (relative to the BarWidth) before drawing a bar for this variable.

For example, if two variables are being drawn in a bar chart and you set the offset of the second variable to be 0.5, that variable will be shifted by half the BarWidth to avoid overlap.

Type float (p. 1432)

Read Only No

BarWidth

This property controls the width of bars in a bar chart. The range of allowable values is 0 to 1, and sets the percentage of the available space used for the bars of the variable.

Type float (p. 1432)

Read Only No

DisplayText

The general property that defines the user-visible name of an entity. This property is defined for all data entities but is used only in those entities that present a label in the user interface.

Type string (p. 1438)

Read Only No

DotStyle

The symbol to be used to at each plot point.

Type DotStyles (p. 1378)

Read Only No

FillColors

Define the fill color for this variable in the plot. All filled regions will use this color except where the style is defined as gradient in which case the GradientColor is used.

Type List (p. 1400)<Color (p. 1365)>

Read Only No

FillStyle

The shading to be used for any filled region.

Type FillStyles (p. 1385)

Read Only No

GradientAxis

The axis that defines plot color if gradient shading is enabled. The axis must be continuous.

Type Axis (p. 1356)

Read Only No

LineColors

Defines the line color of this variable in a plot. The first value in the list will be used if the line style is not gradient. Gradient line style will blend between the provided colors.

Type List (p. 1400)<Color (p. 1365)>

Read Only No

LineStyle

The style of the line.

Type LineStyles (p. 1399)

Read Only No

LineWidth

Sets the width of the line drawn for this variable in pixels.

Type uint (p. 1446)

Read Only No

NumberOfColorBands

Controls the number of color bands to be used in a gradient fill. A value of 0 (the default) will result in a continuous gradient.

Type uint (p. 1446)

Read Only No

OutlineColors

Define the outline colors for symbols. If not set (the default) then LineColors is used.

Type List (p. 1400)<Color (p. 1365)>

Read Only No

ShowLinearInterpolationOfLines

When set to true, causes the ends of a line chart to extend to the edge of the chart. This is primarily used to represent a constant line from a single value.

Type bool (p. 1360)

Read Only No

Smoothing

Enables smoothing of the rendered object. In 3D this results in a smoothed rather than faceted surface. In 2D this results in a smooth line rather than a straight line between points.

Type bool (p. 1360)

Read Only No

SymbolSize

Set the size of a symbol in pixels when a symbol is drawn for this variable. The rendered symbol size may be slightly smaller or larger than expected if symbol does not correctly fit into the specified number of pixels.

Type uint (p. 1446)

Read Only No

Variable

The data entity that defines a variable to be plotted.

Properties

AutoBounds

Defines whether the bounds are to be used from BoundsMin/BoundsMax or whether they are to be automatically generated based on the data.

Type bool (p. 1360)

Read Only No

BoundsMax

Defines the maximum rendered value for the data. Any larger values will be ignored.

Type float (p. 1432)

Read Only No

BoundsMin

Defines the minimum rendered value for the data. Any smaller values will be ignored.

Type float (p. 1432)

Read Only No

DisplayAs

Controls if this variable is displayed using lines, bars, etc.

Type VariableStyle (p. 1449)

Read Only No

FilterBoundsMax

If bounds filtering is enabled, sets the maximum variable value that will cause it to be filtered from the plot. This is different from BoundsMin in that any variable that exceeds this value will be excluded from the plot. This primarily applies to Parallel Coordinate Plots.

Type float (p. 1432)

Read Only No

FilterBoundsMin

If bounds filtering is enabled, sets the minimum variable value that will cause it to be filtered from the plot. This is different from BoundsMin in that any variable that exceeds this value will be excluded from the plot. This primarily applies to Parallel Coordinate Plots.

Type float (p. 1432)

Read Only No

IsFilterBoundsEnabled

When this is true, any variables that are outside the filter bounds will be excluded from the plot. This primarily applies to Parallel Coordinate Plots.

Type bool (p. 1360)

Read Only No

IsIncludedInLegend

Setting this parameter to 'false' will exclude this variable from any legend it may be included in.

This property is only valid for XY and XYZ charts and will be ignored otherwise.

Type bool (p. 1360)

Read Only No

Label

The label of the variable. This is optional and is typically determined from the name of the input variable.

Type string (p. 1438)

Read Only No

RelativeOrder

Define the order of this variable among all the variables in a chart.

Type int (p. 1394)

Read Only No

VariableXY

This is the base class for a data entity that defines a variable to be plotted.

Properties

AutoBounds

Defines whether the bounds are to be used from BoundsMin/BoundsMax or whether they are to be automatically generated based on the data.

Type bool (p. 1360)

Read Only No

BoundsMax

Defines the maximum rendered value for the data. Any larger values will be ignored.

Type float (p. 1432)

Read Only No

BoundsMin

Defines the minimum rendered value for the data. Any smaller values will be ignored.

Type float (p. 1432)

Read Only No

DisplayAs

Controls if this variable is displayed using lines, bars, etc.

Type VariableStyle (p. 1449)

Read Only No

FilterBoundsMax

If bounds filtering is enabled, sets the maximum variable value that will cause it to be filtered from the plot. This is different from BoundsMin in that any variable that exceeds this value will be excluded from the plot. This primarily applies to Parallel Coordinate Plots.

Type float (p. 1432)

Read Only No

FilterBoundsMin

If bounds filtering is enabled, sets the minimum variable value that will cause it to be filtered from the plot. This is different from BoundsMin in that any variable that exceeds this value will be excluded from the plot. This primarily applies to Parallel Coordinate Plots.

Type float (p. 1432)

Read Only No

IsFilterBoundsEnabled

When this is true, any variables that are outside the filter bounds will be excluded from the plot. This primarily applies to Parallel Coordinate Plots.

Type bool (p. 1360)

Read Only No

IsIncludedInLegend

Setting this parameter to 'false' will exclude this variable from any legend it may be included in.

This property is only valid for XY and XYZ charts and will be ignored otherwise.

Type bool (p. 1360)

Read Only No

Label

The label of the variable. This is optional and is typically determined from the name of the input variable.

Type string (p. 1438)

Read Only No

RelativeOrder

Define the order of this variable among all the variables in a chart.

Type int (p. 1394)

Read Only No

VariableXYZ

This is the base class for a data entity that defines a variable to be plotted.

Properties

AutoBounds

Defines whether the bounds are to be used from BoundsMin/BoundsMax or whether they are to be automatically generated based on the data.

Type bool (p. 1360)

Read Only No

BoundsMax

Defines the maximum rendered value for the data. Any larger values will be ignored.

Type float (p. 1432)

Read Only No

BoundsMin

Defines the minimum rendered value for the data. Any smaller values will be ignored.

Type float (p. 1432)

Read Only No

DisplayAs

Controls if this variable is displayed using lines, bars, etc.

Type VariableStyle (p. 1449)

Read Only No

FilterBoundsMax

If bounds filtering is enabled, sets the maximum variable value that will cause it to be filtered from the plot. This is different from BoundsMin in that any variable that exceeds this value will be excluded from the plot. This primarily applies to Parallel Coordinate Plots.

Type float (p. 1432)

Read Only No

FilterBoundsMin

If bounds filtering is enabled, sets the minimum variable value that will cause it to be filtered from the plot. This is different from BoundsMin in that any variable that exceeds this value will be excluded from the plot. This primarily applies to Parallel Coordinate Plots.

Type float (p. 1432)

Read Only No

IsFilterBoundsEnabled

When this is true, any variables that are outside the filter bounds will be excluded from the plot. This primarily applies to Parallel Coordinate Plots.

Type bool (p. 1360)

Read Only No

IsIncludedInLegend

Setting this parameter to 'false' will exclude this variable from any legend it may be included in.

This property is only valid for XY and XYZ charts and will be ignored otherwise.

Type bool (p. 1360)

Read Only No

Label

The label of the variable. This is optional and is typically determined from the name of the input variable.

Type string (p. 1438)

Read Only No

RelativeOrder

Define the order of this variable among all the variables in a chart.

Type int (p. 1394)

Read Only No

ICE

ICE

This container holds ICE data for an instance of IC Engine.

Methods

Refresh

Refresh ICE command.

Optional Arguments

UpstreamList A list of upstream data containers that supply data to this cell

Type List (p. 1400)<DataContainerReference (p. 1371)>

Reset

Reset ICE command.

Update

Update ICE command.

Data Entities

ICEData

ICE Data Object

Properties

CrankRadius

User input: Crank Radius

Type Quantity (p. 1422)

Read Only No

CRLength

User input: Connecting Rod Length

Type Quantity (p. 1422)

Read Only No

DisplayText

The general property that defines the user-visible name of an entity. This property is defined for all data entities but is used only in those entities that present a label in the user interface.

Type string (p. 1438)

Read Only No

EVO

User input: Engine Speed

Type Quantity (p. 1422)

Read Only No

ICCombustionSimulationType

User input: IC Engine Combustion Simulation Type

Type ICCombustionSimulationType (p. 1390)

Read Only No

ICIVCandEVOOption

User input: Option to get IVC and EVO values

Type ICIVCandEVOoption (p. 1390)

Read Only No

ICSimulationType

User input: IC Engine Simulation Type

Type ICSimulationType (p. 1390)

Read Only No

IVC

User input: Engine Speed

Type Quantity (p. 1422)

Read Only No

LiftCurvePath

User input: Path to the valve lift curve

Type string (p. 1438)

Read Only No

MinLift

User input: Mimimum Lift

Type Quantity (p. 1422)

Read Only No

PistonOffset

User input: Piston Offset

Type Quantity (p. 1422)

Read Only No

SetSolverProp

User input: Link to swtch solver from Fluent to Forte

Type string (p. 1438)

Read Only No

ICE Setup

This container holds ICE setup data for an instance of IC Engine.

Methods

Refresh

Refresh ICE solver setup command.

Optional Arguments

UpstreamList A list of upstream data containers that supply data to this cell

Type List (p. 1400)<DataContainerReference (p. 1371)>

Reset

Reset ICE solver setup command.

Update

Update ICE solver setup command.

Data Entities

ICESetupData

ICE Setup Data Object

Properties

CrankAngleSelector

This is link for crank angle selection or KeyGrid dialog.

Type string (p. 1438)

Read Only No

DisplayText

The general property that defines the user-visible name of an entity. This property is defined for all data entities but is used only in those entities that present a label in the user interface.

Type string (p. 1438)

Read Only No

ICKeyGridOption

Yes/No option for key grid

Type YN (p. 1451)

Read Only No

ICResetMeshOnFailure

Option for reset on regen failure

Type bool (p. 1360)

Read Only No

PostIterationJournal

User input: Path to post iteration journal file

Type string (p. 1438)

Read Only No

PreIterationJournal

User input: Path to Pre Iteration Journal File

Type string (p. 1438)

Read Only No

PreIterationJournalForte

User input: Path to Pre Iteration Journal File

Type string (p. 1438)

Read Only No

SolverSettingEditor

This is link for solver setting dialog.

Type string (p. 1438)

Read Only No

SolverSettingEditorForte

This is link for solver setting dialog forte.

Type string (p. 1438)

Read Only No

UserBCProfileFile

User input: Path to User Boundary Condition Profiles

Type string (p. 1438)

Read Only No

UserSettingsFilePath

User input: Path to User Boundary Conditions and Monitor Path

Type string (p. 1438)

Read Only No

ICEM

ICEM CFD

This container holds ICEM CFD data for an instance of ICEM.

Data Entities

MeshingAssemblyTransferType

Mesh data object that resides in the ICEM CFD container.

Properties

DisplayText

The general property that defines the user-visible name of an entity. This property is defined for all data entities but is used only in those entities that present a label in the user interface.

Type string (p. 1438)

Read Only No

License

License preference selection for ICEM

Type string (p. 1438)

Read Only No

Subsets

ICEM CFD creates Subsets instead of Parts from Named Selections if set.

Type bool (p. 1360)

Read Only No

TransferFile

The ICEM CFD downstream files are *.msh "Imported FLUENT Mesh File Type", *.poly "POLYFLOWMesh", *.inp "ANSYS Input File", and *.uns "ICEM CFD Mesh File".

Type DataReference (p. 1371)

Read Only No

SimulationGeneratedMesh

Mesh data object that resides in the ICEM CFD container.

Properties

DisplayText

The general property that defines the user-visible name of an entity. This property is defined for all data entities but is used only in those entities that present a label in the user interface.

Type string (p. 1438)

Read Only No

License

License preference selection for ICEM

Type string (p. 1438)

Read Only No

Subsets

ICEM CFD creates Subsets instead of Parts from Named Selections if set.

Type bool (p. 1360)

Read Only No

TransferFile

The ICEM CFD downstream files are *.msh "Imported FLUENT Mesh File Type", *.poly "POLYFLOWMesh", *.inp "ANSYS Input File", and *.uns "ICEM CFD Mesh File".

Type DataReference (p. 1371)

Read Only No

IcePak

IcePak Setup

This container holds Setup data for an instance of IcePak.

Methods

Edit

User can launch the Icepak application by executing the Edit command

Optional Arguments

Interactive Passing True will run Icepak in interactive mode

Type bool (p. 1360)

Default Value True

SystemCoordinate Optional parameter to specify the system coordinate to be displayed in the application title

Type string (p. 1438)

EnableUpdate

Enables Setup update

Exit

Close the Icepak session

Import

Import on the Setup container with an existing Icepak project or the compressed tzr Icepak project will launch an Icepak session and opens the imported project

Import on the Solution container allow users to set different set of case and data files for post processing.By default, the case and data files from the latest solution are available on the solution component.

Required Arguments

FilePath Path to the project or tzr file when operated on Setup container. Path to the case file to be imported when operated on the Solution container

Type string (p. 1438)

Example

When the Import is called on the Setup container

```
Icepak.Import(FilePath=r"E:\DSModels\ICE\demo1_files\dp0\IPK\Icepak\IcepakProj")
```

When the Import is called on the Setup container with tzr file

```
Icepak.Import(FilePath=r"c:\Temp\test.tzr")
```

When the Import is called on the Solution container

```
Icepak.Import(FilePath=r"D:\IcepakProj00.cfd.cas")
```

Data Entities

ComponentSolveSettingsForAddin

This class contains the solve settings for Addin solution components to use

Properties

ConfiguredQueue

A configured queue used for new RSM architecture

Type string (p. 1438)

Read Only No

DisplayText

The general property that defines the user-visible name of an entity. This property is defined for all data entities but is used only in those entities that present a label in the user interface.

Type string (p. 1438)

Read Only No

EnablePolling

whether enable polling during remote solve

Type bool (p. 1360)

Read Only No

ExecutionMode

execution mode

Type string (p. 1438)

Read Only No

NumberOfProcesses

number of processes for parallel solve

Type int (p. 1394)

Read Only No

PollingInterval

polling internal during remote solve

Type Quantity (p. 1422)

Read Only No

RsmJobName

RSM job name

Type string (p. 1438)

Read Only No

RsmQueueDetails

Configured queue details

Type RsmQueueDetails (p. 1427)

Read Only No

UpdateOption

Update mode

Type JobRunMode (p. 1397)

Read Only No

IcePakSetup

Represents the IcepakSetup data entity

Properties

DisplayText

The general property that defines the user-visible name of an entity. This property is defined for all data entities but is used only in those entities that present a label in the user interface.

Type string (p. 1438)

Read Only No

WorkbenchColorScheme

Setting this property will force Icepak to use the Workbench color scheme. Unsetting this property will allow Icepak to use the default color scheme set inside Icepak

Type bool (p. 1360)

Read Only No

XmlFile

XML Input file

Type DataReference (p. 1371)

Read Only No

IcePak Solution

This container holds Solution data for an instance of IcePak.

Methods

GetComponentSettingsForRsmDpUpdate

This query is used to obtain the ComponentSettingsForRsmDpUpdate object for Journaling and Scripting

GetExpertProperties

This query is used to obtain the ExpertProperties object for Journaling and Scripting

GetSolutionSettings

Query to get fetch the proper solution settings reference.

Import

Import on the Setup container with an existing Icepak project or the compressed tzr Icepak project will launch an Icepak session and opens the imported project

Import on the Solution container allow users to set different set of case and data files for post processing.By default, the case and data files from the latest solution are available on the solution component.

Required Arguments

FilePath Path to the project or tzr file when operated on Setup container. Path to the case file to be imported when operated on the Solution container

Type string (p. 1438)

Example

When the Import is called on the Setup container

```
Icepak.Import(FilePath=r"E:\DSModels\ICE\demo1_files\dp0\IPK\Icepak\IcepakProj")
```

When the Import is called on the Setup container with tzr file

```
Icepak.Import(FilePath=r"c:\Temp\test.tzr")
```

When the Import is called on the Solution container

```
Icepak.Import(FilePath=r"D:\IcepakProj00.cfd.cas")
```

LOST AND FOUND

LOST AND FOUND

A collection of DataEntities that cannot be mapped to a container

Data Entities

DpdbSettings

A data object representing DPDB settings

Properties

DatabaseName

DPS database name to start with.

Type string (p. 1438)

Read Only No

DisplayText

The general property that defines the user-visible name of an entity. This property is defined for all data entities but is used only in those entities that present a label in the user interface.

Type string (p. 1438)

Read Only No

ExecutionTime

execution time limit, in seconds

Type int (p. 1394)

Read Only No

ServerUrl

DPS server URL

Type string (p. 1438)

Read Only No

StartLocalEvaluatorOnUpdate

whether to start local DPS evaluator on update or not

Type bool (p. 1360)

Read Only No

UpdateDesignPointInSteps

whether to split the project into steps prior to DP updates via DPS

Type bool (p. 1360)

Read Only No

MatML31

No details are provided for this entry.

Properties

DisplayText

The general property that defines the user-visible name of an entity. This property is defined for all data entities but is used only in those entities that present a label in the user interface.

Type string (p. 1438)

Read Only No

MatML31

No details are provided for this entry.

Properties

DisplayText

The general property that defines the user-visible name of an entity. This property is defined for all data entities but is used only in those entities that present a label in the user interface.

Type string (p. 1438)

Read Only No

MinervaFile

No details are provided for this entry.

Properties

DisplayText

The general property that defines the user-visible name of an entity. This property is defined for all data entities but is used only in those entities that present a label in the user interface.

Type string (p. 1438)

Read Only No

Location

No details are provided for this entry.

Type string (p. 1438)

Read Only No

NeoHookeanHyperElasticityDependents

Neo-Hookean Hyperelasticity dependents.

Properties

BulkModulus

Gets or sets the Bulk modulus.

Type Expression (p. 1384)<Quantity (p. 1422)>

Read Only No

DisplayText

The general property that defines the user-visible name of an entity. This property is defined for all data entities but is used only in those entities that present a label in the user interface.

Type string (p. 1438)

Read Only No

InitialShearModulus

Gets or sets the initial shear modulus.

Type Expression (p. 1384)<Quantity (p. 1422)>

Read Only No

OrthotropicElasticityDependents

The Orthotropic Elasticity dependents.

Properties

DisplayText

The general property that defines the user-visible name of an entity. This property is defined for all data entities but is used only in those entities that present a label in the user interface.

Type string (p. 1438)

Read Only No

PoissonsRatioXY

Gets or sets the poissons ratio XY.

Type Expression (p. 1384)<Quantity (p. 1422)>

Read Only No

PoissonsRatioXZ

Gets or sets the poissons ratio XZ.

Type Expression (p. 1384)<Quantity (p. 1422)>

Read Only No

PoissonsRatioYZ

Gets or sets the poissons ratio YZ.

Type Expression (p. 1384)<Quantity (p. 1422)>

Read Only No

ShearModulusXY

Gets or sets the shear modulus XY.

Type Expression (p. 1384)<Quantity (p. 1422)>

Read Only No

ShearModulusXZ

Gets or sets the shear modulus XZ.

Type Expression (p. 1384)<Quantity (p. 1422)>

Read Only No

ShearModulusYZ

Gets or sets the shear modulus YZ.

Type Expression (p. 1384)<Quantity (p. 1422)>

Read Only No

YoungsModulusXDirection

Gets or sets the youngs modulus X direction.

Type Expression (p. 1384)<Quantity (p. 1422)>

Read Only No

YoungsModulusYDirection

Gets or sets the youngs modulus Y direction.

Type Expression (p. 1384)<Quantity (p. 1422)>

Read Only No

YoungsModulusZDirection

Gets or sets the youngs modulus Z direction.

Type Expression (p. 1384)<Quantity (p. 1422)>

Read Only No

UpstreamConnectionTracker

No details are provided for this entry.

Properties

DisplayText

The general property that defines the user-visible name of an entity. This property is defined for all data entities but is used only in those entities that present a label in the user interface.

Type string (p. 1438)

Read Only No

MaterialDesigner

Material Designer

This container holds data for an instance of Material Designer.

Methods

ClearGeneratedData

ClearGeneratedData command clears the generated data of a Material Designer cell.

After successful execution of this command, the Material Designer component goes into "update required" state.

Edit

The Edit command starts a Material Designer session, if it is not already running. If a Material Designer session is already running, this command brings the Material Designer window in focus.

If a scdoc file is assigned to Material Designer component, the file is loaded in the Material Designer.

Available options:

Interactive	This optional boolean type argument indicates whether Material Designer is to be started in Interactive mode or batch mode. Its default value is True

Optional Arguments

Interactive Whether to edit in batch mode

Type bool (p. 1360)

Default Value True

StartupArguments Additional arguments to send to Material Designer. These are to be supplied as comand line options when we launch Material Designer.

Type string (p. 1438)

Exit

Exit command shuts down the running session of Material Designer. Before shuting down, Material Designer saves its database.

The session can not be closed if Material Designer is busy in generating features or seeking user's input. In such situations, this command throws an ApplicationBusyException exception.

GetMaterialDesignerProperties

Return a reference to DataEntity managing property settings of Material Designer Container.

Return Reference to DataEntity managing Material Designer properties

Type DataReference (p. 1371)

RefreshMD

Refresh command refreshes the input data in a Material Designer component by consuming all changed data from upstream (source) components. This command also updates the modified parameters in Material Designer.

After successful execution of this command, the Material Designer component goes into "update required" state.

RunScript

Executes a script in the assigned Material Designer, which must be running. It will accept Python (.py), SpaceClaim Script (.scscript) and SpaceClaim Journal (.scjournal) files.

Required Parameters:

ScriptFile	Path of the script file to be run.

Required Arguments

ScriptFile Path of the script file

Type string (p. 1438)

SendCommand

The SendCommand sends Python or SpaceClaim Journal command string to Material Designer for execution. If Material Designer is not open, it will be launched to execute commands, and then closed. If it is already running, it will keep running after the command execution.

If Material Designer is busy and not available for executing the instruction, the SendCommand throws an ApplicationBusyException exception.

Available options:

Command	Python or SpaceClaim journal command string containing scripting commands for Material Designer
Language	Language of the command. Allowed values are: "Python" and "SCJournal". The default language is "Python"

The following command will create a block body in the Material Designer editor.

Required Arguments

Command Command string

Type string (p. 1438)

Optional Arguments

Language Language of the command. The default value of Language is "Python".

Type string (p. 1438)

Default Value Python

Example

```
system1 = GetSystem(Name="MD")
materialDesigner1 = system1.GetContainer(ComponentName="MD")
materialDesigner1.SendCommand( Command = """result = BlockBody.Create(Point.Create(UM(-10), UM(-10), UM(0)
```

Stop

The Stop command shuts down the running session of Material Designer immediately, without saving its unsaved data.

Material Designer can not be stopped if it is busy. In such situations, this command throws an ApplicationBusyException exception.

Data Entities

MaterialDesigner

Material Designer data object

Properties

DisplayText

The general property that defines the user-visible name of an entity. This property is defined for all data entities but is used only in those entities that present a label in the user interface.

Type string (p. 1438)

Read Only No

GeometryFilePath

The location of file currently assigned to the Material Designer component. Material Designer processes this file when started through Edit command.

Type string (p. 1438)

Read Only No

PlugInName

Current PlugInName - Returns plugin name. This is not persisted.

Type string (p. 1438)

Read Only No

Mechanical APDL

Mechanical APDL

This container holds data for a Mechanical APDL analysis.

Methods

AddFile

Copies the specified file to the working directory of the Mechanical APDL editor and registers the file with the Workbench project.

AddInputFile

Specifies an input file containing APDL commands for the Mechanical APDL editor for execution when the editor opens. Copies the file to the application working directory and registers it with the Workbench project.

Edit

Opens the Mechanical APDL editor to allow modification of Analysis data.

Exit

Exits the Mechanical APDL editor.

GetAnalysisSettings

Query to return the reference to the container's AnalysisSettings data entity.

GetComponentSettingsForRsmDpUpdate

This query is used to obtain the ComponentSettingsForRsmDpUpdate object for Journaling and Scripting

GetMapdlInputFile

Query to return the reference to the container's MapdlSetup data entity.

GetMapdlSetup

Query to return the reference to the container's MapdlSetup data entity.

GetSolutionSettings

This query is used to obtain the solution settings object for Journaling and Scripting

SendCommand

Sends commands to the Mechanical APDL editor.

SwitchToBackgroundMode

Switch the Update in progress into background mode. This will enable operations that are not allowed during an Update in foreground mode (e.g. Project Save).

This command is not normally useful in a script. Journals may record the invocation of this command after an Update invoke, as the result of GUI activity while the Update is in progress. However, replay of these journals will always wait for the Update invoke to complete before invoking the next command, rendering this step ineffectual.

Data Entities

ExtendedComponentSettingsForRsmDpUpdate

Extended Component settings when solved as part of design point update via RSM Currently used for Addins with solvers that would like to take advantage of SMP

Properties

DisplayText

The general property that defines the user-visible name of an entity. This property is defined for all data entities but is used only in those entities that present a label in the user interface.

Type string (p. 1438)

Read Only No

LimitOnNumberOfCores

No details are provided for this entry.

Type int (p. 1394)

Read Only No

SerialOnly

No details are provided for this entry.

Type bool (p. 1360)

Read Only No

SharedMemoryParallel

A boolean flag indicating whether or not to restrict the solver to using SMP, instead of distributed parallel

Type bool (p. 1360)

Read Only No

UseLimitOnNumberOfCores

No details are provided for this entry.

Type bool (p. 1360)

Read Only No

HiddenPollingComponentSolveSettings

This class contains a derived version of solve settings where polling is disabled

Properties

ConfiguredQueue

A configured queue used for new RSM architecture

Type string (p. 1438)

Read Only No

DisplayText

The general property that defines the user-visible name of an entity. This property is defined for all data entities but is used only in those entities that present a label in the user interface.

Type string (p. 1438)

Read Only No

EnablePolling

whether enable polling during remote solve

Type bool (p. 1360)

Read Only No

ExecutionMode

execution mode

Type string (p. 1438)

Read Only No

NumberOfProcesses

number of processes for parallel solve

Type int (p. 1394)

Read Only No

PollingInterval

polling internal during remote solve

Type Quantity (p. 1422)

Read Only No

RsmJobName

RSM job name

Type string (p. 1438)

Read Only No

RsmQueueDetails

Configured queue details

Type RsmQueueDetails (p. 1427)

Read Only No

UpdateOption

Update mode

Type JobRunMode (p. 1397)

Read Only No

MapdlInputFile

Represents an input file for the Mechanical APDL editor.

Properties

DisplayText

The general property that defines the user-visible name of an entity. This property is defined for all data entities but is used only in those entities that present a label in the user interface.

Type string (p. 1438)

Read Only No

Methods

Delete

Deletes an input or reference file from the Mechanical APDL editor.

PublishMapdlParameter

Publishes a MAPDL variable located in an input file as a parameter.

SwitchInputOrder

Switches the order of two input files for the Mechanical APDL editor.

UnpublishMapdlParameter

a MAPDL variable located in an input file as a parameter.

MapdlReferenceFile

Represents a reference file for the Mechanical APDL editor.

Properties

DisplayText

The general property that defines the user-visible name of an entity. This property is defined for all data entities but is used only in those entities that present a label in the user interface.

Type string (p. 1438)

Read Only No

Methods

Delete

Deletes an input or reference file from the Mechanical APDL editor.

MapdlSetup

Represents the settings used to launch the Mechanical APDL editor.

Properties

CommandLineOptions

Additional command line options for the Mechanical APDL editor.

Type string (p. 1438)

Read Only No

CustomExecutablePath

Calls a custom ANSYS executable.

Type string (p. 1438)

Read Only No

DatabaseMemory

Defines the portion of workspace (memory) to be used for the database. The default is 512 MB for 64-bit machines, 256 MB for 32-bit machines.

Type int (p. 1394)

Read Only No

DisplayText

The general property that defines the user-visible name of an entity. This property is defined for all data entities but is used only in those entities that present a label in the user interface.

Type string (p. 1438)

Read Only No

Distributed

Enables Distributed ANSYS.

Type bool (p. 1360)

Read Only No

DownloadDistributedFiles

Indicates whether or not to download files from slave node scratches during distributed solves

Type bool (p. 1360)

Read Only No

GPUAccelerator

Specifies the type of GPU Accelerators. By default this is set to none meaning no GPU acceleration is used.

Type GPUAccelerator (p. 1389)

Read Only No

Graphics

Specifies the type of graphics device. This option applies only to interactive mode. For UNIX/Linux systems, graphics device choices are X11, X11C, or 3D. For Windows systems, graphics device options are WIN32 or WIN32C, or 3D.

Type string (p. 1438)

Read Only No

JobName

Specifies the initial jobname, a name assigned to all files generated by the program for a specific model. If you omit the -j option, the jobname is assumed to be file.

Type string (p. 1438)

Read Only No

License

Defines which ANSYS product will run during the session (ANSYS Multiphysics, ANSYS Structural, etc.).

Type string (p. 1438)

Read Only No

MachineList

Specifies the machines on which to run a Distributed ANSYS analysis.

Type string (p. 1438)

Read Only No

MPIType

Defines which type of MPI the solver should use.

Type MPIType (p. 1407)

Read Only No

NumberOfGpusPerMachine

Specifies the number of accelerators to use when running with GPU Acceleration.

Type int (p. 1394)

Read Only No

Processors

Specifies the number of processors to use when running Distributed ANSYS or Shared-memory ANSYS.

Type int (p. 1394)

Read Only No

ReadStartAns

Specifies whether the program reads the start121.ans file at start-up.

Type bool (p. 1360)

Read Only No

WorkspaceMemory

Specifies the total size of the workspace (memory) in megabytes. If you omit the -m option, the default is 1 GB (1024 MB) for 64-bit machines, 512 MB for 32-bit machines.

Type int (p. 1394)

Read Only No

Mechanical

Mechanical Enhanced Model

This container holds Imported Section data for an instance of Ansys Mechanical.

Methods

Edit

Opens the Mechanical editor and attaches geometry.

Optional Arguments

Hidden Specify if Mechanical will open in hidden mode. The default value is false.

Type bool (p. 1360)

Default Value False

Interactive Specify if Mechanical will open in interactive mode. The default value is true.

Type bool (p. 1360)

Default Value True

StartAsReadOnly Specify if Mechanical will open in read-only mode. The default value is false.

Type bool (p. 1360)

Default Value False

Example

To edit the enhanced model component with default optional parameter values:

```
enhancedModel.Edit()
```

Or by specifying optional parameter values:

```
enhancedModel.Edit(Interactive=True)
```

Exit

Exit the Mechanical editor

Optional Arguments

SaveDatabase Indicates whether the Mechanical database will be saved prior to exiting

Type bool (p. 1360)

Default Value True

Mechanical Model

This container holds Model data for an instance of Ansys Mechanical.

Methods

Edit

Opens the Mechanical editor and attaches geometry.

Optional Arguments

Hidden Specify if Mechanical will open in hidden mode. The default value is false.

Type bool (p. 1360)

Default Value False

Interactive Specify if Mechanical will open in interactive mode. The default value is true.

Type bool (p. 1360)

Default Value True

StartAsReadOnly Specify if Mechanical will open in read-only mode. The default value is false.

Type bool (p. 1360)

Default Value False

Example

To edit the model component with default optional parameter values:

```
model.Edit()
```

Or by specifying optional parameter values:

```
model.Edit(Interactive=True)
```

Exit

Exit the Mechanical editor

Optional Arguments

SaveDatabase Indicates whether the Mechanical database will be saved prior to exiting

Type bool (p. 1360)

Default Value True

Export

Exports a .dsdb file for the model component.

Required Arguments

FilePath The path and name of the .dsdb file to be written.

Type string (p. 1438)

ExportASMJournal

Exports an ASM Journal (.wbjn) and supporting files for the model component.

Required Arguments

FilePath The path and name of the .wbjn file to be written. Supporting files written alongside.

Type string (p. 1438)

ExportGeometry

Exports a PartManager database (.pmdb) file for the geometry in the model component.

Required Arguments

FilePath The path and name of the .pmdb file to be written.

Type string (p. 1438)

ExportMesh

Exports a .acmo file for the mesh in the model component.

Required Arguments

FilePath The path and name of the .acmo file to be written.

Type string (p. 1438)

GetACPImportOptions

Query to return the data reference to the acp import options

Return The Data Entity containing settings for this component.

Type DataReference (p. 1371)

Required Arguments

Name The entity of interest.

Type string (p. 1438)

GetMechanicalMesh

Query to return the reference to the container's MechanicalMesh data entity.

Return A reference to the requested MechanicalMesh data entity.

Type DataReference (p. 1371)

GetMechanicalMeshFile

Query to return the reference to the container's MechanicalMeshFile data entity.

Return A reference to the requested MechanicalMeshFile data entity.

Type DataReference (p. 1371)

GetMechanicalModel

Query to return the reference to the container's MechanicalModel data entity.

Return A reference to the requested MechanicalModel data entity.

Type DataReference (p. 1371)

GetMechanicalSystemType

Query to return the reference to the container's MechanicalSystemType data entity.

Return A reference to the requested MechanicalSystemType data entity.

Type DataReference (p. 1371)

GetMeshProperties

Query to return the reference to the container's MeshProperties data entity

Return The Data Entity containing settings for this component.

Type DataReference (p. 1371)

GetModelComponentProperties

Query to return the reference to the container's ModelComponentProperties data entity

Return The Data Entity containing settings for this component.

Type DataReference (p. 1371)

GetSimulationImportOptions

Query to return the data reference to the simulation import options

Return The Data Entity containing settings for this component.

Type DataReference (p. 1371)

Required Arguments

Name The entity of interest.

Type string (p. 1438)

ImportMesh

Imports a mesh file into Mechanical

Required Arguments

FilePath The path and name of the mesh file to be imported.

Type string (p. 1438)

IsMeshingEnabled Enable or disable meshing on the imported mesh

Type bool (p. 1360)

SendCommand

Executes a JScript or python command in the Mechanical editor.

If the Mechanical editor is not open, then the editor's GUI will not be available, causing some commands to fail. In this case, consider calling the container's Edit method to open the editor before using SendCommand.

Furthermore, if the Mechanical editor is not open, SendCommand will start the editor without the GUI, issue the specified command, and then close the editor. For multiple SendCommands, this may degrade performance.

Required Arguments

Command Command argument containing the command.

Type string (p. 1438)

Optional Arguments

ExtensionName Extension name for the command Only available with Python commmands and defaults to empty

Type string (p. 1438)

Language Language of the command. The default value of Language is "Javascript".

Type string (p. 1438)

Default Value Javascript

Example

To execute some arbitrary command (in this case, causing a dialog box to appear) in the Mechanical editor:

```
model.SendCommand(Command="WBScript.Out(\"My Text\",true);" )
```

To run a JScript file already saved to disk using Run Macro from Mechanical:

```
setup.SendCommand(Command="WB.AppletList.Applet(\"DSApplet\").App.Script.doToolsRunMacro(\"C:\\\\macro.js\
```

Data Entities

ACPImportOptions

Import options for model assembly available on the downstream model

Properties

DisplayText

The general property that defines the user-visible name of an entity. This property is defined for all data entities but is used only in those entities that present a label in the user interface.

Type string (p. 1438)

Read Only No

MaterialDesignerLoadCase

No details are provided for this entry.

Type LoadCaseType (p. 1400)

Read Only No

MirrorTransform

Mirror Transformations for the Source Geometry

Type MirrorTransformOptionsForSimulation (p. 1406)

Read Only No

NumberOfCopies

By default this value is 0, specifies how many additional copies of the upstream model are needed

Type int (p. 1394)

Read Only No

RenumberMeshElementsAutomatically

By default this value is true, specifies whether the user would like us to automatically renumber his mesh elements to prevent unique id conflicts during assembly

Type bool (p. 1360)

Read Only No

RigidTransform

Rigid Transformations for the Source Geometry

Type RigidTransformOptionsForSimulation (p. 1426)

Read Only No

SmoothingCdb

Which upstream CDB file generated by mechanical to use as our model source

Type string (p. 1438)

Read Only No

Source

The upstream model source

Type DataReference (p. 1371)

Read Only No

TransferType

Tag specified by the user to name objects in mechanical

Type TransferType (p. 1443)

Read Only No

TransformationType

Which type of transform we should display to the user

Type TransformType (p. 1444)

Read Only No

GeneralModelAssemblyProperties

Class used to expose general properties for the model assembly workflow

Properties

DisplayText

The general property that defines the user-visible name of an entity. This property is defined for all data entities but is used only in those entities that present a label in the user interface.

Type string (p. 1438)

Read Only No

GroupObjectsBySource

Group objects in Mechanical based on source

Type bool (p. 1360)

Read Only No

LengthUnit

Length unit of assembled model

Type string (p. 1438)

Read Only No

ObjectRenaming

Name objects in Mechanical based on selected option

Type ObjectRenamingTypeInMechanical (p. 1409)

Read Only No

MechanicalMesh

This is the mesh data entity object that will exist in the model container.

Properties

Caption

Caption used to identify the mesh.

Type string (p. 1438)

Read Only No

DisplayText

The general property that defines the user-visible name of an entity. This property is defined for all data entities but is used only in those entities that present a label in the user interface.

Type string (p. 1438)

Read Only No

MeshId

Mesh identifier that corresponds with Mesh ID in Mechanical

Type int (p. 1394)

Read Only No

MechanicalModel

The model data entity in the model container.

Properties

AcpRenumberingInformation

ACP automatic renumbering of transfer information

Type Dictionary (p. 1375)<string (p. 1438), List (p. 1400)<DataReference (p. 1371)>>

Read Only No

AllowMeshing

Property used to prevent a re meshing in model assembly workflows

Type bool (p. 1360)

Read Only No

Caption

Caption to identify the model

Type string (p. 1438)

Read Only No

DisplayText

The general property that defines the user-visible name of an entity. This property is defined for all data entities but is used only in those entities that present a label in the user interface.

Type string (p. 1438)

Read Only No

EdaFile

Reference to a material database

Type DataReference (p. 1371)

Read Only No

File

Reference to the Mechanical database

Type DataReference (p. 1371)

Read Only No

IsResetNotRequired

Property used to prevent cascasding resets on downstream containers

Type bool (p. 1360)

Read Only No

MeshFile

Reference to the Mechanical mesh database

Type DataReference (p. 1371)

Read Only No

ModelConversionAssociativityMap

ACP automatic renumbering of transfer information

Type Dictionary (p. 1375)<string (p. 1438), string (p. 1438)>

Read Only No

ModelId

Model identifier that corresponds with the ID on Model tree node in Mechanical

Type int (p. 1394)

Read Only No

PrimeMeshFile

Reference to the Engineering Model database

Type DataReference (p. 1371)

Read Only No

PrototypeId

Prototype identifier which corresponds to the ID on the Geometry tree node in Mechanical

Type int (p. 1394)

Read Only No

MechanicalSystemType

This entity provides string based information about the physics, analysis, and solver settings for the Mechanical system component.

Properties

AnalysisTypeDisplayString

The string which represents current analysis type setting.

Type string (p. 1438)

Read Only Yes

DisplayText

The general property that defines the user-visible name of an entity. This property is defined for all data entities but is used only in those entities that present a label in the user interface.

Type string (p. 1438)

Read Only No

PhysicsTypeDisplayString

The string which represents current physics type setting.

Type string (p. 1438)

Read Only Yes

SolverTypeDisplayString

The string which represents current solver type setting.

Type string (p. 1438)

Read Only Yes

MeshConversionOptionsEntity

(Beta) Entity to control the Skin Detection algorithm for importing cdb files.

Properties

BodyGroupingType

Body grouping type

Type BodyGroupingType (p. 1359)

Read Only No

ComponentKey

If the nodal components will be processed during CDB components the nodal component key will allow filtering of which components to process

Type string (p. 1438)

Read Only No

CreateGeometry

Flag to create geometry

Type bool (p. 1360)

Read Only No

CreateGeometryEdgeComponents

Should the geometry edge named selections be created during CDB conversion

Type bool (p. 1360)

Read Only No

CreateGeometryFaceComponents

Should the geometry face named selections be created during CDB conversion

Type bool (p. 1360)

Read Only No

CreateGeometryVertexComponents

Should the geometry vertex named selections be created during CDB conversion

Type bool (p. 1360)

Read Only No

CutAngle

Only displayed if Forbid Close Components equals Yes. It is the angle used to cut closed surfaces to separate the elements into components.

Type Quantity (p. 1422)

Read Only No

DisplayText

The general property that defines the user-visible name of an entity. This property is defined for all data entities but is used only in those entities that present a label in the user interface.

Type string (p. 1438)

Read Only No

ForbidCloseComponents

Option to split closed surfaces into several components.If Yes then the algorithm cuts closed surfaces into several components. It provides a simple method to avoid problematic faces on closed surfaces.

Type bool (p. 1360)

Read Only No

GeometryImportAnalysisType

The geometry attach type for the CDB file

Type GeometryAttachType (p. 1388)

Read Only No

ProcessLineBodies

Should line bodies be processed during the CDB conversion

Type bool (p. 1360)

Read Only No

ToleranceAngle

The tolerance angle to seperate two elements into seperate components. If the angle between the normals of two adjacent elements is less than or equal to the Tolerance Angle then the two elements are in the same component, otherwise, they are separated.

Type Quantity (p. 1422)

Read Only No

VertexInsertionAngle

The Vertex Insertion Angle is the minimunm angle to insert a vertex between two free edges of mesh. During the generation of the geometry, if two segments of an edge abruptly make an angle greater than the Vertex Insertion Angle, then the edge is split and a vertex is inserted.

Type Quantity (p. 1422)

Read Only No

ModelComponentProperties

No details are provided for this entry.

Properties

DisplayText

The general property that defines the user-visible name of an entity. This property is defined for all data entities but is used only in those entities that present a label in the user interface.

Type string (p. 1438)

Read Only No

License

No details are provided for this entry.

Type string (p. 1438)

Read Only No

ModelOutputSettingsForACP

Entity to control the properties of mesh file generated by Mechanical Model.

Properties

DisplayText

The general property that defines the user-visible name of an entity. This property is defined for all data entities but is used only in those entities that present a label in the user interface.

Type string (p. 1438)

Read Only No

LengthUnit

Length unit to write mesh file.

Type string (p. 1438)

Read Only No

SimulationImportOptions

Import options for model assembly available on the downstream model

Properties

DisplayText

The general property that defines the user-visible name of an entity. This property is defined for all data entities but is used only in those entities that present a label in the user interface.

Type string (p. 1438)

Read Only No

MaterialDesignerLoadCase

No details are provided for this entry.

Type LoadCaseType (p. 1400)

Read Only No

MirrorTransform

Mirror Transformations for the Source Geometry

Type MirrorTransformOptionsForSimulation (p. 1406)

Read Only No

NumberOfCopies

By default this value is 0, specifies how many additional copies of the upstream model are needed

Type int (p. 1394)

Read Only No

RenumberMeshElementsAutomatically

By default this value is true, specifies whether the user would like us to automatically renumber his mesh elements to prevent unique id conflicts during assembly

Type bool (p. 1360)

Read Only No

RigidTransform

Rigid Transformations for the Source Geometry

Type RigidTransformOptionsForSimulation (p. 1426)

Read Only No

SmoothingCdb

Which upstream CDB file generated by mechanical to use as our model source

Type string (p. 1438)

Read Only No

Source

The upstream model source

Type DataReference (p. 1371)

Read Only No

TransformationType

Which type of transform we should display to the user

Type TransformType (p. 1444)

Read Only No

SimulationMeshProperties

Entity to control mesh properties for a Mechanical Model

Properties

DisplayText

The general property that defines the user-visible name of an entity. This property is defined for all data entities but is used only in those entities that present a label in the user interface.

Type string (p. 1438)

Read Only No

saveMeshFileInSeparateFile

Group objects in Meshing based on source

Type bool (p. 1360)

Read Only No

Mechanical Results

This container holds Results data for an instance of Ansys Mechanical.

Methods

Edit

Opens the Mechanical editor and attaches geometry.

Optional Arguments

ForceGui Specify if Mechanical is required to open in GUI mode. This is needed for Topology Optimization to solve correctly during DP updates.

Type bool (p. 1360)

Default Value False

Hidden Specify if Mechanical will open in hidden mode. The default value is false.

Type bool (p. 1360)

Default Value False

Interactive Specify if Mechanical will open in interactive mode. The default value is true.

Type bool (p. 1360)

Default Value True

StartAsReadOnly Specify if Mechanical will open in read-only mode. The default value is false.

Type bool (p. 1360)

Default Value False

Example

To edit the results component with default optional parameter values.

```
result.Edit()
```

Or by specifying optional parameter values:

```
result.Edit(Interactive=True)
```

EditPost

Opens the Mechanical editor and attaches geometry.

Example

To edit the results component with default optional parameter values.

```
result.Edit()
```

Or by specifying optional parameter values:

```
result.Edit(Interactive=True)
```

Exit

Exit the Mechanical editor

Optional Arguments

SaveDatabase Indicates whether the Mechanical database will be saved prior to exiting

Type bool (p. 1360)

Default Value True

GetMechanicalSystemType

Query to return the reference to the container's MechanicalSystemType data entity.

Return A reference to the requested MechanicalSystemType data entity.

Type DataReference (p. 1371)

GetModalOptionsForCdb

Query to return the component reference for a given container and component base name.

Return A reference to the requested data entity.

Type DataReference (p. 1371)

GetPhysicsType

Query to return the reference to the container's MechanicalSystemType data entity.

Return A reference to the requested MechanicalSystemType data entity.

Type DataReference (p. 1371)

GetSimulationImportOptionsForResult

Query to return the component reference for a given container and component base name.

Return A reference to the requested data entity.

Type DataReference (p. 1371)

GetSimulationResultFile

Query to return the component reference for a given container and component base name.

Return A reference to the requested data entity.

Type DataReference (p. 1371)

GetStaticOptionsForCdb

Query to return the component reference for a given container and component base name.

Return A reference to the requested data entity.

Type DataReference (p. 1371)

SendCommand

Executes a JScript or python command in the Mechanical editor.

If the Mechanical editor is not open, then the editor's GUI will not be available, causing some commands to fail. In this case, consider calling the container's Edit method to open the editor before using SendCommand.

Furthermore, if the Mechanical editor is not open, SendCommand will start the editor without the GUI, issue the specified command, and then close the editor. For multiple SendCommands, this may degrade performance.

Required Arguments

Command Command argument containing the command.

Type string (p. 1438)

Optional Arguments

ExtensionName Extension name for the command Only available with Python commmands and defaults to empty

Type string (p. 1438)

Language Language of the command. The default value of Language is "Javascript".

Type string (p. 1438)

Default Value Javascript

Example

To execute some arbitrary command (in this case, causing a dialog box to appear) in the Mechanical editor:

```
model.SendCommand(Command="WBScript.Out(\"My Text\",true);" )
```

To run a JScript file already saved to disk using Run Macro from Mechanical:

```
setup.SendCommand(Command="WB.AppletList.Applet(\"DSApplet\").App.Script.doToolsRunMacro(\"C:\\\\macro.js\
```

Data Entities

ModalOptionsForCdb

ModalOptionsForCdb

Properties

DisplayText

The general property that defines the user-visible name of an entity. This property is defined for all data entities but is used only in those entities that present a label in the user interface.

Type string (p. 1438)

Read Only No

Mode

Time

Type int (p. 1394)

Read Only No

ScaleFactor

Scale Factor

Type double (p. 1378)

Read Only No

SimulationImportOptionsForResult

Import options for model assembly available on the downstream model

Properties

DisplayText

The general property that defines the user-visible name of an entity. This property is defined for all data entities but is used only in those entities that present a label in the user interface.

Type string (p. 1438)

Read Only No

LengthUnit

Length unit of cdb

Type string (p. 1438)

Read Only No

NumberOfCopies

By default this value is 0, specifies how many additional copies of the upstream model are needed

Type int (p. 1394)

Read Only No

RenumberMeshElementsAutomatically

By default this value is true, specifies whether the user would like us to automatically renumber his mesh elements to prevent unique id conflicts during assembly

Type bool (p. 1360)

Read Only No

RigidTransform

Rigid Transformations for the Source Geometry

Type RigidTransformOptionsForSimulation (p. 1426)

Read Only No

Source

The upstream model source

Type string (p. 1438)

Read Only No

SimulationResultsStlZip

DataTransfer provider that holds the file reference to a zipped folder of STL and pmdb files.

Properties

DisplayText

The general property that defines the user-visible name of an entity. This property is defined for all data entities but is used only in those entities that present a label in the user interface.

Type string (p. 1438)

Read Only No

ExportTopOptAction

No details are provided for this entry.

Type SimulationTopOptAction (p. 1432)

Read Only No

TransferDataObject

The data transfer object provided by this provider

Type DataReference (p. 1371)

Read Only No

TransferFile

No details are provided for this entry.

Type DataReference (p. 1371)

Read Only Yes

VersionNumber

No details are provided for this entry.

Type int (p. 1394)

Read Only Yes

StaticOptionsForCdb

StaticOptionsForCdb

Properties

DisplayText

The general property that defines the user-visible name of an entity. This property is defined for all data entities but is used only in those entities that present a label in the user interface.

Type string (p. 1438)

Read Only No

ScaleFactor

Scale Factor

Type double (p. 1378)

Read Only No

Time

Time

Type double (p. 1378)

Read Only No

Mechanical Setup

This container holds Set Up data for an instance of Ansys Mechanical.

Methods

Edit

Opens the Mechanical editor and attaches geometry.

Optional Arguments

Hidden Specify if Mechanical will open in hidden mode. The default value is false.

Type bool (p. 1360)

Default Value False

Interactive Specify if Mechanical will open in interactive mode. The default value is true.

Type bool (p. 1360)

Default Value True

StartAsReadOnly Specify if Mechanical will open in read-only mode. The default value is false.

Type bool (p. 1360)

Default Value False

Example

To edit the setup component with default optional parameter values.

```
setup.Edit()
```

Or by specifying optional parameter values:

```
setup.Edit(Interactive=True)
```

Exit

Exit the Mechanical editor

Optional Arguments

SaveDatabase Indicates whether the Mechanical database will be saved prior to exiting

Type bool (p. 1360)

Default Value True

Export

Writes either an APDL input file (for use with the ANSYS solver) or a CAE Representation file (for use with any solver).

Required Arguments

Path If the SetupDataType is 'InputFile', this should be a file path. If the SetupDataType is 'CAERepresentation', this should be a folder path.

Type string (p. 1438)

SetupDataType Type of setup data to write to disk. Available options are: 'InputFile' and 'CAERepresentation'.

Type string (p. 1438)

GetMechanicalSetupFile

Query to return the reference to the container's MechanicalSetupFile data entity.

Return A reference to the requested MechanicalSetupFile data entity.

Type DataReference (p. 1371)

GetMechanicalSystemType

Query to return the reference to the container's MechanicalSystemType data entity.

Return A reference to the requested MechanicalSystemType data entity.

Type DataReference (p. 1371)

GetPhysicsType

Query to return the reference to the container's MechanicalSystemType data entity.

Return A reference to the requested MechanicalSystemType data entity.

Type DataReference (p. 1371)

SendCommand

Executes a JScript or python command in the Mechanical editor.

If the Mechanical editor is not open, then the editor's GUI will not be available, causing some commands to fail. In this case, consider calling the container's Edit method to open the editor before using SendCommand.

Furthermore, if the Mechanical editor is not open, SendCommand will start the editor without the GUI, issue the specified command, and then close the editor. For multiple SendCommands, this may degrade performance.

Required Arguments

Command Command argument containing the command.

Type string (p. 1438)

Optional Arguments

ExtensionName Extension name for the command Only available with Python commmands and defaults to empty

Type string (p. 1438)

Language Language of the command. The default value of Language is "Javascript".

Type string (p. 1438)

Default Value Javascript

Example

To execute some arbitrary command (in this case, causing a dialog box to appear) in the Mechanical editor:

```
model.SendCommand(Command="WBScript.Out(\"My Text\",true);" )
```

To run a JScript file already saved to disk using Run Macro from Mechanical:

```
setup.SendCommand(Command="WB.AppletList.Applet(\"DSApplet\").App.Script.doToolsRunMacro(\"C:\\\\macro.js\
```

SetDesignAssessmentFile

Set the design assessment

Required Arguments

FilePath The design assessment file's path.

Type string (p. 1438)

Data Entities

DesignAssessmentSetupSettingsType

The data entity that holds the setup properties for a Design Assessment.

Properties

DisplayText

The general property that defines the user-visible name of an entity. This property is defined for all data entities but is used only in those entities that present a label in the user interface.

Type string (p. 1438)

Read Only No

SolverTarget

The type of assessment to be performed.

Type string (p. 1438)

Read Only No

UserAttributeFile

The file to specify attributes to perform a User Defined design assessment.

Type string (p. 1438)

Read Only No

Mechanical Solution

This container holds Solution data for an instance of Ansys Mechanical.

Methods

Edit

Opens the Mechanical editor and attaches geometry.

Optional Arguments

Hidden Specify if Mechanical will open in hidden mode. The default value is false.

Type bool (p. 1360)

Default Value False

Interactive Specify if Mechanical will open in interactive mode. The default value is true.

Type bool (p. 1360)

Default Value True

StartAsReadOnly Specify if Mechanical will open in read-only mode. The default value is false.

Type bool (p. 1360)

Default Value False

Example

To edit the solution component with default optional parameter values.

```
solution.Edit()
```

Or by specifying optional parameter values:

```
solution.Edit(Interactive=True)
```

Exit

Exit the Mechanical editor

Optional Arguments

SaveDatabase Indicates whether the Mechanical database will be saved prior to exiting

Type bool (p. 1360)

Default Value True

Export

Exports the results(*.db, *.rth, *.rmg, and *.rst) files to given DirectoryPath.

Required Arguments

DirectoryPath Directory path to export the result files to.

Type string (p. 1438)

GetComponentSettingsForRsmDpUpdate

This query is used to obtain the ComponentSettingsForRsmDpUpdate object for Journaling and Scripting

GetExpertProperties

This query is used to obtain the ExpertProperties object for Journaling and Scripting

GetMechanicalSystemType

Query to return the reference to the container's MechanicalSystemType data entity.

Return A reference to the requested MechanicalSystemType data entity.

Type DataReference (p. 1371)

GetModalUpdateOptionsForCdbTransfer

Query to return the data reference to the modal options for solution to model

Return A reference to the requested data entity.

Type DataReference (p. 1371)

Required Arguments

Name The entity of interest.

Type string (p. 1438)

GetPhysicsType

Query to return the reference to the container's MechanicalSystemType data entity.

Return A reference to the requested MechanicalSystemType data entity.

Type DataReference (p. 1371)

GetRBDUpdateOptionsForMechdbTransfer

Query to return the data reference to the static options for solution to model

Return A reference to the requested data entity.

Type DataReference (p. 1371)

Required Arguments

Name The entity of interest.

Type string (p. 1438)

GetSolutionSettings

Returns a DataReference to the Solution Settings object for this container

Return The Data Entity containing settings for this component.

Type DataReference (p. 1371)

GetStaticUpdateOptionsForCdbTransfer

Query to return the data reference to the static options for solution to model

Return A reference to the requested data entity.

Type DataReference (p. 1371)

Required Arguments

Name The entity of interest.

Type string (p. 1438)

SendCommand

Executes a JScript or python command in the Mechanical editor.

If the Mechanical editor is not open, then the editor's GUI will not be available, causing some commands to fail. In this case, consider calling the container's Edit method to open the editor before using SendCommand.

Furthermore, if the Mechanical editor is not open, SendCommand will start the editor without the GUI, issue the specified command, and then close the editor. For multiple SendCommands, this may degrade performance.

Required Arguments

Command Command argument containing the command.

Type string (p. 1438)

Optional Arguments

ExtensionName Extension name for the command Only available with Python commmands and defaults to empty

Type string (p. 1438)

Language Language of the command. The default value of Language is "Javascript".

Type string (p. 1438)

Default Value Javascript

Example

To execute some arbitrary command (in this case, causing a dialog box to appear) in the Mechanical editor:

```
model.SendCommand(Command="WBScript.Out(\"My Text\",true);" )
```

To run a JScript file already saved to disk using Run Macro from Mechanical:

```
setup.SendCommand(Command="WB.AppletList.Applet(\"DSApplet\").App.Script.doToolsRunMacro(\"C:\\\\macro.js\
```

Data Entities

ExtendedComponentSettingsForRsmDpUpdate

Extended Component settings when solved as part of design point update via RSM Currently used for Addins with solvers that would like to take advantage of SMP

Properties

DisplayText

The general property that defines the user-visible name of an entity. This property is defined for all data entities but is used only in those entities that present a label in the user interface.

Type string (p. 1438)

Read Only No

LimitOnNumberOfCores

No details are provided for this entry.

Type int (p. 1394)

Read Only No

SerialOnly

No details are provided for this entry.

Type bool (p. 1360)

Read Only No

SharedMemoryParallel

A boolean flag indicating whether or not to restrict the solver to using SMP, instead of distributed parallel

Type bool (p. 1360)

Read Only No

UseLimitOnNumberOfCores

No details are provided for this entry.

Type bool (p. 1360)

Read Only No

ModalUpdateOptionsForCdbTransfer

ModalUpdateOptionsForCdbTransfer

Properties

DisplayText

The general property that defines the user-visible name of an entity. This property is defined for all data entities but is used only in those entities that present a label in the user interface.

Type string (p. 1438)

Read Only No

ElementComponentKey

Filtering the element components

Type string (p. 1438)

Read Only No

FaceComponentKey

Filtering the face components

Type string (p. 1438)

Read Only No

Mode

Time

Type int (p. 1394)

Read Only No

NodalComponentKey

Filtering the nodal components

Type string (p. 1438)

Read Only No

ProcessElementComponents

Should we process the element components for the system

Type bool (p. 1360)

Read Only No

ProcessFaceComponents

Should we process the face components for the system

Type bool (p. 1360)

Read Only No

ProcessModelData

Should we process the coordinate systems for the system

Type bool (p. 1360)

Read Only No

ProcessNodalComponents

Should we process the nodal components for the system

Type bool (p. 1360)

Read Only No

ScaleFactor

Scale Factor

Type double (p. 1378)

Read Only No

RBDUpdateOptionsForMechdbTransfer

RBDUpdateOptionsForMechdbTransfer

Properties

DisplayText

The general property that defines the user-visible name of an entity. This property is defined for all data entities but is used only in those entities that present a label in the user interface.

Type string (p. 1438)

Read Only No

ElementComponentKey

Filtering the element components

Type string (p. 1438)

Read Only No

FaceComponentKey

Filtering the face components

Type string (p. 1438)

Read Only No

NodalComponentKey

Filtering the nodal components

Type string (p. 1438)

Read Only No

ProcessElementComponents

Should we process the element components for the system

Type bool (p. 1360)

Read Only No

ProcessFaceComponents

Should we process the face components for the system

Type bool (p. 1360)

Read Only No

ProcessModelData

Should we process the coordinate systems for the system

Type bool (p. 1360)

Read Only No

ProcessNodalComponents

Should we process the nodal components for the system

Type bool (p. 1360)

Read Only No

ScaleFactor

Scale Factor

Type double (p. 1378)

Read Only No

Time

Actual time

Type double (p. 1378)

Read Only No

TimeOption

Time option

Type TimeOption (p. 1442)

Read Only No

SimulationSolutionSettings

Mechanical specific simulation solution settings entity used for Journaling and Scripting. Not based on the common Infrastructure.Rsm.Queries entity.

Properties

DisplayText

The general property that defines the user-visible name of an entity. This property is defined for all data entities but is used only in those entities that present a label in the user interface.

Type string (p. 1438)

Read Only No

Queue

The queue that will be used on RSM task

Type string (p. 1438)

Read Only Yes

SolveManager

Solve Manager used on RSM task.

Type string (p. 1438)

Read Only Yes

SolveProcessSetting

Solve process setting used on RSM task.

Type string (p. 1438)

Read Only No

UpdateOption

Used to specify the type of update, local or RSM.

Type string (p. 1438)

Read Only No

SimulationSolutionSpeosObj

DataTransfer provider that holds the file reference to a obj file.

Properties

DisplayText

The general property that defines the user-visible name of an entity. This property is defined for all data entities but is used only in those entities that present a label in the user interface.

Type string (p. 1438)

Read Only No

TransferDataObject

The data transfer object provided by this provider

Type DataReference (p. 1371)

Read Only No

TransferFile

No details are provided for this entry.

Type DataReference (p. 1371)

Read Only Yes

VersionNumber

No details are provided for this entry.

Type int (p. 1394)

Read Only Yes

SimulationSolutionToSpaceClaimObj

DataTransfer provider that holds the DataTransfer Objects from Simulation Solution to SpaceClaim.

Properties

DisplayText

The general property that defines the user-visible name of an entity. This property is defined for all data entities but is used only in those entities that present a label in the user interface.

Type string (p. 1438)

Read Only No

VersionNumber

No details are provided for this entry.

Type int (p. 1394)

Read Only Yes

StaticUpdateOptionsForCdbTransfer

StaticUpdateOptionsForCdbTransfer

Properties

DisplayText

The general property that defines the user-visible name of an entity. This property is defined for all data entities but is used only in those entities that present a label in the user interface.

Type string (p. 1438)

Read Only No

ElementComponentKey

Filtering the element components

Type string (p. 1438)

Read Only No

FaceComponentKey

Filtering the face components

Type string (p. 1438)

Read Only No

NodalComponentKey

Filtering the nodal components

Type string (p. 1438)

Read Only No

ProcessElementComponents

Should we process the element components for the system

Type bool (p. 1360)

Read Only No

ProcessFaceComponents

Should we process the face components for the system

Type bool (p. 1360)

Read Only No

ProcessModelData

Should we process the coordinate systems for the system

Type bool (p. 1360)

Read Only No

ProcessNodalComponents

Should we process the nodal components for the system

Type bool (p. 1360)

Read Only No

ScaleFactor

Scale Factor

Type double (p. 1378)

Read Only No

Time

Actual time

Type double (p. 1378)

Read Only No

TimeOption

Time option

Type TimeOption (p. 1442)

Read Only No

Mesh

Mesh

This container holds Mesh data for an instance of the Meshing application.

Methods

Edit

Opens the Meshing editor. If the editor has never been opened, the command will import the geometry file associated with the Geometry cell in the Mesh system.

Optional Arguments

Interactive Specifies whether the editor should open in interactive mode. The default value is true.

Type bool (p. 1360)

Default Value True

Example

To edit the mesh component with default optional parameter values:

```
mesh.Edit()
```

Or by specifying optional parameter values:

```
mesh.Edit(Interactive=True )
```

Exit

Exits the Meshing editor.

Optional Arguments

SaveDatabase Bool flag to save the database on exit of the editor

Type bool (p. 1360)

Default Value True

Export

Saves the current state of the Meshing editor and exports a *.meshdat file. If the Meshing editor is not currently open, the cached state (the state of the editor the last time it was closed) of the Meshing editor will be exported.

Required Arguments

FilePath The path and name of the .meshdat file to be written.

Type string (p. 1438)

ExportASMJournal

Exports an ASM Journal (.wbjn) and supporting files for the model component.

Required Arguments

FilePath The path and name of the .wbjn file to be written. Supporting files written alongside.

Type string (p. 1438)

ExportGeometry

Exports a PartManager database (.pmdb) file for the geometry in the model component.

Required Arguments

FilePath The path and name of the .pmdb file to be written.

Type string (p. 1438)

ExportMesh

Exports a .acmo file for the mesh in the model component.

Required Arguments

FilePath The path and name of the .acmo file to be written.

Type string (p. 1438)

GetMechanicalMeshFile

Returns the data reference to the container's MechanicalMeshFile data entity.

Return A reference to the requested MechanicalMeshFile data entity.

Type DataReference (p. 1371)

GetMechanicalModel

Returns the data reference to the container's MechanicalModel data entity.

Return A reference to the requested MechanicalModel data entity.

Type DataReference (p. 1371)

GetMechanicalSystemType

Returns the data reference to the container's MechanicalSystemType data entity.

Return A reference to the requested MechanicalSystemType data entity.

Type DataReference (p. 1371)

GetMeshingImportOptions

Query to return the data reference to the meshing import options

Return The Data Entity containing settings for this component.

Type DataReference (p. 1371)

Required Arguments

Name The entity of interest.

Type string (p. 1438)

GetMeshProperties

Query to return the data reference to the meshing properties

Return The Data Entity containing settings for this component.

Type DataReference (p. 1371)

Import

Import a mesh data file. This action will delete the geometry container from the Mesh system and convert the Mesh cell into a file container. It is not possible to view meshes that are imported through this command in the Meshing editor; however, the files imported can be transferred to downstream mesh consumers (such as CFX, FLUENT, etc.) by way of normal component to component data transfer.

Required Arguments

FilePath The data file to be imported.

Type string (p. 1438)

MeshType The type of mesh data file.

Type MeshFileType (p. 1403)

Example

To import a data file into the mesh component with default optional parameter values:

```
mesh.Import(FilePath=r"C:\temp\data.cfx", MeshType="CFX")
```

SendCommand

Sends commands to the Meshing editor using JScript or Python syntax.

If the Meshing editor is not open, then the editor's GUI will not be available to the script, causing some commands to fail. In this case, consider calling Edit() to open the editor before using SendCommand. Furthermore,if the Meshing editor is not open,SendCommand will start the editor without GUI,issue the specified command,and then close the editor. For multiple SendCommands,this may degrade performance.

Required Arguments

Command The command(s) to execute in the Meshing editor.

Type string (p. 1438)

Optional Arguments

Language Language of the command. The default value of Language is "Javascript".

Type string (p. 1438)

Default Value Javascript

Example

To execute some arbitrary command (in this case, causing a dialog box to appear) in the Meshing editor:

```
mesh.SendCommand(Command=r"WBScript.Out("My Text",true);" )
```

Data Entities

GeneralMeshAssemblyProperties

Class used to expose general properties for the mesh assembly workflow

Properties

DisplayText

The general property that defines the user-visible name of an entity. This property is defined for all data entities but is used only in those entities that present a label in the user interface.

Type string (p. 1438)

Read Only No

GroupObjectsBySource

Group objects in Meshing based on source

Type bool (p. 1360)

Read Only No

LengthUnit

Target Length Unit for Assembled Model

Type string (p. 1438)

Read Only No

ObjectRenaming

Name objects in Meshing based on selected option

Type ObjectRenamingTypeInMeshing (p. 1409)

Read Only No

MechanicalModel

The data entity that contains the identifiers which maintain a relationship between the Mesh data contain and the objects in the Meshing editor tree.

Properties

Caption

Caption to identify the model.

Type string (p. 1438)

Read Only No

DisplayText

The general property that defines the user-visible name of an entity. This property is defined for all data entities but is used only in those entities that present a label in the user interface.

Type string (p. 1438)

Read Only No

ModelId

Identifier which corresponds to the Model object in the Meshing editor tree. This ID can be used in conjunction with SendCommand to perform operations such as inserting mesh controls or manipulating mesh settings within the Meshing editor itself.

Type int (p. 1394)

Read Only No

PrototypeId

Identifier which corresponds to the Geometry object in the Meshing editor tree. This ID can be used in conjunction with SendCommand to perform operations such as inserting mesh controls or manipulating mesh settings within the Meshing editor itself.

Type int (p. 1394)

Read Only No

MechanicalSystemType

This entity provides string based information about the physics, analysis, and solver settings for the Mesh component. As the Mesh component may be used independent of any specific physics, analysis, or solver, the properties exposed by this entity may take a value of "Any", meaning simply that it is up to the user to initialize and correctly configure the mesh settings within the Meshing editor.

Properties

AnalysisTypeDisplayString

The string which represents the current analysis type setting - this value will always be "Any".

Type string (p. 1438)

Read Only Yes

DisplayText

The general property that defines the user-visible name of an entity. This property is defined for all data entities but is used only in those entities that present a label in the user interface.

Type string (p. 1438)

Read Only No

ICEngineAutoSetup

The boolean which represents the IC Engine mesh setup options.

Type bool (p. 1360)

Read Only No

ICEngineMeshSettings

The string is a hyper link which opens the IC Engine mesh setting dialog.

Type string (p. 1438)

Read Only No

PhysicsTypeDisplayString

The string which represents current physics type setting - the value can be "Any" or "CFD".

Type string (p. 1438)

Read Only Yes

RefMeshSize

The quantity represent the reference mesh size for IC Engine mesh settings.

Type Quantity (p. 1422)

Read Only No

SolverTypeDisplayString

The string which represents the current solver setting - the value can be "Any", "FLUENT", "CFX", or "POLYFLOW".

Type string (p. 1438)

Read Only Yes

MeshConversionOptions

(Beta) Entity to control the Skin Detection algorithm for importing cdb files.

Properties

BodyGroupingType

Body grouping type

Type BodyGroupingType (p. 1359)

Read Only No

ComponentKey

If the nodal components will be processed during CDB components the nodal component key will allow filtering of which components to process

Type string (p. 1438)

Read Only No

CreateGeometry

Flag to create geometry

Type bool (p. 1360)

Read Only No

CreateGeometryEdgeComponents

Should the geometry edge named selections be created during CDB conversion

Type bool (p. 1360)

Read Only No

CreateGeometryFaceComponents

Should the geometry face named selections be created during CDB conversion

Type bool (p. 1360)

Read Only No

CreateGeometryVertexComponents

Should the geometry vertex named selections be created during CDB conversion

Type bool (p. 1360)

Read Only No

CutAngle

Only displayed if Forbid Close Components equals Yes. It is the angle used to cut closed surfaces to separate the elements into components.

Type Quantity (p. 1422)

Read Only No

DisplayText

The general property that defines the user-visible name of an entity. This property is defined for all data entities but is used only in those entities that present a label in the user interface.

Type string (p. 1438)

Read Only No

ForbidCloseComponents

Option to split closed surfaces into several components.If Yes then the algorithm cuts closed surfaces into several components. It provides a simple method to avoid problematic faces on closed surfaces.

Type bool (p. 1360)

Read Only No

GeometryImportAnalysisType

The geometry attach type for the CDB file

Type GeometryAttachType (p. 1388)

Read Only No

ProcessLineBodies

Should line bodies be processed during the CDB conversion

Type bool (p. 1360)

Read Only No

ToleranceAngle

The tolerance angle to seperate two elements into seperate components. If the angle between the normals of two adjacent elements is less than or equal to the Tolerance Angle then the two elements are in the same component, otherwise, they are separated.

Type Quantity (p. 1422)

Read Only No

VertexInsertionAngle

The Vertex Insertion Angle is the minimunm angle to insert a vertex between two free edges of mesh. During the generation of the geometry, if two segments of an edge abruptly make an angle greater than the Vertex Insertion Angle, then the edge is split and a vertex is inserted.

Type Quantity (p. 1422)

Read Only No

MeshingImportOptions

Import options for model assembly available on the downstream model

Properties

DisplayText

The general property that defines the user-visible name of an entity. This property is defined for all data entities but is used only in those entities that present a label in the user interface.

Type string (p. 1438)

Read Only No

MirrorTransform

Mirror Transformations for the Source Geometry

Type MirrorTransformOptionsForMeshing (p. 1405)

Read Only No

NumberOfCopies

By default this value is 0, specifies how many additional copies of the upstream mesh are needed

Type int (p. 1394)

Read Only No

RenumberMeshElementsAutomatically

By default this value is true, specifies whether the user would like us to automatically renumber his mesh elements to prevent unique id conflicts during assembly

Type bool (p. 1360)

Read Only No

RigidTransform

Rigid Transformations for the Source Geometry

Type RigidTransformOptionsForMeshing (p. 1426)

Read Only No

Source

The upstream mesh source

Type DataReference (p. 1371)

Read Only No

TransformationType

Which type of transform we should display to the user

Type TransformType (p. 1444)

Read Only No

MeshProperties

Contains properties to control mesh generation in Meshing application

Properties

DisplayText

The general property that defines the user-visible name of an entity. This property is defined for all data entities but is used only in those entities that present a label in the user interface.

Type string (p. 1438)

Read Only No

saveMeshFileInSeparateFile

Group objects in Meshing based on source

Type bool (p. 1360)

Read Only No

Microsoft Office Excel Analysis

Microsoft Office Excel Analysis

This container holds information to expose data from an instance of Microsoft Excel as Workbench parameters.

Methods

GetExcelSetup

Get the DataReference of the MSExcelSetup entity. There is only one MSExcelSetup entity per Analysis container. An exception is thrown if the entity is not found.

Return The DataReference of the MSExcelSetup entity.

Type DataReference (p. 1371)

Example

The following example shows how the user can get a setup entity to set a different NamedRangeKey value.

```
system1 = GetSystem(Name="XLS")
analysis1 = system1.GetContainer(ComponentName="Analysis")
setup1 = analysis1.GetExcelSetup()
setup1.NamedRangeKey = "MyPrefix"
```

Data Entities

MSExcelFile

This entity represents an Excel file added to the Analysis container in order to expose data as Workbench parameters and perform calculations based on parameters. It is created by the MSExcelSetup.AddFile method which copies and registers the original user's file into the project files.

Properties

DisplayText

The general property that defines the user-visible name of an entity. This property is defined for all data entities but is used only in those entities that present a label in the user interface.

Type string (p. 1438)

Read Only No

ErrorMessage

Error message if the calculation failed.

Type string (p. 1438)

Read Only No

FileName

Name of the file.

Type string (p. 1438)

Read Only No

MacroName

Name of the macro used to calculate the workbook.

Type string (p. 1438)

Read Only No

OriginalFilePath

Original Path of the file.

Type string (p. 1438)

Read Only No

State

Calculation state of the file.

Type Status (p. 1436)

Read Only No

UseMacro

Set to True if a Visual Basic macro is used to calculate the workbook.

Type bool (p. 1360)

Read Only No

Methods

GetRange

Gets the DataReference of an MSExcelRange entity from its name. An exception is thrown if the entity is not found.

Return The DataRefence of the MSExcelRange entity.

Type DataReference (p. 1371)

Required Arguments

Name The Name of the MSExcelRange to retrieve.

Type string (p. 1438)

Example

The following example shows how the user can get an MSExcelRange named "WB_Thickness".

```
system1 = GetSystem(Name="XLS")
analysis1 = system1.GetContainer(ComponentName="Analysis")
setup1 = analysis1.GetExcelSetup()
file1 = setup1.GetFile()
range1 = file1.GetRange(Name="WB_Thickness")
write range1.CellRange
write range1.Value
```

GetRanges

Gets the list of all MSExcelRange entities.

Return The DataReferenceSet containing all the MSExcelRange entities associated with the MSExcelFile.

Type DataReferenceSet (p. 1371)

Example

The following example shows how the user can get all the MSExcelRange entities of an MSExcelFile.

```
system1 = GetSystem(Name="XLS")
analysis1 = system1.GetContainer(ComponentName="Analysis")
setup1 = analysis1.GetExcelSetup()
file1 = setup1.GetFile()
ranges = file1.GetRanges()
```

OpenInApplication

No details are provided for this entry.

PublishParameter

Publishes an MSExcelRange as a Parameter in the Workbench project. The IsOutput argument allows to publish the range as an input or an output parameter. The method returns the created Parameter entity.

Return DataReference of the created parameter.

Type DataReference (p. 1371)

Required Arguments

ExcelRange The MSExcelRange to be published as a parameter.

Type DataReference (p. 1371)

Optional Arguments

IsOutput True to specify that the range is published as an output parameter, or false for input parameter.

Type bool (p. 1360)

Default Value False

Example

The following example shows how to publish ranges as input or output parameters.

```
system1 = GetSystem(Name="XLS")
analysis1 = system1.GetContainer(ComponentName="Analysis")
setup1 = analysis1.GetExcelSetup()
file = setup1.AddFile(FilePath="C:\Test\ProcessCalculation.xlsx")
ValvePosition = file.GetRange(Name="WB_Valve_Position")
T1 = file.GetRange(Name="WB_T1")
input1 = file.PublishParameter( ValvePosition )
output1 = file.PublishParameter( T1 )
```

Reload

Reloads an MSExcelFile to synchronize the data between the Microsoft Office Excel application and Workbench. For instance, it is necessary to call this method when the ExcelSetup.NamedRangeKey has been modified, in order to filter the named ranges based on the new NamedRangeKey's value. It is also necessary to reload the file if the workbook has been edited outside of Workbench, for instance to change a formula. The method recreates MSExcelRange and Parameter entities as required. The existing results in Workbench are invalidated.

Example

The following example shows how to reload a file.

```
system1 = GetSystem(Name="XLS")
analysis1 = system1.GetContainer(ComponentName="Analysis")
setup1 = analysis1.GetExcelSetup()
file = setup1.GetFile()
```

```
file.Reload()
```

UnpublishParameter

Unpublishes an MSExcelRange as a Parameter from the Workbench project, which means deleting the Parameter entity that was created in association to this MSExcelRange.

Required Arguments

ExcelRange The MSExcelRange to unpublish.

Type DataReference (p. 1371)

Example

The following example shows how to publish and unpublish ranges as input or output parameters.

```
system1 = GetSystem(Name="XLS")
analysis1 = system1.GetContainer(ComponentName="Analysis")
setup1 = analysis1.GetExcelSetup()
file = setup1.AddFile(FilePath="C:\Test\ProcessCalculation.xlsx")
ValvePosition = file.GetRange(Name="WB_Valve_Position")
input1 = file.PublishParameter( ValvePosition )
T1 = file.GetRange(Name="WB_T1")
output1 = file.PublishParameter( T1 )
file.UnpublishParameter(ValvePosition)
file.UnpublishParameter(T1)
```

MSExcelRange

This entity represents an Excel named range that matches the prefix string define by the Parameter Key. The MSExcelRange entities are created automatically when an Excel file is added, or reloaded, by filtering all the named ranges found in the Excel workbook with the Parameter Key. The MSExcelRange is not published as a parameter by default: use the MSExcelFile.PublishParameter method to expose the range as an input or output parameter in the Workbench project. The named ranges can contain a single cell for the value, or two cells for the value and the unit string.

Properties

CellRange

The coordinates defining the Excel range in the workbook.

Type string (p. 1438)

Read Only No

DisplayText

The general property that defines the user-visible name of an entity. This property is defined for all data entities but is used only in those entities that present a label in the user interface.

Type string (p. 1438)

Read Only No

Value

The current value of the range.

Type Quantity (p. 1422)

Read Only No

ValueQuantityName

The quantity name.

Type string (p. 1438)

Read Only No

Methods

GetParameter

Gets the Parameter entity associated with a published MSExcelRange entity. If the MSExcelRange is not published as a parameter, the method returns null.

Return The Parameter associated with the MSExcelRange, or null if it is not published.

Type DataReference (p. 1371)

Example

The following example shows how the user can get the parameter associated with a range.

```
system1 = GetSystem(Name="XLS")
analysis1 = system1.GetContainer(ComponentName="Analysis")
setup1 = analysis1.GetExcelSetup()
file1 = setup1.GetFile()
range1 = file1.GetRange(Name="WB_Thickness")
parameter1 = range1.GetParameter()
write parameter1.Name
```

MSExcelSetup

This entity holds properties to setup the data exchange with the Microsoft Office Excel application and information about the state of the connection with the instance of Microsoft Office Excel. There is one unique MSExcelSetup entity instance per Analysis container.

Properties

DisplayText

The general property that defines the user-visible name of an entity. This property is defined for all data entities but is used only in those entities that present a label in the user interface.

Type string (p. 1438)

Read Only No

ExcelStatus

Status of the connection with the instance of the Microsoft Office Excel application.

Type ExcelConnectionState (p. 1383)

Read Only Yes

ExcelVersion

Version number of the Microsoft Office Excel instance connected with Workbench.

Type string (p. 1438)

Read Only Yes

NamedRangeKey

The Named Ranges Key is a prefix string used to filter the named ranges found in the Excel workbook. The default value is set using the value of the related user preference. It is possible to use an empty string in order to retrieve all named ranges.

Type string (p. 1438)

Read Only No

UnitSystemName

The name of the unit system used for the Analysis container.

Type string (p. 1438)

Read Only No

Methods

AddFile

Adds a file to an MSExcelSetup entity by providing its FilePath. The method copies and registers the file into the Workbench project and returns an MSExcelFile entity if successful. The method also creates an MSExcelRange entity for each named range matching the ExcelSetup.NamedRangeKey prefix string ("" by default). If the file does not exist, is not an Excel file or cannot be registered into the project, the method throws an exception. If a file was already added to the MSExcelSetup, the method throws an

exception as well because only one file can be handled in each Analysis container. To use another file, the user has to Reset the data container first.

Return The DataReference of the added file.

Type DataReference (p. 1371)

Required Arguments

FilePath The Path of the original Microsoft Office Excel file to add.

Type string (p. 1438)

Example

The following example shows how to add a file to the MSExcelSetup entity.

```
system1 = GetSystem(Name="XLS")
analysis1 = system1.GetContainer(ComponentName="Analysis")
setup1 = analysis1.GetExcelSetup()
file = setup1.AddFile(FilePath="C:\Test\ProcessCalculation.xlsx")
write file.FileName
ranges = file.GetRanges()
```

DeleteFile

Deletes a file from an MSExcelSetup entity and from the Workbench project file management. All the MSExcelRange entities and associated Parameter entities are deleted as well.

Required Arguments

File The DataReference of the MSExcelFile entity to be deleted.

Type DataReference (p. 1371)

Example

The following example shows how to delete a file from an MSExcelSetup entity.

```
system1 = GetSystem(Name="XLS")
analysis1 = system1.GetContainer(ComponentName="Analysis")
setup1 = analysis1.GetExcelSetup()
file = setup1.GetFile()
setup1.DeleteFile(file)
```

GetFile

Gets the MSExcelFile entity associated to the MSExcelSetup entity. If the MSExcelSetup entity has no file, the method returns null. If the optional argument Name is specified but does not correspond to a file associated to the MSExcelSetup, the method throws an exception.

Return The DataReference of the retrieved file entity.

Type DataReference (p. 1371)

Optional Arguments

Name Name of the MSExcelFile entity to retrieve.

Type string (p. 1438)

Example

The following example shows how the user can get a file entity from a setup entity to retrieve one of its properties.

```
system1 = GetSystem(Name="XLS")
analysis1 = system1.GetContainer(ComponentName="Analysis")
setup1 = analysis1.GetExcelSetup()
file1 = setup1.GetFile()
write file1.Name
```

Parameters

Parameters

This container hold project-level Parameters and Design Points.

Methods

GetAllNamedExpressions

Returns all named expressions associated with a given container.

Return Named expressions in the given container.

Type DataReferenceSet (p. 1371)

GetAllParameters

Returns the set of all parameters associated with all entity properties in a given container. Parameters not associated with containers are not returned.

Return The set of parameters present in the given container.

Type DataReferenceSet (p. 1371)

Example

In this example 'paramSet' becomes the set of parameters associated with the properties of any entity that resides within the Results container of system1.

```
results1 = system1.GetContainer(ComponentName="Results")
paramSet = results1.GetAllParameters()
```

Data Entities

DesignPoint

The data entity which describes a project-level design point.

Properties

DisplayText

The general property that defines the user-visible name of an entity. This property is defined for all data entities but is used only in those entities that present a label in the user interface.

Type string (p. 1438)

Read Only No

Exported

Specifies whether the design point is exported.

Type bool (p. 1360)

Read Only No

HasValidRetainedData

Indicates whether the data model and files for this design point is retained in the project and valid. This can happen for a retained DP; or a non-retained DP before previously retained data is deleted.

Type bool (p. 1360)

Read Only Yes

IsUpToDate

True if the whole project is up to date for this design point.

Type bool (p. 1360)

Read Only Yes

Note

Contains user-defined notes about the design point.

Type string (p. 1438)

Read Only No

Retained

Gets or sets whether the design point is retained.

Type bool (p. 1360)

Read Only No

StateOfParameters

Indicates the current state of the design point's parameters.

Type DesignPointState (p. 1374)

Read Only Yes

UpdateOrder

The update order of the design point.

Type double (p. 1378)

Read Only No

Methods

AreParameterValuesEqual

Comparison to see if all parameter values within two different design points are equivalent.

Return True if all parameter values are equal.

Type bool (p. 1360)

Required Arguments

DesignPoint2 The second design point

Type DataReference (p. 1371)

CopyParameterExpressions

Copies all input parameter expressions from one design point to another.

Invalidates all output parameter values in the desination design point.

Required Arguments

ToDesignPoint The destination design point.

Type DataReference (p. 1371)

Delete

Deletes a design point, the associated directory, and all design point files from the project.

Note that the active design point cannot be deleted.

Duplicate

Creates a new design point in which all parameters have the same values as in the specified original design point.

Return The created design point entity.

Type DataReference (p. 1371)

Optional Arguments

Name name of new Design Point

Type string (p. 1438)

GetParameterValue

Returns the value of a specified parameter in a given design point.

Return The value of the parameter in the specified design point.

Type Object (p. 1409)

Required Arguments

Parameter The parameter data reference.

Type DataReference (p. 1371)

SetParameterExpression

Sets the expression of the specified input parameter in the given design point.

Required Arguments

Expression The string with the expression for the parameter.

Type string (p. 1438)

Parameter The input parameter data reference.

Type DataReference (p. 1371)

Example

The following example illustrates the setting of a parameter expression for a given design point.

```
dp = Parameters.GetDesignPoint("0")
param = Parameters.GetParameter("P1")
dp.SetParameterExpression(param, "cos(1)")
```

SetParameterExpressions

Sets expressions of the specified parameters in the specified design point.

Required Arguments

ParameterExpressions The parameters and expressions

Type IDictionary (p. 1375)<DataReference (p. 1371), string (p. 1438)>

DesignPointReportSetting

Design point report setting entity.

Properties

DesignPointReportImage

Report image showing options

Type string (p. 1438)

Read Only No

DisplayText

The general property that defines the user-visible name of an entity. This property is defined for all data entities but is used only in those entities that present a label in the user interface.

Type string (p. 1438)

Read Only No

Parameter

The data entity which describes a project-level Parameter.

Properties

Description

A human-readable description of the parameter.

Type string (p. 1438)

Read Only No

DisplayText

The general property that defines the user-visible name of an entity. This property is defined for all data entities but is used only in those entities that present a label in the user interface.

Type string (p. 1438)

Read Only No

ErrorMessage

An error message as a result of a validation or expression evaluation error.

Type string (p. 1438)

Read Only Yes

Expression

The parameter definition as an expression. This property may be a constant expression, or it may depend on the values of other parameters.

Type string (p. 1438)

Read Only No

ExpressionType

Indicates whether the expression is constant, derived from another expression, or undefined.

Type ExpressionType (p. 1384)

Read Only Yes

PersistedNamedValueEntityReference

the data reference of the entity that the named value refers to

Type DataReference (p. 1371)

Read Only No

Usage

Specifies how the paramter is used in the data model.

Type ParameterUsage (p. 1413)

Read Only Yes

Value

The current value of the parameter. For derived parameters, this value is the evaluated expression result.

Type Object (p. 1409)

Read Only Yes

ValueQuantityName

Gets the value quantity name if ValueSpec is a QuantitySpec. Null otherwise. If the entity is a parameter then setting this property will constrain the dimensions of values of the parameter to be consistent with the quantity name. This property does not constrain the values of named expressions.

Type string (p. 1438)

Read Only No

Methods

Delete

Deletes a parameter.

A parameter can only be deleted when it is no longer associated with any properties/entities in the project.

Disassociate

Disassociates a data model property from an existing parameter.

To associate the parameter with a property, call AssociateParameter.

Required Arguments

Entity The data model entity that holds the property to be disassociated.

Type DataReference (p. 1371)

PropertyName The name of the property to be disassociated.

Type string (p. 1438)

DisassociateInContext

Disassociates a data model property for a given context from an existing parameter.

To associate the parameter with a property in context, call AssociateParameterInContext.

Required Arguments

Entity The data model entity that holds the property to be disassociated.

Type DataReference (p. 1371)

PropertyName The name of the property to be disassociated.

Type string (p. 1438)

Optional Arguments

Context The context of the association.

Type DataReference (p. 1371)

Example

The following example illustrates proper DisassociateParameterInContext invocation:

```
entity1 = ...
parameter1 = Parameters.GetParameter(Name="p1")
parameter1.DisassociateInContext(entity1, "SampleProperty")
```

Callers may provide a Context argument.

```
contextEntity1 = ...
parameter1.DisassociateInContext(entity1, "SampleProperty", contextEntity1)
```

GetAssociatedEntityProperties

Returns the entity properties associated with a given parameter. If a container is specified, returns only the entity properties for the given container.

Return A ParameterizedEntityPropertiesCollection dictionary containing a data reference to each entity with parameterized properties and the list of property names within that entity.

Type ParameterizedEntityPropertiesCollection (p. 1413)

Optional Arguments

Container Optionally specifies a container for which the entity properties should be returned.

Type DataContainerReference (p. 1371)

GetDependencies

For a derived parameter, returns the list of parameters referred to by this parameter's expression.

Return Data references to the parameters that this parameter is dependent upon.

Type List (p. 1400)<DataReference (p. 1371)>

SetQuantityName

Sets a parameter's quantity name.

Required Arguments

QuantityName The quantity name.

Type string (p. 1438)

SetQuantityUnit

Set the unit string for a parameter across all design points.

Required Arguments

Unit The new units of the parameter.

Type string (p. 1438)

ParameterSummaryChart

This is a summary chart that will display all parameters against all design points. This is useful for understanding and visualizing the full parameter space, but is not very useful for comparing one parameter against another. For parameter vs. parameter charts, see CreateParameterVsParameterChart.

Properties

Chart

This stores the data reference to any generated charts from the Graphics system. e.g. a result of CreateMultiAxisChart.

Type DataReference (p. 1371)

Read Only No

DisplayText

The general property that defines the user-visible name of an entity. This property is defined for all data entities but is used only in those entities that present a label in the user interface.

Type string (p. 1438)

Read Only No

IsBaseDesignPointExcluded

If this property is set the base design point will be excluded.

Type bool (p. 1360)

Read Only No

Variables

The variables that have been added to the chart.

Type DataReferenceSet (p. 1371)

Read Only No

Methods

Delete

Deletes a parameter chart.

ParameterVsParameterChart

This chart type can be used to plot one parameter against another or parameters vs. design points.

There are 4 accessable axes, X-Top, X-Bottom, Y-Left and Y-Right.

```
chart1 = Parameters.CreateParameterVsParameterChart()
chart1.XAxisBottom = Parameters.GetParameter("P1")
chart1.XAxisTop = Parameters.GetParameter("P2")
chart1.YAxisLeft = Parameters.GetParameter("P3")
chart1.YAxisRight = Parameters.GetParameter("P4")
```

Properties

Chart

This stores the data reference to any generated charts from the Graphics system. e.g. a result of CreateChartXY.

Type DataReference (p. 1371)

Read Only No

DisplayText

The general property that defines the user-visible name of an entity. This property is defined for all data entities but is used only in those entities that present a label in the user interface.

Type string (p. 1438)

Read Only No

IsBaseDesignPointExcluded

If this property is set the base design point will be excluded.

Type bool (p. 1360)

Read Only No

XAxisBottom

The parameter entity to be plotted on the primary (bottom) x-axis.

Type DataReference (p. 1371)

Read Only No

XAxisTop

The parameter entity to be plotted on the secondary (top) x-axis.

Type DataReference (p. 1371)

Read Only No

YAxisLeft

The parameter entity to be plotted on the left (primary) y-axis.

Type DataReference (p. 1371)

Read Only No

YAxisRight

The parameter entity to be plotted on the right (secondary) y-axis.

Type DataReference (p. 1371)

Read Only No

Methods

Delete

Deletes a parameter chart.

Polyflow

Polyflow Setup

This container holds Setup data for an instance of Polyflow.

Methods

AddFile

Allows to import any kind of file into the Inputs sub-directory of the current Polyflow system

```
setup1.AddFile(FilePath="D:/temp/viscosity.crv")
  This command will copy the viscosity.crv file from the D:\temp directory
  into the Inputs directory of the current Polyflow system.
```

Required Arguments

FilePath Input : Path of file to be added.

Type string (p. 1438)

Edit

Starts the editor (Polydata) given a Polyflow Setup Container

```
setup1.Edit()
This command will launch the editor (Polydata) of the selected Setup cell.
```

Exit

Stops the editor (Polydata) or the solver (Polyflow) given a Polyflow Setup or Solution Container

```
setup1.Exit()
This command will stop the editor (Polydata) of the selected Setup cell.
```

```
solver1.Exit()
This command will stop the solver (Polyflow) of the selected Solution cell.
```

GetPolyflowMesh

Returns the PolyflowMesh object stored in the Polyflow Setup component.

Return The PolyflowMesh object stored in the Polyflow Setup component.

Type DataReference (p. 1371)

ImportMesh

Allows to import a mesh file (gambit *.neu file, *.poly file, *.msh Fluent Mesh file or Polyflow Mesh file). For *.neu and *.poly and a *.msh fluent file, a mesh conversion is automatically performed to get eventually a Polyflow Mesh file (*.msh)

```
setup1.ImportMesh(FilePath="D:/temp/swell.msh")
This command will copy the swell.msh file from the D:\temp directory
into the directory of the current Polyflow system.
```

Required Arguments

FilePath Input : path of mesh file to be imported (*.neu, *.poly or *.msh)

Type string (p. 1438)

ImportSetup

Allows to import a setup file (*.dat file) into the current Polyflow system

```
setup1.ImportSetup(FilePath="D:/temp/swell.dat")
This command will copy the swell.dat file from the D:/temp directory
into the directory of the current Polyflow system.
```

Required Arguments

FilePath Input : Path of Polyflow setup file to be imported.

Type string (p. 1438)

ImportUDF

Allows to import the udf file into the Inputs sub-directory of the current Polyflow system

```
setup1.ImportUDF(UdfFilePath="D:/temp/swell.udf")
This command will copy the swell.udf file from the D:/temp directory
into the Inputs directory of the current Polyflow system.
```

Required Arguments

UdfFilePath Input : Path of Polyflow udf file to be imported

Type string (p. 1438)

SetMeshPreferences

Allows to specify which mesh of upstream polyflow system will be used in the current polyflow system : one does not provide a mesh file name, but a specific key to the corresponding mesh. The Mesh restart mode can take two values : "NoUpstreamMeshFile" or "SingleMeshFile"

```
setup1.SetMeshPreferences(
meshRestartMode="SingleMeshFile",
meshKey="mesh (t=0.5000000E+00)[formatted]"))
This command will take the mesh generated by the upstream polyflow system
at time t=0.5 (and that is formatted)
```

Note : the mesh keys can be found in the "last_result.pub" file generated by the Polyflow solver in the upstream system!

Required Arguments

meshKey Mesh key (among the list of available meshes created by an upstream polyflow system)

Type string (p. 1438)

meshRestartMode Mesh restart mode

Type MeshRestartMode (p. 1404)

StartFuseTool

Starts Polyfuse given a Polyflow Setup Container

```
setup1.StartFuseTool()
This command will launch the mesh manipulation tool (Polyfuse) on the selected Setup cell.
```

StartMaterialsTool

Starts Polymat given a Polyflow Setup Container

```
setup1.StartMaterialsTool()
This command will launch the material parameters fitting tool (Polymat) on the selected Setup cell.
```

StartPreferences

Starts the Editor of Polydata preferences given a Polyflow Setup Container or Starts the Editor of Polyflow preferences given a Polyflow Solution Container

```
This command applies to Setup and Solution cells.
    setup1.StartPreferences()
    This command will launch the Preferences setting tool (Polypref) of the selected Setup cell.
    solution1.StartPreferences()
    This command will launch the Preferences setting tool (Polypref) of the selected Solution cell.
```

Data Entities

PolydataPreference

Contains a reference to the preferences file for Polydata (setup editor). It is a provider of the preferences file.

Properties

DisplayText

The general property that defines the user-visible name of an entity. This property is defined for all data entities but is used only in those entities that present a label in the user interface.

Type string (p. 1438)

Read Only No

File

Data reference to the preferences file for Polydata.

Type DataReference (p. 1371)

Read Only No

PolyflowMesh

Contains a reference to a Polyflow mesh file. It is a provider for a Polyflow mesh file.

Properties

DisplayText

The general property that defines the user-visible name of an entity. This property is defined for all data entities but is used only in those entities that present a label in the user interface.

Type string (p. 1438)

Read Only No

File

Data reference to the mesh file

Type DataReference (p. 1371)

Read Only No

MeshReadingMode

Tells if one wants a conversion of the input mesh (*.poly or fluent *.msh) into a polyflow *.msh or if one wants a direct reading by polydata (when editing setup)

Type MeshReading (p. 1404)

Read Only No

PolyflowSetup

Contains a reference to a Polyflow data file. It is a provider of the data file (containing the setup).

Properties

DisplayText

The general property that defines the user-visible name of an entity. This property is defined for all data entities but is used only in those entities that present a label in the user interface.

Type string (p. 1438)

Read Only No

File

Data reference to the Polyflow Dat file

Type DataReference (p. 1371)

Read Only No

Polyflow Solution

This container holds Solution data for an instance of Polyflow.

Methods

ClearTemporaryFiles

Clear temporary files generated by Polyflow solver (if the solver is not running!) given a Polyflow Solution Container

```
solution1.ClearTemporaryFiles()
```

Exit

Stops the editor (Polydata) or the solver (Polyflow) given a Polyflow Setup or Solution Container

```
setup1.Exit()
This command will stop the editor (Polydata) of the selected Setup cell.
```

```
solver1.Exit()
This command will stop the solver (Polyflow) of the selected Solution cell.
```

GetComponentSettingsForRsmDpUpdate

This query is used to obtain the ComponentSettingsForRsmDpUpdate object for Journaling and Scripting

GetPolyflowSolution

Returns the PolyflowSolution object stored in the Polyflow Solution component.

Return The PolyflowSolution object stored in the Polyflow Solution component.

Type DataReference (p. 1371)

GetSolutionSettings

This query is used to obtain the component solve settings object for Journaling and Scripting

SetResultsPreferences

Allows to specify which result (res/rst/csv) of upstream polyflow system will be used in the current polyflow system : one does not provide filenames, but specific keys to the corresponding result files.

The solver restart mode can take the following values:

"No initialization from upstream system",
"Restart with a single Polyflow results file",
"Restart with Polyflow results and restart files",
"Restart with a single Polyflow CSV file",
"Restart with Polyflow CSV and restart files",
"Restart with a list of Polyflow results files (for transient mixing tasks only!)",
"Restart with a list of Polyflow CSV files (for conversion of a list of Polyflow CSV files)"

Note : the result (res,rst,csv) keys can be found in the "last_result.pub" file generated by the Polyflow solver in the upstream system!

Required Arguments

csvKey Csv key (among the list of available csv created by an upstream polyflow system)

Type string (p. 1438)

restartKey Restart key (among the list of available restart created by an upstream polyflow system)

Type string (p. 1438)

resultKey Result key (among the list of available results created by an upstream polyflow system)

Type string (p. 1438)

solverRestartMode Solver restart mode

Type SolverRestartMode (p. 1434)

Example

```
solution1.SetResultsPreferences(
solverRestartMode="Restart with a single Polyflow CSV file",
resultKey="undefined",
restartKey="undefined",
csvKey="csv (t=0.5000000E+00)[formatted]")
```

This command will take the csv generated by the upstream polyflow system at time t=0.5 (and that is formatted)

StartCFDPost

Starts CFD-Post given a Polyflow Solution Container in order to see the initial state of a simulation.

```
solution1.StartCFDPost()
This command will launch CFD-Post on the selected Solution cell.
```

StartCurveTool

Starts Polycurve (curves viewer) given a Polyflow Solution Container

```
solution1.StartCurveTool()
This command will launch Polycurve of the selected Solution cell.
```

StartDiagnosticsTool

Starts Diagnostics tool (Polydiag) given a Polyflow Solution Container

```
solution1.StartDiagnosticsTool()
This command will launch the Diagnostics tool (Polydiag) of the selected Solution cell.
```

StartListingViewer

Starts the listing viewer, given a Polyflow Solution Container, to present the content of a text file

```
solution1.StartListingViewer(FilePath="D:/temp/polyflow.lst")
This command will launch the listing viewer tool (Polylst) on the selected Solution cell.
```

Required Arguments

ListingFilePath Input : listing filename (absolute path)

Type string (p. 1438)

StartPreferences

Starts the Editor of Polydata preferences given a Polyflow Setup Container or Starts the Editor of Polyflow preferences given a Polyflow Solution Container

```
This command applies to Setup and Solution cells.
    setup1.StartPreferences()
    This command will launch the Preferences setting tool (Polypref) of the selected Setup cell.
    solution1.StartPreferences()
    This command will launch the Preferences setting tool (Polypref) of the selected Solution cell.
```

StartStatisticsTool

Starts Polystat given a Polyflow Solution Container

```
solution1.StartStatisticsTool()
This command will launch the statistical analyzer tool (Polystat) on the selected Solution cell.
```

SwitchToBackgroundMode

Switch the Update in progress into background mode. This will enable operations that are not allowed during an Update in foreground mode (e.g. Project Save).

This command is not normally useful in a script. Journals may record the invocation of this command after an Update invoke, as the result of GUI activity while the Update is in progress. However, replay of these journals will always wait for the Update invoke to complete before invoking the next command, rendering this step ineffectual.

Data Entities

HiddenPollingComponentSolveSettings

This class contains a derived version of solve settings where polling is disabled

Properties

ConfiguredQueue

A configured queue used for new RSM architecture

Type string (p. 1438)

Read Only No

DisplayText

The general property that defines the user-visible name of an entity. This property is defined for all data entities but is used only in those entities that present a label in the user interface.

Type string (p. 1438)

Read Only No

EnablePolling

whether enable polling during remote solve

Type bool (p. 1360)

Read Only No

ExecutionMode

execution mode

Type string (p. 1438)

Read Only No

NumberOfProcesses

number of processes for parallel solve

Type int (p. 1394)

Read Only No

PollingInterval

polling internal during remote solve

Type Quantity (p. 1422)

Read Only No

RsmJobName

RSM job name

Type string (p. 1438)

Read Only No

RsmQueueDetails

Configured queue details

Type RsmQueueDetails (p. 1427)

Read Only No

UpdateOption

Update mode

Type JobRunMode (p. 1397)

Read Only No

PolyflowPreference

Contains a reference to the preferences file for Polyflow (solver). It is a provider of the preferences file.

Properties

DisplayText

The general property that defines the user-visible name of an entity. This property is defined for all data entities but is used only in those entities that present a label in the user interface.

Type string (p. 1438)

Read Only No

File

Data reference to the preferences file for Polyflow.

Type DataReference (p. 1371)

Read Only No

PolyflowSolution

Contains a reference to a Polyflow listing file. It is a provider of the listing file (containing a summary of the simulation).

Properties

DisplayText

The general property that defines the user-visible name of an entity. This property is defined for all data entities but is used only in those entities that present a label in the user interface.

Type string (p. 1438)

Read Only No

File

Data reference to the listing file generated by the Polyflow solver and containing a summary of the simulation.

Type DataReference (p. 1371)

Read Only No

Project

Project

This container holds the Systems, Components and Templates in the project.

Data Entities

Component

No details are provided for this entry.

Properties

DirectoryName

No details are provided for this entry.

Type string (p. 1438)

Read Only Yes

DisplayText

The general property that defines the user-visible name of an entity. This property is defined for all data entities but is used only in those entities that present a label in the user interface.

Type string (p. 1438)

Read Only No

Notes

No details are provided for this entry.

Type string (p. 1438)

Read Only No

UserId

No details are provided for this entry.

Type string (p. 1438)

Read Only Yes

Methods

Clean

No details are provided for this entry.

DeleteShare

No details are provided for this entry.

Optional Arguments

System No details are provided for this entry.

Type DataReference (p. 1371)

DeleteTransfer

No details are provided for this entry.

Required Arguments

TargetComponent No details are provided for this entry.

Type DataReference (p. 1371)

GetAIMMechanicalPhysicsDefinitionOutput

This method is available only in AIM. For the given Physics Definition schematic component, return it's valid Mechanical Physics Definition output

Return The AIMMechanicalPhysicsDefinitionOutput object associated with the project schematic component.

Type DataReference (p. 1371)

GetContainer

No details are provided for this entry.

Return No details are provided for this entry.

Type DataContainerReference (p. 1371)

GetFluentMeshOutput

This method is available only in AIM. For the given Meshing schematic component, return it's valid FluentMesh output

Return The AIMFluentMeshOutput object associated with the project schematic component.

Type DataReference (p. 1371)

GetFluentPartMeshOutputProvider

This method is available only in AIM. For the given Meshing schematic component, return it's valid FluentMesh output

Return The AIMFluentPartMeshOutputProvider object associated with the project schematic component.

Type DataReference (p. 1371)

GetFluentPhysicsDefinitionOutput

This method is available only in AIM. For the given Meshing schematic component, return it's valid FluentPhysicsDefinition output for Fluent

Return The AIMFluentPhysicsDefinitionOutput object associated with the project schematic component.

Type DataReference (p. 1371)

GetGeometryMeshOutput

This method is available only in AIM. For the given Meshing schematic component, return it's valid GeometryMesh output

Return The AIMGeometryMeshOutput object associated with the project schematic component.

Type DataReference (p. 1371)

GetTaskObject

This method is available only in AIM. For the given project schematic component, return it's anchor object.

Return The anchor object associated with the project schematic component.

Type DataReference (p. 1371)

HasValidPartMesh

For the given Meshing schematic component, check whether it supports Part Mesh

Return Returns true if it supports Part Mesh otherwise false

Type bool (p. 1360)

HasValidVolumeFillMesh

For the given Meshing schematic component, check whether it supports volume Fill Mesh

Return Returns true if it supports Volume Fill Mesh otherwise false

Type bool (p. 1360)

Refresh

No details are provided for this entry.

RemoveFromSystem

No details are provided for this entry.

Required Arguments

System No details are provided for this entry.

Type DataReference (p. 1371)

ReplaceWithShare

No details are provided for this entry.

Required Arguments

ComponentToShare No details are provided for this entry.

Type DataReference (p. 1371)

Optional Arguments

SourceSystem No details are provided for this entry.

Type DataReference (p. 1371)

TargetSystem No details are provided for this entry.

Type DataReference (p. 1371)

Example

Required example does not exist.

Reset

No details are provided for this entry.

TransferData

No details are provided for this entry.

Required Arguments

TargetComponent No details are provided for this entry.

Type DataReference (p. 1371)

TransferSpecificData

No details are provided for this entry.

Required Arguments

TargetComponent No details are provided for this entry.

Type DataReference (p. 1371)

TransferDataName No details are provided for this entry.

Type string (p. 1438)

Update

No details are provided for this entry.

Optional Arguments

AllDependencies No details are provided for this entry.

Type bool (p. 1360)

Continue No details are provided for this entry.

Type UpdateContinuation (p. 1446)

Default Value None

UpdateUpstreamComponents

No details are provided for this entry.

ComponentTemplate

No details are provided for this entry.

Properties

DisplayText

The general property that defines the user-visible name of an entity. This property is defined for all data entities but is used only in those entities that present a label in the user interface.

Type string (p. 1438)

Read Only No

Methods

CreateComponent

No details are provided for this entry.

Required Arguments

System No details are provided for this entry.

Type DataReference (p. 1371)

Optional Arguments

AllSystemProperties No details are provided for this entry.

Type SystemPropertyDictionary (p. 1440)

CreatedComponent No details are provided for this entry.

Type Output (p. 1411)<DataReference (p. 1371)>

Name No details are provided for this entry.

Type string (p. 1438)

UpstreamComponent No details are provided for this entry.

Type List (p. 1400)<DataReference (p. 1371)>

Example

Required example does not exist.

CustomProjectTemplate

No details are provided for this entry.

Properties

DisplayText

The general property that defines the user-visible name of an entity. This property is defined for all data entities but is used only in those entities that present a label in the user interface.

Type string (p. 1438)

Read Only No

SystemNames

No details are provided for this entry.

Type List (p. 1400)<string (p. 1438)>

Read Only No

SystemTemplates

No details are provided for this entry.

Type List (p. 1400)<DataReference (p. 1371)>

Read Only No

Methods

CreateProject

No details are provided for this entry.

Delete

No details are provided for this entry.

DesignPointSolveSettings

No details are provided for this entry.

Properties

ComponentExecutionMode

No details are provided for this entry.

Type string (p. 1438)

Read Only No

ConfiguredQueue

A configured queue used for new RSM architecture

Type string (p. 1438)

Read Only No

DesignPointRetainedOrExportedUpdate

No details are provided for this entry.

Type DesignPointRetainedOrExportedUpdate (p. 1374)

Read Only No

DisplayText

The general property that defines the user-visible name of an entity. This property is defined for all data entities but is used only in those entities that present a label in the user interface.

Type string (p. 1438)

Read Only No

JobStatusCheckIntervalInSeconds

No details are provided for this entry.

Type int (p. 1394)

Read Only No

NumberOfCoresPerJob

No details are provided for this entry.

Type int (p. 1394)

Read Only No

NumberOfJobsToBeCreated

No details are provided for this entry.

Type int (p. 1394)

Read Only No

NumberOfTasksToBeCreated

No details are provided for this entry.

Type int (p. 1394)

Read Only No

PartialUpdate

No details are provided for this entry.

Type PartialDesignPointUpdate (p. 1414)

Read Only No

PreRSMUpdate

No details are provided for this entry.

Type PartialDesignPointUpdate (p. 1414)

Read Only No

RetainPartialUpdate

No details are provided for this entry.

Type PartialDesignPointUpdate (p. 1414)

Read Only No

RsmJobName

RSM job name

Type string (p. 1438)

Read Only No

RsmQueueDetails

Configured queue details

Type RsmQueueDetails (p. 1427)

Read Only No

UpdateMethod

No details are provided for this entry.

Type string (p. 1438)

Read Only No

UpdateOption

Update mode

Type JobRunMode (p. 1397)

Read Only No

UpdateOrder

No details are provided for this entry.

Type DesignPointUpdateOrder (p. 1374)

Read Only No

UpfrontLicenseCheckout

No details are provided for this entry.

Type string (p. 1438)

Read Only No

SchematicSettings

No details are provided for this entry.

Properties

DisplayText

The general property that defines the user-visible name of an entity. This property is defined for all data entities but is used only in those entities that present a label in the user interface.

Type string (p. 1438)

Read Only No

Notes

No details are provided for this entry.

Type string (p. 1438)

Read Only No

System

No details are provided for this entry.

Properties

AnalysisType

No details are provided for this entry.

Type string (p. 1438)

Read Only No

DisplayText

The general property that defines the user-visible name of an entity. This property is defined for all data entities but is used only in those entities that present a label in the user interface.

Type string (p. 1438)

Read Only No

Notes

No details are provided for this entry.

Type string (p. 1438)

Read Only No

Physics

No details are provided for this entry.

Type List (p. 1400)<string (p. 1438)>

Read Only No

Solver

No details are provided for this entry.

Type List (p. 1400)<string (p. 1438)>

Read Only Yes

UserId

No details are provided for this entry.

Type string (p. 1438)

Read Only Yes

Methods

Copy

No details are provided for this entry.

Return No details are provided for this entry.

Type DataReference (p. 1371)

Required Arguments

KeepConnections No details are provided for this entry.

Type bool (p. 1360)

Delete

No details are provided for this entry.

Duplicate

No details are provided for this entry.

Return No details are provided for this entry.

Type DataReference (p. 1371)

Optional Arguments

ComponentsToShare No details are provided for this entry.

Type List (p. 1400)<DataReference (p. 1371)>

Name No details are provided for this entry.

Type string (p. 1438)

Position No details are provided for this entry.

Type PositionType (p. 1418)

RelativeTo No details are provided for this entry.

Type DataReference (p. 1371)

Example

Required example does not exist.

Export

No details are provided for this entry.

Required Arguments

FilePath No details are provided for this entry.

Type string (p. 1438)

IncludeExternalFiles No details are provided for this entry.

Type bool (p. 1360)

IncludeResultFiles No details are provided for this entry.

Type bool (p. 1360)

IncludeUserFiles No details are provided for this entry.

Type bool (p. 1360)

Example

Required example does not exist.

GetComponent

No details are provided for this entry.

Return No details are provided for this entry.

Type DataReference (p. 1371)

Required Arguments

Name No details are provided for this entry.

Type string (p. 1438)

GetContainer

No details are provided for this entry.

Return No details are provided for this entry.

Type DataContainerReference (p. 1371)

Required Arguments

ComponentName No details are provided for this entry.

Type string (p. 1438)

GetReplaceableTemplates

No details are provided for this entry.

Return No details are provided for this entry.

Type List (p. 1400)<DataReference (p. 1371)>

Move

No details are provided for this entry.

Required Arguments

Position No details are provided for this entry.

Type PositionType (p. 1418)

RelativeTo No details are provided for this entry.

Type DataReference (p. 1371)

RecreateDeletedComponents

No details are provided for this entry.

Refresh

No details are provided for this entry.

Update

No details are provided for this entry.

Optional Arguments

AllDependencies No details are provided for this entry.

Type bool (p. 1360)

Template

No details are provided for this entry.

Properties

AnalysisType

No details are provided for this entry.

Type string (p. 1438)

Read Only No

DisplayName

No details are provided for this entry.

Type string (p. 1438)

Read Only No

DisplayText

The general property that defines the user-visible name of an entity. This property is defined for all data entities but is used only in those entities that present a label in the user interface.

Type string (p. 1438)

Read Only No

Physics

No details are provided for this entry.

Type List (p. 1400)<string (p. 1438)>

Read Only No

Solver

No details are provided for this entry.

Type List (p. 1400)<string (p. 1438)>

Read Only Yes

SystemType

No details are provided for this entry.

Type string (p. 1438)

Read Only No

SystemTypeAbbreviation

No details are provided for this entry.

Type string (p. 1438)

Read Only No

Methods

CreateSystem

No details are provided for this entry.

Return No details are provided for this entry.

Type DataReference (p. 1371)

Optional Arguments

ComponentsToShare No details are provided for this entry.

Type List (p. 1400)<DataReference (p. 1371)>

DataTransferFrom No details are provided for this entry.

Type List (p. 1400)<TransferDataToNewComponentSpec (p. 1443)>

DataTransferTo No details are provided for this entry.

Type List (p. 1400)<TransferDataFromNewComponentSpec (p. 1443)>

Name No details are provided for this entry.

Type string (p. 1438)

Position No details are provided for this entry.

Type PositionType (p. 1418)

Default Value Default

RelativeTo No details are provided for this entry.

Type DataReference (p. 1371)

Example

Required example does not exist.

ReplaceSystem

No details are provided for this entry.

Return No details are provided for this entry.

Type DataReference (p. 1371)

Required Arguments

System No details are provided for this entry.

Type DataReference (p. 1371)

Optional Arguments

Name No details are provided for this entry.

Type string (p. 1438)

Project File Types

This container holds File Types registered with the File Manager.

Data Entities

FileType

FileType defines file type information including a short description.

Properties

Description

The description of the file type.

Type string (p. 1438)

Read Only Yes

DisplayText

The general property that defines the user-visible name of an entity. This property is defined for all data entities but is used only in those entities that present a label in the user interface.

Type string (p. 1438)

Read Only No

ExtensionPatterns

The possible file extension patterns to match in a regular expression. The patterns are used to check whether this file type can handle a set of file extensions.

Type List (p. 1400)<string (p. 1438)>

Read Only Yes

Extensions

The possible file extensions (with the leading "."). These extensions can be displayed to the user in a file dialog filter, for example.

Type List (p. 1400)<string (p. 1438)>

Read Only Yes

Project Files

This container holds Files registered with the File Manager.

Methods

GetFiles

Gets all the File References that are associated with the specified container.

Return The set of file references associated with the container.

Type DataReferenceSet (p. 1371)

Example

This example prints the location of files for two different components of a system.

```
system1 = GetSystem(Name="Static1")
geometry1 = system1.GetContainer(ComponentName="Geometry")
for fileRef in geometry1.GetFiles():
   print fileRef.Location
>>> C:\Users\myUser\Projects\static1_files\dp0\SYS\DM\SYS.agdb
>>> E:\data\Models\pipe.x_t
model1 = system1.GetContainer(ComponentName="Model")
for fileRef in model1.GetFiles():
   print fileRef.Location
>>> C:\Users\myUser\Projects\static1_files\dp0\global\MECH\SYS.engd
>>> C:\Users\myUser\Projects\static1_files\dp0\global\MECH\SYS.mechdb
```

Data Entities

FileReference

The data entity that represents a file that is part of the project.

Properties

Directory

A string that contains the path to the directory containing the file.

Type string (p. 1438)

Read Only Yes

DisplayText

The general property that defines the user-visible name of an entity. This property is defined for all data entities but is used only in those entities that present a label in the user interface.

Type string (p. 1438)

Read Only No

Exists

A value indicating whether the file exists.

Type bool (p. 1360)

Read Only Yes

FileName

The name of the file, including the extension.

Type string (p. 1438)

Read Only Yes

LastModifiedTime

The time at which this file was most recently modified.

Type DateTime (p. 1372)

Read Only Yes

Location

The full path to the local file including the directory and file name.

Type string (p. 1438)

Read Only No

Size

The size of the file in bytes.

Type long (p. 1394)

Read Only Yes

Methods

Copy

Copies a file to a target directory.

Return A reference to the created file is returned in this argument.

Type DataReference (p. 1371)

Required Arguments

DestinationDirectoryPath The full path to the destination directory.

Type string (p. 1438)

Overwrite Specifies whether to overwrite an existing files of the same name. Note: A registered file may not be overwritten.

Type bool (p. 1360)

Example

The following example illustrates proper CopyFileByReference command invocation:

```
fileRef = GetRegisteredFileQuery(FilePath=r"path_to_file")
fileRefCopy = fileRef.Copy(DestinationDirectoryPath=r"path_to_copy_file", Overwrite=True, CopyReferenceCou
```

Delete

Deletes the specified file

Optional Arguments

BackUp Specifies whether to back up the file before deletion. This optional argument's default value is true.

Type bool (p. 1360)

Default Value True

DeleteIfShared Specifies whether a registered file should be deleted even if it is still in use. This optional argument's default value is true. If this argument's value is false, an exception will be thrown if a shared file is encountered. To avoid the exception, set ErrorIfShared to false.

Type bool (p. 1360)

Default Value False

ErrorIfShared Specifies whether to throw an exception when trying to delete a shared file if DeleteIfShared is set to false.

Type bool (p. 1360)

Example

The following example illustrates a deletion of a registered file via a file reference. The file will be deleted if even if it is still in use. Note that the file will be backed up.

```
fileRef = GetRegisteredFile(FilePath=r"C:\Users\anyuser\path-to-file.extension")
fileRef.Delete(DeleteIfShared=True,
               BackUp=True)
```

The next example illustrates a deletion of a registered file via a file reference without a forced deletion. If the file is shared, an error will not occur, and the file reference count will be decremented.

```
fileRef = GetRegisteredFile(FilePath=r"C:\Users\anyuser\path-to-file.extension")
fileRef.Delete(ErrorIfShared=False)
```

Move

Moves a file to a new directory.

Required Arguments

NewDirectoryPath The full path to the destination directory.

Type string (p. 1438)

Optional Arguments

BackUp Specifies whether to back up the file before the move. This optional argument's default value is true.

Type bool (p. 1360)

Default Value True

Example

The following example illustrates proper MoveFileByReference command invocation:

```
fileRef = GetRegisteredFileQuery(FilePath=r"path_to_file")
fileRefCopy = fileRef.Move(NewDirectoryPath=r"path_to_move_file", Backup=False)
```

Rename

Renames a file.

Required Arguments

New-Name The new file name, excluding the directory path. To change the directory, see MoveFileCommand.

Type string (p. 1438)

Optional Arguments

BackUp Specifies whether to back up the file before renaming it. This optional argument's default value is true.

Type bool (p. 1360)

Default Value True

Example

The following example illustrates proper RenameFileByReference command invocation:

```
fileRef = GetRegisteredFileQuery(FilePath=r"path_to_file")
fileRefCopy = fileRef.Rename(NeName="new_file_name", BackUp=True)
```

Repair

Repairs a file that is referenced in the project but cannot be found on disk. If the file is expected to be found under the project directory, then this command will copy the new file to the expected location rather than link to the new file in its current location. Use SetFilePathCommand to link to a file in a new location.

Required Arguments

FilePath A full path to the file in its updated location.

Type string (p. 1438)

Project Messages

This container holds current project messages.

Data Entities

StoredMessage

A data entity representing a message (error, warning, information, etc.) that is visible to the user.

Properties

Association

The data model entity to which the message applies (a component, for example).

Type string (p. 1438)

Read Only Yes

DateTimeStamp

The publication time.

Type DateTime (p. 1372)

Read Only Yes

Details

The detailed text string of the message.

Type string (p. 1438)

Read Only Yes

DisplayText

The general property that defines the user-visible name of an entity. This property is defined for all data entities but is used only in those entities that present a label in the user interface.

Type string (p. 1438)

Read Only No

MessageCategory

The category of the message

Type MessageCategory (p. 1404)

Read Only Yes

MessageType

The type of the message (e.g., Information, Warning, Error, etc.).

Type MessageType (p. 1404)

Read Only Yes

Summary

The summary text string of the message.

Type string (p. 1438)

Read Only Yes

Study

Study

This container holds Study data for an instance of Study.

Common Methods

CreateDataManager

The Data Manager is a utility object intended for use by the ImportEngineeringData command. It is used to load an applicable manager for the type of data being managed as well the source of the data which the manager will use to bring the data into the project.

Return The newly created Data Manager DataReference.

Type DataReference (p. 1371)

Optional Arguments

Provider The management provider for the requested Source of data. The provider understands how to convert the data in the sources to AIM.

Type string (p. 1438)

Default Value ANSYS

Sources The source of the data which will be managed by the Data Manager. If this argument is not provided the Source must be set or provided before the Data Manager is used.

Type List (p. 1400)<string (p. 1438)>

Example

This example creates a Data Manager in the study.

```
dataManagerThermal = study1.CreateDataManager(
    Sources=[r"C:\ANSYSDev\Program Files\Ansys Inc\v151\Addins\EngineeringData\Samples\Thermal_Materials.x
    Provider="ANSYS")
```

CreateEntity

Create AIM objects of the requested Type.

Return An instance of the Type.

Type DataReference (p. 1371)

Required Arguments

Type The entity Type to create. The entity Type can be seen in the GUI by hovering over the entity's object icon. Also using dir(Study) will show object types for Get<Type> queries, for which the <Type> may be able to be created.

Type string (p. 1438)

Optional Arguments

AssociatedObject The object to associate this object to in the AIM dependency graph, which will create a directed edge in the AIM dependency graph. See discussion on Study and Tasks. This is deprecated in favor of Association.

Type DataReference (p. 1371)

Association The object to associate this object to in the AIM dependency graph, which will create a directed edge in the AIM dependency graph. See discussion on Study and Tasks.

Type DataReference (p. 1371)

Location If the entity has a location property, this will set the property to the provided LocationSet.

Type Object (p. 1409)

Default Value Ansys.Study.Core.Common.Referencing.LocationSet

Location2 If the entity has a second location property, this will set the property to the provided LocationSet.

Type Object (p. 1409)

Default Value Ansys.Study.Core.Common.Referencing.LocationSet

LocationWithHitPoints If the entity has a location property with hit points, this will set the property to the provided LocationSet.

Type Object (p. 1409)

Default Value Ansys.Study.Core.Common.Referencing.LocationSet

LocationWithHitPoints2 If the entity has a second location property with hit points, this will set the property to the provided LocationSet.

Type Object (p. 1409)

Default Value Ansys.Study.Core.Common.Referencing.Location-Set

ObjectReferences Future use

Type DataReferenceSet (p. 1371)

Example

```
This example shows the steps preceding the CreateEntity, and then creates a PhysicsRegion object.
system1 = GetSystem(Name="Study")
physicsDefinitionComponent1 = Study.CreateTask(
    Type="Physics Definition",
    System=system1)
study1 = system1.GetContainer(ComponentName="Study")
physicsDefinition1 = physicsDefinitionComponent1.GetTaskObject()
physicsRegion1 = study1.CreateEntity(
    Type="PhysicsRegion",
    Association=physicsDefinition1)
```

CreateTable

CreateTable is a convenience command which performs the following commands:

Creates a table and associates with the Parent. See the CreateEntity command, where Type="Table" and AssociatedObject=Parent.
Adds columns corresponding to the Columns parameter. See the AddTableColumn command.

Return The DataReference of the newly created table.

Type DataReference (p. 1371)

Required Arguments

Columns A List of tuple (name and physical quantity). The name is used to identify the column and must be unique to this table. The physical quantity (e.g. Density, Pressure, etc.) is used to verify the data has the correct units. The use of integer indexes in other table related commands correspond to the order of the columns, with 0 (zero) being the first item.

Type List (p. 1400)<List (p. 1400)<string (p. 1438)>>

Parent The DataReference of the object to which this table should be created in the scope of

Type DataReference (p. 1371)

Example

This example illustrates creating a table and setting data in the table.

```
# drContainer is the DataContainerReference of an object that has been previously created (e.g. study1)
# drParent is the DataReference of an object that has been previously created (e.g. material1)
table1 = drContainer.CreateTable(drParent, Columns=[("Temperature","Temperature"),("Displacement","Length"
table1.AddRow(Data=["100 [C]","1.0 [m]"])
table1.AddRow(Data=["200 [C]",".02 [m]"])
table1.AddRow(Data=["300 [C]",".06 [m]"])
```

```
table1.GetData()
```

GetAllTaskGroupsInProject

Returns all tasks in the project.

Return The task groups in the project.

Type DataReferenceSet (p. 1371)

GetBeamEndRelease

Obtains a BeamEndRelease instance by name from the supplied container.

Return The BeamEndRelease instance.

Type DataReference (p. 1371)

Required Arguments

Name The entity name for which to search. Include the @ prefix if searching by display text

Type string (p. 1438)

GetBeamSection

Obtains a BeamSection instance by name from the supplied container.

Return The BeamSection instance.

Type DataReference (p. 1371)

Required Arguments

Name The entity name for which to search. Include the @ prefix if searching by display text

Type string (p. 1438)

GetBearingLoad

Obtains a BearingLoad instance by name from the supplied container.

Return The BearingLoad instance.

Type DataReference (p. 1371)

Required Arguments

Name The entity name for which to search. Include the @ prefix if searching by display text

Type string (p. 1438)

GetBoltPretension

Obtains a BoltPretension instance by name from the supplied container.

Return The BoltPretension instance.

Type DataReference (p. 1371)

Required Arguments

Name The entity name for which to search. Include the @ prefix if searching by display text

Type string (p. 1438)

GetBoundaryCondition

Obtains a BoundaryCondition instance by name from the supplied container.

Return The BoundaryCondition instance.

Type DataReference (p. 1371)

Required Arguments

Name The entity name for which to search. Include the @ prefix if searching by display text

Type string (p. 1438)

GetBoundaryLayerRefinement

Obtains a BoundaryLayerRefinement instance by name from the supplied container.

Return The BoundaryLayerRefinement instance.

Type DataReference (p. 1371)

Required Arguments

Name The entity name for which to search. Include the @ prefix if searching by display text

Type string (p. 1438)

GetCappingSurface

Obtains a CappingSurface instance by name from the supplied container.

Return The CappingSurface instance.

Type DataReference (p. 1371)

Required Arguments

Name The entity name for which to search. Include the @ prefix if searching by display text

Type string (p. 1438)

GetCharge

Obtains a Charge instance by name from the supplied container.

Return The Charge instance.

Type DataReference (p. 1371)

Required Arguments

Name The entity name for which to search. Include the @ prefix if searching by display text

Type string (p. 1438)

GetConditionGroup

Obtains a ConditionGroup instance by name from the supplied container.

Return The ConditionGroup instance.

Type DataReference (p. 1371)

Required Arguments

Name The entity name for which to search. Include the @ prefix if searching by display text

Type string (p. 1438)

GetConfiguration

Obtains a Configuration instance by name from the supplied container.

Return The Configuration instance.

Type DataReference (p. 1371)

Required Arguments

Name The entity name for which to search. Include the @ prefix if searching by display text

Type string (p. 1438)

GetConnection

Obtains a Connection instance by name from the supplied container.

Return The Connection instance.

Type DataReference (p. 1371)

Required Arguments

Name The entity name for which to search. Include the @ prefix if searching by display text

Type string (p. 1438)

GetConnectionBehavior

Obtains a ConnectionBehavior instance by name from the supplied container.

Return The ConnectionBehavior instance.

Type DataReference (p. 1371)

Required Arguments

Name The entity name for which to search. Include the @ prefix if searching by display text

Type string (p. 1438)

GetConstraint

Obtains a Constraint instance by name from the supplied container.

Return The Constraint instance.

Type DataReference (p. 1371)

Required Arguments

Name The entity name for which to search. Include the @ prefix if searching by display text

Type string (p. 1438)

GetContact

Obtains a Contact instance by name from the supplied container.

Return The Contact instance.

Type DataReference (p. 1371)

Required Arguments

Name The entity name for which to search. Include the @ prefix if searching by display text

Type string (p. 1438)

GetContactBehavior

Obtains a ContactBehavior instance by name from the supplied container.

Return The ContactBehavior instance.

Type DataReference (p. 1371)

Required Arguments

Name The entity name for which to search. Include the @ prefix if searching by display text

Type string (p. 1438)

GetContactDetection

Obtains a ContactDetection instance by name from the supplied container.

Return The ContactDetection instance.

Type DataReference (p. 1371)

Required Arguments

Name The entity name for which to search. Include the @ prefix if searching by display text

Type string (p. 1438)

GetContourResult

Obtains a ContourResult instance by name from the supplied container.

Return The ContourResult instance.

Type DataReference (p. 1371)

Required Arguments

Name The entity name for which to search. Include the @ prefix if searching by display text

Type string (p. 1438)

GetConvection

Obtains a Convection instance by name from the supplied container.

Return The Convection instance.

Type DataReference (p. 1371)

Required Arguments

Name The entity name for which to search. Include the @ prefix if searching by display text

Type string (p. 1438)

GetCurrent

Obtains a Current instance by name from the supplied container.

Return The Current instance.

Type DataReference (p. 1371)

Required Arguments

Name The entity name for which to search. Include the @ prefix if searching by display text

Type string (p. 1438)

GetCyclicSymmetryConstraint

Obtains a CyclicSymmetryConstraint instance by name from the supplied container.

Return The CyclicSymmetryConstraint instance.

Type DataReference (p. 1371)

Required Arguments

Name The entity name for which to search. Include the @ prefix if searching by display text

Type string (p. 1438)

GetDisplacement

Obtains a Displacement instance by name from the supplied container.

Return The Displacement instance.

Type DataReference (p. 1371)

Required Arguments

Name The entity name for which to search. Include the @ prefix if searching by display text

Type string (p. 1438)

GetDistributedMass

Obtains a DistributedMass instance by name from the supplied container.

Return The DistributedMass instance.

Type DataReference (p. 1371)

Required Arguments

Name The entity name for which to search. Include the @ prefix if searching by display text

Type string (p. 1438)

GetEquipotential

Obtains a Equipotential instance by name from the supplied container.

Return The Equipotential instance.

Type DataReference (p. 1371)

Required Arguments

Name The entity name for which to search. Include the @ prefix if searching by display text

Type string (p. 1438)

GetFatigueSettings

Obtains a FatigueSettings instance by name from the supplied container.

Return The FatigueSettings instance.

Type DataReference (p. 1371)

Required Arguments

Name The entity name for which to search. Include the @ prefix if searching by display text

Type string (p. 1438)

GetFidelityRefinement

Obtains a FidelityRefinement instance by name from the supplied container.

Return The FidelityRefinement instance.

Type DataReference (p. 1371)

Required Arguments

Name The entity name for which to search. Include the @ prefix if searching by display text

Type string (p. 1438)

GetForce

Obtains a Force instance by name from the supplied container.

Return The Force instance.

Type DataReference (p. 1371)

Required Arguments

Name The entity name for which to search. Include the @ prefix if searching by display text

Type string (p. 1438)

GetGeometryImportSource

Obtains a GeometryImportSource instance by name from the supplied container.

Return The GeometryImportSource instance.

Type DataReference (p. 1371)

Required Arguments

Name The entity name for which to search. Include the @ prefix if searching by display text

Type string (p. 1438)

GetGravityDefinition

Obtains a GravityDefinition instance by name from the supplied container.

Return The GravityDefinition instance.

Type DataReference (p. 1371)

Required Arguments

Name The entity name for which to search. Include the @ prefix if searching by display text

Type string (p. 1438)

GetGroundLocation

Obtains a GroundLocation instance by name from the supplied container.

Return The GroundLocation instance.

Type DataReference (p. 1371)

Required Arguments

Name The entity name for which to search. Include the @ prefix if searching by display text

Type string (p. 1438)

GetHeatFlow

Obtains a HeatFlow instance by name from the supplied container.

Return The HeatFlow instance.

Type DataReference (p. 1371)

Required Arguments

Name The entity name for which to search. Include the @ prefix if searching by display text

Type string (p. 1438)

GetHeatFlux

Obtains a HeatFlux instance by name from the supplied container.

Return The HeatFlux instance.

Type DataReference (p. 1371)

Required Arguments

Name The entity name for which to search. Include the @ prefix if searching by display text

Type string (p. 1438)

GetHeatGeneration

Obtains a HeatGeneration instance by name from the supplied container.

Return The HeatGeneration instance.

Type DataReference (p. 1371)

Required Arguments

Name The entity name for which to search. Include the @ prefix if searching by display text

Type string (p. 1438)

GetHeatSrc

Obtains a HeatSrc instance by name from the supplied container.

Return The HeatSrc instance.

Type DataReference (p. 1371)

Required Arguments

Name The entity name for which to search. Include the @ prefix if searching by display text

Type string (p. 1438)

GetImport

Obtains a Import instance by name from the supplied container.

Return The Import instance.

Type DataReference (p. 1371)

Required Arguments

Name The entity name for which to search. Include the @ prefix if searching by display text

Type string (p. 1438)

GetInertiaLoad

Obtains a InertiaLoad instance by name from the supplied container.

Return The InertiaLoad instance.

Type DataReference (p. 1371)

Required Arguments

Name The entity name for which to search. Include the @ prefix if searching by display text

Type string (p. 1438)

GetInitialCondition

Obtains a InitialCondition instance by name from the supplied container.

Return The InitialCondition instance.

Type DataReference (p. 1371)

Required Arguments

Name The entity name for which to search. Include the @ prefix if searching by display text

Type string (p. 1438)

GetInitialTemperature

Obtains a InitialTemperature instance by name from the supplied container.

Return The InitialTemperature instance.

Type DataReference (p. 1371)

Required Arguments

Name The entity name for which to search. Include the @ prefix if searching by display text

Type string (p. 1438)

GetInterfaceGenerator

Obtains a InterfaceGenerator instance by name from the supplied container.

Return The InterfaceGenerator instance.

Type DataReference (p. 1371)

Required Arguments

Name The entity name for which to search. Include the @ prefix if searching by display text

Type string (p. 1438)

GetIsoSurface

Obtains a IsoSurface instance by name from the supplied container.

Return The IsoSurface instance.

Type DataReference (p. 1371)

Required Arguments

Name The entity name for which to search. Include the @ prefix if searching by display text

Type string (p. 1438)

GetJoint

Obtains a Joint instance by name from the supplied container.

Return The Joint instance.

Type DataReference (p. 1371)

Required Arguments

Name The entity name for which to search. Include the @ prefix if searching by display text

Type string (p. 1438)

GetJointBehavior

Obtains a JointBehavior instance by name from the supplied container.

Return The JointBehavior instance.

Type DataReference (p. 1371)

Required Arguments

Name The entity name for which to search. Include the @ prefix if searching by display text

Type string (p. 1438)

GetLaunchSettings

Obtains a LaunchSettings instance by name from the supplied container.

Return The LaunchSettings instance.

Type DataReference (p. 1371)

Required Arguments

Name The entity name for which to search. Include the @ prefix if searching by display text

Type string (p. 1438)

GetLine

Obtains a Line instance by name from the supplied container.

Return The Line instance.

Type DataReference (p. 1371)

Required Arguments

Name The entity name for which to search. Include the @ prefix if searching by display text

Type string (p. 1438)

GetLinearizedStressChart

Obtains a LinearizedStressChart instance by name from the supplied container.

Return The LinearizedStressChart instance.

Type DataReference (p. 1371)

Required Arguments

Name The entity name for which to search. Include the @ prefix if searching by display text

Type string (p. 1438)

GetLineChart

Obtains a LineChart instance by name from the supplied container.

Return The LineChart instance.

Type DataReference (p. 1371)

Required Arguments

Name The entity name for which to search. Include the @ prefix if searching by display text

Type string (p. 1438)

GetLoad

Obtains a Load instance by name from the supplied container.

Return The Load instance.

Type DataReference (p. 1371)

Required Arguments

Name The entity name for which to search. Include the @ prefix if searching by display text

Type string (p. 1438)

GetMaterial

Obtains a Material instance by name from the supplied container.

Return The Material instance.

Type DataReference (p. 1371)

Required Arguments

Name The entity name for which to search. Include the @ prefix if searching by display text

Type string (p. 1438)

GetMaterialAssignment

Obtains a MaterialAssignment instance by name from the supplied container.

Return The MaterialAssignment instance.

Type DataReference (p. 1371)

Required Arguments

Name The entity name for which to search. Include the @ prefix if searching by display text

Type string (p. 1438)

GetMaterialAtState

Obtains a MaterialAtState instance by name from the supplied container.

Return The MaterialAtState instance.

Type DataReference (p. 1371)

Required Arguments

Name The entity name for which to search. Include the @ prefix if searching by display text

Type string (p. 1438)

GetMechanicalPhysicsOptions

Obtains a MechanicalPhysicsOptions instance by name from the supplied container.

Return The MechanicalPhysicsOptions instance.

Type DataReference (p. 1371)

Required Arguments

Name The entity name for which to search. Include the @ prefix if searching by display text

Type string (p. 1438)

GetMemberSizeConstraint

Obtains a MemberSizeConstraint instance by name from the supplied container.

Return The MemberSizeConstraint instance.

Type DataReference (p. 1371)

Required Arguments

Name The entity name for which to search. Include the @ prefix if searching by display text

Type string (p. 1438)

GetMesh

Obtains a Mesh instance by name from the supplied container.

Return The Mesh instance.

Type DataReference (p. 1371)

Required Arguments

Name The entity name for which to search. Include the @ prefix if searching by display text

Type string (p. 1438)

GetMeshControl

Obtains a MeshControl instance by name from the supplied container.

Return The MeshControl instance.

Type DataReference (p. 1371)

Required Arguments

Name The entity name for which to search. Include the @ prefix if searching by display text

Type string (p. 1438)

GetMeshControlBodySizing

Obtains a MeshControlBodySizing instance by name from the supplied container.

Return The MeshControlBodySizing instance.

Type DataReference (p. 1371)

Required Arguments

Name The entity name for which to search. Include the @ prefix if searching by display text

Type string (p. 1438)

GetMeshControlEdgeSizing

Obtains a MeshControlEdgeSizing instance by name from the supplied container.

Return The MeshControlEdgeSizing instance.

Type DataReference (p. 1371)

Required Arguments

Name The entity name for which to search. Include the @ prefix if searching by display text

Type string (p. 1438)

GetMeshControlElementShape

Obtains a MeshControlElementShape instance by name from the supplied container.

Return The MeshControlElementShape instance.

Type DataReference (p. 1371)

Required Arguments

Name The entity name for which to search. Include the @ prefix if searching by display text

Type string (p. 1438)

GetMeshControlFaceSizing

Obtains a MeshControlFaceSizing instance by name from the supplied container.

Return The MeshControlFaceSizing instance.

Type DataReference (p. 1371)

Required Arguments

Name The entity name for which to search. Include the @ prefix if searching by display text

Type string (p. 1438)

GetMeshControlLocalBoundaryLayer

Obtains a MeshControlLocalBoundaryLayer instance by name from the supplied container.

Return The MeshControlLocalBoundaryLayer instance.

Type DataReference (p. 1371)

Required Arguments

Name The entity name for which to search. Include the @ prefix if searching by display text

Type string (p. 1438)

GetMeshDiagnostics

Obtains a MeshDiagnostics instance by name from the supplied container.

Return The MeshDiagnostics instance.

Type DataReference (p. 1371)

Required Arguments

Name The entity name for which to search. Include the @ prefix if searching by display text

Type string (p. 1438)

GetMeshing

Obtains a Meshing instance by name from the supplied container.

Return The Meshing instance.

Type DataReference (p. 1371)

Required Arguments

Name The entity name for which to search. Include the @ prefix if searching by display text

Type string (p. 1438)

GetMeshModeling

Obtains a MeshModeling instance by name from the supplied container.

Return The MeshModeling instance.

Type DataReference (p. 1371)

Required Arguments

Name The entity name for which to search. Include the @ prefix if searching by display text

Type string (p. 1438)

GetModeling

Obtains a Modeling instance by name from the supplied container.

Return The Modeling instance.

Type DataReference (p. 1371)

Required Arguments

Name The entity name for which to search. Include the @ prefix if searching by display text

Type string (p. 1438)

GetModelingImportSource

Obtains a ModelingImportSource instance by name from the supplied container.

Return The ModelingImportSource instance.

Type DataReference (p. 1371)

Required Arguments

Name The entity name for which to search. Include the @ prefix if searching by display text

Type string (p. 1438)

GetMoment

Obtains a Moment instance by name from the supplied container.

Return The Moment instance.

Type DataReference (p. 1371)

Required Arguments

Name The entity name for which to search. Include the @ prefix if searching by display text

Type string (p. 1438)

GetMomentumSrc

Obtains a MomentumSrc instance by name from the supplied container.

Return The MomentumSrc instance.

Type DataReference (p. 1371)

Required Arguments

Name The entity name for which to search. Include the @ prefix if searching by display text

Type string (p. 1438)

GetMonitorChart

Obtains a MonitorChart instance by name from the supplied container.

Return The MonitorChart instance.

Type DataReference (p. 1371)

Required Arguments

Name The entity name for which to search. Include the @ prefix if searching by display text

Type string (p. 1438)

GetMoveRotateControl

Obtains a MoveRotateControl instance by name from the supplied container.

Return The MoveRotateControl instance.

Type DataReference (p. 1371)

Required Arguments

Name The entity name for which to search. Include the @ prefix if searching by display text

Type string (p. 1438)

GetOptimizationConstraint

Obtains a OptimizationConstraint instance by name from the supplied container.

Return The OptimizationConstraint instance.

Type DataReference (p. 1371)

Required Arguments

Name The entity name for which to search. Include the @ prefix if searching by display text

Type string (p. 1438)

GetOptimizationOptions

Obtains a OptimizationOptions instance by name from the supplied container.

Return The OptimizationOptions instance.

Type DataReference (p. 1371)

Required Arguments

Name The entity name for which to search. Include the @ prefix if searching by display text

Type string (p. 1438)

GetOutputControls

Obtains a OutputControls instance by name from the supplied container.

Return The OutputControls instance.

Type DataReference (p. 1371)

Required Arguments

Name The entity name for which to search. Include the @ prefix if searching by display text

Type string (p. 1438)

GetParticleInjectionCnd

Obtains a ParticleInjectionCnd instance by name from the supplied container.

Return The ParticleInjectionCnd instance.

Type DataReference (p. 1371)

Required Arguments

Name The entity name for which to search. Include the @ prefix if searching by display text

Type string (p. 1438)

GetParticleTrack

Obtains a ParticleTrack instance by name from the supplied container.

Return The ParticleTrack instance.

Type DataReference (p. 1371)

Required Arguments

Name The entity name for which to search. Include the @ prefix if searching by display text

Type string (p. 1438)

GetPerimeterWeld

Obtains a PerimeterWeld instance by name from the supplied container.

Return The PerimeterWeld instance.

Type DataReference (p. 1371)

Required Arguments

Name The entity name for which to search. Include the @ prefix if searching by display text

Type string (p. 1438)

GetPhysicsCoupling

Obtains a PhysicsCoupling instance by name from the supplied container.

Return The PhysicsCoupling instance.

Type DataReference (p. 1371)

Required Arguments

Name The entity name for which to search. Include the @ prefix if searching by display text

Type string (p. 1438)

GetPhysicsDefinition

Obtains a PhysicsDefinition instance by name from the supplied container.

Return The PhysicsDefinition instance.

Type DataReference (p. 1371)

Required Arguments

Name The entity name for which to search. Include the @ prefix if searching by display text

Type string (p. 1438)

GetPhysicsOptions

Obtains a PhysicsOptions instance by name from the supplied container.

Return The PhysicsOptions instance.

Type DataReference (p. 1371)

Required Arguments

Name The entity name for which to search. Include the @ prefix if searching by display text

Type string (p. 1438)

GetPhysicsRegion

Obtains a PhysicsRegion instance by name from the supplied container.

Return The PhysicsRegion instance.

Type DataReference (p. 1371)

Required Arguments

Name The entity name for which to search. Include the @ prefix if searching by display text

Type string (p. 1438)

GetPlane

Obtains a Plane instance by name from the supplied container.

Return The Plane instance.

Type DataReference (p. 1371)

Required Arguments

Name The entity name for which to search. Include the @ prefix if searching by display text

Type string (p. 1438)

GetPoint

Obtains a Point instance by name from the supplied container.

Return The Point instance.

Type DataReference (p. 1371)

Required Arguments

Name The entity name for which to search. Include the @ prefix if searching by display text

Type string (p. 1438)

GetPolyflowCondition

Obtains a PolyflowCondition instance by name from the supplied container.

Return The PolyflowCondition instance.

Type DataReference (p. 1371)

Required Arguments

Name The entity name for which to search. Include the @ prefix if searching by display text

Type string (p. 1438)

GetPolyflowContact

Obtains a PolyflowContact instance by name from the supplied container.

Return The PolyflowContact instance.

Type DataReference (p. 1371)

Required Arguments

Name The entity name for which to search. Include the @ prefix if searching by display text

Type string (p. 1438)

GetPolyflowContactBehavior

Obtains a PolyflowContactBehavior instance by name from the supplied container.

Return The PolyflowContactBehavior instance.

Type DataReference (p. 1371)

Required Arguments

Name The entity name for which to search. Include the @ prefix if searching by display text

Type string (p. 1438)

GetPolyflowConvection

Obtains a PolyflowConvection instance by name from the supplied container.

Return The PolyflowConvection instance.

Type DataReference (p. 1371)

Required Arguments

Name The entity name for which to search. Include the @ prefix if searching by display text

Type string (p. 1438)

GetPolyflowDieDeformation

Obtains a PolyflowDieDeformation instance by name from the supplied container.

Return The PolyflowDieDeformation instance.

Type DataReference (p. 1371)

Required Arguments

Name The entity name for which to search. Include the @ prefix if searching by display text

Type string (p. 1438)

GetPolyflowFixedMold

Obtains a PolyflowFixedMold instance by name from the supplied container.

Return The PolyflowFixedMold instance.

Type DataReference (p. 1371)

Required Arguments

Name The entity name for which to search. Include the @ prefix if searching by display text

Type string (p. 1438)

GetPolyflowFluidSolidInterfaceBehavior

Obtains a PolyflowFluidSolidInterfaceBehavior instance by name from the supplied container.

Return The PolyflowFluidSolidInterfaceBehavior instance.

Type DataReference (p. 1371)

Required Arguments

Name The entity name for which to search. Include the @ prefix if searching by display text

Type string (p. 1438)

GetPolyflowForceDrivenMold

Obtains a PolyflowForceDrivenMold instance by name from the supplied container.

Return The PolyflowForceDrivenMold instance.

Type DataReference (p. 1371)

Required Arguments

Name The entity name for which to search. Include the @ prefix if searching by display text

Type string (p. 1438)

GetPolyflowFreejetExit

Obtains a PolyflowFreejetExit instance by name from the supplied container.

Return The PolyflowFreejetExit instance.

Type DataReference (p. 1371)

Required Arguments

Name The entity name for which to search. Include the @ prefix if searching by display text

Type string (p. 1438)

GetPolyflowFreeSurface

Obtains a PolyflowFreeSurface instance by name from the supplied container.

Return The PolyflowFreeSurface instance.

Type DataReference (p. 1371)

Required Arguments

Name The entity name for which to search. Include the @ prefix if searching by display text

Type string (p. 1438)

GetPolyflowHeatFlux

Obtains a PolyflowHeatFlux instance by name from the supplied container.

Return The PolyflowHeatFlux instance.

Type DataReference (p. 1371)

Required Arguments

Name The entity name for which to search. Include the @ prefix if searching by display text

Type string (p. 1438)

GetPolyflowInflow

Obtains a PolyflowInflow instance by name from the supplied container.

Return The PolyflowInflow instance.

Type DataReference (p. 1371)

Required Arguments

Name The entity name for which to search. Include the @ prefix if searching by display text

Type string (p. 1438)

GetPolyflowInitialLayerThickness

Obtains a PolyflowInitialLayerThickness instance by name from the supplied container.

Return The PolyflowInitialLayerThickness instance.

Type DataReference (p. 1371)

Required Arguments

Name The entity name for which to search. Include the @ prefix if searching by display text

Type string (p. 1438)

GetPolyflowInitialTemperature

Obtains a PolyflowInitialTemperature instance by name from the supplied container.

Return The PolyflowInitialTemperature instance.

Type DataReference (p. 1371)

Required Arguments

Name The entity name for which to search. Include the @ prefix if searching by display text

Type string (p. 1438)

GetPolyflowInsulated

Obtains a PolyflowInsulated instance by name from the supplied container.

Return The PolyflowInsulated instance.

Type DataReference (p. 1371)

Required Arguments

Name The entity name for which to search. Include the @ prefix if searching by display text

Type string (p. 1438)

GetPolyflowLaunchControls

Obtains a PolyflowLaunchControls instance by name from the supplied container.

Return The PolyflowLaunchControls instance.

Type DataReference (p. 1371)

Required Arguments

Name The entity name for which to search. Include the @ prefix if searching by display text

Type string (p. 1438)

GetPolyflowMold

Obtains a PolyflowMold instance by name from the supplied container.

Return The PolyflowMold instance.

Type DataReference (p. 1371)

Required Arguments

Name The entity name for which to search. Include the @ prefix if searching by display text

Type string (p. 1438)

GetPolyflowNumericalControls

Obtains a PolyflowNumericalControls instance by name from the supplied container.

Return The PolyflowNumericalControls instance.

Type DataReference (p. 1371)

Required Arguments

Name The entity name for which to search. Include the @ prefix if searching by display text

Type string (p. 1438)

GetPolyflowOutflow

Obtains a PolyflowOutflow instance by name from the supplied container.

Return The PolyflowOutflow instance.

Type DataReference (p. 1371)

Required Arguments

Name The entity name for which to search. Include the @ prefix if searching by display text

Type string (p. 1438)

GetPolyflowOutputControls

Obtains a PolyflowOutputControls instance by name from the supplied container.

Return The PolyflowOutputControls instance.

Type DataReference (p. 1371)

Required Arguments

Name The entity name for which to search. Include the @ prefix if searching by display text

Type string (p. 1438)

GetPolyflowPhysicsOptions

Obtains a PolyflowPhysicsOptions instance by name from the supplied container.

Return The PolyflowPhysicsOptions instance.

Type DataReference (p. 1371)

Required Arguments

Name The entity name for which to search. Include the @ prefix if searching by display text

Type string (p. 1438)

GetPolyflowPressure

Obtains a PolyflowPressure instance by name from the supplied container.

Return The PolyflowPressure instance.

Type DataReference (p. 1371)

Required Arguments

Name The entity name for which to search. Include the @ prefix if searching by display text

Type string (p. 1438)

GetPolyflowRegionInterface

Obtains a PolyflowRegionInterface instance by name from the supplied container.

Return The PolyflowRegionInterface instance.

Type DataReference (p. 1371)

Required Arguments

Name The entity name for which to search. Include the @ prefix if searching by display text

Type string (p. 1438)

GetPolyflowRemeshing

Obtains a PolyflowRemeshing instance by name from the supplied container.

Return The PolyflowRemeshing instance.

Type DataReference (p. 1371)

Required Arguments

Name The entity name for which to search. Include the @ prefix if searching by display text

Type string (p. 1438)

GetPolyflowRestrictor

Obtains a PolyflowRestrictor instance by name from the supplied container.

Return The PolyflowRestrictor instance.

Type DataReference (p. 1371)

Required Arguments

Name The entity name for which to search. Include the @ prefix if searching by display text

Type string (p. 1438)

GetPolyflowSolutionControls

Obtains a PolyflowSolutionControls instance by name from the supplied container.

Return The PolyflowSolutionControls instance.

Type DataReference (p. 1371)

Required Arguments

Name The entity name for which to search. Include the @ prefix if searching by display text

Type string (p. 1438)

GetPolyflowSupport

Obtains a PolyflowSupport instance by name from the supplied container.

Return The PolyflowSupport instance.

Type DataReference (p. 1371)

Required Arguments

Name The entity name for which to search. Include the @ prefix if searching by display text

Type string (p. 1438)

GetPolyflowSymmetry

Obtains a PolyflowSymmetry instance by name from the supplied container.

Return The PolyflowSymmetry instance.

Type DataReference (p. 1371)

Required Arguments

Name The entity name for which to search. Include the @ prefix if searching by display text

Type string (p. 1438)

GetPolyflowTemperatureImposed

Obtains a PolyflowTemperatureImposed instance by name from the supplied container.

Return The PolyflowTemperatureImposed instance.

Type DataReference (p. 1371)

Required Arguments

Name The entity name for which to search. Include the @ prefix if searching by display text

Type string (p. 1438)

GetPolyflowVelocityDrivenMold

Obtains a PolyflowVelocityDrivenMold instance by name from the supplied container.

Return The PolyflowVelocityDrivenMold instance.

Type DataReference (p. 1371)

Required Arguments

Name The entity name for which to search. Include the @ prefix if searching by display text

Type string (p. 1438)

GetPolyflowWall

Obtains a PolyflowWall instance by name from the supplied container.

Return The PolyflowWall instance.

Type DataReference (p. 1371)

Required Arguments

Name The entity name for which to search. Include the @ prefix if searching by display text

Type string (p. 1438)

GetPolyflowWallBase

Obtains a PolyflowWallBase instance by name from the supplied container.

Return The PolyflowWallBase instance.

Type DataReference (p. 1371)

Required Arguments

Name The entity name for which to search. Include the @ prefix if searching by display text

Type string (p. 1438)

GetPorositySrc

Obtains a PorositySrc instance by name from the supplied container.

Return The PorositySrc instance.

Type DataReference (p. 1371)

Required Arguments

Name The entity name for which to search. Include the @ prefix if searching by display text

Type string (p. 1438)

GetPressure

Obtains a Pressure instance by name from the supplied container.

Return The Pressure instance.

Type DataReference (p. 1371)

Required Arguments

Name The entity name for which to search. Include the @ prefix if searching by display text

Type string (p. 1438)

GetPressureDrop

Obtains a PressureDrop instance by name from the supplied container.

Return The PressureDrop instance.

Type DataReference (p. 1371)

Required Arguments

Name The entity name for which to search. Include the @ prefix if searching by display text

Type string (p. 1438)

GetPrimitiveBox

Obtains a PrimitiveBox instance by name from the supplied container.

Return The PrimitiveBox instance.

Type DataReference (p. 1371)

Required Arguments

Name The entity name for which to search. Include the @ prefix if searching by display text

Type string (p. 1438)

GetPsdAcceleration

Obtains a PsdAcceleration instance by name from the supplied container.

Return The PsdAcceleration instance.

Type DataReference (p. 1371)

Required Arguments

Name The entity name for which to search. Include the @ prefix if searching by display text

Type string (p. 1438)

GetPsdDisplacement

Obtains a PsdDisplacement instance by name from the supplied container.

Return The PsdDisplacement instance.

Type DataReference (p. 1371)

Required Arguments

Name The entity name for which to search. Include the @ prefix if searching by display text

Type string (p. 1438)

GetPsdGAcceleration

Obtains a PsdGAcceleration instance by name from the supplied container.

Return The PsdGAcceleration instance.

Type DataReference (p. 1371)

Required Arguments

Name The entity name for which to search. Include the @ prefix if searching by display text

Type string (p. 1438)

GetPsdLoad

Obtains a PsdLoad instance by name from the supplied container.

Return The PsdLoad instance.

Type DataReference (p. 1371)

Required Arguments

Name The entity name for which to search. Include the @ prefix if searching by display text

Type string (p. 1438)

GetPsdVelocity

Obtains a PsdVelocity instance by name from the supplied container.

Return The PsdVelocity instance.

Type DataReference (p. 1371)

Required Arguments

Name The entity name for which to search. Include the @ prefix if searching by display text

Type string (p. 1438)

GetPullOutDirection

Obtains a PullOutDirection instance by name from the supplied container.

Return The PullOutDirection instance.

Type DataReference (p. 1371)

Required Arguments

Name The entity name for which to search. Include the @ prefix if searching by display text

Type string (p. 1438)

GetRadiation

Obtains a Radiation instance by name from the supplied container.

Return The Radiation instance.

Type DataReference (p. 1371)

Required Arguments

Name The entity name for which to search. Include the @ prefix if searching by display text

Type string (p. 1438)

GetReferenceFrame

Obtains a ReferenceFrame instance by name from the supplied container.

Return The ReferenceFrame instance.

Type DataReference (p. 1371)

Required Arguments

Name The entity name for which to search. Include the @ prefix if searching by display text

Type string (p. 1438)

GetRegionInterface

Obtains a RegionInterface instance by name from the supplied container.

Return The RegionInterface instance.

Type DataReference (p. 1371)

Required Arguments

Name The entity name for which to search. Include the @ prefix if searching by display text

Type string (p. 1438)

GetRegionInterfaceGenerator

Obtains a RegionInterfaceGenerator instance by name from the supplied container.

Return The RegionInterfaceGenerator instance.

Type DataReference (p. 1371)

Required Arguments

Name The entity name for which to search. Include the @ prefix if searching by display text

Type string (p. 1438)

GetResponsePsdResult

Obtains a ResponsePsdResult instance by name from the supplied container.

Return The ResponsePsdResult instance.

Type DataReference (p. 1371)

Required Arguments

Name The entity name for which to search. Include the @ prefix if searching by display text

Type string (p. 1438)

GetResultObject

Obtains a ResultObject instance by name from the supplied container.

Return The ResultObject instance.

Type DataReference (p. 1371)

Required Arguments

Name The entity name for which to search. Include the @ prefix if searching by display text

Type string (p. 1438)

GetResults

Obtains a Results instance by name from the supplied container.

Return The Results instance.

Type DataReference (p. 1371)

Required Arguments

Name The entity name for which to search. Include the @ prefix if searching by display text

Type string (p. 1438)

GetScalarConstraint

Obtains a ScalarConstraint instance by name from the supplied container.

Return The ScalarConstraint instance.

Type DataReference (p. 1371)

Required Arguments

Name The entity name for which to search. Include the @ prefix if searching by display text

Type string (p. 1438)

GetScalarInitialCondition

Obtains a ScalarInitialCondition instance by name from the supplied container.

Return The ScalarInitialCondition instance.

Type DataReference (p. 1371)

Required Arguments

Name The entity name for which to search. Include the @ prefix if searching by display text

Type string (p. 1438)

GetScalarLoad

Obtains a ScalarLoad instance by name from the supplied container.

Return The ScalarLoad instance.

Type DataReference (p. 1371)

Required Arguments

Name The entity name for which to search. Include the @ prefix if searching by display text

Type string (p. 1438)

GetSectionLayer

Obtains a SectionLayer instance by name from the supplied container.

Return The SectionLayer instance.

Type DataReference (p. 1371)

Required Arguments

Name The entity name for which to search. Include the @ prefix if searching by display text

Type string (p. 1438)

GetSelectionSet

Obtains a SelectionSet instance by name from the supplied container.

Return The SelectionSet instance.

Type DataReference (p. 1371)

Required Arguments

Name The entity name for which to search. Include the @ prefix if searching by display text

Type string (p. 1438)

GetShape

Obtains a Shape instance by name from the supplied container.

Return The Shape instance.

Type DataReference (p. 1371)

Required Arguments

Name The entity name for which to search. Include the @ prefix if searching by display text

Type string (p. 1438)

GetShapeTaskStatistics

Obtains a ShapeTaskStatistics instance by name from the supplied container.

Return The ShapeTaskStatistics instance.

Type DataReference (p. 1371)

Required Arguments

Name The entity name for which to search. Include the @ prefix if searching by display text

Type string (p. 1438)

GetShearMomentDiagram

Obtains a ShearMomentDiagram instance by name from the supplied container.

Return The ShearMomentDiagram instance.

Type DataReference (p. 1371)

Required Arguments

Name The entity name for which to search. Include the @ prefix if searching by display text

Type string (p. 1438)

GetShellThickness

Obtains a ShellThickness instance by name from the supplied container.

Return The ShellThickness instance.

Type DataReference (p. 1371)

Required Arguments

Name The entity name for which to search. Include the @ prefix if searching by display text

Type string (p. 1438)

GetSimulation

Obtains a Simulation instance by name from the supplied container.

Return The Simulation instance.

Type DataReference (p. 1371)

Required Arguments

Name The entity name for which to search. Include the @ prefix if searching by display text

Type string (p. 1438)

GetSingleValueResult

Obtains a SingleValueResult instance by name from the supplied container.

Return The SingleValueResult instance.

Type DataReference (p. 1371)

Required Arguments

Name The entity name for which to search. Include the @ prefix if searching by display text

Type string (p. 1438)

GetSolutionProgression

Obtains a SolutionProgression instance by name from the supplied container.

Return The SolutionProgression instance.

Type DataReference (p. 1371)

Required Arguments

Name The entity name for which to search. Include the @ prefix if searching by display text

Type string (p. 1438)

GetSolutionStep

Obtains a SolutionStep instance by name from the supplied container.

Return The SolutionStep instance.

Type DataReference (p. 1371)

Required Arguments

Name The entity name for which to search. Include the @ prefix if searching by display text

Type string (p. 1438)

GetSolvePhysics

Obtains a SolvePhysics instance by name from the supplied container.

Return The SolvePhysics instance.

Type DataReference (p. 1371)

Required Arguments

Name The entity name for which to search. Include the @ prefix if searching by display text

Type string (p. 1438)

GetSolverSettings

Obtains a SolverSettings instance by name from the supplied container.

Return The SolverSettings instance.

Type DataReference (p. 1371)

Required Arguments

Name The entity name for which to search. Include the @ prefix if searching by display text

Type string (p. 1438)

GetSpotWeld

Obtains a SpotWeld instance by name from the supplied container.

Return The SpotWeld instance.

Type DataReference (p. 1371)

Required Arguments

Name The entity name for which to search. Include the @ prefix if searching by display text

Type string (p. 1438)

GetSpring

Obtains a Spring instance by name from the supplied container.

Return The Spring instance.

Type DataReference (p. 1371)

Required Arguments

Name The entity name for which to search. Include the @ prefix if searching by display text

Type string (p. 1438)

GetSpringBehavior

Obtains a SpringBehavior instance by name from the supplied container.

Return The SpringBehavior instance.

Type DataReference (p. 1371)

Required Arguments

Name The entity name for which to search. Include the @ prefix if searching by display text

Type string (p. 1438)

GetStatusGroup

Obtains a StatusGroup instance by name from the supplied container.

Return The StatusGroup instance.

Type DataReference (p. 1371)

Required Arguments

Name The entity name for which to search. Include the @ prefix if searching by display text

Type string (p. 1438)

GetStreamLine

Obtains a StreamLine instance by name from the supplied container.

Return The StreamLine instance.

Type DataReference (p. 1371)

Required Arguments

Name The entity name for which to search. Include the @ prefix if searching by display text

Type string (p. 1438)

GetStressConstraint

Obtains a StressConstraint instance by name from the supplied container.

Return The StressConstraint instance.

Type DataReference (p. 1371)

Required Arguments

Name The entity name for which to search. Include the @ prefix if searching by display text

Type string (p. 1438)

GetStructuralMass

Obtains a StructuralMass instance by name from the supplied container.

Return The StructuralMass instance.

Type DataReference (p. 1371)

Required Arguments

Name The entity name for which to search. Include the @ prefix if searching by display text

Type string (p. 1438)

GetSupport

Obtains a Support instance by name from the supplied container.

Return The Support instance.

Type DataReference (p. 1371)

Required Arguments

Name The entity name for which to search. Include the @ prefix if searching by display text

Type string (p. 1438)

GetSuppressControl

Obtains a SuppressControl instance by name from the supplied container.

Return The SuppressControl instance.

Type DataReference (p. 1371)

Required Arguments

Name The entity name for which to search. Include the @ prefix if searching by display text

Type string (p. 1438)

GetSymmetryConstraint

Obtains a SymmetryConstraint instance by name from the supplied container.

Return The SymmetryConstraint instance.

Type DataReference (p. 1371)

Required Arguments

Name The entity name for which to search. Include the @ prefix if searching by display text

Type string (p. 1438)

GetTableResult

Obtains a TableResult instance by name from the supplied container.

Return The TableResult instance.

Type DataReference (p. 1371)

Required Arguments

Name The entity name for which to search. Include the @ prefix if searching by display text

Type string (p. 1438)

GetTaskGroup

Obtains a TaskGroup instance by name from the supplied container.

Return The TaskGroup instance.

Type DataReference (p. 1371)

Required Arguments

Name The entity name for which to search.

Type string (p. 1438)

GetTemperatureCondition

Obtains a TemperatureCondition instance by name from the supplied container.

Return The TemperatureCondition instance.

Type DataReference (p. 1371)

Required Arguments

Name The entity name for which to search. Include the @ prefix if searching by display text

Type string (p. 1438)

GetTemperatureConstraint

Obtains a TemperatureConstraint instance by name from the supplied container.

Return The TemperatureConstraint instance.

Type DataReference (p. 1371)

Required Arguments

Name The entity name for which to search. Include the @ prefix if searching by display text

Type string (p. 1438)

GetThermalMass

Obtains a ThermalMass instance by name from the supplied container.

Return The ThermalMass instance.

Type DataReference (p. 1371)

Required Arguments

Name The entity name for which to search. Include the @ prefix if searching by display text

Type string (p. 1438)

GetTranscript

Obtains a Transcript instance by name from the supplied container.

Return The Transcript instance.

Type DataReference (p. 1371)

Required Arguments

Name The entity name for which to search. Include the @ prefix if searching by display text

Type string (p. 1438)

GetUserCommands

Obtains a UserCommands instance by name from the supplied container.

Return The UserCommands instance.

Type DataReference (p. 1371)

Required Arguments

Name The entity name for which to search. Include the @ prefix if searching by display text

Type string (p. 1438)

GetValidMaterialContent

Query to return a list of valid material model tuples, where each tuple has the strings for use on the CreateMaterialContent command. The tuple is in the following form (Model, Definition, Behavior)

Return A list of the valid material model tuples. The default list returned is in the following form [(Model, Definition, Behavior), ..., (Model, Definition, Behavior)].

Type List (p. 1400)<Object (p. 1409)>

Optional Arguments

IncludeName Include the user friendly name as the first element of a sub-list and second element is the material model tuple. The list returned is in the following form [[Name, (Model, Definition, Behavior)], ..., [Name, (Model, Definition, Behavior)]]

Type bool (p. 1360)

Default Value False

Example

This example requests all the valid material properties including the names.

```
validContent = Study.GetValidMaterialContent(IncludeName=true)
```

GetVectorConstraint

Obtains a VectorConstraint instance by name from the supplied container.

Return The VectorConstraint instance.

Type DataReference (p. 1371)

Required Arguments

Name The entity name for which to search. Include the @ prefix if searching by display text

Type string (p. 1438)

GetVectorInitialCondition

Obtains a VectorInitialCondition instance by name from the supplied container.

Return The VectorInitialCondition instance.

Type DataReference (p. 1371)

Required Arguments

Name The entity name for which to search. Include the @ prefix if searching by display text

Type string (p. 1438)

GetVectorLoad

Obtains a VectorLoad instance by name from the supplied container.

Return The VectorLoad instance.

Type DataReference (p. 1371)

Required Arguments

Name The entity name for which to search. Include the @ prefix if searching by display text

Type string (p. 1438)

GetVectorResult

Obtains a VectorResult instance by name from the supplied container.

Return The VectorResult instance.

Type DataReference (p. 1371)

Required Arguments

Name The entity name for which to search. Include the @ prefix if searching by display text

Type string (p. 1438)

GetVoltage

Obtains a Voltage instance by name from the supplied container.

Return The Voltage instance.

Type DataReference (p. 1371)

Required Arguments

Name The entity name for which to search. Include the @ prefix if searching by display text

Type string (p. 1438)

GetVolumeExtractionRegion

Obtains a VolumeExtractionRegion instance by name from the supplied container.

Return The VolumeExtractionRegion instance.

Type DataReference (p. 1371)

Required Arguments

Name The entity name for which to search. Include the @ prefix if searching by display text

Type string (p. 1438)

GetVolumeSimplificationRegion

Obtains a VolumeSimplificationRegion instance by name from the supplied container.

Return The VolumeSimplificationRegion instance.

Type DataReference (p. 1371)

Required Arguments

Name The entity name for which to search. Include the @ prefix if searching by display text

Type string (p. 1438)

GetWrap

Obtains a Wrap instance by name from the supplied container.

Return The Wrap instance.

Type DataReference (p. 1371)

Required Arguments

Name The entity name for which to search. Include the @ prefix if searching by display text

Type string (p. 1438)

MigrateDefaultBoltModelAsToOneBoltForAllLocations

Create AIM objects of the requested Type.

Return An instance of the Type.

Type DataReference (p. 1371)

Required Arguments

Type The entity Type to create. The entity Type can be seen in the GUI by hovering over the entity's object icon. Also using dir(Study) will show object types for Get<Type> queries, for which the <Type> may be able to be created.

Type string (p. 1438)

Optional Arguments

AssociatedObject The object to associate this object to in the AIM dependency graph, which will create a directed edge in the AIM dependency graph. See discussion on Study and Tasks. This is deprecated in favor of Association.

Type DataReference (p. 1371)

Association The object to associate this object to in the AIM dependency graph, which will create a directed edge in the AIM dependency graph. See discussion on Study and Tasks.

Type DataReference (p. 1371)

Location If the entity has a location property, this will set the property to the provided LocationSet.

Type Object (p. 1409)

Default Value Ansys.Study.Core.Common.Referencing.LocationSet

Location2 If the entity has a second location property, this will set the property to the provided LocationSet.

Type Object (p. 1409)

Default Value Ansys.Study.Core.Common.Referencing.LocationSet

LocationWithHitPoints If the entity has a location property with hit points, this will set the property to the provided LocationSet.

Type Object (p. 1409)

Default Value Ansys.Study.Core.Common.Referencing.LocationSet

LocationWithHitPoints2 If the entity has a second location property with hit points, this will set the property to the provided LocationSet.

Type Object (p. 1409)

Default Value Ansys.Study.Core.Common.Referencing.Location-Set

ObjectReferences Future use

Type DataReferenceSet (p. 1371)

Example

```
This example shows the steps preceding the CreateEntity, and then creates a PhysicsRegion object.
system1 = GetSystem(Name="Study")
physicsDefinitionComponent1 = Study.CreateTask(
    Type="Physics Definition",
    System=system1)
study1 = system1.GetContainer(ComponentName="Study")
physicsDefinition1 = physicsDefinitionComponent1.GetTaskObject()
physicsRegion1 = study1.CreateEntity(
    Type="PhysicsRegion",
    Association=physicsDefinition1)
```

MigrateDefaultSupportTypeToUserdefined

Create AIM objects of the requested Type.

Return An instance of the Type.

Type DataReference (p. 1371)

Required Arguments

Type The entity Type to create. The entity Type can be seen in the GUI by hovering over the entity's object icon. Also using dir(Study) will show object types for Get<Type> queries, for which the <Type> may be able to be created.

Type string (p. 1438)

Optional Arguments

AssociatedObject The object to associate this object to in the AIM dependency graph, which will create a directed edge in the AIM dependency graph. See discussion on Study and Tasks. This is deprecated in favor of Association.

Type DataReference (p. 1371)

Association The object to associate this object to in the AIM dependency graph, which will create a directed edge in the AIM dependency graph. See discussion on Study and Tasks.

Type DataReference (p. 1371)

Location If the entity has a location property, this will set the property to the provided LocationSet.

Type Object (p. 1409)

Default Value Ansys.Study.Core.Common.Referencing.LocationSet

Location2 If the entity has a second location property, this will set the property to the provided LocationSet.

Type Object (p. 1409)

Default Value Ansys.Study.Core.Common.Referencing.LocationSet

LocationWithHitPoints If the entity has a location property with hit points, this will set the property to the provided LocationSet.

Type Object (p. 1409)

Default Value Ansys.Study.Core.Common.Referencing.LocationSet

LocationWithHitPoints2 If the entity has a second location property with hit points, this will set the property to the provided LocationSet.

Type Object (p. 1409)

Default Value Ansys.Study.Core.Common.Referencing.LocationSet

ObjectReferences Future use

Type DataReferenceSet (p. 1371)

Example

```
This example shows the steps preceding the CreateEntity, and then creates a PhysicsRegion object.
system1 = GetSystem(Name="Study")
physicsDefinitionComponent1 = Study.CreateTask(
    Type="Physics Definition",
    System=system1)
study1 = system1.GetContainer(ComponentName="Study")
physicsDefinition1 = physicsDefinitionComponent1.GetTaskObject()
physicsRegion1 = study1.CreateEntity(
    Type="PhysicsRegion",
    Association=physicsDefinition1)
```

MigrateDefaultVectorDefineByToComponents

Create AIM objects of the requested Type.

Return An instance of the Type.

Type DataReference (p. 1371)

Required Arguments

Type The entity Type to create. The entity Type can be seen in the GUI by hovering over the entity's object icon. Also using dir(Study) will show object types for Get<Type> queries, for which the <Type> may be able to be created.

Type string (p. 1438)

Optional Arguments

AssociatedObject The object to associate this object to in the AIM dependency graph, which will create a directed edge in the AIM dependency graph. See discussion on Study and Tasks. This is deprecated in favor of Association.

Type DataReference (p. 1371)

Association The object to associate this object to in the AIM dependency graph, which will create a directed edge in the AIM dependency graph. See discussion on Study and Tasks.

Type DataReference (p. 1371)

Location If the entity has a location property, this will set the property to the provided LocationSet.

Type Object (p. 1409)

Default Value Ansys.Study.Core.Common.Referencing.LocationSet

Location2 If the entity has a second location property, this will set the property to the provided LocationSet.

Type Object (p. 1409)

Default Value Ansys.Study.Core.Common.Referencing.LocationSet

LocationWithHitPoints If the entity has a location property with hit points, this will set the property to the provided LocationSet.

Type Object (p. 1409)

Default Value Ansys.Study.Core.Common.Referencing.LocationSet

LocationWithHitPoints2 If the entity has a second location property with hit points, this will set the property to the provided LocationSet.

Type Object (p. 1409)

Default Value Ansys.Study.Core.Common.Referencing.LocationSet

ObjectReferences Future use

Type DataReferenceSet (p. 1371)

Example

```
This example shows the steps preceding the CreateEntity, and then creates a PhysicsRegion object.
system1 = GetSystem(Name="Study")
physicsDefinitionComponent1 = Study.CreateTask(
    Type="Physics Definition",
    System=system1)
study1 = system1.GetContainer(ComponentName="Study")
physicsDefinition1 = physicsDefinitionComponent1.GetTaskObject()
physicsRegion1 = study1.CreateEntity(
    Type="PhysicsRegion",
    Association=physicsDefinition1)
```

TryGetBeamEndRelease

Obtains a BeamEndRelease instance by name from the supplied container, falling back to an alternate name if the first is not found.

Return The BeamEndRelease instance.

Type DataReference (p. 1371)

Required Arguments

AlternateName The entity name for which to search. Include the @ prefix if searching by display text.

Type string (p. 1438)

Name The entity name for which to search. Include the @ prefix if searching by display text.

Type string (p. 1438)

TryGetBeamSection

Obtains a BeamSection instance by name from the supplied container, falling back to an alternate name if the first is not found.

Return The BeamSection instance.

Type DataReference (p. 1371)

Required Arguments

AlternateName The entity name for which to search. Include the @ prefix if searching by display text.

Type string (p. 1438)

Name The entity name for which to search. Include the @ prefix if searching by display text.

Type string (p. 1438)

TryGetBearingLoad

Obtains a BearingLoad instance by name from the supplied container, falling back to an alternate name if the first is not found.

Return The BearingLoad instance.

Type DataReference (p. 1371)

Required Arguments

AlternateName The entity name for which to search. Include the @ prefix if searching by display text.

Type string (p. 1438)

Name The entity name for which to search. Include the @ prefix if searching by display text.

Type string (p. 1438)

TryGetBoltPretension

Obtains a BoltPretension instance by name from the supplied container, falling back to an alternate name if the first is not found.

Return The BoltPretension instance.

Type DataReference (p. 1371)

Required Arguments

AlternateName The entity name for which to search. Include the @ prefix if searching by display text.

Type string (p. 1438)

Name The entity name for which to search. Include the @ prefix if searching by display text.

Type string (p. 1438)

TryGetBoundaryCondition

Obtains a BoundaryCondition instance by name from the supplied container, falling back to an alternate name if the first is not found.

Return The BoundaryCondition instance.

Type DataReference (p. 1371)

Required Arguments

AlternateName The entity name for which to search. Include the @ prefix if searching by display text.

Type string (p. 1438)

Name The entity name for which to search. Include the @ prefix if searching by display text.

Type string (p. 1438)

TryGetBoundaryLayerRefinement

Obtains a BoundaryLayerRefinement instance by name from the supplied container, falling back to an alternate name if the first is not found.

Return The BoundaryLayerRefinement instance.

Type DataReference (p. 1371)

Required Arguments

AlternateName The entity name for which to search. Include the @ prefix if searching by display text.

Type string (p. 1438)

Name The entity name for which to search. Include the @ prefix if searching by display text.

Type string (p. 1438)

TryGetCappingSurface

Obtains a CappingSurface instance by name from the supplied container, falling back to an alternate name if the first is not found.

Return The CappingSurface instance.

Type DataReference (p. 1371)

Required Arguments

AlternateName The entity name for which to search. Include the @ prefix if searching by display text.

Type string (p. 1438)

Name The entity name for which to search. Include the @ prefix if searching by display text.

Type string (p. 1438)

TryGetCharge

Obtains a Charge instance by name from the supplied container, falling back to an alternate name if the first is not found.

Return The Charge instance.

Type DataReference (p. 1371)

Required Arguments

AlternateName The entity name for which to search. Include the @ prefix if searching by display text.

Type string (p. 1438)

Name The entity name for which to search. Include the @ prefix if searching by display text.

Type string (p. 1438)

TryGetConditionGroup

Obtains a ConditionGroup instance by name from the supplied container, falling back to an alternate name if the first is not found.

Return The ConditionGroup instance.

Type DataReference (p. 1371)

Required Arguments

AlternateName The entity name for which to search. Include the @ prefix if searching by display text.

Type string (p. 1438)

Name The entity name for which to search. Include the @ prefix if searching by display text.

Type string (p. 1438)

TryGetConfiguration

Obtains a Configuration instance by name from the supplied container, falling back to an alternate name if the first is not found.

Return The Configuration instance.

Type DataReference (p. 1371)

Required Arguments

AlternateName The entity name for which to search. Include the @ prefix if searching by display text.

Type string (p. 1438)

Name The entity name for which to search. Include the @ prefix if searching by display text.

Type string (p. 1438)

TryGetConnection

Obtains a Connection instance by name from the supplied container, falling back to an alternate name if the first is not found.

Return The Connection instance.

Type DataReference (p. 1371)

Required Arguments

AlternateName The entity name for which to search. Include the @ prefix if searching by display text.

Type string (p. 1438)

Name The entity name for which to search. Include the @ prefix if searching by display text.

Type string (p. 1438)

TryGetConnectionBehavior

Obtains a ConnectionBehavior instance by name from the supplied container, falling back to an alternate name if the first is not found.

Return The ConnectionBehavior instance.

Type DataReference (p. 1371)

Required Arguments

AlternateName The entity name for which to search. Include the @ prefix if searching by display text.

Type string (p. 1438)

Name The entity name for which to search. Include the @ prefix if searching by display text.

Type string (p. 1438)

TryGetConstraint

Obtains a Constraint instance by name from the supplied container, falling back to an alternate name if the first is not found.

Return The Constraint instance.

Type DataReference (p. 1371)

Required Arguments

AlternateName The entity name for which to search. Include the @ prefix if searching by display text.

Type string (p. 1438)

Name The entity name for which to search. Include the @ prefix if searching by display text.

Type string (p. 1438)

TryGetContact

Obtains a Contact instance by name from the supplied container, falling back to an alternate name if the first is not found.

Return The Contact instance.

Type DataReference (p. 1371)

Required Arguments

AlternateName The entity name for which to search. Include the @ prefix if searching by display text.

Type string (p. 1438)

Name The entity name for which to search. Include the @ prefix if searching by display text.

Type string (p. 1438)

TryGetContactBehavior

Obtains a ContactBehavior instance by name from the supplied container, falling back to an alternate name if the first is not found.

Return The ContactBehavior instance.

Type DataReference (p. 1371)

Required Arguments

AlternateName The entity name for which to search. Include the @ prefix if searching by display text.

Type string (p. 1438)

Name The entity name for which to search. Include the @ prefix if searching by display text.

Type string (p. 1438)

TryGetContactDetection

Obtains a ContactDetection instance by name from the supplied container, falling back to an alternate name if the first is not found.

Return The ContactDetection instance.

Type DataReference (p. 1371)

Required Arguments

AlternateName The entity name for which to search. Include the @ prefix if searching by display text.

Type string (p. 1438)

Name The entity name for which to search. Include the @ prefix if searching by display text.

Type string (p. 1438)

TryGetContourResult

Obtains a ContourResult instance by name from the supplied container, falling back to an alternate name if the first is not found.

Return The ContourResult instance.

Type DataReference (p. 1371)

Required Arguments

AlternateName The entity name for which to search. Include the @ prefix if searching by display text.

Type string (p. 1438)

Name The entity name for which to search. Include the @ prefix if searching by display text.

Type string (p. 1438)

TryGetConvection

Obtains a Convection instance by name from the supplied container, falling back to an alternate name if the first is not found.

Return The Convection instance.

Type DataReference (p. 1371)

Required Arguments

AlternateName The entity name for which to search. Include the @ prefix if searching by display text.

Type string (p. 1438)

Name The entity name for which to search. Include the @ prefix if searching by display text.

Type string (p. 1438)

TryGetCurrent

Obtains a Current instance by name from the supplied container, falling back to an alternate name if the first is not found.

Return The Current instance.

Type DataReference (p. 1371)

Required Arguments

AlternateName The entity name for which to search. Include the @ prefix if searching by display text.

Type string (p. 1438)

Name The entity name for which to search. Include the @ prefix if searching by display text.

Type string (p. 1438)

TryGetCyclicSymmetryConstraint

Obtains a CyclicSymmetryConstraint instance by name from the supplied container, falling back to an alternate name if the first is not found.

Return The CyclicSymmetryConstraint instance.

Type DataReference (p. 1371)

Required Arguments

AlternateName The entity name for which to search. Include the @ prefix if searching by display text.

Type string (p. 1438)

Name The entity name for which to search. Include the @ prefix if searching by display text.

Type string (p. 1438)

TryGetDisplacement

Obtains a Displacement instance by name from the supplied container, falling back to an alternate name if the first is not found.

Return The Displacement instance.

Type DataReference (p. 1371)

Required Arguments

AlternateName The entity name for which to search. Include the @ prefix if searching by display text.

Type string (p. 1438)

Name The entity name for which to search. Include the @ prefix if searching by display text.

Type string (p. 1438)

TryGetDistributedMass

Obtains a DistributedMass instance by name from the supplied container, falling back to an alternate name if the first is not found.

Return The DistributedMass instance.

Type DataReference (p. 1371)

Required Arguments

AlternateName The entity name for which to search. Include the @ prefix if searching by display text.

Type string (p. 1438)

Name The entity name for which to search. Include the @ prefix if searching by display text.

Type string (p. 1438)

TryGetEquipotential

Obtains a Equipotential instance by name from the supplied container, falling back to an alternate name if the first is not found.

Return The Equipotential instance.

Type DataReference (p. 1371)

Required Arguments

AlternateName The entity name for which to search. Include the @ prefix if searching by display text.

Type string (p. 1438)

Name The entity name for which to search. Include the @ prefix if searching by display text.

Type string (p. 1438)

TryGetFatigueSettings

Obtains a FatigueSettings instance by name from the supplied container, falling back to an alternate name if the first is not found.

Return The FatigueSettings instance.

Type DataReference (p. 1371)

Required Arguments

AlternateName The entity name for which to search. Include the @ prefix if searching by display text.

Type string (p. 1438)

Name The entity name for which to search. Include the @ prefix if searching by display text.

Type string (p. 1438)

TryGetFidelityRefinement

Obtains a FidelityRefinement instance by name from the supplied container, falling back to an alternate name if the first is not found.

Return The FidelityRefinement instance.

Type DataReference (p. 1371)

Required Arguments

AlternateName The entity name for which to search. Include the @ prefix if searching by display text.

Type string (p. 1438)

Name The entity name for which to search. Include the @ prefix if searching by display text.

Type string (p. 1438)

TryGetForce

Obtains a Force instance by name from the supplied container, falling back to an alternate name if the first is not found.

Return The Force instance.

Type DataReference (p. 1371)

Required Arguments

AlternateName The entity name for which to search. Include the @ prefix if searching by display text.

Type string (p. 1438)

Name The entity name for which to search. Include the @ prefix if searching by display text.

Type string (p. 1438)

TryGetGeometryImportSource

Obtains a GeometryImportSource instance by name from the supplied container, falling back to an alternate name if the first is not found.

Return The GeometryImportSource instance.

Type DataReference (p. 1371)

Required Arguments

AlternateName The entity name for which to search. Include the @ prefix if searching by display text.

Type string (p. 1438)

Name The entity name for which to search. Include the @ prefix if searching by display text.

Type string (p. 1438)

TryGetGravityDefinition

Obtains a GravityDefinition instance by name from the supplied container, falling back to an alternate name if the first is not found.

Return The GravityDefinition instance.

Type DataReference (p. 1371)

Required Arguments

AlternateName The entity name for which to search. Include the @ prefix if searching by display text.

Type string (p. 1438)

Name The entity name for which to search. Include the @ prefix if searching by display text.

Type string (p. 1438)

TryGetGroundLocation

Obtains a GroundLocation instance by name from the supplied container, falling back to an alternate name if the first is not found.

Return The GroundLocation instance.

Type DataReference (p. 1371)

Required Arguments

AlternateName The entity name for which to search. Include the @ prefix if searching by display text.

Type string (p. 1438)

Name The entity name for which to search. Include the @ prefix if searching by display text.

Type string (p. 1438)

TryGetHeatFlow

Obtains a HeatFlow instance by name from the supplied container, falling back to an alternate name if the first is not found.

Return The HeatFlow instance.

Type DataReference (p. 1371)

Required Arguments

AlternateName The entity name for which to search. Include the @ prefix if searching by display text.

Type string (p. 1438)

Name The entity name for which to search. Include the @ prefix if searching by display text.

Type string (p. 1438)

TryGetHeatFlux

Obtains a HeatFlux instance by name from the supplied container, falling back to an alternate name if the first is not found.

Return The HeatFlux instance.

Type DataReference (p. 1371)

Required Arguments

AlternateName The entity name for which to search. Include the @ prefix if searching by display text.

Type string (p. 1438)

Name The entity name for which to search. Include the @ prefix if searching by display text.

Type string (p. 1438)

TryGetHeatGeneration

Obtains a HeatGeneration instance by name from the supplied container, falling back to an alternate name if the first is not found.

Return The HeatGeneration instance.

Type DataReference (p. 1371)

Required Arguments

AlternateName The entity name for which to search. Include the @ prefix if searching by display text.

Type string (p. 1438)

Name The entity name for which to search. Include the @ prefix if searching by display text.

Type string (p. 1438)

TryGetHeatSrc

Obtains a HeatSrc instance by name from the supplied container, falling back to an alternate name if the first is not found.

Return The HeatSrc instance.

Type DataReference (p. 1371)

Required Arguments

AlternateName The entity name for which to search. Include the @ prefix if searching by display text.

Type string (p. 1438)

Name The entity name for which to search. Include the @ prefix if searching by display text.

Type string (p. 1438)

TryGetImport

Obtains a Import instance by name from the supplied container, falling back to an alternate name if the first is not found.

Return The Import instance.

Type DataReference (p. 1371)

Required Arguments

AlternateName The entity name for which to search. Include the @ prefix if searching by display text.

Type string (p. 1438)

Name The entity name for which to search. Include the @ prefix if searching by display text.

Type string (p. 1438)

TryGetInertiaLoad

Obtains a InertiaLoad instance by name from the supplied container, falling back to an alternate name if the first is not found.

Return The InertiaLoad instance.

Type DataReference (p. 1371)

Required Arguments

AlternateName The entity name for which to search. Include the @ prefix if searching by display text.

Type string (p. 1438)

Name The entity name for which to search. Include the @ prefix if searching by display text.

Type string (p. 1438)

TryGetInitialCondition

Obtains a InitialCondition instance by name from the supplied container, falling back to an alternate name if the first is not found.

Return The InitialCondition instance.

Type DataReference (p. 1371)

Required Arguments

AlternateName The entity name for which to search. Include the @ prefix if searching by display text.

Type string (p. 1438)

Name The entity name for which to search. Include the @ prefix if searching by display text.

Type string (p. 1438)

TryGetInitialTemperature

Obtains a InitialTemperature instance by name from the supplied container, falling back to an alternate name if the first is not found.

Return The InitialTemperature instance.

Type DataReference (p. 1371)

Required Arguments

AlternateName The entity name for which to search. Include the @ prefix if searching by display text.

Type string (p. 1438)

Name The entity name for which to search. Include the @ prefix if searching by display text.

Type string (p. 1438)

TryGetInterfaceGenerator

Obtains a InterfaceGenerator instance by name from the supplied container, falling back to an alternate name if the first is not found.

Return The InterfaceGenerator instance.

Type DataReference (p. 1371)

Required Arguments

AlternateName The entity name for which to search. Include the @ prefix if searching by display text.

Type string (p. 1438)

Name The entity name for which to search. Include the @ prefix if searching by display text.

Type string (p. 1438)

TryGetIsoSurface

Obtains a IsoSurface instance by name from the supplied container, falling back to an alternate name if the first is not found.

Return The IsoSurface instance.

Type DataReference (p. 1371)

Required Arguments

AlternateName The entity name for which to search. Include the @ prefix if searching by display text.

Type string (p. 1438)

Name The entity name for which to search. Include the @ prefix if searching by display text.

Type string (p. 1438)

TryGetJoint

Obtains a Joint instance by name from the supplied container, falling back to an alternate name if the first is not found.

Return The Joint instance.

Type DataReference (p. 1371)

Required Arguments

AlternateName The entity name for which to search. Include the @ prefix if searching by display text.

Type string (p. 1438)

Name The entity name for which to search. Include the @ prefix if searching by display text.

Type string (p. 1438)

TryGetJointBehavior

Obtains a JointBehavior instance by name from the supplied container, falling back to an alternate name if the first is not found.

Return The JointBehavior instance.

Type DataReference (p. 1371)

Required Arguments

AlternateName The entity name for which to search. Include the @ prefix if searching by display text.

Type string (p. 1438)

Name The entity name for which to search. Include the @ prefix if searching by display text.

Type string (p. 1438)

TryGetLaunchSettings

Obtains a LaunchSettings instance by name from the supplied container, falling back to an alternate name if the first is not found.

Return The LaunchSettings instance.

Type DataReference (p. 1371)

Required Arguments

AlternateName The entity name for which to search. Include the @ prefix if searching by display text.

Type string (p. 1438)

Name The entity name for which to search. Include the @ prefix if searching by display text.

Type string (p. 1438)

TryGetLine

Obtains a Line instance by name from the supplied container, falling back to an alternate name if the first is not found.

Return The Line instance.

Type DataReference (p. 1371)

Required Arguments

AlternateName The entity name for which to search. Include the @ prefix if searching by display text.

Type string (p. 1438)

Name The entity name for which to search. Include the @ prefix if searching by display text.

Type string (p. 1438)

TryGetLinearizedStressChart

Obtains a LinearizedStressChart instance by name from the supplied container, falling back to an alternate name if the first is not found.

Return The LinearizedStressChart instance.

Type DataReference (p. 1371)

Required Arguments

AlternateName The entity name for which to search. Include the @ prefix if searching by display text.

Type string (p. 1438)

Name The entity name for which to search. Include the @ prefix if searching by display text.

Type string (p. 1438)

TryGetLineChart

Obtains a LineChart instance by name from the supplied container, falling back to an alternate name if the first is not found.

Return The LineChart instance.

Type DataReference (p. 1371)

Required Arguments

AlternateName The entity name for which to search. Include the @ prefix if searching by display text.

Type string (p. 1438)

Name The entity name for which to search. Include the @ prefix if searching by display text.

Type string (p. 1438)

TryGetLoad

Obtains a Load instance by name from the supplied container, falling back to an alternate name if the first is not found.

Return The Load instance.

Type DataReference (p. 1371)

Required Arguments

AlternateName The entity name for which to search. Include the @ prefix if searching by display text.

Type string (p. 1438)

Name The entity name for which to search. Include the @ prefix if searching by display text.

Type string (p. 1438)

TryGetMaterial

Obtains a Material instance by name from the supplied container, falling back to an alternate name if the first is not found.

Return The Material instance.

Type DataReference (p. 1371)

Required Arguments

AlternateName The entity name for which to search. Include the @ prefix if searching by display text.

Type string (p. 1438)

Name The entity name for which to search. Include the @ prefix if searching by display text.

Type string (p. 1438)

TryGetMaterialAssignment

Obtains a MaterialAssignment instance by name from the supplied container, falling back to an alternate name if the first is not found.

Return The MaterialAssignment instance.

Type DataReference (p. 1371)

Required Arguments

AlternateName The entity name for which to search. Include the @ prefix if searching by display text.

Type string (p. 1438)

Name The entity name for which to search. Include the @ prefix if searching by display text.

Type string (p. 1438)

TryGetMaterialAtState

Obtains a MaterialAtState instance by name from the supplied container, falling back to an alternate name if the first is not found.

Return The MaterialAtState instance.

Type DataReference (p. 1371)

Required Arguments

AlternateName The entity name for which to search. Include the @ prefix if searching by display text.

Type string (p. 1438)

Name The entity name for which to search. Include the @ prefix if searching by display text.

Type string (p. 1438)

TryGetMechanicalPhysicsOptions

Obtains a MechanicalPhysicsOptions instance by name from the supplied container, falling back to an alternate name if the first is not found.

Return The MechanicalPhysicsOptions instance.

Type DataReference (p. 1371)

Required Arguments

AlternateName The entity name for which to search. Include the @ prefix if searching by display text.

Type string (p. 1438)

Name The entity name for which to search. Include the @ prefix if searching by display text.

Type string (p. 1438)

TryGetMemberSizeConstraint

Obtains a MemberSizeConstraint instance by name from the supplied container, falling back to an alternate name if the first is not found.

Return The MemberSizeConstraint instance.

Type DataReference (p. 1371)

Required Arguments

AlternateName The entity name for which to search. Include the @ prefix if searching by display text.

Type string (p. 1438)

Name The entity name for which to search. Include the @ prefix if searching by display text.

Type string (p. 1438)

TryGetMesh

Obtains a Mesh instance by name from the supplied container, falling back to an alternate name if the first is not found.

Return The Mesh instance.

Type DataReference (p. 1371)

Required Arguments

AlternateName The entity name for which to search. Include the @ prefix if searching by display text.

Type string (p. 1438)

Name The entity name for which to search. Include the @ prefix if searching by display text.

Type string (p. 1438)

TryGetMeshControl

Obtains a MeshControl instance by name from the supplied container, falling back to an alternate name if the first is not found.

Return The MeshControl instance.

Type DataReference (p. 1371)

Required Arguments

AlternateName The entity name for which to search. Include the @ prefix if searching by display text.

Type string (p. 1438)

Name The entity name for which to search. Include the @ prefix if searching by display text.

Type string (p. 1438)

TryGetMeshControlBodySizing

Obtains a MeshControlBodySizing instance by name from the supplied container, falling back to an alternate name if the first is not found.

Return The MeshControlBodySizing instance.

Type DataReference (p. 1371)

Required Arguments

AlternateName The entity name for which to search. Include the @ prefix if searching by display text.

Type string (p. 1438)

Name The entity name for which to search. Include the @ prefix if searching by display text.

Type string (p. 1438)

TryGetMeshControlEdgeSizing

Obtains a MeshControlEdgeSizing instance by name from the supplied container, falling back to an alternate name if the first is not found.

Return The MeshControlEdgeSizing instance.

Type DataReference (p. 1371)

Required Arguments

AlternateName The entity name for which to search. Include the @ prefix if searching by display text.

Type string (p. 1438)

Name The entity name for which to search. Include the @ prefix if searching by display text.

Type string (p. 1438)

TryGetMeshControlElementShape

Obtains a MeshControlElementShape instance by name from the supplied container, falling back to an alternate name if the first is not found.

Return The MeshControlElementShape instance.

Type DataReference (p. 1371)

Required Arguments

AlternateName The entity name for which to search. Include the @ prefix if searching by display text.

Type string (p. 1438)

Name The entity name for which to search. Include the @ prefix if searching by display text.

Type string (p. 1438)

TryGetMeshControlFaceSizing

Obtains a MeshControlFaceSizing instance by name from the supplied container, falling back to an alternate name if the first is not found.

Return The MeshControlFaceSizing instance.

Type DataReference (p. 1371)

Required Arguments

AlternateName The entity name for which to search. Include the @ prefix if searching by display text.

Type string (p. 1438)

Name The entity name for which to search. Include the @ prefix if searching by display text.

Type string (p. 1438)

TryGetMeshControlLocalBoundaryLayer

Obtains a MeshControlLocalBoundaryLayer instance by name from the supplied container, falling back to an alternate name if the first is not found.

Return The MeshControlLocalBoundaryLayer instance.

Type DataReference (p. 1371)

Required Arguments

AlternateName The entity name for which to search. Include the @ prefix if searching by display text.

Type string (p. 1438)

Name The entity name for which to search. Include the @ prefix if searching by display text.

Type string (p. 1438)

TryGetMeshDiagnostics

Obtains a MeshDiagnostics instance by name from the supplied container, falling back to an alternate name if the first is not found.

Return The MeshDiagnostics instance.

Type DataReference (p. 1371)

Required Arguments

AlternateName The entity name for which to search. Include the @ prefix if searching by display text.

Type string (p. 1438)

Name The entity name for which to search. Include the @ prefix if searching by display text.

Type string (p. 1438)

TryGetMeshing

Obtains a Meshing instance by name from the supplied container, falling back to an alternate name if the first is not found.

Return The Meshing instance.

Type DataReference (p. 1371)

Required Arguments

AlternateName The entity name for which to search. Include the @ prefix if searching by display text.

Type string (p. 1438)

Name The entity name for which to search. Include the @ prefix if searching by display text.

Type string (p. 1438)

TryGetMeshModeling

Obtains a MeshModeling instance by name from the supplied container, falling back to an alternate name if the first is not found.

Return The MeshModeling instance.

Type DataReference (p. 1371)

Required Arguments

AlternateName The entity name for which to search. Include the @ prefix if searching by display text.

Type string (p. 1438)

Name The entity name for which to search. Include the @ prefix if searching by display text.

Type string (p. 1438)

TryGetModeling

Obtains a Modeling instance by name from the supplied container, falling back to an alternate name if the first is not found.

Return The Modeling instance.

Type DataReference (p. 1371)

Required Arguments

AlternateName The entity name for which to search. Include the @ prefix if searching by display text.

Type string (p. 1438)

Name The entity name for which to search. Include the @ prefix if searching by display text.

Type string (p. 1438)

TryGetModelingImportSource

Obtains a ModelingImportSource instance by name from the supplied container, falling back to an alternate name if the first is not found.

Return The ModelingImportSource instance.

Type DataReference (p. 1371)

Required Arguments

AlternateName The entity name for which to search. Include the @ prefix if searching by display text.

Type string (p. 1438)

Name The entity name for which to search. Include the @ prefix if searching by display text.

Type string (p. 1438)

TryGetMoment

Obtains a Moment instance by name from the supplied container, falling back to an alternate name if the first is not found.

Return The Moment instance.

Type DataReference (p. 1371)

Required Arguments

AlternateName The entity name for which to search. Include the @ prefix if searching by display text.

Type string (p. 1438)

Name The entity name for which to search. Include the @ prefix if searching by display text.

Type string (p. 1438)

TryGetMomentumSrc

Obtains a MomentumSrc instance by name from the supplied container, falling back to an alternate name if the first is not found.

Return The MomentumSrc instance.

Type DataReference (p. 1371)

Required Arguments

AlternateName The entity name for which to search. Include the @ prefix if searching by display text.

Type string (p. 1438)

Name The entity name for which to search. Include the @ prefix if searching by display text.

Type string (p. 1438)

TryGetMonitorChart

Obtains a MonitorChart instance by name from the supplied container, falling back to an alternate name if the first is not found.

Return The MonitorChart instance.

Type DataReference (p. 1371)

Required Arguments

AlternateName The entity name for which to search. Include the @ prefix if searching by display text.

Type string (p. 1438)

Name The entity name for which to search. Include the @ prefix if searching by display text.

Type string (p. 1438)

TryGetMoveRotateControl

Obtains a MoveRotateControl instance by name from the supplied container, falling back to an alternate name if the first is not found.

Return The MoveRotateControl instance.

Type DataReference (p. 1371)

Required Arguments

AlternateName The entity name for which to search. Include the @ prefix if searching by display text.

Type string (p. 1438)

Name The entity name for which to search. Include the @ prefix if searching by display text.

Type string (p. 1438)

TryGetOptimizationConstraint

Obtains a OptimizationConstraint instance by name from the supplied container, falling back to an alternate name if the first is not found.

Return The OptimizationConstraint instance.

Type DataReference (p. 1371)

Required Arguments

AlternateName The entity name for which to search. Include the @ prefix if searching by display text.

Type string (p. 1438)

Name The entity name for which to search. Include the @ prefix if searching by display text.

Type string (p. 1438)

TryGetOptimizationOptions

Obtains a OptimizationOptions instance by name from the supplied container, falling back to an alternate name if the first is not found.

Return The OptimizationOptions instance.

Type DataReference (p. 1371)

Required Arguments

AlternateName The entity name for which to search. Include the @ prefix if searching by display text.

Type string (p. 1438)

Name The entity name for which to search. Include the @ prefix if searching by display text.

Type string (p. 1438)

TryGetOutputControls

Obtains a OutputControls instance by name from the supplied container, falling back to an alternate name if the first is not found.

Return The OutputControls instance.

Type DataReference (p. 1371)

Required Arguments

AlternateName The entity name for which to search. Include the @ prefix if searching by display text.

Type string (p. 1438)

Name The entity name for which to search. Include the @ prefix if searching by display text.

Type string (p. 1438)

TryGetParticleInjectionCnd

Obtains a ParticleInjectionCnd instance by name from the supplied container, falling back to an alternate name if the first is not found.

Return The ParticleInjectionCnd instance.

Type DataReference (p. 1371)

Required Arguments

AlternateName The entity name for which to search. Include the @ prefix if searching by display text.

Type string (p. 1438)

Name The entity name for which to search. Include the @ prefix if searching by display text.

Type string (p. 1438)

TryGetParticleTrack

Obtains a ParticleTrack instance by name from the supplied container, falling back to an alternate name if the first is not found.

Return The ParticleTrack instance.

Type DataReference (p. 1371)

Required Arguments

AlternateName The entity name for which to search. Include the @ prefix if searching by display text.

Type string (p. 1438)

Name The entity name for which to search. Include the @ prefix if searching by display text.

Type string (p. 1438)

TryGetPerimeterWeld

Obtains a PerimeterWeld instance by name from the supplied container, falling back to an alternate name if the first is not found.

Return The PerimeterWeld instance.

Type DataReference (p. 1371)

Required Arguments

AlternateName The entity name for which to search. Include the @ prefix if searching by display text.

Type string (p. 1438)

Name The entity name for which to search. Include the @ prefix if searching by display text.

Type string (p. 1438)

TryGetPhysicsCoupling

Obtains a PhysicsCoupling instance by name from the supplied container, falling back to an alternate name if the first is not found.

Return The PhysicsCoupling instance.

Type DataReference (p. 1371)

Required Arguments

AlternateName The entity name for which to search. Include the @ prefix if searching by display text.

Type string (p. 1438)

Name The entity name for which to search. Include the @ prefix if searching by display text.

Type string (p. 1438)

TryGetPhysicsDefinition

Obtains a PhysicsDefinition instance by name from the supplied container, falling back to an alternate name if the first is not found.

Return The PhysicsDefinition instance.

Type DataReference (p. 1371)

Required Arguments

AlternateName The entity name for which to search. Include the @ prefix if searching by display text.

Type string (p. 1438)

Name The entity name for which to search. Include the @ prefix if searching by display text.

Type string (p. 1438)

TryGetPhysicsOptions

Obtains a PhysicsOptions instance by name from the supplied container, falling back to an alternate name if the first is not found.

Return The PhysicsOptions instance.

Type DataReference (p. 1371)

Required Arguments

AlternateName The entity name for which to search. Include the @ prefix if searching by display text.

Type string (p. 1438)

Name The entity name for which to search. Include the @ prefix if searching by display text.

Type string (p. 1438)

TryGetPhysicsRegion

Obtains a PhysicsRegion instance by name from the supplied container, falling back to an alternate name if the first is not found.

Return The PhysicsRegion instance.

Type DataReference (p. 1371)

Required Arguments

AlternateName The entity name for which to search. Include the @ prefix if searching by display text.

Type string (p. 1438)

Name The entity name for which to search. Include the @ prefix if searching by display text.

Type string (p. 1438)

TryGetPlane

Obtains a Plane instance by name from the supplied container, falling back to an alternate name if the first is not found.

Return The Plane instance.

Type DataReference (p. 1371)

Required Arguments

AlternateName The entity name for which to search. Include the @ prefix if searching by display text.

Type string (p. 1438)

Name The entity name for which to search. Include the @ prefix if searching by display text.

Type string (p. 1438)

TryGetPoint

Obtains a Point instance by name from the supplied container, falling back to an alternate name if the first is not found.

Return The Point instance.

Type DataReference (p. 1371)

Required Arguments

AlternateName The entity name for which to search. Include the @ prefix if searching by display text.

Type string (p. 1438)

Name The entity name for which to search. Include the @ prefix if searching by display text.

Type string (p. 1438)

TryGetPolyflowCondition

Obtains a PolyflowCondition instance by name from the supplied container, falling back to an alternate name if the first is not found.

Return The PolyflowCondition instance.

Type DataReference (p. 1371)

Required Arguments

AlternateName The entity name for which to search. Include the @ prefix if searching by display text.

Type string (p. 1438)

Name The entity name for which to search. Include the @ prefix if searching by display text.

Type string (p. 1438)

TryGetPolyflowContact

Obtains a PolyflowContact instance by name from the supplied container, falling back to an alternate name if the first is not found.

Return The PolyflowContact instance.

Type DataReference (p. 1371)

Required Arguments

AlternateName The entity name for which to search. Include the @ prefix if searching by display text.

Type string (p. 1438)

Name The entity name for which to search. Include the @ prefix if searching by display text.

Type string (p. 1438)

TryGetPolyflowContactBehavior

Obtains a PolyflowContactBehavior instance by name from the supplied container, falling back to an alternate name if the first is not found.

Return The PolyflowContactBehavior instance.

Type DataReference (p. 1371)

Required Arguments

AlternateName The entity name for which to search. Include the @ prefix if searching by display text.

Type string (p. 1438)

Name The entity name for which to search. Include the @ prefix if searching by display text.

Type string (p. 1438)

TryGetPolyflowConvection

Obtains a PolyflowConvection instance by name from the supplied container, falling back to an alternate name if the first is not found.

Return The PolyflowConvection instance.

Type DataReference (p. 1371)

Required Arguments

AlternateName The entity name for which to search. Include the @ prefix if searching by display text.

Type string (p. 1438)

Name The entity name for which to search. Include the @ prefix if searching by display text.

Type string (p. 1438)

TryGetPolyflowDieDeformation

Obtains a PolyflowDieDeformation instance by name from the supplied container, falling back to an alternate name if the first is not found.

Return The PolyflowDieDeformation instance.

Type DataReference (p. 1371)

Required Arguments

AlternateName The entity name for which to search. Include the @ prefix if searching by display text.

Type string (p. 1438)

Name The entity name for which to search. Include the @ prefix if searching by display text.

Type string (p. 1438)

TryGetPolyflowFixedMold

Obtains a PolyflowFixedMold instance by name from the supplied container, falling back to an alternate name if the first is not found.

Return The PolyflowFixedMold instance.

Type DataReference (p. 1371)

Required Arguments

AlternateName The entity name for which to search. Include the @ prefix if searching by display text.

Type string (p. 1438)

Name The entity name for which to search. Include the @ prefix if searching by display text.

Type string (p. 1438)

TryGetPolyflowFluidSolidInterfaceBehavior

Obtains a PolyflowFluidSolidInterfaceBehavior instance by name from the supplied container, falling back to an alternate name if the first is not found.

Return The PolyflowFluidSolidInterfaceBehavior instance.

Type DataReference (p. 1371)

Required Arguments

AlternateName The entity name for which to search. Include the @ prefix if searching by display text.

Type string (p. 1438)

Name The entity name for which to search. Include the @ prefix if searching by display text.

Type string (p. 1438)

TryGetPolyflowForceDrivenMold

Obtains a PolyflowForceDrivenMold instance by name from the supplied container, falling back to an alternate name if the first is not found.

Return The PolyflowForceDrivenMold instance.

Type DataReference (p. 1371)

Required Arguments

AlternateName The entity name for which to search. Include the @ prefix if searching by display text.

Type string (p. 1438)

Name The entity name for which to search. Include the @ prefix if searching by display text.

Type string (p. 1438)

TryGetPolyflowFreejetExit

Obtains a PolyflowFreejetExit instance by name from the supplied container, falling back to an alternate name if the first is not found.

Return The PolyflowFreejetExit instance.

Type DataReference (p. 1371)

Required Arguments

AlternateName The entity name for which to search. Include the @ prefix if searching by display text.

Type string (p. 1438)

Name The entity name for which to search. Include the @ prefix if searching by display text.

Type string (p. 1438)

TryGetPolyflowFreeSurface

Obtains a PolyflowFreeSurface instance by name from the supplied container, falling back to an alternate name if the first is not found.

Return The PolyflowFreeSurface instance.

Type DataReference (p. 1371)

Required Arguments

AlternateName The entity name for which to search. Include the @ prefix if searching by display text.

Type string (p. 1438)

Name The entity name for which to search. Include the @ prefix if searching by display text.

Type string (p. 1438)

TryGetPolyflowHeatFlux

Obtains a PolyflowHeatFlux instance by name from the supplied container, falling back to an alternate name if the first is not found.

Return The PolyflowHeatFlux instance.

Type DataReference (p. 1371)

Required Arguments

AlternateName The entity name for which to search. Include the @ prefix if searching by display text.

Type string (p. 1438)

Name The entity name for which to search. Include the @ prefix if searching by display text.

Type string (p. 1438)

TryGetPolyflowInflow

Obtains a PolyflowInflow instance by name from the supplied container, falling back to an alternate name if the first is not found.

Return The PolyflowInflow instance.

Type DataReference (p. 1371)

Required Arguments

AlternateName The entity name for which to search. Include the @ prefix if searching by display text.

Type string (p. 1438)

Name The entity name for which to search. Include the @ prefix if searching by display text.

Type string (p. 1438)

TryGetPolyflowInitialLayerThickness

Obtains a PolyflowInitialLayerThickness instance by name from the supplied container, falling back to an alternate name if the first is not found.

Return The PolyflowInitialLayerThickness instance.

Type DataReference (p. 1371)

Required Arguments

AlternateName The entity name for which to search. Include the @ prefix if searching by display text.

Type string (p. 1438)

Name The entity name for which to search. Include the @ prefix if searching by display text.

Type string (p. 1438)

TryGetPolyflowInitialTemperature

Obtains a PolyflowInitialTemperature instance by name from the supplied container, falling back to an alternate name if the first is not found.

Return The PolyflowInitialTemperature instance.

Type DataReference (p. 1371)

Required Arguments

AlternateName The entity name for which to search. Include the @ prefix if searching by display text.

Type string (p. 1438)

Name The entity name for which to search. Include the @ prefix if searching by display text.

Type string (p. 1438)

TryGetPolyflowInsulated

Obtains a PolyflowInsulated instance by name from the supplied container, falling back to an alternate name if the first is not found.

Return The PolyflowInsulated instance.

Type DataReference (p. 1371)

Required Arguments

AlternateName The entity name for which to search. Include the @ prefix if searching by display text.

Type string (p. 1438)

Name The entity name for which to search. Include the @ prefix if searching by display text.

Type string (p. 1438)

TryGetPolyflowLaunchControls

Obtains a PolyflowLaunchControls instance by name from the supplied container, falling back to an alternate name if the first is not found.

Return The PolyflowLaunchControls instance.

Type DataReference (p. 1371)

Required Arguments

AlternateName The entity name for which to search. Include the @ prefix if searching by display text.

Type string (p. 1438)

Name The entity name for which to search. Include the @ prefix if searching by display text.

Type string (p. 1438)

TryGetPolyflowMold

Obtains a PolyflowMold instance by name from the supplied container, falling back to an alternate name if the first is not found.

Return The PolyflowMold instance.

Type DataReference (p. 1371)

Required Arguments

AlternateName The entity name for which to search. Include the @ prefix if searching by display text.

Type string (p. 1438)

Name The entity name for which to search. Include the @ prefix if searching by display text.

Type string (p. 1438)

TryGetPolyflowNumericalControls

Obtains a PolyflowNumericalControls instance by name from the supplied container, falling back to an alternate name if the first is not found.

Return The PolyflowNumericalControls instance.

Type DataReference (p. 1371)

Required Arguments

AlternateName The entity name for which to search. Include the @ prefix if searching by display text.

Type string (p. 1438)

Name The entity name for which to search. Include the @ prefix if searching by display text.

Type string (p. 1438)

TryGetPolyflowOutflow

Obtains a PolyflowOutflow instance by name from the supplied container, falling back to an alternate name if the first is not found.

Return The PolyflowOutflow instance.

Type DataReference (p. 1371)

Required Arguments

AlternateName The entity name for which to search. Include the @ prefix if searching by display text.

Type string (p. 1438)

Name The entity name for which to search. Include the @ prefix if searching by display text.

Type string (p. 1438)

TryGetPolyflowOutputControls

Obtains a PolyflowOutputControls instance by name from the supplied container, falling back to an alternate name if the first is not found.

Return The PolyflowOutputControls instance.

Type DataReference (p. 1371)

Required Arguments

AlternateName The entity name for which to search. Include the @ prefix if searching by display text.

Type string (p. 1438)

Name The entity name for which to search. Include the @ prefix if searching by display text.

Type string (p. 1438)

TryGetPolyflowPhysicsOptions

Obtains a PolyflowPhysicsOptions instance by name from the supplied container, falling back to an alternate name if the first is not found.

Return The PolyflowPhysicsOptions instance.

Type DataReference (p. 1371)

Required Arguments

AlternateName The entity name for which to search. Include the @ prefix if searching by display text.

Type string (p. 1438)

Name The entity name for which to search. Include the @ prefix if searching by display text.

Type string (p. 1438)

TryGetPolyflowPressure

Obtains a PolyflowPressure instance by name from the supplied container, falling back to an alternate name if the first is not found.

Return The PolyflowPressure instance.

Type DataReference (p. 1371)

Required Arguments

AlternateName The entity name for which to search. Include the @ prefix if searching by display text.

Type string (p. 1438)

Name The entity name for which to search. Include the @ prefix if searching by display text.

Type string (p. 1438)

TryGetPolyflowRegionInterface

Obtains a PolyflowRegionInterface instance by name from the supplied container, falling back to an alternate name if the first is not found.

Return The PolyflowRegionInterface instance.

Type DataReference (p. 1371)

Required Arguments

AlternateName The entity name for which to search. Include the @ prefix if searching by display text.

Type string (p. 1438)

Name The entity name for which to search. Include the @ prefix if searching by display text.

Type string (p. 1438)

TryGetPolyflowRemeshing

Obtains a PolyflowRemeshing instance by name from the supplied container, falling back to an alternate name if the first is not found.

Return The PolyflowRemeshing instance.

Type DataReference (p. 1371)

Required Arguments

AlternateName The entity name for which to search. Include the @ prefix if searching by display text.

Type string (p. 1438)

Name The entity name for which to search. Include the @ prefix if searching by display text.

Type string (p. 1438)

TryGetPolyflowRestrictor

Obtains a PolyflowRestrictor instance by name from the supplied container, falling back to an alternate name if the first is not found.

Return The PolyflowRestrictor instance.

Type DataReference (p. 1371)

Required Arguments

AlternateName The entity name for which to search. Include the @ prefix if searching by display text.

Type string (p. 1438)

Name The entity name for which to search. Include the @ prefix if searching by display text.

Type string (p. 1438)

TryGetPolyflowSolutionControls

Obtains a PolyflowSolutionControls instance by name from the supplied container, falling back to an alternate name if the first is not found.

Return The PolyflowSolutionControls instance.

Type DataReference (p. 1371)

Required Arguments

AlternateName The entity name for which to search. Include the @ prefix if searching by display text.

Type string (p. 1438)

Name The entity name for which to search. Include the @ prefix if searching by display text.

Type string (p. 1438)

TryGetPolyflowSupport

Obtains a PolyflowSupport instance by name from the supplied container, falling back to an alternate name if the first is not found.

Return The PolyflowSupport instance.

Type DataReference (p. 1371)

Required Arguments

AlternateName The entity name for which to search. Include the @ prefix if searching by display text.

Type string (p. 1438)

Name The entity name for which to search. Include the @ prefix if searching by display text.

Type string (p. 1438)

TryGetPolyflowSymmetry

Obtains a PolyflowSymmetry instance by name from the supplied container, falling back to an alternate name if the first is not found.

Return The PolyflowSymmetry instance.

Type DataReference (p. 1371)

Required Arguments

AlternateName The entity name for which to search. Include the @ prefix if searching by display text.

Type string (p. 1438)

Name The entity name for which to search. Include the @ prefix if searching by display text.

Type string (p. 1438)

TryGetPolyflowTemperatureImposed

Obtains a PolyflowTemperatureImposed instance by name from the supplied container, falling back to an alternate name if the first is not found.

Return The PolyflowTemperatureImposed instance.

Type DataReference (p. 1371)

Required Arguments

AlternateName The entity name for which to search. Include the @ prefix if searching by display text.

Type string (p. 1438)

Name The entity name for which to search. Include the @ prefix if searching by display text.

Type string (p. 1438)

TryGetPolyflowVelocityDrivenMold

Obtains a PolyflowVelocityDrivenMold instance by name from the supplied container, falling back to an alternate name if the first is not found.

Return The PolyflowVelocityDrivenMold instance.

Type DataReference (p. 1371)

Required Arguments

AlternateName The entity name for which to search. Include the @ prefix if searching by display text.

Type string (p. 1438)

Name The entity name for which to search. Include the @ prefix if searching by display text.

Type string (p. 1438)

TryGetPolyflowWall

Obtains a PolyflowWall instance by name from the supplied container, falling back to an alternate name if the first is not found.

Return The PolyflowWall instance.

Type DataReference (p. 1371)

Required Arguments

AlternateName The entity name for which to search. Include the @ prefix if searching by display text.

Type string (p. 1438)

Name The entity name for which to search. Include the @ prefix if searching by display text.

Type string (p. 1438)

TryGetPolyflowWallBase

Obtains a PolyflowWallBase instance by name from the supplied container, falling back to an alternate name if the first is not found.

Return The PolyflowWallBase instance.

Type DataReference (p. 1371)

Required Arguments

AlternateName The entity name for which to search. Include the @ prefix if searching by display text.

Type string (p. 1438)

Name The entity name for which to search. Include the @ prefix if searching by display text.

Type string (p. 1438)

TryGetPorositySrc

Obtains a PorositySrc instance by name from the supplied container, falling back to an alternate name if the first is not found.

Return The PorositySrc instance.

Type DataReference (p. 1371)

Required Arguments

AlternateName The entity name for which to search. Include the @ prefix if searching by display text.

Type string (p. 1438)

Name The entity name for which to search. Include the @ prefix if searching by display text.

Type string (p. 1438)

TryGetPressure

Obtains a Pressure instance by name from the supplied container, falling back to an alternate name if the first is not found.

Return The Pressure instance.

Type DataReference (p. 1371)

Required Arguments

AlternateName The entity name for which to search. Include the @ prefix if searching by display text.

Type string (p. 1438)

Name The entity name for which to search. Include the @ prefix if searching by display text.

Type string (p. 1438)

TryGetPressureDrop

Obtains a PressureDrop instance by name from the supplied container, falling back to an alternate name if the first is not found.

Return The PressureDrop instance.

Type DataReference (p. 1371)

Required Arguments

AlternateName The entity name for which to search. Include the @ prefix if searching by display text.

Type string (p. 1438)

Name The entity name for which to search. Include the @ prefix if searching by display text.

Type string (p. 1438)

TryGetPrimitiveBox

Obtains a PrimitiveBox instance by name from the supplied container, falling back to an alternate name if the first is not found.

Return The PrimitiveBox instance.

Type DataReference (p. 1371)

Required Arguments

AlternateName The entity name for which to search. Include the @ prefix if searching by display text.

Type string (p. 1438)

Name The entity name for which to search. Include the @ prefix if searching by display text.

Type string (p. 1438)

TryGetPsdAcceleration

Obtains a PsdAcceleration instance by name from the supplied container, falling back to an alternate name if the first is not found.

Return The PsdAcceleration instance.

Type DataReference (p. 1371)

Required Arguments

AlternateName The entity name for which to search. Include the @ prefix if searching by display text.

Type string (p. 1438)

Name The entity name for which to search. Include the @ prefix if searching by display text.

Type string (p. 1438)

TryGetPsdDisplacement

Obtains a PsdDisplacement instance by name from the supplied container, falling back to an alternate name if the first is not found.

Return The PsdDisplacement instance.

Type DataReference (p. 1371)

Required Arguments

AlternateName The entity name for which to search. Include the @ prefix if searching by display text.

Type string (p. 1438)

Name The entity name for which to search. Include the @ prefix if searching by display text.

Type string (p. 1438)

TryGetPsdGAcceleration

Obtains a PsdGAcceleration instance by name from the supplied container, falling back to an alternate name if the first is not found.

Return The PsdGAcceleration instance.

Type DataReference (p. 1371)

Required Arguments

AlternateName The entity name for which to search. Include the @ prefix if searching by display text.

Type string (p. 1438)

Name The entity name for which to search. Include the @ prefix if searching by display text.

Type string (p. 1438)

TryGetPsdLoad

Obtains a PsdLoad instance by name from the supplied container, falling back to an alternate name if the first is not found.

Return The PsdLoad instance.

Type DataReference (p. 1371)

Required Arguments

AlternateName The entity name for which to search. Include the @ prefix if searching by display text.

Type string (p. 1438)

Name The entity name for which to search. Include the @ prefix if searching by display text.

Type string (p. 1438)

TryGetPsdVelocity

Obtains a PsdVelocity instance by name from the supplied container, falling back to an alternate name if the first is not found.

Return The PsdVelocity instance.

Type DataReference (p. 1371)

Required Arguments

AlternateName The entity name for which to search. Include the @ prefix if searching by display text.

Type string (p. 1438)

Name The entity name for which to search. Include the @ prefix if searching by display text.

Type string (p. 1438)

TryGetPullOutDirection

Obtains a PullOutDirection instance by name from the supplied container, falling back to an alternate name if the first is not found.

Return The PullOutDirection instance.

Type DataReference (p. 1371)

Required Arguments

AlternateName The entity name for which to search. Include the @ prefix if searching by display text.

Type string (p. 1438)

Name The entity name for which to search. Include the @ prefix if searching by display text.

Type string (p. 1438)

TryGetRadiation

Obtains a Radiation instance by name from the supplied container, falling back to an alternate name if the first is not found.

Return The Radiation instance.

Type DataReference (p. 1371)

Required Arguments

AlternateName The entity name for which to search. Include the @ prefix if searching by display text.

Type string (p. 1438)

Name The entity name for which to search. Include the @ prefix if searching by display text.

Type string (p. 1438)

TryGetReferenceFrame

Obtains a ReferenceFrame instance by name from the supplied container, falling back to an alternate name if the first is not found.

Return The ReferenceFrame instance.

Type DataReference (p. 1371)

Required Arguments

AlternateName The entity name for which to search. Include the @ prefix if searching by display text.

Type string (p. 1438)

Name The entity name for which to search. Include the @ prefix if searching by display text.

Type string (p. 1438)

TryGetRegionInterface

Obtains a RegionInterface instance by name from the supplied container, falling back to an alternate name if the first is not found.

Return The RegionInterface instance.

Type DataReference (p. 1371)

Required Arguments

AlternateName The entity name for which to search. Include the @ prefix if searching by display text.

Type string (p. 1438)

Name The entity name for which to search. Include the @ prefix if searching by display text.

Type string (p. 1438)

TryGetRegionInterfaceGenerator

Obtains a RegionInterfaceGenerator instance by name from the supplied container, falling back to an alternate name if the first is not found.

Return The RegionInterfaceGenerator instance.

Type DataReference (p. 1371)

Required Arguments

AlternateName The entity name for which to search. Include the @ prefix if searching by display text.

Type string (p. 1438)

Name The entity name for which to search. Include the @ prefix if searching by display text.

Type string (p. 1438)

TryGetResponsePsdResult

Obtains a ResponsePsdResult instance by name from the supplied container, falling back to an alternate name if the first is not found.

Return The ResponsePsdResult instance.

Type DataReference (p. 1371)

Required Arguments

AlternateName The entity name for which to search. Include the @ prefix if searching by display text.

Type string (p. 1438)

Name The entity name for which to search. Include the @ prefix if searching by display text.

Type string (p. 1438)

TryGetResultObject

Obtains a ResultObject instance by name from the supplied container, falling back to an alternate name if the first is not found.

Return The ResultObject instance.

Type DataReference (p. 1371)

Required Arguments

AlternateName The entity name for which to search. Include the @ prefix if searching by display text.

Type string (p. 1438)

Name The entity name for which to search. Include the @ prefix if searching by display text.

Type string (p. 1438)

TryGetResults

Obtains a Results instance by name from the supplied container, falling back to an alternate name if the first is not found.

Return The Results instance.

Type DataReference (p. 1371)

Required Arguments

AlternateName The entity name for which to search. Include the @ prefix if searching by display text.

Type string (p. 1438)

Name The entity name for which to search. Include the @ prefix if searching by display text.

Type string (p. 1438)

TryGetScalarConstraint

Obtains a ScalarConstraint instance by name from the supplied container, falling back to an alternate name if the first is not found.

Return The ScalarConstraint instance.

Type DataReference (p. 1371)

Required Arguments

AlternateName The entity name for which to search. Include the @ prefix if searching by display text.

Type string (p. 1438)

Name The entity name for which to search. Include the @ prefix if searching by display text.

Type string (p. 1438)

TryGetScalarInitialCondition

Obtains a ScalarInitialCondition instance by name from the supplied container, falling back to an alternate name if the first is not found.

Return The ScalarInitialCondition instance.

Type DataReference (p. 1371)

Required Arguments

AlternateName The entity name for which to search. Include the @ prefix if searching by display text.

Type string (p. 1438)

Name The entity name for which to search. Include the @ prefix if searching by display text.

Type string (p. 1438)

TryGetScalarLoad

Obtains a ScalarLoad instance by name from the supplied container, falling back to an alternate name if the first is not found.

Return The ScalarLoad instance.

Type DataReference (p. 1371)

Required Arguments

AlternateName The entity name for which to search. Include the @ prefix if searching by display text.

Type string (p. 1438)

Name The entity name for which to search. Include the @ prefix if searching by display text.

Type string (p. 1438)

TryGetSectionLayer

Obtains a SectionLayer instance by name from the supplied container, falling back to an alternate name if the first is not found.

Return The SectionLayer instance.

Type DataReference (p. 1371)

Required Arguments

AlternateName The entity name for which to search. Include the @ prefix if searching by display text.

Type string (p. 1438)

Name The entity name for which to search. Include the @ prefix if searching by display text.

Type string (p. 1438)

TryGetSelectionSet

Obtains a SelectionSet instance by name from the supplied container, falling back to an alternate name if the first is not found.

Return The SelectionSet instance.

Type DataReference (p. 1371)

Required Arguments

AlternateName The entity name for which to search. Include the @ prefix if searching by display text.

Type string (p. 1438)

Name The entity name for which to search. Include the @ prefix if searching by display text.

Type string (p. 1438)

TryGetShape

Obtains a Shape instance by name from the supplied container, falling back to an alternate name if the first is not found.

Return The Shape instance.

Type DataReference (p. 1371)

Required Arguments

AlternateName The entity name for which to search. Include the @ prefix if searching by display text.

Type string (p. 1438)

Name The entity name for which to search. Include the @ prefix if searching by display text.

Type string (p. 1438)

TryGetShapeTaskStatistics

Obtains a ShapeTaskStatistics instance by name from the supplied container, falling back to an alternate name if the first is not found.

Return The ShapeTaskStatistics instance.

Type DataReference (p. 1371)

Required Arguments

AlternateName The entity name for which to search. Include the @ prefix if searching by display text.

Type string (p. 1438)

Name The entity name for which to search. Include the @ prefix if searching by display text.

Type string (p. 1438)

TryGetShearMomentDiagram

Obtains a ShearMomentDiagram instance by name from the supplied container, falling back to an alternate name if the first is not found.

Return The ShearMomentDiagram instance.

Type DataReference (p. 1371)

Required Arguments

AlternateName The entity name for which to search. Include the @ prefix if searching by display text.

Type string (p. 1438)

Name The entity name for which to search. Include the @ prefix if searching by display text.

Type string (p. 1438)

TryGetShellThickness

Obtains a ShellThickness instance by name from the supplied container, falling back to an alternate name if the first is not found.

Return The ShellThickness instance.

Type DataReference (p. 1371)

Required Arguments

AlternateName The entity name for which to search. Include the @ prefix if searching by display text.

Type string (p. 1438)

Name The entity name for which to search. Include the @ prefix if searching by display text.

Type string (p. 1438)

TryGetSimulation

Obtains a Simulation instance by name from the supplied container, falling back to an alternate name if the first is not found.

Return The Simulation instance.

Type DataReference (p. 1371)

Required Arguments

AlternateName The entity name for which to search. Include the @ prefix if searching by display text.

Type string (p. 1438)

Name The entity name for which to search. Include the @ prefix if searching by display text.

Type string (p. 1438)

TryGetSingleValueResult

Obtains a SingleValueResult instance by name from the supplied container, falling back to an alternate name if the first is not found.

Return The SingleValueResult instance.

Type DataReference (p. 1371)

Required Arguments

AlternateName The entity name for which to search. Include the @ prefix if searching by display text.

Type string (p. 1438)

Name The entity name for which to search. Include the @ prefix if searching by display text.

Type string (p. 1438)

TryGetSolutionProgression

Obtains a SolutionProgression instance by name from the supplied container, falling back to an alternate name if the first is not found.

Return The SolutionProgression instance.

Type DataReference (p. 1371)

Required Arguments

AlternateName The entity name for which to search. Include the @ prefix if searching by display text.

Type string (p. 1438)

Name The entity name for which to search. Include the @ prefix if searching by display text.

Type string (p. 1438)

TryGetSolutionStep

Obtains a SolutionStep instance by name from the supplied container, falling back to an alternate name if the first is not found.

Return The SolutionStep instance.

Type DataReference (p. 1371)

Required Arguments

AlternateName The entity name for which to search. Include the @ prefix if searching by display text.

Type string (p. 1438)

Name The entity name for which to search. Include the @ prefix if searching by display text.

Type string (p. 1438)

TryGetSolvePhysics

Obtains a SolvePhysics instance by name from the supplied container, falling back to an alternate name if the first is not found.

Return The SolvePhysics instance.

Type DataReference (p. 1371)

Required Arguments

AlternateName The entity name for which to search. Include the @ prefix if searching by display text.

Type string (p. 1438)

Name The entity name for which to search. Include the @ prefix if searching by display text.

Type string (p. 1438)

TryGetSolverSettings

Obtains a SolverSettings instance by name from the supplied container, falling back to an alternate name if the first is not found.

Return The SolverSettings instance.

Type DataReference (p. 1371)

Required Arguments

AlternateName The entity name for which to search. Include the @ prefix if searching by display text.

Type string (p. 1438)

Name The entity name for which to search. Include the @ prefix if searching by display text.

Type string (p. 1438)

TryGetSpotWeld

Obtains a SpotWeld instance by name from the supplied container, falling back to an alternate name if the first is not found.

Return The SpotWeld instance.

Type DataReference (p. 1371)

Required Arguments

AlternateName The entity name for which to search. Include the @ prefix if searching by display text.

Type string (p. 1438)

Name The entity name for which to search. Include the @ prefix if searching by display text.

Type string (p. 1438)

TryGetSpring

Obtains a Spring instance by name from the supplied container, falling back to an alternate name if the first is not found.

Return The Spring instance.

Type DataReference (p. 1371)

Required Arguments

AlternateName The entity name for which to search. Include the @ prefix if searching by display text.

Type string (p. 1438)

Name The entity name for which to search. Include the @ prefix if searching by display text.

Type string (p. 1438)

TryGetSpringBehavior

Obtains a SpringBehavior instance by name from the supplied container, falling back to an alternate name if the first is not found.

Return The SpringBehavior instance.

Type DataReference (p. 1371)

Required Arguments

AlternateName The entity name for which to search. Include the @ prefix if searching by display text.

Type string (p. 1438)

Name The entity name for which to search. Include the @ prefix if searching by display text.

Type string (p. 1438)

TryGetStatusGroup

Obtains a StatusGroup instance by name from the supplied container, falling back to an alternate name if the first is not found.

Return The StatusGroup instance.

Type DataReference (p. 1371)

Required Arguments

AlternateName The entity name for which to search. Include the @ prefix if searching by display text.

Type string (p. 1438)

Name The entity name for which to search. Include the @ prefix if searching by display text.

Type string (p. 1438)

TryGetStreamLine

Obtains a StreamLine instance by name from the supplied container, falling back to an alternate name if the first is not found.

Return The StreamLine instance.

Type DataReference (p. 1371)

Required Arguments

AlternateName The entity name for which to search. Include the @ prefix if searching by display text.

Type string (p. 1438)

Name The entity name for which to search. Include the @ prefix if searching by display text.

Type string (p. 1438)

TryGetStressConstraint

Obtains a StressConstraint instance by name from the supplied container, falling back to an alternate name if the first is not found.

Return The StressConstraint instance.

Type DataReference (p. 1371)

Required Arguments

AlternateName The entity name for which to search. Include the @ prefix if searching by display text.

Type string (p. 1438)

Name The entity name for which to search. Include the @ prefix if searching by display text.

Type string (p. 1438)

TryGetStructuralMass

Obtains a StructuralMass instance by name from the supplied container, falling back to an alternate name if the first is not found.

Return The StructuralMass instance.

Type DataReference (p. 1371)

Required Arguments

AlternateName The entity name for which to search. Include the @ prefix if searching by display text.

Type string (p. 1438)

Name The entity name for which to search. Include the @ prefix if searching by display text.

Type string (p. 1438)

TryGetSupport

Obtains a Support instance by name from the supplied container, falling back to an alternate name if the first is not found.

Return The Support instance.

Type DataReference (p. 1371)

Required Arguments

AlternateName The entity name for which to search. Include the @ prefix if searching by display text.

Type string (p. 1438)

Name The entity name for which to search. Include the @ prefix if searching by display text.

Type string (p. 1438)

TryGetSuppressControl

Obtains a SuppressControl instance by name from the supplied container, falling back to an alternate name if the first is not found.

Return The SuppressControl instance.

Type DataReference (p. 1371)

Required Arguments

AlternateName The entity name for which to search. Include the @ prefix if searching by display text.

Type string (p. 1438)

Name The entity name for which to search. Include the @ prefix if searching by display text.

Type string (p. 1438)

TryGetSymmetryConstraint

Obtains a SymmetryConstraint instance by name from the supplied container, falling back to an alternate name if the first is not found.

Return The SymmetryConstraint instance.

Type DataReference (p. 1371)

Required Arguments

AlternateName The entity name for which to search. Include the @ prefix if searching by display text.

Type string (p. 1438)

Name The entity name for which to search. Include the @ prefix if searching by display text.

Type string (p. 1438)

TryGetTableResult

Obtains a TableResult instance by name from the supplied container, falling back to an alternate name if the first is not found.

Return The TableResult instance.

Type DataReference (p. 1371)

Required Arguments

AlternateName The entity name for which to search. Include the @ prefix if searching by display text.

Type string (p. 1438)

Name The entity name for which to search. Include the @ prefix if searching by display text.

Type string (p. 1438)

TryGetTemperatureCondition

Obtains a TemperatureCondition instance by name from the supplied container, falling back to an alternate name if the first is not found.

Return The TemperatureCondition instance.

Type DataReference (p. 1371)

Required Arguments

AlternateName The entity name for which to search. Include the @ prefix if searching by display text.

Type string (p. 1438)

Name The entity name for which to search. Include the @ prefix if searching by display text.

Type string (p. 1438)

TryGetTemperatureConstraint

Obtains a TemperatureConstraint instance by name from the supplied container, falling back to an alternate name if the first is not found.

Return The TemperatureConstraint instance.

Type DataReference (p. 1371)

Required Arguments

AlternateName The entity name for which to search. Include the @ prefix if searching by display text.

Type string (p. 1438)

Name The entity name for which to search. Include the @ prefix if searching by display text.

Type string (p. 1438)

TryGetThermalMass

Obtains a ThermalMass instance by name from the supplied container, falling back to an alternate name if the first is not found.

Return The ThermalMass instance.

Type DataReference (p. 1371)

Required Arguments

AlternateName The entity name for which to search. Include the @ prefix if searching by display text.

Type string (p. 1438)

Name The entity name for which to search. Include the @ prefix if searching by display text.

Type string (p. 1438)

TryGetTranscript

Obtains a Transcript instance by name from the supplied container, falling back to an alternate name if the first is not found.

Return The Transcript instance.

Type DataReference (p. 1371)

Required Arguments

AlternateName The entity name for which to search. Include the @ prefix if searching by display text.

Type string (p. 1438)

Name The entity name for which to search. Include the @ prefix if searching by display text.

Type string (p. 1438)

TryGetUserCommands

Obtains a UserCommands instance by name from the supplied container, falling back to an alternate name if the first is not found.

Return The UserCommands instance.

Type DataReference (p. 1371)

Required Arguments

AlternateName The entity name for which to search. Include the @ prefix if searching by display text.

Type string (p. 1438)

Name The entity name for which to search. Include the @ prefix if searching by display text.

Type string (p. 1438)

TryGetVectorConstraint

Obtains a VectorConstraint instance by name from the supplied container, falling back to an alternate name if the first is not found.

Return The VectorConstraint instance.

Type DataReference (p. 1371)

Required Arguments

AlternateName The entity name for which to search. Include the @ prefix if searching by display text.

Type string (p. 1438)

Name The entity name for which to search. Include the @ prefix if searching by display text.

Type string (p. 1438)

TryGetVectorInitialCondition

Obtains a VectorInitialCondition instance by name from the supplied container, falling back to an alternate name if the first is not found.

Return The VectorInitialCondition instance.

Type DataReference (p. 1371)

Required Arguments

AlternateName The entity name for which to search. Include the @ prefix if searching by display text.

Type string (p. 1438)

Name The entity name for which to search. Include the @ prefix if searching by display text.

Type string (p. 1438)

TryGetVectorLoad

Obtains a VectorLoad instance by name from the supplied container, falling back to an alternate name if the first is not found.

Return The VectorLoad instance.

Type DataReference (p. 1371)

Required Arguments

AlternateName The entity name for which to search. Include the @ prefix if searching by display text.

Type string (p. 1438)

Name The entity name for which to search. Include the @ prefix if searching by display text.

Type string (p. 1438)

TryGetVectorResult

Obtains a VectorResult instance by name from the supplied container, falling back to an alternate name if the first is not found.

Return The VectorResult instance.

Type DataReference (p. 1371)

Required Arguments

AlternateName The entity name for which to search. Include the @ prefix if searching by display text.

Type string (p. 1438)

Name The entity name for which to search. Include the @ prefix if searching by display text.

Type string (p. 1438)

TryGetVoltage

Obtains a Voltage instance by name from the supplied container, falling back to an alternate name if the first is not found.

Return The Voltage instance.

Type DataReference (p. 1371)

Required Arguments

AlternateName The entity name for which to search. Include the @ prefix if searching by display text.

Type string (p. 1438)

Name The entity name for which to search. Include the @ prefix if searching by display text.

Type string (p. 1438)

TryGetVolumeExtractionRegion

Obtains a VolumeExtractionRegion instance by name from the supplied container, falling back to an alternate name if the first is not found.

Return The VolumeExtractionRegion instance.

Type DataReference (p. 1371)

Required Arguments

AlternateName The entity name for which to search. Include the @ prefix if searching by display text.

Type string (p. 1438)

Name The entity name for which to search. Include the @ prefix if searching by display text.

Type string (p. 1438)

TryGetVolumeSimplificationRegion

Obtains a VolumeSimplificationRegion instance by name from the supplied container, falling back to an alternate name if the first is not found.

Return The VolumeSimplificationRegion instance.

Type DataReference (p. 1371)

Required Arguments

AlternateName The entity name for which to search. Include the @ prefix if searching by display text.

Type string (p. 1438)

Name The entity name for which to search. Include the @ prefix if searching by display text.

Type string (p. 1438)

TryGetWrap

Obtains a Wrap instance by name from the supplied container, falling back to an alternate name if the first is not found.

Return The Wrap instance.

Type DataReference (p. 1371)

Required Arguments

AlternateName The entity name for which to search. Include the @ prefix if searching by display text.

Type string (p. 1438)

Name The entity name for which to search. Include the @ prefix if searching by display text.

Type string (p. 1438)

Common Data Entities

AlternatingStress

The Alternating Stress material model.

Properties

Interpolation

The Interpolation for the AlternatingStress charting can be chosen between: Linear, Semi-Log and Log-Log.

Type InterpolationType (p. 1396)

Read Only No

Magnitude

The Magnitude is an expresion which evaluates to a Quantity with the correct units.

Type Expression (p. 1384)<Quantity (p. 1422)>

Read Only No

Appearance

The Appearance Properties material model.

Properties

ColorComponentBlue

Material Color Blue Value

Type double (p. 1378)

Read Only No

ColorComponentGreen

Material Color Green Value

Type double (p. 1378)

Read Only No

ColorComponentRed

Material Color Red Value

Type double (p. 1378)

Read Only No

ColorSpace

Material Color Space Value

Type ColorSpaceSetting (p. 1365)

Read Only No

MaterialMetallic

Material Metallic Value

Type double (p. 1378)

Read Only No

MaterialOpacity

Material Color Alpha Value

Type double (p. 1378)

Read Only No

MaterialRoughness

Material Roughness Value

Type double (p. 1378)

Read Only No

MaterialTransmittance

Material Density - used to define how much contribution diffuse color has on the output.

Type double (p. 1378)

Read Only No

ArbitrarySection

This class represents arbitrary beam cross section with following properties

Properties

Area

Area of the section

Type Quantity (p. 1422)

Read Only No

AreaMomentOfInertiaAboutYAxis

Moment of inertia about the y axis

Type Quantity (p. 1422)

Read Only No

AreaMomentOfInertiaAboutZAxis

Moment of inertia about the z axis

Type Quantity (p. 1422)

Read Only No

CentroidAbscissa

y coordinate of the centroid

Type Quantity (p. 1422)

Read Only No

CentroidOrdinate

z coordinate of the centroid

Type Quantity (p. 1422)

Read Only No

ProductOfInertiaAboutYZAxes

Product of inertia

Type Quantity (p. 1422)

Read Only No

ShearCenterAbscissa

y coordinate of the shear center

Type Quantity (p. 1422)

Read Only No

ShearCenterOrdinate

z coordinate of the shear center

Type Quantity (p. 1422)

Read Only No

Torsion

Torsion constant

Type Quantity (p. 1422)

Read Only No

Warping

Warping constant

Type Quantity (p. 1422)

Read Only No

AxisDirectionDefinition

Defines an axis from a vector or point.

Properties

Automatic

Property to automatically compute the axis direction.

Type bool (p. 1360)

Read Only No

Axis

Axis being defined.

Type AxisType (p. 1357)

Read Only No

DefineBy

Option to define the axis.

Type AxisDefineBy (p. 1357)

Read Only No

BeamSection

BeamSection is a class that holds beam section type, offset type and cross section properties.

Properties

Location

Defined locations for the object.

Type LocationSet (p. 1401)

Read Only No

OffsetType

Offset of the beam cross section, also the location of the nodes in the section

Type BeamSectionOffsetType (p. 1357)

Read Only No

SectionType

Shape of the beam section

Type BeamSectionType (p. 1357)

Read Only No

XOffset

X offset of the beam cross section

Type Expression (p. 1384)<Quantity (p. 1422)>

Read Only No

YOffset

Y offset of the beam cross section

Type Expression (p. 1384)<Quantity (p. 1422)>

Read Only No

BHCurve

The B-H Curve model.

Can be applied to Magnetic analyses

Properties

Value

The Magnitude allows for tabular input through the expresson language.

Type Expression (p. 1384)<Quantity (p. 1422)>

Read Only No

BilinearIsotropicHardening

Implementation for IProvideChartData

Properties

Value

The Magnitude.

Type Expression (p. 1384)<BilinearIsotropicHardeningDependents (p. 1358)>

Read Only No

BilinearIsotropicHardeningDependents

The Bilinear Isotropic Hardening dependents.

Properties

DisplayText

The general property that defines the user-visible name of an entity. This property is defined for all data entities but is used only in those entities that present a label in the user interface.

Type string (p. 1438)

Read Only No

TangentModulus

Gets or sets the tangent modulus.

Type Expression (p. 1384)<Quantity (p. 1422)>

Read Only No

YieldStrength

Gets or sets the yield strength.

Type Expression (p. 1384)<Quantity (p. 1422)>

Read Only No

BilinearKinematicHardeningDependents

The Bilinear Kinematic Hardening dependents.

Properties

DisplayText

The general property that defines the user-visible name of an entity. This property is defined for all data entities but is used only in those entities that present a label in the user interface.

Type string (p. 1438)

Read Only No

TangentModulus

Gets or sets the tangent modulus.

Type Expression (p. 1384)<Quantity (p. 1422)>

Read Only No

YieldStrength

Gets or sets the yield strength.

Type Expression (p. 1384)<Quantity (p. 1422)>

Read Only No

BoundaryLayerRefinement

No details are provided for this entry.

Properties

AspectRatio

Specifies relative thickness of adjacent boundary layers.

Type double (p. 1378)

Read Only No

BoundaryLayerAlgorithm

Specifies pre or post boundary layer algorithm to be used by the mesher.

Available options:

Post
Pre

Type BoundaryLayerAlgorithm (p. 1360)

Read Only No

DefineBy

Specifies the way heights of the boundary layers are determined.

Available options:

SmoothTransition	Mesher ensures smooth rate of volume change using local tetrahedral element size.
TotalThickness	Creates constant boundary layers using the values of Number of Layers and Growth Rate.
FirstLayerThickness	Creates constant boundary layers using values of First Layer Height, Maximum Layers and Growth Rate.
FirstAspectRatio	Creates boundary layers using values of First Aspect Ratio, Maximum Layers and Growth Rate.
LastAspectRatio	Creates constant boundary layers using values of First Layer Height, Maximum Layers and Aspect Ratio.

Type BoundaryLayerOption (p. 1360)

Read Only No

FirstAspectRatio

Specifies the aspect ratio of the boundaries that are extruded from the boundary base.

Type double (p. 1378)

Read Only No

FirstLayerHeight

Specifies the height of the first boundary layer.

Type Quantity (p. 1422)

Read Only No

GrowthRate

Specifies the value for Growth Rate.

Type double (p. 1378)

Read Only No

Location

A common property for all sub meshing objects with a defined Geometry Filter

Type LocationSet (p. 1401)

Read Only No

MaximumLayers

Specifies the maximum number of boundary layers to be created.

Type int (p. 1394)

Read Only No

MaximumThickness

Specifies the desired maximum thickness of the boundary layer.

Type Quantity (p. 1422)

Read Only No

NumberOfLayers

Specifies the number of boundary layers.

Type int (p. 1394)

Read Only No

TransitionRatio

Specifies the rate at which adjacent elements grow.

Type double (p. 1378)

Read Only No

UseAutomaticallyDefinedLocation

A common toggle for all sub meshing objects with a Location property so that it can be automatically set. Currently only visible in Boundary Layer object.

Type bool (p. 1360)

Read Only No

CalculatedValueMonitor

Calculated value monitor objects are used to present graphical charts tracking aspects of the solution results, such as max(displacements.x)@Body1.

Properties

Component

For non-scalar Calculated Value Monitors, defines the component

Type string (p. 1438)

Read Only No

Expression

For Calculated Value Monitors, defines the expression

Type string (p. 1438)

Read Only No

Function

For Calcualted Value Monitors, defines the function

Type string (p. 1438)

Read Only No

Location

For scoped Calculated Value Monitors, defines the location

Type LocationSet (p. 1401)

Read Only No

Method

The method by which the value is calculated.

Type TrackerMethod (p. 1442)

Read Only No

MonitorTypeName

Gives a type of Calculated Value Monitor

Type string (p. 1438)

Read Only No

SubCategory

Gives the name of a set of fields

Type string (p. 1438)

Read Only No

CappingSurface

A CappingSurface creates an analytic surface based on a set of bounding edges or faces.

Properties

CreateSelectionSet

Flag to enable/disable automatic selection set creation.

Type bool (p. 1360)

Read Only No

SelectionMethod

Specifies the method used to create the Capping Surface.

Type CappingSurfaceConstructionMethod (p. 1362)

Read Only No

CappingSurfaceEdgesConstraint

Provides additional information to a Capping Surface when it is constructed based on a set of edges.

Properties

Algorithm

The algorithm used to construct the capping surface.

Type ConstructionAlgorithm (p. 1367)

Read Only No

Edges

The set of selected edges.

Type LocationSet (p. 1401)

Read Only No

FlipLoopOrientation

Sets the orientation of the edge loop, if the algorithm is a Convex Hull.

Type ReverseEdgeLoopOrientation (p. 1426)

Read Only No

CappingSurfaceFacesConstraint

Provides additional information to a Capping Surface when it is constructed based on the internal loops of a set of faces.

Properties

Algorithm

The algorithm used to construct the capping surface.

Type ConstructionAlgorithm (p. 1367)

Read Only No

Faces

The set of selected faces.

Type LocationSet (p. 1401)

Read Only No

FlipLoopOrientation

Sets the orientation of the edge loop, if the algorithm is a Convex Hull.

Type ReverseEdgeLoopOrientation (p. 1426)

Read Only No

CappingSurfaceLoopsConstraint

Provides additional information to a Capping Surface when it is constructed based on a set of edge loops.

Properties

Algorithm

The algorithm used to construct the capping surface.

Type ConstructionAlgorithm (p. 1367)

Read Only No

EdgeLoopCount

Gets or sets edge loop count.

Type int (p. 1394)

Read Only No

EdgeLoops

The set of selected edge loops.

Type List (p. 1400)<LocationSet (p. 1401)>

Read Only No

FlipLoopOrientation

Sets the orientation of the edge loop, if the algorithm is a Convex Hull.

Type ReverseEdgeLoopOrientation (p. 1426)

Read Only No

CappingSurfaceVerticesConstraint

Provides additional information to a Capping Surface when it is constructed based on the internal loops of a set of faces.

Properties

Algorithm

The algorithm used to construct the capping surface.

Type ConstructionAlgorithm (p. 1367)

Read Only No

FlipLoopOrientation

Sets the orientation of the edge loop, if the algorithm is a Convex Hull.

Type ReverseEdgeLoopOrientation (p. 1426)

Read Only No

Vertices

The set of selected Vertices.

Type LocationSet (p. 1401)

Read Only No

CartesianTriplet

Triplet represented in cartesian coordinates

Properties

X

X coordinate

Type Expression (p. 1384)<Quantity (p. 1422)>

Read Only No

Y

Y coordinate

Type Expression (p. 1384)<Quantity (p. 1422)>

Read Only No

Z

Z coordinate

Type Expression (p. 1384)<Quantity (p. 1422)>

Read Only No

ChannelSection

This class represents channel beam cross section

Properties

LowerFlangeThickness

Lower Flange Thickness

Type Quantity (p. 1422)

Read Only No

LowerFlangeWidth

Lower Flange Width

Type Quantity (p. 1422)

Read Only No

OverallHeight

Overall Height

Type Quantity (p. 1422)

Read Only No

StemThickness

Stem Thickness

Type Quantity (p. 1422)

Read Only No

UpperFlangeThickness

Upper Flange Thickness

Type Quantity (p. 1422)

Read Only No

UpperFlangeWidth

Upper Flange Width

Type Quantity (p. 1422)

Read Only No

CircularSolidSection

This class represents circular solid beam cross section

Properties

Radius

Radius

Type Quantity (p. 1422)

Read Only No

CircularTubeSection

This class represents circular tube beam cross section

Properties

InnerRadius

Inner radius of the tube

Type Quantity (p. 1422)

Read Only No

OuterRadius

Outer radius of the tube

Type Quantity (p. 1422)

Read Only No

ConditionGroup

Collection of UserObjects (Loads, Constraints, Contacts) and their selected StatusGroup for each step
Key is SolutionStep DataReference, Value is UserObject DataReference

Properties

ConditionDataReferences

List of DataReferences for UserObjects (Loads, Constraints, Contacts, etc.)

Type List (p. 1400)<DataReference (p. 1371)>

Read Only No

SimulationStepStatusGroup

Dictionary of SolutionStep DataReference and StatusGroup DataReferences

Type Dictionary (p. 1375)<DataReference (p. 1371), DataReference (p. 1371)>

Read Only No

Configuration

The entity that represents a Configuration Task. Holds the list of configure controls objects.

Configure controls:

Suppress control	Allows the suppression selected part components.
Move/rotate control	Allows the transformation selected part components.

Properties

No Properties.

Methods

InsertPrimitiveBox

Inserts a primitive axis-alinged box that can be used to define an enclosure.

Required Arguments

CurrentSelection Location set used to define the box bounds.

Type Object (p. 1409)

ContourResult

Displays the contour of a selected variable on a specified location.

Properties

CalculateAverage

Set to true if averages are to be calculated as part of the Result evaluation.

Type bool (p. 1360)

Read Only No

EvaluateFullRange

Whether or not to evaluate all time points in the evaluation

Type bool (p. 1360)

Read Only No

LastAutomaticDisplayText

The last automatically-generated DisplayText assigned to this Result Object

Type string (p. 1438)

Read Only No

Location

The Location over which the Result is to be evaluated.

Type LocationSet (p. 1401)

Read Only No

RelativeTo

The Reference Frame which the result is relative to

Type ReferenceFrame (p. 1424)

Read Only No

ResultName

Name of the Result.

Type string (p. 1438)

Read Only No

Settings

This allows for the required settings (such as Fatigue Settings) to be added to the contour result.

Type GeneralSettings (p. 1388)

Read Only No

SimulationStep

The Simulation Step object reference for a multi-step analysis.

Type SolutionStep (p. 1434)

Read Only No

Variable

The solution variable to be displayed

Type Expression (p. 1384)<Quantity (p. 1422)>

Read Only No

CylindricalTriplet

Triplet represented in cylindrical coordinates

Properties

Phi

Angle from the +X axis in the XY plane (a.k.a. theta or phi) phi = arctan(y / x) 0 <= theta < 2PI

Type Expression (p. 1384)<Quantity (p. 1422)>

Read Only No

Rho

Distance from the origin in the XY plane (positive) r = sqrt(x^2 + y^2)

Type Expression (p. 1384)<Quantity (p. 1422)>

Read Only No

Z

Distance from the XY plane (z coordinate)

Type Expression (p. 1384)<Quantity (p. 1422)>

Read Only No

DataManager

The DataManager is used to access readers and writers. This allows engineering data to be read and written to and from files.

Properties

DisplayText

The general property that defines the user-visible name of an entity. This property is defined for all data entities but is used only in those entities that present a label in the user interface.

Type string (p. 1438)

Read Only No

ProviderName

Gets or sets the name of the provider.

Type string (p. 1438)

Read Only Yes

Methods

DeleteDataManager

Deletes the specified DataManager.

Example

This example creates and then deletes a DataManager

```
dataManagerThermal = study1.CreateDataManager(
    Sources=[r"C:\ANSYSDev\Program Files\Ansys Inc\v151\Addins\EngineeringData\Samples\Thermal_Materials.x
    Provider="ANSYS")
Study.DeleteDataManager(dataManagerThermal)
```

FindByDescription

Finds items in the Data Manager, where the item's Description contains the supplied text.

Return A list of the names of the items the search found.

Type List (p. 1400)<string (p. 1438)>

Required Arguments

Description A string containing the text to search for.

Type string (p. 1438)

Example

This example create a data manager and searches for a match in the description.

```
dataManagerThermal = study1.CreateDataManager(
```

```
    Sources=[r"C:\ANSYSDev\Program Files\Ansys Inc\v151\Addins\EngineeringData\Samples\Thermal_Materials.x
    Provider="ANSYS")
matList = Study.FindByDescription(dataManagerThermal, "Thermal Properties")
```

FindByModel

Finds materials in the Data Manager, where the material contains a matching model for the supplied arguments.

Return A list of the names of the items the search found.

Type List (p. 1400)<string (p. 1438)>

Required Arguments

Model The string to identify the material property/model to be searched for.

Type string (p. 1438)

Optional Arguments

Behavior The optional string to identify the behavior of the model.

Type string (p. 1438)

Definition The optional string to identify the way in which model is defined.

Type string (p. 1438)

Example

This example creates a data manager and searches for materials that have an Isotropic Elasticity model.

```
dataManagerGeneral = study1.CreateDataManager(
    Sources=[r"C:\ANSYSDev\Program Files\Ansys Inc\v151\Addins\EngineeringData\Samples\General_Materials.x
    Provider="ANSYS")
matList = Study.FindByModel(dataManagerGeneral, "Elasticity", Behavior="Isotropic")
```

FindByName

Finds items in the Data Manager, where the item's name contains the supplied string.

Return A list of the names of the items the search found.

Type List (p. 1400)<string (p. 1438)>

Required Arguments

Name The string to identify the name of the item to be searched for.

Type string (p. 1438)

Example

This example creates a data manager and searches for any material having Alloy in the name.

```
dataManagerGeneral = study1.CreateDataManager(
    Sources=[r"C:\ANSYSDev\Program Files\Ansys Inc\v151\Addins\EngineeringData\Samples\General_Materials.x
    Provider="ANSYS")
matList = Study.FindByName(dataManagerGeneral, "Alloy")
```

FindByRange

Finds items in the Data Manager, where the item has the property and a value within the specified range.

Return A list of the names of the items the search found.

Type List (p. 1400)<string (p. 1438)>

Required Arguments

MaximumQuantity The maximum quantity in the range to search for. i.e. 1 [kPa]

Type string (p. 1438)

MinimumQuantity The minimum quantity in the range to search for. i.e. 10 [Pa]

Type string (p. 1438)

PropertyName The name of the property to use. i.e Bulk Modulus

Type string (p. 1438)

Example

This example creates a data manager and searches for any material having density in the range of 7000 [kg m^-3] to 9000 [kg m^-3].

```
dataManagerGeneral = study1.CreateDataManager(
    Sources=[r"C:\ANSYSDev\Program Files\Ansys Inc\v151\Addins\EngineeringData\Samples\General_Materials.x
    Provider="ANSYS")
matList = Study.FindByRange(dataManagerGeneral, "Density", "7000 [kg m^-3]" ,"9000 [kg m^-3]")
```

GetValidContentNames

Gets the names of all the items in the Data Manager.

Return A List of strings containing the names of the items loaded into the DataManager.

Type List (p. 1400)<string (p. 1438)>

Optional Arguments

DataType The type of data to be returned.

Type string (p. 1438)

Default Value Material

Example

This example creates a data manager and returns the items it is managing.

```
dataManagerGeneral = study1.CreateDataManager(
    Sources=[r"C:\ANSYSDev\Program Files\Ansys Inc\v151\Addins\EngineeringData\Samples\General_Materials.x
    Provider="ANSYS")
contents = Study.GetValidContentNames(dataManagerGeneral)
```

Density

The Density model provided via a specified expression.

Properties

Magnitude

The Magnitude is an expresion which evaluates to a Quantity with the correct units.

Type Expression (p. 1384)<Quantity (p. 1422)>

Read Only No

FidelityRefinement

No details are provided for this entry.

Properties

Behavior

Specifies whether size control settings can be changed by the mesher (Soft) or not (Hard).

Available options:

Soft
Hard

Type SizingBehavior (p. 1433)

Read Only No

CurvatureNormalAngle

Specifies maximum angle that one element edge is allowed to span.

Type Quantity (p. 1422)

Read Only No

Detail

No details are provided for this entry.

Type Quantity (p. 1422)

Read Only No

GrowthRate

Specifies increase in element edge length with each succeeding layer of elements.

Type double (p. 1378)

Read Only No

LocalMinSize

Specifies value to override global Min Size on local entities.

Type Quantity (p. 1422)

Read Only No

Location

A common property for all sub meshing objects with a defined Geometry Filter

Type LocationSet (p. 1401)

Read Only No

NumberOfCells

Specifies value to override global Number of cells across gap on local entities

Type int (p. 1394)

Read Only No

ProximityMinimumSize

Specifies value to override global Proximity Min size on local entities

Type Quantity (p. 1422)

Read Only No

ProximitySizeFunctionSource

Specifies value to override global Number of cells across gap on local entities

Type ProximitySizeFunctionSources (p. 1422)

Read Only No

SizeFunctionMethod

Specifies which size function to use.

Available options:

Program controlled
Adaptive
Curvature and proximity
Curvature
Proximity
Fixed

Type UseAdvancedLocalSizeFunction (p. 1447)

Read Only No

GeometryDimension

Dimension parameter imported from CAD.

Properties

Value

Dimension parameter value.

Type Quantity (p. 1422)

Read Only No

GeometryImportSource

Defines an imported geometry source through direct file browsing or an active CAD attach source. Provides the ability to update the geometry from the source system.

Properties

ActiveCADAttachSource

The Active CAD Source chosen from the option list

Type string (p. 1438)

Read Only No

FilePath

Import source location.

Type string (p. 1438)

Read Only No

LineModelTypes

No details are provided for this entry.

Type LineModelType (p. 1399)

Read Only No

MaterialAssignments

Material assignments brought in by this import.

Type List (p. 1400)<MaterialAssignment (p. 1402)>

Read Only No

PlugInName

Displays the current plugin name when the source has been defined by an active CAD attach.

Type string (p. 1438)

Read Only Yes

SharedEdgesAmongBeams

No details are provided for this entry.

Type bool (p. 1360)

Read Only No

SourceSelectionMode

Controls how the source selection if defined.

Selection Types:

BrowseSource	Source is specified using the file browser dialog.
ActiveCADAttach	Source is specified by an active CAD attach source.

Type SourceSelectionType (p. 1434)

Read Only No

SourceType

Import source type.

Type string (p. 1438)

Read Only Yes

Methods

BrowseImportSourceOperation

Sets an import source file location through a file browser dialog.

Required Arguments

FilePath Path to the chosen file.

Type string (p. 1438)

GenerateImportSourceOperation

Generates the output for a specific Geometry Import Source.

UpdateGeometryImportSource

Updates a specific Geometry Import Source from the source CAD system. Any changed parameter values associated with this import source are pushed to the source CAD system.

UpdateGeometryImportSourceFromCAD

Updates a specific Geometry Import Source from the source CAD system. Parameter values associated with the import source are synchronized with the current values defined in the source CAD system.

GeometryPreferences

Defines geometry import preferences used during the initial geometry import and any subsequent updates.

Properties

CleanGeometryOnImport

No details are provided for this entry.

Type bool (p. 1360)

Read Only No

CoordinateSystemKey

Allows the specification of a key that is used to filter processed CAD System coordinate systema during import. The key must be present at the beginning or the end of a coordinate system's name to be valid for import. The Coordinate System Key supports multiple prefixes/suffixes with each value separated by a semicolon. If the value is empty, all coordinate systems are imported. The key is empty by default.

Type string (p. 1438)

Read Only No

DecomposeDisjointFaces

Enables the decomposition of disjoint faces into separate face entities. The default is No.

Type bool (p. 1360)

Read Only No

DimensionKey

Allows the specification of a key that is used to filter processed CAD System Parameters during import. The key must be present at the beginning or the end of a CAD Parameter's name to be valid for import. The Dimension Key supports multiple prefixes/suffixes with each value separated by a semicolon. If the value is empty, all Dimensions are imported. The default key is "DS".

Type string (p. 1438)

Read Only No

ImportCoordinateSystems

Specifies whether coordinate systems created in the CAD System are imported as Reference Frame objects. The default is No.

Type bool (p. 1360)

Read Only No

ImportDimensions

Enables Dimension processing. Dimension processing can have a negative impact on overall import performance. The default is Independent.

Type ImportParameterType (p. 1392)

Read Only No

ImportNamedSelections

Enables processing of CAD System Named Selections that result in the creation of Selection Sets. The default is No.

Type bool (p. 1360)

Read Only No

ImportSheets

Enables the import of surface bodies. The default is Yes.

Type bool (p. 1360)

Read Only No

ImportSolids

Enables the import of solid bodies. The default is Yes.

Type bool (p. 1360)

Read Only No

ImportWires

Enables the import of wire/line bodies. The default is No.

Type bool (p. 1360)

Read Only No

MixedResolution

Allows parts of mixed dimension to be imported as components of assemblies which have parts of different dimensions. The default is None.

The following options control what is imported when there are bodies of mixed dimension in a multi-body part:

None	Nothing is imported.
Solid	Only solids are imported.
Sheet	Only sheet surfaces are imported.
SolidSheet	Only solids and sheet surfaces are imported.

Type MixedResolutionType (p. 1406)

Read Only No

NamedSelectionKey

Allows the specification of a key that is used to filter processed CAD System Named Selections during import. The key must be present at the beginning or end of a CAD Named Selection's name to be valid for import. The Selection Set Key supports multiple prefixes/suffixes with each value separated by a semicolon. If the value is empty, all Selection Sets are imported. The default key is "NS".

Type string (p. 1438)

Read Only No

ProcessAssociativity

Indicates if action should be taken to allow associativity. Associativity processing can have a negative impact on import performance. The default is Yes.

Type bool (p. 1360)

Read Only No

ProcessEnclosures

Enables the processing of enclosure and symmetry CAD System Named Selections. The default is Yes.

Type bool (p. 1360)

Read Only No

ReaderSaveUpdate

Enables saving of the internal part files generated from the geometry import during the import/update action. The default is No.

Type bool (p. 1360)

Read Only No

StitchSurfacesOnImport

No details are provided for this entry.

Type ImportStitchType (p. 1392)

Read Only No

StitchTolerance

No details are provided for this entry.

Type double (p. 1378)

Read Only No

UseInstances

Honors a geometry import's part instances during processing to produce faster import times and allow smaller database sizes. The default is Yes.

Type bool (p. 1360)

Read Only No

GlobalMeshAdvancedAttributes

GlobalMeshAdvancedAttributes specifies the global advanced properties for Part Meshing.

Available options:

ShapeChecking	Determines the shape checking algorithm to be used by the mesher.
TriangleSurfaceMesher	Determines surface meshing algorithm to be used by patch conforming mesher.
ElementMidsideNodes	Determines whether elements are with or without midside nodes.

StraightSidedElements	Specifies meshing to straight edge elements.
LoopRemovalTolerance	Specifies the tolerance value for sheet loop removal.
MeshBasedDefeaturing	Specifies automatic defeaturing of dirty geometries.
DefeaturingTolerance	Specifies the defeaturing tolerance value.
UseAllProcessors	Specifies if all available processors are used for parallel part-based meshing.
ProcessorLimit	Specifies the maximum number of processors to be utilized for parallel part-based meshing.

Properties

DefeaturingTolerance

Specifies the defeaturing tolerance value.

Type Quantity (p. 1422)

Read Only No

ElementMidsideNodes

Determines whether elements are with or without midside nodes.

Available options:

UseEngineeringIntent	Determine whether midside nodes are kept or dropped based on the Engineering Intent.
Dropped	Elements without midside nodes.
Kept	Elements with midside nodes.

Type ElementMidsideNodes (p. 1380)

Read Only No

LoopRemovalTolerance

Specifies the tolerance value for sheet loop removal. Any loop with radius less than or equal to Loop Removal Tolerance value is removed by mesher. Setting the tolerance value to 0 means that no loops will be removed.

Type Quantity (p. 1422)

Read Only No

MeshBasedDefeaturing

Specifies automatic defeaturing of dirty geometries.

Available options:

AutomaticallyDetermined	Automatically removes features smaller than or equal to the calculated default tolerance value on dirty geometry.
UserDefined	Automatically removes features smaller than or equal to the user defined defeaturing tolerance value on dirty geometry.
Off	No defeaturing done on dirty geometry.

Type MeshBasedDefeaturing (p. 1403)

Read Only No

ProcessorLimit

Specifies the number of processors to be used by part-based meshing task to enhance performance and ranges from 0 to 64. Default value of 0 (when Use all processors = Yes) automatically ensures that maximum number of available CPU cores are utilized.

Type uint (p. 1446)

Read Only No

ShapeChecking

Determines the shape checking algorithm to be used by the mesher.

Available options:

UseEngineeringIntent	Determine which algorithm to use based on the Engineering Intent.
StandardMechanical	Specifies the standard mechanical shape checking algorithm for structural simulations.
AggressiveMechanical	Specifies the aggressive mechanical shape checking algorithm for structural simulations.
CFD	Specifies the CFD shape checking algorithm for flow simulations.
None	Disable shape checking.

Type ShapeChecking (p. 1431)

Read Only No

StraightSidedElements

Specifies meshing to straight edge elements.

Available options:

No
Yes

Type StraightSidedElements (p. 1437)

Read Only No

TriangleSurfaceMesher

Determines surface meshing algorithm to be used by patch conforming mesher.

Available options:

Program Controlled	Mesher determines usage of Delaunay or Advancing Front algorithm.
Advancing Front	Mesher uses Advancing Front algorithm but can switch to Delaunay if problems occur.

Type TriangleSurfaceMesher (p. 1444)

Read Only No

UseAllProcessors

Specifies usage of all available processors for parallel part-based meshing.

Available options:

No	Manually specify the limit to the number of processors that can be utilized for parallel part-based meshing.
Yes	Automatically detects and utilizes all available processors for meshing parts of an assembly in parallel.

Type UseAllProcessors (p. 1447)

Read Only No

GlobalMeshBoundaryLayerAttributes

GlobalMeshBoundaryLayerAttributes specifies the global boundary layer properties.

Available options:

CollisionAvoidance	Determines strategy to avoid collision between inflated surface meshes.
FixFirstLayer	Determines whether the heights or ratios of the first boundary layer will be modified to avoid collision.
GapFactor	Specifies gap between intersecting prisms.
MaxHeightOverBase	Specifies maximum allowable prism aspect ratio.
GrowthRateType	Determines height of boundary layer for given initial height and height ratio.
MaxAngle	Specifies the maximum prism layer growth angle.
FilletRatio	Specifies whether fillets proportional to the prism height will be created in corner zones of tetrahedral mesh.

UsePostSmoothing	Specifies whether post boundary layer smoothing will be performed.
SmoothIterations	Specifies the number of post boundary layer smoothing iterations.

Properties

CollisionAvoidance

Determines strategy to avoid collision between inflated surface meshes.

Available options:

None	No check for layer collisions.
LayerCompression	Boundary layers are compressed in collision areas.
StairStepping	Prism layers are stair stepped to avoid collision and maintain the gap defined by gap factor.

Type CollisionAvoidance (p. 1365)

Read Only No

FilletRatio

Specifies whether fillets proportional to the prism height will be created in corner zones of tetrahedral mesh.

Type double (p. 1378)

Read Only No

FixFirstLayer

Determines whether the heights or ratios of the first boundary layer will be modified to avoid collision.

Available options:

No
Yes

Type FixFirstLayer (p. 1386)

Read Only No

GapFactor

Specifies gap between intersecting prisms.

Type double (p. 1378)

Read Only No

GrowthRateType

Determines height of boundary layers for given initial height and height ratio.

Available options:

Geometric	Height of boundary layer determined geometrically.
Linear	Height of boundary layer determined linearly.
Exponential	Height of boundary layer determined exponentially.

Type GrowthRateType (p. 1389)

Read Only No

MaxAngle

Specifies the maximum prism layer growth angle.

Type Quantity (p. 1422)

Read Only No

MaxHeightOverBase

Specifies maximum allowable prism aspect ratio.

Type double (p. 1378)

Read Only No

SmoothIterations

Specifies the number of post boundary layer smoothing iterations.

Type int (p. 1394)

Read Only No

UsePostSmoothing

Specifies whether post boundary layer smoothing will be performed.

Available options:

No
Yes

Type UsePostSmoothing (p. 1448)

Read Only No

GlobalMeshSizingAttributes

GlobalMeshSizingAttributes specifies the global mesh sizing attributes.

Available options:

InheritanceWrap	Specifies whether the sizing function for sew is inherited from the predecessor wrap task or not.
UseAdvancedSizeFunction	Specifies options for advanced size function.
MinSize	Specifies the minimum size returned to the mesher.
ProximityMinSize	Specifies global minimum size when Advanced Size Function is Proximity and Curvature or Proximity.
MaxFaceSize	Specifies maximum size returned to the surface mesher.
MaxSize	Specifies the maximum size returned to the mesher.
ProximitySizeFunctionSources	Specifies the source type to be used for size function calculation when Advanced Size Function is Proximity or Curvature and proximity.
GrowthRate	Specifies the increase in edge length with each succeeding layer of elements.
CurvatureNormalAngle	Specifies maximum allowable angle that one element is allowed to span when Advanced Size Function is Proximity and Curvature or Curvature.
NumberOfCellsAcrossGap	Specifies minimum number of elements to be generated in gaps when Advanced Size Function is Proximity and Curvature or Proximity.
Smoothing	Specifies the number of smoothing iterations.
AdaptiveResolution	The scaling value for the arguments of various functions for the adaptive size function.

Properties

AdaptiveResolution

Specifies adaptive resolution

Type double (p. 1378)

Read Only No

CurvatureNormalAngle

Specifies maximum allowable angle that one element is allowed to span when Advanced Size Function is Proximity and Curvature or Curvature.

Type Quantity (p. 1422)

Read Only No

ElementSeedSize

Specifies the element seed size returned to the mesher.

Type Quantity (p. 1422)

Read Only No

GrowthRate

Specifies the increase in edge length with each succeeding layer of elements.

Type double (p. 1378)

Read Only No

MaxFaceSize

Specifies maximum size returned to the surface mesher.

Type Quantity (p. 1422)

Read Only No

MaxSize

Specifies the maximum size returned to the mesher.

Type Quantity (p. 1422)

Read Only No

MinSize

Specifies the minimum size returned to the mesher.

Type Quantity (p. 1422)

Read Only No

NumberOfCellsAcrossGap

Specifies minimum number of elements to be generated in gaps when Advanced Size Function is Proximity and Curvature or Proximity.

Type int (p. 1394)

Read Only No

ProximityMinSize

Specifies global minimum size when Advanced Size Function is Proximity and Curvature or Proximity.

Type Quantity (p. 1422)

Read Only No

ProximitySizeFunctionSources

Specifies the source type to be used for size function calculation when Advanced Size Function is Proximity or Curvature and proximity.

Available options:

Faces
Edges
FacesAndEdges

Type ProximitySizeFunctionSources (p. 1422)

Read Only No

Smoothing

Specifies the number of smoothing iterations.

Available options:

Low
Medium
High

Type Smoothing (p. 1433)

Read Only No

UseAdvancedSizeFunction

Specifies options for advanced size function.

Available options:

Adaptive
Curvature
Proximity
Fixed
ProximityCurvature

Type UseAdvancedSizeFunction (p. 1447)

Read Only No

UseFixedSizeFunctionForSheets

Specifies use of Fixed Size function for sheets if there are present in the geometry.

Type bool (p. 1360)

Read Only No

GlobalPAMAdvanceAttributes

No details are provided for this entry.

Properties

MidSideNodes

Mid side nodes info to determine the element order either linear or quadratic.

Type APAMMidsideNodeOrder (p. 1355)

Read Only No

GlobalPAMSizingAttributes

No details are provided for this entry.

Properties

AdvancedSizeFunction

No details are provided for this entry.

Type APAMAdvancedSizeFunction (p. 1354)

Read Only No

CurvatureNormalAngle

No details are provided for this entry.

Type Quantity (p. 1422)

Read Only Yes

DisableAutomaticControls

Disables automatic (proxy) controls allowing only mesh controls and

to control the mesh

Type bool (p. 1360)

Read Only No

GrowthRate

Specifies the increase in edge length with each succeeding layer of elements.

Type double (p. 1378)

Read Only Yes

MaxFaceSize

No details are provided for this entry.

Type Quantity (p. 1422)

Read Only Yes

MaxSize

No details are provided for this entry.

Type Quantity (p. 1422)

Read Only Yes

MinSize

No details are provided for this entry.

Type Quantity (p. 1422)

Read Only Yes

NumberOfCellsAcrossGap

Specifies minimum number of elements to be generated in gaps when Advanced Size Function is Proximity and Curvature or Proximity.

Type int (p. 1394)

Read Only Yes

ProximitySizeFunctionSources

Specifies the source type to be used for size function calculation when Advanced Size Function is Proximity or Curvature and proximity.

Available options:

Faces
Edges
FacesAndEdges

Type APAMProximitySizeFunctionSources (p. 1355)

Read Only Yes

UsePredefinedSettings

Automatically calculates the global sizing values based on the resolution value and the choice of size function.

Type bool (p. 1360)

Read Only No

GlobalParametersExtension

Frequency parameters for Results

Properties

Frequency

The computed Frequency for this Result.

Type Quantity (p. 1422)

Read Only Yes

FrequencyImaginary

The Imaginary (damping) component in the computed Frequency.

Type Quantity (p. 1422)

Read Only Yes

FrequencyReal

The Real (stability) component in the computed Frequency.

Type Quantity (p. 1422)

Read Only Yes

GlobalWrapSewAdvancedAttributes

GlobalWrapSewAdvancedAttributes specifies the global advanced properties for wrap and sew.

Available options:

TessellationRefinement	Specifies value to be used for tessellation refinement.
RefinementTolerance	Specifies the user-defined tolerance for tessellation refinement.
EdgeExtractionAngle	Determines which CAD features are captured.
CreateIntersectingEdgesForOverlappingBodies	Create intersecting edges for overlapping bodies.
Zoning	Specifies options for zoning.
TargetMetric	Specifies the measure for mesh quality.
MetricValue	Specifies the value at which wrapper surface will be improved.

Aggressive	Specifies whether mesh may deviate from geometry as a result of wrapper surface improvement.

Properties

Aggressive

Specifies whether mesh may deviate from geometry as a result of wrapper surface improvement.

Available options:

MeetTarget
MaintainGeometry

Type Aggressive (p. 1353)

Read Only No

CreateIntersectingEdgesForOverlappingBodies

Create intersecting edges for overlapping bodies.

Available options:

No
Yes

Type CreateIntersectingEdgesForOverlappingBodies (p. 1370)

Read Only No

EdgeExtractionAngle

Determines which CAD features are captured.

Type Quantity (p. 1422)

Read Only No

MetricValue

Specifies the value at which wrapper surface will be improved.

Type double (p. 1378)

Read Only No

RefinementTolerance

Specifies the refinement tolerance for user defined tessellation refinement.

Type Quantity (p. 1422)

Read Only No

TargetMetric

Specifies the measure for mesh quality.

Available options:

Skewness
SizeChange
AspectRatio

Type Metric (p. 1405)

Read Only No

TessellationRefinement

Specifies value to be used for tessellation refinement.

Available options:

AutomaticallyDetermined	Sets tessellation refinement to 10% of the value of Min Size/Proximity Min Size.
UserDefined	Sets tessellation refinement to user defined value.
Off	Tessellation refinement is not performed.

Type TessellationRefinement (p. 1441)

Read Only No

Zoning

Specifies options for zoning.

Available options:

Part
Body
Face

Type Zoning (p. 1452)

Read Only No

GravityDefinition

GravityDefinition defines the three components of gravity and the reference frame it is relative to.

Properties

RelativeToName

Relative to

Type string (p. 1438)

Read Only Yes

XComponent

Component in the x-axis direction

Type Quantity (p. 1422)

Read Only No

YComponent

Component in the y-axis direction

Type Quantity (p. 1422)

Read Only No

ZComponent

Component in the z-axis direction

Type Quantity (p. 1422)

Read Only No

HatSection

This class represents HAT beam cross section

Properties

LeftBrimThickness

Left Brim Thickness

Type Quantity (p. 1422)

Read Only No

LeftBrimWidth

Left Brim Width

Type Quantity (p. 1422)

Read Only No

LeftStemThickness

Left Stem Thickness

Type Quantity (p. 1422)

Read Only No

OverallHeight

Overall Height

Type Quantity (p. 1422)

Read Only No

RightBrimThickness

Right Brim Thickness

Type Quantity (p. 1422)

Read Only No

RightBrimWidth

Right Brim Width

Type Quantity (p. 1422)

Read Only No

RightStemThickness

Right Stem Thickness

Type Quantity (p. 1422)

Read Only No

TopHatThickness

Top Hat Thickness

Type Quantity (p. 1422)

Read Only No

TopHatWidth

Top Hat Width

Type Quantity (p. 1422)

Read Only No

HollowRectangularSection

This class represents hollow rectangular beam cross section

Properties

LowerThickness

Lower Thickness

Type Quantity (p. 1422)

Read Only No

OuterHeightOfTheBox

Outer Height of the Box

Type Quantity (p. 1422)

Read Only No

OuterWidthOfTheBox

Outer Width of the Box

Type Quantity (p. 1422)

Read Only No

UpperThickness

Upper Thickness

Type Quantity (p. 1422)

Read Only No

VerticalLeftThickness

Vertical Left Thickness

Type Quantity (p. 1422)

Read Only No

VerticalRightThickness

Vertical Right Thickness

Type Quantity (p. 1422)

Read Only No

Import

Main Data Import Task user object. Holds the list of import source objects

Properties

No Properties.

Methods

AddActiveCADAttachSourceOperation

Creates a single Geometry Import Source within an Import Task. The import source object will be initialized with its source selection mode set to Active CAD Attach.

AddGeometryImportSourceOperation

Adds a single Geometry Import Source to an Import Task.

Return Reference to the Geometry Import Source.

Type DataReference (p. 1371)

Optional Arguments

FilePath Path to the chosen file.

Type string (p. 1438)

ChooseImportSources

Multiple selection of import source files. The file list is processed and import source objects are created within the Import Task.

Required Arguments

FilePaths List of paths to the chosen files.

Type List (p. 1400)<string (p. 1438)>

ISection

This class represents I beam cross section

Properties

LowerFlangeThickness

Lower Flange Thickness

Type Quantity (p. 1422)

Read Only No

LowerFlangeWidth

Lower Flange Width

Type Quantity (p. 1422)

Read Only No

OverallHeight

Overall height

Type Quantity (p. 1422)

Read Only No

StemThickness

Stem Thickness

Type Quantity (p. 1422)

Read Only No

UpperFlangeThickness

Upper Flange Thickness

Type Quantity (p. 1422)

Read Only No

UpperFlangeWidth

Upper Flange Width

Type Quantity (p. 1422)

Read Only No

IsoSurface

Creates surfaces of constant value of the specified variable.

Properties

EvaluateFullRange

Whether or not to evaluate all time points in the evaluation

Type bool (p. 1360)

Read Only No

IsovalueMethod

The method by which isovalues are defined.

Type IsovalueMethod (p. 1397)

Read Only No

Isovalues

The list of isovalues.

Type Expression (p. 1384)<List (p. 1400)<Quantity (p. 1422)>>

Read Only No

LastAutomaticDisplayText

The last automatically-generated DisplayText assigned to this Result Object

Type string (p. 1438)

Read Only No

Location

The Location over which the Result is to be evaluated.

Type LocationSet (p. 1401)

Read Only No

NumberOfIsovalues

The number of isovalues.

Type int (p. 1394)

Read Only No

ResultName

Result name

Type string (p. 1438)

Read Only No

SimulationStep

The Simulation Step object reference for a multi-step analysis.

Type SolutionStep (p. 1434)

Read Only No

Variable

The variable of interest.

Type Expression (p. 1384)<Quantity (p. 1422)>

Read Only No

IsotropicElasticity

The Isotropic Elasticity material model.

Properties

Derivation

The Isotropic Elasticity values can be entered for one of the following combinations:

YoungsModulusPoissonsRatio	Young's Modulus and Poisson's Ratio
ShearModulusPoissonsRatio	Shear Modulus and Poisson's Ratio
BulkModulusPoissonsRatio	Bulk Modulus and Poisson's Ratio
YoungsModulusShearModulus	Young's Modulus and Shear Modulus
YoungsModulusBulkModulus	Young's Modulus and Bulk Modulus
BulkModulusShearModulus	Bulk Modulus and Shear Modulus

which will then compute the other values using Hooke's law.

Type DerivationType (p. 1373)

Read Only No

Magnitude

The Magnitude is used for an expresion to include variations of the data for any independent variable (e.g. Temperature).

Type Expression (p. 1384)<IsotropicElasticityDependents (p. 1397)>

Read Only No

IsotropicElectricalConductivity

Isotropic bulk conductivity material model.

Properties

Value

The Value is an expresion which evaluates to a Quantity with the correct units.

Type Expression (p. 1384)<Quantity (p. 1422)>

Read Only No

IsotropicMagneticLossTangent

Magnetic Loss Tangent material model.

Properties

Value

The Value is an expresion which evaluates to a Quantity with the correct units.

Type Expression (p. 1384)<Quantity (p. 1422)>

Read Only No

IsotropicRelativePermeability

The Isotropic Relative Permeability model.

Can be applied to Magnetic analyses

Properties

Magnitude

Gets or sets the magnitude.

Type Expression (p. 1384)<Quantity (p. 1422)>

Read Only No

IsotropicRelativePermittivity

Isotropic relative permittivity material model.

Properties

Magnitude

The Magnitude is an expresion which evaluates to a Quantity with the correct units.

Type Expression (p. 1384)<Quantity (p. 1422)>

Read Only No

IsotropicResistivity

The input for resistivity which is isotropic.

This can be used in an electric/magnetic analysis.

Properties

Magnitude

Gets or sets the magnitude.

Type Expression (p. 1384)<Quantity (p. 1422)>

Read Only No

IsotropicSecantCoefficientofThermalExpansion

The Isotropic Secant Coefficient of Thermal Expansion material model.

Properties

Magnitude

The Magnitude is an expresion which evaluates to a Quantity with the correct units.

Type Expression (p. 1384)<Quantity (p. 1422)>

Read Only No

IsotropicSeebeckCoefficient

Seebeck Coefficient : Isotropic model.

Can be applied to Thermal analyses

Properties

Magnitude

Gets or sets the magnitude.

Type Expression (p. 1384)<Quantity (p. 1422)>

Read Only No

IsotropicThermalConductivity

The Isotropic Thermal Conductivity material model.

Properties

Magnitude

The Magnitude is an expresion which evaluates to a Quantity with the correct units.

Type Expression (p. 1384)<Quantity (p. 1422)>

Read Only No

Legend

Specifies how the legend for the current object should be displayed.

Properties

ClipToRange

Specifies whether to clip out-of-range values. If not set, the out-of-range values are colored by the Invalid Min/Max Colors.

Type bool (p. 1360)

Read Only No

ColorDistribution

The distribution used to compute the size of each color band.

Type ColorDistributionOption (p. 1365)

Read Only No

Coloring

Specifies smooth or banded coloring.

Type LegendColoring (p. 1399)

Read Only No

NumberOfColors

The number of colors to be used.

Type int (p. 1394)

Read Only No

RangeMaximum

Specifies the maximum user range value when variable range is 'User Specified'.

Type Quantity (p. 1422)

Read Only No

RangeMinimum

Specifies the minimum user range value when variable range is 'User Specified'.

Type Quantity (p. 1422)

Read Only No

VariableRange

Specifies 'Local' or 'User Specified' variable range.

Type VariableRangeOption (p. 1449)

Read Only No

Line

Line represented by 2 end points

Properties

Point1

Line start point

Type Point (p. 1417)

Read Only No

Point2

Line end point

Type Point (p. 1417)

Read Only No

LineChart

Displays the quantity of a variable on a line

Properties

EvaluateFullRange

Whether or not to evaluate all time points in the evaluation

Type bool (p. 1360)

Read Only No

Location

The Location over which the Result is to be evaluated.

Type LocationSet (p. 1401)

Read Only No

NormalizeData

Normalize data so they are displayed as percentages

Type bool (p. 1360)

Read Only No

ReverseDirection

Reverse the direction of the data when it's sent to the graph

Type bool (p. 1360)

Read Only No

SimulationStep

The Simulation Step object reference for a multi-step analysis.

Type SolutionStep (p. 1434)

Read Only No

Variable

Data Model support for variable selector.

Type Expression (p. 1384)<Quantity (p. 1422)>

Read Only No

Variables

List of variables to be plotted

Type List (p. 1400)<Expression (p. 1384)<Quantity (p. 1422)>>

Read Only No

LSection

This class represents L beam cross section

Properties

HorizontalLegLength

Horizontal Leg Length

Type Quantity (p. 1422)

Read Only No

HorizontalLegThickness

Horizontal Leg Thickness

Type Quantity (p. 1422)

Read Only No

VerticalLegLength

Vertical Leg Length

Type Quantity (p. 1422)

Read Only No

VerticalLegThickness

Vertical Leg Thickness

Type Quantity (p. 1422)

Read Only No

MagneticCoercivity

Magnetic Coercivity material model.

Properties

MagnetizationDirectionType

Magnetization direction type with dropdown.

Type DirectionType (p. 1376)

Read Only No

Magnitude

Magnetic coercivity magnitude

Type Expression (p. 1384)<Quantity (p. 1422)>

Read Only No

Primary

First component

Type Expression (p. 1384)<Quantity (p. 1422)>

Read Only No

Secondary

Second component

Type Expression (p. 1384)<Quantity (p. 1422)>

Read Only No

Tertiary

Third component

Type Expression (p. 1384)<Quantity (p. 1422)>

Read Only No

Material

The material object holds all information that defines the behavior for a specific material.

Properties

DefaultStateOfMatter

A material can have data for the states; solid, liquid and gas. The default specifies that when this material is assigned to the model it will by default use this state.

Type StateOfMatter (p. 1436)

Read Only No

Description

A description of the material contents.

Type string (p. 1438)

Read Only No

SourceInformation

Information about the original source of this managed engineering data.

Type SourceInformation (p. 1434)

Read Only Yes

Methods

AddPropertyTable

This command is used to allow the expression of a property to reference data in the named table. This is a reference and not a copy, and so if the table is used in the expression, and a change is made on the referenced table and the change is in the data being referenced the value of the property will be modified. The command creates table functions in the context of this expression which can be used with constants, field variables, or another named expression. If the property does not support an expression an error will occur.

Return An int which corresponds to the Index passed in to the command and indicates the comand successfully completed.

Type int (p. 1394)

Required Arguments

Dependents A List of the column index(es) to use from the Table which correspond to the dependent variable(s) of the Property. The indexes can be an Integer index or the String name of the column.

Type List (p. 1400)<Object (p. 1409)>

Independents A List of the column index(es) to use from the Table for use in the expression and correspond to the independent variables of the Property. The indexes can be an Integer index or the String name of the column.

Type List (p. 1400)<Object (p. 1409)>

Index An integer which uniquely identifies the Table for use in the expression of the Property. In the expression this table can be accessed via tableIndex(args). The table with an index of one can be accessed in the expression as table(args) or table1(args).

Type uint (p. 1446)

Property The String which provides the path of the property on the Parent.

Type string (p. 1438)

Table The DataReference of the table to add to the Property.

Type DataReference (p. 1371)

Optional Arguments

Bounds An enum which describes what will occur when the index is outside of the bounds of the table. The valid enum values are; Constant, Fit, Zero, and Error. The default is Error.

Type TableInterpolationBeyondBounds (p. 1441)

Default Value Error

Interpolation An enum of the type of interpolation to use when accessing data from the table at a given index. The valid enum values are; None, Linear, Cubic Spline. The default is Linear.

Type TableInterpolation (p. 1440)

Default Value Linear

Example

```
tbl1 = pressure1.CreateTable(Columns=[[“X”, “Length”], [“Y”, “Length”], [“Z”, “Length”], [“Pressure”, “Pre
#
# Use three of the columns from tbl1
bc1.AddPropertyTable(Property=“Magnitude”, Table=tbl1, Index=1, Independents=[“X”, “Y”], Dependents=[“Pres
# In the table function the x and y refer to field variables
bc1.SetPropertyExpression(Entity=bc1, Name=“Magnitude”, Expresssion=“table1(x, y)”)
#
# Use four of the columns from tbl1
```

```
bc1.AddPropertyTable(Property="Temperature", Table=tbl1, Index=1, Independents=["X", "Y", "Z"], Dependents
# In the table function the x, y, and z refer to field variables
bc1.SetPropertyExpression(Entity=bc1, Name="Temperature", Expression="table1(x, y, z)")
#
# Use two of the columns from tbl1
bc1.AddPropertyTable(Property="Magnitude", Table=tbl1, Index=1, Independents=["X"], Dependents=["Pressure"
# In the table function the x refers to field variables
bc1.SetPropertyExpression(Entity=bc1, Name="Magnitude", Expression="table1(x)")
#
# Use of a constant
bc1.SetPropertyExpression(Entity=bc1, Name="Magnitude", Expression="table1(20)")
```

CreateContent

Includes a physical quantity or the constitutive relation for the physical response of a material.

The material property is created based on the specified optional parameters "Definition" and "Behavior".

You can use the GetValidMaterialContentQuery query to obtain a list of available property/models with the corresponding behavior and/or definition.

Optional Arguments

Behavior The optional string to identify the behavior of the property/model.

The behavior typically refers to how the material responds, e.g., the density property could behave as an Ideal Gas or a Real Gas.

Type string (p. 1438)

Definition The optional string to identify the way in which new material property/model will be defined.

In some cases the material property/model may be defined in different ways, e.g., Thermal Expansion can be defined using Secant or Instantaneous.

Type string (p. 1438)

Model The string to identify the material property/model to be included.

Type string (p. 1438)

State The optional string to identify the physical state of the property/model.

Type string (p. 1438)

Example

This example assumes that a material has been created.

It then adds the Isotropic Elasticity model to the material

```
material1.CreateContent(Model="Elasticity", Definition="", Behavior="Isotropic")
```

CreatePropertyTable

CreatePropertyTable is a convenience command which performs the following commands:

Creates a table. See the CreateEntity command, where Type="Table".
Adds columns corresponding to the Fields parameter and the dependent variables of the Property parameter. See the and AddTableColumn command.
Adds the table to the Property parameter. See the AddPropertyTable command.
Sets the expression on the Property parameter to “table(Fields)”. See the SetPropertyExpression command.

Return The DataReference of the newly created table.

Type DataReference (p. 1371)

Required Arguments

Fields A List of field variable names to be used as the independent variables for this property. The dependent variables are defined by the Property.

Type List (p. 1400)<string (p. 1438)>

Property The String which provides the path of the property on the Parent.

Type string (p. 1438)

Example

This example creates a new Table for the Isotropic Resistivity property in a material.

```
# drMaterial is the DataReference of the material having an Isotropic Resistivity property
table1 = material1.CreatePropertyTable(
    Property="IsotropicResistivity.Magnitude",
    Fields=["Temperature"])
table1.AddRow(Data=["100 [C]",".01 [ohm m]"])
table1.AddRow(Data=["200 [C]",".02 [ohm m]"])
table1.AddRow(Data=["300 [C]",".06 [ohm m]"])
table1.GetData()
```

DeleteContent

Deletes a material model from the specified material.

Required Arguments

Model The string to identify the material property/model to be removed.

Type string (p. 1438)

Optional Arguments

Behavior The optional string to identify the behavior of the property/model that is to be deleted.

The behavior typically refers to how the material responds, e.g., the density property could behave as an Ideal Gas or a Real Gas.

Type string (p. 1438)

Definition The optional string to identify the way in which material property/model to be deleted was defined.

In some cases the material property/model may be defined in different ways, e.g., Thermal Expansion can be defined using Secant or Instantaneous.

Type string (p. 1438)

State The optional string to identify the physical state to add the property/model.

Type string (p. 1438)

Example

This example deletes the Isotropic Elasticity model from an already created material.

```
material1.DeleteContent(Model="Elasticity", Definition="", Behavior="Isotropic")
```

ExportEngineeringData

ExportEngineeringData allows an engineering data object which can be managed to be exported to a source via a DataManager

Required Arguments

Source The source that this data will be exported to, via the Data Manager.

Type string (p. 1438)

Optional Arguments

DataManager The data manager which is managing the requested engineering data. If not provided then the DataManager associated with the Entity will be used. If the Entity has no associated DataManager the default ANSYS Data Manager will be used.

Type DataReference (p. 1371)

Format The format which the Data Manager should use to save the entity. If it is not provided the DataManager will determine it based on the Source. If the Source does not yet exist the Data Manager will determine the format if it is a single type or throw an exception if the Format is not provided.

Type string (p. 1438)

Default Value HDF5

Example

This example assumes that a material and data manager exist in the Study

```
dataManager1 = GetDataEntity("/MaterialDataContainer/DataManager:DataManager 1")
```

```
material1.ExportEngineeringData(
    Source=r"C:\Users\epc\AppData\Roaming\Ansys\v151\Engineering_Data_Library.xml",
    DataManager=dataManager1)
```

ImportEngineeringData

ImportEngineeringData allows an engineering data object which can be managed to import data from a source via a DataManager

Required Arguments

Name The name of the engineering data to import from the DataManager.

Type string (p. 1438)

Optional Arguments

DataManager The data manager which is managing the requested engineering data. If not provided then the DataManager associated with the Entity will be used. If the Entity has no associated DataManager the default ANSYS Data Manager will be used.

Type DataReference (p. 1371)

Source The source of the data which will be managed by the Data Manager. If it is not provided the Source property of the DataManager will be used, if it is not set then an exception will occur.

Type string (p. 1438)

Example

The example assumes that a material has already been created.

```
material1.ImportEngineeringData(Name="Aluminum Alloy")
```

RemovePropertyTable

Remove a table reference from a property to prevent its use in an expression. If the property does not support an expression an error will occur.

Required Arguments

Index The integer which uniquely identifies the Table for use in the expression of the Property.

Type uint (p. 1446)

Property The String which provides the path of the property on the Parent.

Type string (p. 1438)

Example

```
tbl1 = pressure1.CreateTable(Columns=[[“X”, “Length”], [“Y”, “Length”], [“Pressure”, “Pressure”]])
bc1.AddPropertyTable(Property=“Magnitude”, Table=tbl1, Independents=[“X”, “Y”], Dependents=[“Pressure”])
# In the table function the x and y refer to field variables
bc1.SetPropertyExpression(Entity=bc1, Name=“Magnitude”, Expression=“table1(x, y)”)
bc1.RemovePropertyTable(Property=“Magnitude”, Index=1);
```

MaterialAssignment

Assigns a Material to a specified Location.

Properties

Location

The Location for the Material.

Type LocationSet (p. 1401)

Read Only No

Material

For UI access only to the reference to the Material, other areas should use MaterialAtState

Type Material (p. 1402)

Read Only No

ReferenceFrame

When applicable, for example orthotropic data is present, the reference frame is used to determine the material's x, y, and z coordinates.

Type ReferenceFrame (p. 1424)

Read Only No

StateOfMatter

The state of matter to use for this context.

Type StateOfMatter (p. 1436)

Read Only No

MaterialAtState

The material object holds all information that defines the behavior for a specific material.

Properties

Description

A description of the material contents.

Type string (p. 1438)

Read Only No

SourceInformation

Information about the original source of this managed engineering data.

Type SourceInformation (p. 1434)

Read Only Yes

Methods

AddPropertyTable

This command is used to allow the expression of a property to reference data in the named table. This is a reference and not a copy, and so if the table is used in the expression, and a change is made on the referenced table and the change is in the data being referenced the value of the property will be modified. The command creates table functions in the context of this expression which can be used with constants, field variables, or another named expression. If the property does not support an expression an error will occur.

Return An int which corresponds to the Index passed in to the command and indicates the comand successfully completed.

Type int (p. 1394)

Required Arguments

Dependents A List of the column index(es) to use from the Table which correspond to the dependent variable(s) of the Property. The indexes can be an Integer index or the String name of the column.

Type List (p. 1400)<Object (p. 1409)>

Independents A List of the column index(es) to use from the Table for use in the expression and correspond to the independent variables of the Property. The indexes can be an Integer index or the String name of the column.

Type List (p. 1400)<Object (p. 1409)>

Index An integer which uniquely identifies the Table for use in the expression of the Property. In the expression this table can be accessed via tableIndex(args). The table with an index of one can be accessed in the expression as table(args) or table1(args).

Type uint (p. 1446)

Property The String which provides the path of the property on the Parent.

Type string (p. 1438)

Table The DataReference of the table to add to the Property.

Type DataReference (p. 1371)

Optional Arguments

Bounds An enum which describes what will occur when the index is outside of the bounds of the table. The valid enum values are; Constant, Fit, Zero, and Error. The default is Error.

Type TableInterpolationBeyondBounds (p. 1441)

Default Value Error

Interpolation An enum of the type of interpolation to use when accessing data from the table at a given index. The valid enum values are; None, Linear, Cubic Spline. The default is Linear.

Type TableInterpolation (p. 1440)

Default Value Linear

Example

```
tbl1 = pressure1.CreateTable(Columns=[[“X”, “Length”], [“Y”, “Length”], [“Z”, “Length”], [“Pressure”, “Pre
#
# Use three of the columns from tbl1
bc1.AddPropertyTable(Property=“Magnitude”, Table=tbl1, Index=1, Independents=[“X”, “Y”], Dependents=[“Pres
# In the table function the x and y refer to field variables
bc1.SetPropertyExpression(Entity=bc1, Name=“Magnitude”, Expresssion=“table1(x, y)”)
#
# Use four of the columns from tbl1
bc1.AddPropertyTable(Property=“Temperature”, Table=tbl1, Index=1, Independents=[“X”, “Y”, “Z”], Dependents
# In the table function the x, y, and z refer to field variables
bc1.SetPropertyExpression(Entity=bc1, Name=“Temperature”, Expression=“table1(x, y, z)”)
#
# Use two of the columns from tbl1
bc1.AddPropertyTable(Property=“Magnitude”, Table=tbl1, Index=1, Independents=[“X”], Dependents=[“Pressure”
# In the table function the x refers to field variables
bc1.SetPropertyExpression(Entity=bc1, Name=“Magnitude”, Expression=“table1(x)”)
#
# Use of a constant
bc1.SetPropertyExpression(Entity=bc1, Name=“Magnitude”, Expression=“table1(20)”)
```

RemovePropertyTable

Remove a table reference from a property to prevent its use in an expression. If the property does not support an expression an error will occur.

Required Arguments

Index The integer which uniquely identifies the Table for use in the expression of the Property.

Type uint (p. 1446)

Property The String which provides the path of the property on the Parent.

Type string (p. 1438)

Example

```
tbl1 = pressure1.CreateTable(Columns=[[“X”, “Length”], [“Y”, “Length”], [“Pressure”, “Pressure”]])
bc1.AddPropertyTable(Property=“Magnitude”, Table=tbl1, Independents=[“X”, “Y”], Dependents=[“Pressure”])
# In the table function the x and y refer to field variables
bc1.SetPropertyExpression(Entity=bc1, Name=“Magnitude”, Expression=“table1(x, y)”)
bc1.RemovePropertyTable(Property=“Magnitude”, Index=1);
```

MaterialOrientation

No details are provided for this entry.

Properties

DefineBy

No details are provided for this entry.

Type OrientationDefineBy (p. 1410)

Read Only No

EdgeGuide

No details are provided for this entry.

Type LocationSet (p. 1401)

Read Only No

MaximumNumberOfDisplayPoints

No details are provided for this entry.

Type int (p. 1394)

Read Only No

RelativeTo

No details are provided for this entry.

Type ReferenceFrame (p. 1424)

Read Only No

SurfaceGuide

No details are provided for this entry.

Type LocationSet (p. 1401)

Read Only No

MeshControlBodySizing

Body Sizing control which contains all sizing options a user can place on a Body reference

Properties

Behavior

Specifies whether size control settings can be changed by the mesher (Soft) or not (Hard).

Available options:

Soft
Hard

Type SizingBehavior (p. 1433)

Read Only No

CurvatureNormalAngle

Specifies maximum angle that one element edge is allowed to span.

Type Quantity (p. 1422)

Read Only No

ElementSize

Specifies the maximum size for the edge, face or body sizing control.

Type Quantity (p. 1422)

Read Only No

GrowthRate

Specifies increase in element edge length with each succeeding layer of elements.

Type double (p. 1378)

Read Only No

LocalMinSize

Specifies value to override global Min Size on local entities.

Type Quantity (p. 1422)

Read Only No

Location

A common property for all sub meshing objects with a defined Geometry Filter

Type LocationSet (p. 1401)

Read Only No

NumberOfCells

Specifies value to override global Number of cells across gap on local entities

Type int (p. 1394)

Read Only No

ProximityMinimumSize

Specifies value to override global Proximity Min size on local entities

Type Quantity (p. 1422)

Read Only No

ProximitySizeFunctionSource

Specifies value to override global Number of cells across gap on local entities

Type ProximitySizeFunctionSources (p. 1422)

Read Only No

SizeFunctionMethod

Specifies which size function to use.

Available options:

Program controlled
Adaptive
Curvature and proximity
Curvature
Proximity
Fixed

Type UseAdvancedLocalSizeFunction (p. 1447)

Read Only No

MeshControlEdgeSizing

Specifies attributes for edge sizing.

Available options:

EdgeSizingMethod	Specifies the edge sizing method.
EdgeDivisions	Specifies the number of divisions on each edge.
BiasMethod	Specifies bias option which is always smooth transition.
BiasType	Specifies option for bias type.
BiasGrowthRate	Specifies value for bias growth rate.

Properties

Behavior

Specifies whether size control settings can be changed by the mesher (Soft) or not (Hard).

Available options:

Soft
Hard

Type SizingBehavior (p. 1433)

Read Only No

BiasGrowthRate

Specifies value for bias growth rate.

Type double (p. 1378)

Read Only No

BiasMethod

Specifies bias option which is always smooth transition.

Type BiasMethod (p. 1358)

Read Only No

BiasType

Specifies option for bias type.

Available options:

Right
Left
CenterIn

CenterOut
Constant

Type BiasType (p. 1358)

Read Only No

CurvatureNormalAngle

Specifies maximum angle that one element edge is allowed to span.

Type Quantity (p. 1422)

Read Only No

EdgeDivisions

Specifies the number of divisions on each edge.

Type int (p. 1394)

Read Only No

EdgeSizingMethod

Specifies the edge sizing method.

Available options:

ElementSize
EdgeDivisions

Type EdgeSizingMethod (p. 1379)

Read Only No

ElementSize

Specifies the maximum size for the edge, face or body sizing control.

Type Quantity (p. 1422)

Read Only No

GrowthRate

Specifies increase in element edge length with each succeeding layer of elements.

Type double (p. 1378)

Read Only No

LocalMinSize

Specifies value to override global Min Size on local entities.

Type Quantity (p. 1422)

Read Only No

Location

A common property for all sub meshing objects with a defined Geometry Filter

Type LocationSet (p. 1401)

Read Only No

NumberOfCells

Specifies value to override global Number of cells across gap on local entities

Type int (p. 1394)

Read Only No

ProximityMinimumSize

Specifies value to override global Proximity Min size on local entities

Type Quantity (p. 1422)

Read Only No

ProximitySizeFunctionSource

Specifies value to override global Number of cells across gap on local entities

Type ProximitySizeFunctionSources (p. 1422)

Read Only No

SizeFunctionMethod

Specifies which size function to use.

Available options:

Program controlled
Adaptive
Curvature and proximity
Curvature
Proximity
Fixed

Type UseAdvancedLocalSizeFunction (p. 1447)

Read Only No

MeshControlElementShape

Mesh Method properties

Properties

LocalElementMidsideNodes

Show options to select the Local Element Midside Nodes

Type LocalElementMidsideNodes (p. 1400)

Read Only No

Location

A common property for all sub meshing objects with a defined Geometry Filter

Type LocationSet (p. 1401)

Read Only No

Shape

Show options to select the Mesh Method

Type ElementShape (p. 1380)

Read Only No

SourceLocation

Source Location for MultiZone/thin Sweep/Sweep Methods

Type LocationSet (p. 1401)

Read Only No

TargetLocation

Target Location for sweep Method

Type LocationSet (p. 1401)

Read Only No

MeshControlFaceSizing

Face Sizing control to be scoped to a topological selection of faces.

Properties

Behavior

Specifies whether size control settings can be changed by the mesher (Soft) or not (Hard).

Available options:

Soft
Hard

Type SizingBehavior (p. 1433)

Read Only No

CurvatureNormalAngle

Specifies maximum angle that one element edge is allowed to span.

Type Quantity (p. 1422)

Read Only No

ElementSize

Specifies the maximum size for the edge, face or body sizing control.

Type Quantity (p. 1422)

Read Only No

GrowthRate

Specifies increase in element edge length with each succeeding layer of elements.

Type double (p. 1378)

Read Only No

LocalMinSize

Specifies value to override global Min Size on local entities.

Type Quantity (p. 1422)

Read Only No

Location

A common property for all sub meshing objects with a defined Geometry Filter

Type LocationSet (p. 1401)

Read Only No

NumberOfCells

Specifies value to override global Number of cells across gap on local entities

Type int (p. 1394)

Read Only No

ProximityMinimumSize

Specifies value to override global Proximity Min size on local entities

Type Quantity (p. 1422)

Read Only No

ProximitySizeFunctionSource

Specifies value to override global Number of cells across gap on local entities

Type ProximitySizeFunctionSources (p. 1422)

Read Only No

SizeFunctionMethod

Specifies which size function to use.

Available options:

Program controlled
Adaptive
Curvature and proximity
Curvature
Proximity
Fixed

Type UseAdvancedLocalSizeFunction (p. 1447)

Read Only No

MeshControlLocalBoundaryLayer

Specifies local boundary layer attributes.

Available options:

BoundaryLayerOption	Specifies the way heights of the boundary layers are determined.
FirstLayerHeight	Specifies the height of the first boundary layer.
FirstAspectRatio	Specifies the aspect ratio of the boundaries that are extruded from the boundary base.
TransitionRatio	Specifies the rate at which adjacent elements grow.

NumberOfLayers	Specifies the number of boundary layers.
MaximumLayers	Specifies the maximum number of boundary layers to be created.
GrowthRate	Specifies the value for Growth Rate.
AspectRatio	Specifies relative thickness of adjacent boundary layers.
MaximumThickness	Specifies the desired maximum thickness of the boundary layer.
BoundaryLayerAlgorithm	Specifies pre or post boundary layer algorithm to be used by the mesher.

Properties

AspectRatio

Specifies relative thickness of adjacent boundary layers.

Type double (p. 1378)

Read Only No

BoundaryLayerAlgorithm

Specifies pre or post boundary layer algorithm to be used by the mesher.

Available options:

Post
Pre

Type BoundaryLayerAlgorithm (p. 1360)

Read Only No

DefineBy

Specifies the way heights of the boundary layers are determined.

Available options:

SmoothTransition	Mesher ensures smooth rate of volume change using local tetrahedral element size.
TotalThickness	Creates constant boundary layers using the values of Number of Layers and Growth Rate.
FirstLayerThickness	Creates constant boundary layers using values of First Layer Height, Maximum Layers and Growth Rate.
FirstAspectRatio	Creates boundary layers using values of First Aspect Ratio, Maximum Layers and Growth Rate.
LastAspectRatio	Creates constant boundary layers using values of First Layer Height, Maximum Layers and Aspect Ratio.

Type BoundaryLayerOption (p. 1360)

Read Only No

FirstAspectRatio

Specifies the aspect ratio of the boundaries that are extruded from the boundary base.

Type double (p. 1378)

Read Only No

FirstLayerHeight

Specifies the height of the first boundary layer.

Type Quantity (p. 1422)

Read Only No

GrowthRate

Specifies the value for Growth Rate.

Type double (p. 1378)

Read Only No

Location

A common property for all sub meshing objects with a defined Geometry Filter

Type LocationSet (p. 1401)

Read Only No

MaximumLayers

Specifies the maximum number of boundary layers to be created.

Type int (p. 1394)

Read Only No

MaximumThickness

Specifies the desired maximum thickness of the boundary layer.

Type Quantity (p. 1422)

Read Only No

NumberOfLayers

Specifies the number of boundary layers.

Type int (p. 1394)

Read Only No

TransitionRatio

Specifies the rate at which adjacent elements grow.

Type double (p. 1378)

Read Only No

UseAutomaticallyDefinedLocation

A common toggle for all sub meshing objects with a Location property so that it can be automatically set. Currently only visible in Boundary Layer object.

Type bool (p. 1360)

Read Only No

MeshCrossSection

No details are provided for this entry.

Properties

Area

Area of the section

Type Quantity (p. 1422)

Read Only No

AreaMomentOfInertiaAboutYAxis

Moment of inertia about the y axis

Type Quantity (p. 1422)

Read Only No

AreaMomentOfInertiaAboutZAxis

Moment of inertia about the z axis

Type Quantity (p. 1422)

Read Only No

CentroidAbscissa

y coordinate of the centroid

Type Quantity (p. 1422)

Read Only No

CentroidOrdinate

z coordinate of the centroid

Type Quantity (p. 1422)

Read Only No

ProductOfInertiaAboutYZAxes

Product of inertia

Type Quantity (p. 1422)

Read Only No

SectionIntegrationOptions

No details are provided for this entry.

Type IntegrationOptions (p. 1395)

Read Only No

ShearCenterAbscissa

y coordinate of the shear center

Type Quantity (p. 1422)

Read Only No

ShearCenterOrdinate

z coordinate of the shear center

Type Quantity (p. 1422)

Read Only No

Torsion

Torsion constant

Type Quantity (p. 1422)

Read Only No

Warping

Warping constant

Type Quantity (p. 1422)

Read Only No

MeshDiagnostics

MeshDiagnostics specifies the settings used to evaluate mesh quality.

Available options:

GeometryLocation	Specifies the selected topology set or the entire model for which mesh diagnostics are evaluated during part-based Meshing.
MeshMetric	Specifies the metric type based on which mesh quality is evaluated.
NumNodes	Specifies number of nodes in the mesh.
NumElements	Specifies number of elements in the mesh.
MinMetric	Specifies the minimum value for chosen metric type.
MaxMetric	Specifies the maximum value for chosen metric type.
AverageMetric	Specifies the average value for chosen metric type.
StandardDeviation	Specifies the standard deviation value for chosen metric type.

Properties

AverageMetric

Specifies the average value for chosen metric type.

Type double (p. 1378)

Read Only No

Location

A common property for all sub meshing objects with a defined Geometry Filter

Type LocationSet (p. 1401)

Read Only No

MaxMetric

Specifies the maximum value for chosen metric type.

Type double (p. 1378)

Read Only No

MeshMetric

Specifies the metric type based on which mesh quality is evaluated.

Available options:

ElementQuality	A composite quality metric is computed ranging between 0 and 1.
AspectRatio	Aspect ratio is calculated based only on corner nodes of elements.
JacobianRatio	Jacobian ratio is computed for all elements except those with no midside nodes or perfectly centered midside nodes.
WarpingFactor	Warping factor is computed for quadrilateral shell elements and quadrilateral faces of bricks, wedges and pyramids.
ParallelDeviation	Parallel deviation is computed ignoring midside nodes and based on unit vectors along each element size.
MaximumCornerAngle	Maximum corner angles are computed between adjacent edges and high corner angles can degrade element performance.
Skewness	Skewness is one primary quality measure that determines how close to ideal (that is, equilateral or equiangular) a face or cell is.
OrthogonalQuality	Orthogonal quality is computed using face normal vectors and ranges from 0 to 1.
None	No mesh metric type is selected.

Type MeshMetric (p. 1403)

Read Only No

MinMetric

Specifies the minimum value for chosen metric type.

Type double (p. 1378)

Read Only No

NumElements

Specifies number of elements in the mesh.

Type int (p. 1394)

Read Only No

NumNodes

Specifies number of nodes in the mesh.

Type int (p. 1394)

Read Only No

StandardDeviation

Specifies the standard deviation value for chosen metric type.

Type double (p. 1378)

Read Only No

Methods

EvaluateMeshDiagnostics

Evaluates the output mesh metrics for a given Mesh Diagnostics control.

MeshDisplayProperties

Base class for display state objects with reference Ids or reference names.

Properties

No Properties.

MeshDisplayPropertiesWithIds

Object returned by mesh display state queries with reference Ids and their corresponding display state properties.

Available properties:

State	Meshing state for entity. [0 = Unmeshed, 1 = Meshed, 2 = OutOfDate, 3 = Failed, 4 = Suppressed, 5 = Connected, 6 = NeedsConnected]
MeshColor	Color with which the mesh element edges should be drawn by graphics.
Translucency	Translucency value with which the mesh element faces should be drawn by graphics.
ShowMesh	Flag to tell graphics whether to overlay the mesh on the entity or not.
ReferenceIds	A list of entity reference Ids that are associated with a given meshing state.

Properties

No Properties.

MeshDisplayPropertiesWithStrings

Object returned by mesh display state queries with reference names and their corresponding display state properties.

Available properties:

State	Meshing state for entity. [0 = Unmeshed, 1 = Meshed, 2 = OutOfDate, 3 = Failed, 4 = Suppressed, 5 = Connected, 6 = NeedsConnected]
MeshColor	Color with which the mesh element edges should be drawn by graphics.
Translucency	Translucency value with which the mesh element faces should be drawn by graphics.
ShowMesh	Flag to tell graphics whether to overlay the mesh on the entity or not.
ReferenceNames	A list of entity reference names that are associated with a given meshing state.

Properties

No Properties.

Meshing

Meshing Task is used to determine (in a smart way) which meshing technology to use for meshing the model. A mesh can be generated by either (1) selectively meshing parts or entire assemblies of parts, or (2) filling in the volume of a surface mesh created by a Wrap task.

Available options:

EngineeringIntent	Defines the kind of physics that is being studied in this simulation process (used in part-based Meshing).

Properties

DeriveZoneTypeFromName

Used only when there is a connection from AIM Mesh Modeling component to Fluent Setup - part mesh file tranfer

Type bool (p. 1360)

Read Only No

EngineeringIntent

Defines the kind of physics that is being studied in this simulation process (used in part-based Meshing).

Available options:

StructuralOrThermal	Intent of this process is a structural and/or thermal simulation.
FluidFlow	Intent of this process is a fluid flow or fluid solid conduction simulation.

Type EngineeringIntent (p. 1381)

Read Only No

MeshingRetries

Will tell the mesher to try again with finer sizing inputs if the first attempt at meshing fails.

Type bool (p. 1360)

Read Only No

MeshResolution

Defines the resolution for automatically calculating global meshing inputs. Selecting a higher resolution generates a better quality mesh.

Type int (p. 1394)

Read Only No

ThinStructures

Specifies to use defaults for models with shells

Type bool (p. 1360)

Read Only No

UsePredefinedSettings

Automatically calculates the global sizing values based on the resolution value.

Type bool (p. 1360)

Read Only No

MeshModeling

Main MeshModeling Task user object

Properties

DeriveZoneTypeFromName

Used only when there is a connection from AIM Mesh Modeling component to Fluent Setup - part mesh file transfer

Type bool (p. 1360)

Read Only No

MeshSectionCell

No details are provided for this entry.

Properties

No Properties.

MeshSectionNode

No details are provided for this entry.

Properties

No Properties.

Modeling

Main Modeling Task user object

Properties

No Properties.

Methods

ChooseGeometryImport

Browses for the geometry import file path and sets it for a Modeling Task.

Required Arguments

FilePath Path to the chosen file.

Type string (p. 1438)

ResyncGeometry

Forces a resynchronization of the AIM model with the SC document.

SetImportPath

Sets the import file path on a Modeling Task.

Required Arguments

FilePath No details are provided for this entry.

Type string (p. 1438)

ModelingImportSource

Defines an imported modeling source associated with the Modeling Task.

Properties

ActiveCADAttachSource

The Active CAD Source chosen from the option list

Type string (p. 1438)

Read Only No

FilePath

Import source location.

Type string (p. 1438)

Read Only No

LineModelTypes

No details are provided for this entry.

Type LineModelType (p. 1399)

Read Only No

MaterialAssignments

Material assignments brought in by this import.

Type List (p. 1400)<MaterialAssignment (p. 1402)>

Read Only No

PlugInName

Displays the current plugin name when the source has been defined by an active CAD attach.

Type string (p. 1438)

Read Only Yes

SharedEdgesAmongBeams

No details are provided for this entry.

Type bool (p. 1360)

Read Only No

SourceSelectionMode

Controls how the source selection if defined.

Selection Types:

BrowseSource	Source is specified using the file browser dialog.
ActiveCADAttach	Source is specified by an active CAD attach source.

Type SourceSelectionType (p. 1434)

Read Only No

SourceType

Import source type.

Type string (p. 1438)

Read Only Yes

ModelsElasticity

Elasticity of the active model.

Properties

ModelType

Elasticity options displayed in UI

Type ElasticityType (p. 1380)

Read Only No

ModelsPlasticity

Plasticity of the active model.

Properties

As

Plasticity options displayed in UI

Type PlasticityType (p. 1417)

Read Only No

MolarMass

The Molar Mass model.

Can be applied to Fluid and Thermal analyses

Properties

Magnitude

Gets or sets the magnitude.

Type Expression (p. 1384)<Quantity (p. 1422)>

Read Only No

MonitorChart

Class containing a list of monitors to be displayed in a single chart.

Properties

No Properties.

MoveRotateControl

The Configuration control to move or rotate the geometry to a new Reference Frame.

Properties

PartSelections

Part selection for suppression

Type LocationSet (p. 1401)

Read Only No

RelativeTo

The target Reference Frame.

Type ReferenceFrame (p. 1424)

Read Only No

Methods

GenerateConfigureControlCommand

Generates the output for a specific Configure Control.

MultipleTableEditor

Class that edits multiple tables.

Properties

ReferenceValue

A dictionary of reference values.

Type Dictionary (p. 1375)<string (p. 1438), Object (p. 1409)>

Read Only No

Table

A Table Expression.

Type Expression (p. 1384)<Quantity (p. 1422)>

Read Only No

NeoHookeanHyperElasticity

The Neo-Hookean Hyperelasticity model.

Can be applied to Structural analyses

Properties

Magnitude

The Magnitude.

Type Expression (p. 1384)<NeoHookeanHyperElasticityDependents (p. 1408)>

Read Only No

OriginAndOrientationDefinitionMethod

Defines a reference frame from a local origin and 2 direction vectors

Properties

No Properties.

OrthotropicElasticity

The Orthotropic Elasticity model.

Can be applied to Structural analyses

Properties

Magnitude

The Magnitude.

Type Expression (p. 1384)<OrthotropicElasticityDependents (p. 1411)>

Read Only No

PhysicsCoupling

Provides the definition of a Physics Coupling Interface.

Properties

PhysicsRegion1

The source Physics Region that sends data via the Physics Coupling Interface.

Type PhysicsRegion (p. 1415)

Read Only No

PhysicsRegion2

The target Physics Region that receives data via the Physics Coupling Interface.

Type PhysicsRegion (p. 1415)

Read Only No

Side1Location

The source locations for the Physics Coupling Interface. This location can be set to AllCouplingSourceFaces(), AllCouplingSourceBodies(), or specific boundary conditions or selection sets from the Source solution.

Type string (p. 1438)

Read Only No

Side2Locations

The target locations that receive data via the Physics Coupling Interface.

Type LocationSet (p. 1401)

Read Only No

SourceFrequency

The source frequency for the Physics Coupling Interface.

Type string (p. 1438)

Read Only No

PhysicsDefinition

Represents a physics solution.

Properties

CalculationType

The calculation type for the solution.

Type CalculationType (p. 1361)

Read Only No

PhysicsFidelity

Defines the resolution for automatically calculated solver inputs. Selecting a higher fidelity gives better accuracy.

Type int (p. 1394)

Read Only No

PreStressedModal

Switch to include pre-stressed effect in a modal analysis.

Type bool (p. 1360)

Read Only No

RandomVibration

Switch to include random vibration in an analysis.

Type bool (p. 1360)

Read Only No

Methods

EvaluateVolumeResult

Generates voxel result data for the given variable(s).

Optional Arguments

VolumeDataInputs The variables to be evaluated.

Type List (p. 1400)<VolumeDataInput (p. 1450)>

GetAllBoundaryConditions

Provides a list of all Boundary Conditions in this entity. (Deprecated; use GetAllConditionsQuery instead)

Return A list of DataReferences containing the requested Boundary Conditions.

Type DataReferenceSet (p. 1371)

GetAllConditions

Provides a list of all (Boundary) Conditions in this entity.

Return A list of DataReferences containing the requested (Boundary) Conditions.

Type DataReferenceSet (p. 1371)

Optional Arguments

SelectedLocationSet A list of Locations to filter the boundary conditions on.

Type LocationSet (p. 1401)

GetAllContacts

A query to return all Contact objects in the Solution.

Return A list of DataReferences containing the requested Contacts.

Type DataReferenceSet (p. 1371)

GetAllObjects

GetAllObjects may be used to get all of the objects for the given search criteria.

Return A DataReferenceSet of all the Parent children objects of the given Type.

Type DataReferenceSet (p. 1371)

Optional Arguments

SearchLocation The criteria to use to do the match for the SearchString. This can be Exact, BeginsWith, Contains, or EndsWith and the default is Exact.

Type SearchPosition (p. 1430)

Default Value Exact

SearchString A search string to filter the return Components for the requested Type.

Type string (p. 1438)

Example

This example displays all of the objects in a default solution.

```
system1 = GetSystem(Name="Study")
```

```
solutionComponent1 = Study.CreateTask(
   Name="Solution",
   System=system1)
study1 = system1.GetContainer(ComponentName="Study")
solution1 = study1.GetSolution(Name="Solution")
for item in solution1.GetAllObjects():
   print item
```

GetAllPhysicsRegions

A query to return all Physics Regions in the Solution.

Return A list of DataReferences containing the requested Physics Regions.

Type DataReferenceSet (p. 1371)

GetBodiesOfUnknownPhysicsRegion

Gets a list of bodies that are not associated with a physics region.

Return A list of bodies that are not associated with any physics regions.

Type List (p. 1400)<string (p. 1438)>

Required Arguments

Model The DataReference of the model to lookup the bodies in.

Type DataReference (p. 1371)

Optional Arguments

ExcludeSurfaceBodies Should exclude surface bodies

Type bool (p. 1360)

GetBodiesOfUnknownPhysicsRegionAndTotalBodyCount

Gets a list of bodies that are not associated with a physics region and also the total number of bodies.

Return Total count of bodies and a list of bodies that are not associated with any physics regions.

Type Dictionary (p. 1375)<long (p. 1394), List (p. 1400)<string (p. 1438)>>

Required Arguments

Model The DataReference of the model to lookup the bodies in.

Type DataReference (p. 1371)

GetDuplicateMaterialAssignments

Provides the list of overlapping Material Assignments given an active Model.

Return A DataReferenceSet of the duplicate material assignments.

Type DataReferenceSet (p. 1371)

Required Arguments

ActiveModel The DataReference of the model of interest.

Type DataReference (p. 1371)

ReferenceTopologyType The ReferenceTopologyType of interest.

Type ReferenceTopologyType (p. 1424)

GetMaterialAssignments

A query to return all Material Assignments in the Analysis.

Return A list of DataReferences with the requested Material Assignments.

Type DataReferenceSet (p. 1371)

GetTopologyWithoutMaterial

Locates the topologies (represented by their reference strings) that currently don't have a material assignment given an active Model.

Return A list of topology reference strings containing the names of the found bodies.

Type List (p. 1400)<string (p. 1438)>

Required Arguments

ActiveModel The DataReference of the model of interest.

Type DataReference (p. 1371)

ReferenceTopologyType The ReferenceTopologyType of interest.

Type ReferenceTopologyType (p. 1424)

WriteSolverInputFile

No details are provided for this entry.

Return Flag whether query is successful.

Type string (p. 1438)

PhysicsOptions

The common set of Physics Options.

Properties

No Properties.

PhysicsRegion

Specifies the type and location of the fundamental physics in the simulation.

Properties

Location

The location of the Physics Region.

Type LocationSet (p. 1401)

Read Only No

PhysicsType

The physics type for the physics region.

Type PhysicsType (p. 1415)

Read Only No

Methods

GetAllBoundaryConditions

Provides a list of all Boundary Conditions in this entity. (Deprecated; use GetAllConditionsQuery instead)

Return A list of DataReferences containing the requested Boundary Conditions.

Type DataReferenceSet (p. 1371)

GetAllConditions

Provides a list of all (Boundary) Conditions in this entity.

Return A list of DataReferences containing the requested (Boundary) Conditions.

Type DataReferenceSet (p. 1371)

Optional Arguments

SelectedLocationSet A list of Locations to filter the boundary conditions on.

Type LocationSet (p. 1401)

GetDuplicateMaterialAssignments

Provides the list of overlapping Material Assignments given an active Model.

Return A DataReferenceSet of the duplicate material assignments.

Type DataReferenceSet (p. 1371)

Required Arguments

ActiveModel The DataReference of the model of interest.

Type DataReference (p. 1371)

ReferenceTopologyType The ReferenceTopologyType of interest.

Type ReferenceTopologyType (p. 1424)

GetMaterialAssignments

A query to return all Material Assignments in the Analysis.

Return A list of DataReferences with the requested Material Assignments.

Type DataReferenceSet (p. 1371)

GetTopologyWithoutMaterial

Locates the topologies (represented by their reference strings) that currently don't have a material assignment given an active Model.

Return A list of topology reference strings containing the names of the found bodies.

Type List (p. 1400)<string (p. 1438)>

Required Arguments

ActiveModel The DataReference of the model of interest.

Type DataReference (p. 1371)

ReferenceTopologyType The ReferenceTopologyType of interest.

Type ReferenceTopologyType (p. 1424)

Plane

The analytic definition of a flat plane where the Z axis is the normal direction.

Properties

ConstructionMethod

Method for constructing the plane.

Type PlaneConstructionMethod (p. 1416)

Read Only No

ScopedBodies

Bodies which this plane is scoped to.

Type LocationSet (p. 1401)

Read Only No

Transform

Plane transform.

Type Transform4x4 (p. 1443)

Read Only Yes

PlaneFromPlaneDefinition

Definition type for defining a plane from another plane.

Properties

BasePlane

Plane to define from.

Type Plane (p. 1416)

Read Only No

PlaneFromReferenceFrameDefinition

Definition type for defining a plane from a reference frame.

Properties

Plane

The reference frame axes used to define the plane.

Type PlaneOption (p. 1417)

Read Only No

ReferenceFrame

Reference Frame to define from.

Type ReferenceFrame (p. 1424)

Read Only No

PlaneOriginAndOrientationDefinition

Definition type for defining a plane from an origin and 2 axes.

Properties

No Properties.

PlaneTransformation

Extension for plane offset and rotation

Properties

RotationAngle

Rotation angle about the rotation axis

Type Expression (p. 1384)<Quantity (p. 1422)>

Read Only No

Suppress

Flag to suppress the transformation

Type bool (p. 1360)

Read Only No

Point

Point represented by local coordinates relative to a reference frame.

Properties

CalculationMethod

Calculation method for defining the point from geometry.

Type PointCalculationMethod (p. 1417)

Read Only No

DefineBy

Method for defining the point.

Type CoordinateInputMethod (p. 1369)

Read Only No

Location

Location to define the point.

Type LocationSet (p. 1401)

Read Only No

RelativeTo

Frame of reference for the point.

Type ReferenceFrame (p. 1424)

Read Only No

PointDefinition

Collection of properties to define a UniversalPoint

Properties

CalculationMethod

Calculation method for defining the point from geometry

Type PointCalculationMethod (p. 1417)

Read Only No

DefineBy

Method for defining the point

Type CoordinateInputMethod (p. 1369)

Read Only No

Location

Location to define the point

Type LocationSet (p. 1401)

Read Only No

RelativeTo

Frame of reference for the point

Type ReferenceFrame (p. 1424)

Read Only No

PowerFerriteCoreLoss

Core Loss Power Ferrite material model.

Properties

CM

Gets or sets the CM.

Type Expression (p. 1384)<Quantity (p. 1422)>

Read Only No

X

Gets or sets the X.

Type Expression (p. 1384)<Quantity (p. 1422)>

Read Only No

Y

Gets or sets the Y.

Type Expression (p. 1384)<Quantity (p. 1422)>

Read Only No

PrimitiveBox

This class represents a primitive box component

Properties

CreateSelectionSets

Flag to enable/disable selection set creation

Type bool (p. 1360)

Read Only No

PrimitiveBoxConstraint

This class extends the PrimitiveBox class. Contains properties like Selection Method, Cushioning, Location etc.

Properties

CushionXminus

Gets or sets the -X cushion value for non-uniform cushion.

Type Expression (p. 1384)<Quantity (p. 1422)>

Read Only No

CushionXplus

Gets or sets the +X cushion value for non-uniform cushion.

Type Expression (p. 1384)<Quantity (p. 1422)>

Read Only No

CushionYminus

Gets or sets the -Y cushion value for non-uniform cushion.

Type Expression (p. 1384)<Quantity (p. 1422)>

Read Only No

CushionYplus

Gets or sets the +Y cushion value for non-uniform cushion.

Type Expression (p. 1384)<Quantity (p. 1422)>

Read Only No

CushionZminus

Gets or sets the -Z cushion value for non-uniform cushion.

Type Expression (p. 1384)<Quantity (p. 1422)>

Read Only No

CushionZplus

Gets or sets the +Z cushion value for non-uniform cushion.

Type Expression (p. 1384)<Quantity (p. 1422)>

Read Only No

EntitiesInSelection

Gets or sets the entites to be enclosed.

Type LocationSet (p. 1401)

Read Only No

SelectionMethod

Gets or sets the options for choosing between box creation methods.

Type PrimitiveBoxCreationMethod (p. 1420)

Read Only No

TypeOfCushion

Gets or sets the option to select between Uniform or Non-Uniform cushion around box.

Type BoxCushionType (p. 1360)

Read Only No

UniformCushionValue

Gets or sets the uniform cushion value for uniform cushion.

Type Expression (p. 1384)<Quantity (p. 1422)>

Read Only No

QuadrilateralSection

This class represents quadrilateral beam cross section. It consists of four coordinates for the corners(i,j,k,l), each having an abscissa and ordinate.

Properties

Abscissa1

Abscissa 1

Type Quantity (p. 1422)

Read Only No

Abscissa2

Abscissa 2

Type Quantity (p. 1422)

Read Only No

Abscissa3

Abscissa 3

Type Quantity (p. 1422)

Read Only No

Abscissa4

Abscissa 4

Type Quantity (p. 1422)

Read Only No

Ordinate1

Ordinate 1

Type Quantity (p. 1422)

Read Only No

Ordinate2

Ordinate 2

Type Quantity (p. 1422)

Read Only No

Ordinate3

Ordinate 3

Type Quantity (p. 1422)

Read Only No

Ordinate4

Ordinate 4

Type Quantity (p. 1422)

Read Only No

RectangularSection

This class represents retangular beam cross section

Properties

Height

Overall height

Type Quantity (p. 1422)

Read Only No

Width

Overall width

Type Quantity (p. 1422)

Read Only No

ReferenceFrame

Frame of reference for modeling and simulation data.

Properties

ConfigurationTransform

Transform relative to the global reference frame without motion

Type Transform4x4 (p. 1443)

Read Only Yes

DefinitionType

Reference frame definition type.

Type ReferenceFrameDefinitionType (p. 1424)

Read Only No

EnablePivot

Flag to Enable/Disable the GUI "Pivot".

Type bool (p. 1360)

Read Only No

ParentFrame

Parent reference frame.

Type ReferenceFrame (p. 1424)

Read Only No

PreferredCoordinateType

Default coordinate type for points/vectors in this reference frame.

Type TripletType (p. 1445)

Read Only No

ResultObject

Used to define and evaluate a quantitative result in an Analysis.

Properties

CalculateAverage

Set to true if averages are to be calculated as part of the Result evaluation.

Type bool (p. 1360)

Read Only No

EvaluateFullRange

Whether or not to evaluate all time points in the evaluation

Type bool (p. 1360)

Read Only No

LastAutomaticDisplayText

The last automatically-generated DisplayText assigned to this Result Object

Type string (p. 1438)

Read Only No

Location

The Location over which the Result is to be evaluated.

Type LocationSet (p. 1401)

Read Only No

RelativeTo

The Reference Frame which the result is relative to

Type ReferenceFrame (p. 1424)

Read Only No

ResultName

Name of the Result.

Type string (p. 1438)

Read Only No

SimulationStep

The Simulation Step object reference for a multi-step analysis.

Type SolutionStep (p. 1434)

Read Only No

Variable

The solution variable to be displayed

Type Expression (p. 1384)<Quantity (p. 1422)>

Read Only No

Methods

Evaluate

Evaluates the Result's calculated and displayed values based on the current solution.

Results

The top-level object for a Results task.

Properties

AnimationDuration

The length of time an animation takes to play

Type Quantity (p. 1422)

Read Only No

CustomScaling

Deformation scaling setting for custom scaling

Type double (p. 1378)

Read Only No

Scaling

How much to deform the Results display

Type DeformationScaling (p. 1373)

Read Only No

ResultSetParametersExtension

Parameters for Results

Properties

Mode

The mode number for a Modal analysis.

Type int (p. 1394)

Read Only No

PhaseAngle

The Phase Angle representation of a Complex result.

Type double (p. 1378)

Read Only No

Substep

The substep number within the simulation step.

Type int (p. 1394)

Read Only No

ReverseAboutXTransform

Rotates about the X axis by 180 degrees.

Properties

Suppress

Suppress/Unsuppress the transform

Type bool (p. 1360)

Read Only No

ReverseAboutYTransform

Rotates about the Y axis by 180 degrees

Properties

Suppress

Suppress/Unsuppress the transform

Type bool (p. 1360)

Read Only No

ReverseAboutZTransform

Rotates about the Z axis by 180 degrees

Properties

Suppress

Suppress/Unsuppress the transform

Type bool (p. 1360)

Read Only No

RotationTransform

Rotates the matrix about a vector relative to the local reference frame.

Properties

RelativeToType

Define the axis direction relative to Local or Parent.

Type RelativeToType (p. 1425)

Read Only No

RotationAngle

Angle to rotate by.

Type Expression (p. 1384)<Quantity (p. 1422)>

Read Only No

Suppress

Suppress/Unsuppress the transform

Type bool (p. 1360)

Read Only No

SectionLayer

SectionLayer is a class that represents various section properties as available to a given layer. In general composite shell and beam elements can have multiple layers and each layer can have its own orientation, thickness and offset.

Properties

Offset

The offset within the stack up

Type Quantity (p. 1422)

Read Only No

RelativeTo

Orientation with respect to the element reference frame

Type ReferenceFrame (p. 1424)

Read Only No

Thickness

Thickness

Type Quantity (p. 1422)

Read Only No

SelectionSet

An object that can be used to reference any Location.

Properties

Location

The selected location: evaluates to a set of reference ids. See LocationSet for more detail.

Type LocationSet (p. 1401)

Read Only No

Methods

GetSelectionSetSelectableEntityCount

Returns the number of selectable entities of the specified SelectionSet.

Return The number of selectable entities of the specified SelectionSet.

Type uint (p. 1446)

Shape

The top-level object for a Shape task.

Properties

DensityThreshold

The density threshold for the shape.

Type double (p. 1378)

Read Only No

ResultName

The name of the result.

Type string (p. 1438)

Read Only No

ShapeAdjustment

The level of shape adjustment set on a range.

Type Expression (p. 1384)<Quantity (p. 1422)>

Read Only No

SmoothShape

A flag to turn on/off optimized topological smoothing of a shape.

Type bool (p. 1360)

Read Only No

Methods

ExportNewModel

Exports a Model updated with changes made by STL scripts.

ShapeTaskStatistics

Class to optimize Shape statistics.

Properties

MassReduction

The mass removed from the shape after reduction.

Type Quantity (p. 1422)

Read Only Yes

OriginalMass

The shape's original mass.

Type Quantity (p. 1422)

Read Only Yes

OriginalVolume

The shape's original volume.

Type Quantity (p. 1422)

Read Only Yes

ReducedMass

The shape's mass after reduction.

Type Quantity (p. 1422)

Read Only Yes

ReducedVolume

The shape's volume after reduction.

Type Quantity (p. 1422)

Read Only Yes

VolumeReduction

The volume removed from the shape after reduction.

Type Quantity (p. 1422)

Read Only Yes

ShellThickness

ShellThickness is a class that holds thickness and offset information of surface bodies.

Properties

IsImportedThickness

Flag if the thickness and offset information are imported data from a modeler as opposed to user input data.

Type bool (p. 1360)

Read Only No

Location

Defined locations for the object.

Type LocationSet (p. 1401)

Read Only No

Offset

Input user offset value when the Thickness distribution is set to User defined offset.

Type Quantity (p. 1422)

Read Only No

ReverseOffsetDirection

Switch to reverse the offset direction.

Type bool (p. 1360)

Read Only No

Thickness

Total thickness

Type Expression (p. 1384)<Quantity (p. 1422)>

Read Only No

ThicknessDistribution

Offset of the shell section, also the location of the nodes in the section.

Type SectionOffsetType (p. 1430)

Read Only No

Simulation

The top-level object that manages all data in a simulation study. Depends on the Reference Manager ,Configuration Manager and ETRM

Properties

No Properties.

Methods

GetAllModels

A query to return all available models in the Simulation.

Return A list of DataReferences containing the requested Models.

Type DataReferenceSet (p. 1371)

GetObjectsForReference

A query to get all data objects in the simulation which reference the topological entity with the given reference ID.

Return The list of data objects that references the given input Reference string.

Type List (p. 1400)<DataReference (p. 1371)>

Required Arguments

Reference The toplogical entities reference string such as "BODY1", "FACE16", etc

Type string (p. 1438)

GetPhysicsRegionForSelection

Returns the Physics Regions that contain any of the specified journal reference strings.

Return The Physics Regions whose location contains the selection.

Type List (p. 1400)<DataReference (p. 1371)>

Required Arguments

GeometryStrings The reference strings of the geometry selection of interest.

Type List (p. 1400)<string (p. 1438)>

SingleValueResult

A result object that can be used to evaluate a single calculated value.

Properties

CalculateAverage

Set to true if averages are to be calculated as part of the Result evaluation.

Type bool (p. 1360)

Read Only No

Condition

The boolean condition used in CountIf and SumIf expressions

Type string (p. 1438)

Read Only No

EvaluateFullRange

Whether or not to evaluate all time points in the evaluation

Type bool (p. 1360)

Read Only No

Expression

An expression to be calculated.

Type Expression (p. 1384)<Quantity (p. 1422)>

Read Only No

Function

The function used to calculate the value.

Type string (p. 1438)

Read Only No

LastAutomaticDisplayText

The last automatically-generated DisplayText assigned to this Result Object

Type string (p. 1438)

Read Only No

Location

The Location over which the Result is to be evaluated.

Type LocationSet (p. 1401)

Read Only No

MeshEntity

Specifies the type of mesh location (node, element, face, edge) to be used to calculate the value.

Type EntityType (p. 1381)

Read Only No

Method

The method by which the value is calculated.

Type Method (p. 1405)

Read Only No

RelativeTo

The Reference Frame which the result is relative to

Type ReferenceFrame (p. 1424)

Read Only No

ResultName

Name of the Result.

Type string (p. 1438)

Read Only No

SimulationStep

The Simulation Step object reference for a multi-step analysis.

Type SolutionStep (p. 1434)

Read Only No

Value

The current value of the result.

Type Expression (p. 1384)<Quantity (p. 1422)>

Read Only No

Variable

The solution variable to be displayed

Type Expression (p. 1384)<Quantity (p. 1422)>

Read Only No

WeightType

The weight method used to calculate an average value.

Type WeightingType (p. 1451)

Read Only No

SolutionProgression

SolutionProgression is a class that specifes convergence, stabilization of solution etc, the typical attributes for monitoring and controlling a solution.

Properties

MaxNumberDesignIterations

Sets the maximum number of design iteration in optimization process

Type int (p. 1394)

Read Only No

SolutionQualityMonitor

Solution quality monitor objects are used to present graphical charts tracking aspects of the solution quality, such as convergence.

Properties

Component

For non-scalar Solution Quality Metrics, defines the component

Type string (p. 1438)

Read Only No

Location

For scoped Solution Quality Metrics, defines the location

Type LocationSet (p. 1401)

Read Only No

MonitorTypeName

Gives a type of Solution Quality Metric

Type string (p. 1438)

Read Only No

SubCategory

Gives the name of a set of fields (assembly of fields of same nature : for contact, for thermal, ..)

Type string (p. 1438)

Read Only No

SolutionStep

Solution Step

Properties

SolutionProgression

It displays the solution progression associated with this solution step

Type SolutionProgression (p. 1433)

Read Only No

SolutionVariableMonitor

Solution variable monitor objects are used to present graphical charts tracking aspects of the solution results, such as displacements.

Properties

Component

Gives the component of the field

Type string (p. 1438)

Read Only No

Location

The Location over which the monitor is to be evaluated

Type LocationSet (p. 1401)

Read Only No

MonitorTypeName

Gives the name of a field

Type string (p. 1438)

Read Only No

SubCategory

Gives the name of a set of fields (assembly of fields of same nature : for contact, for thermal, ..)

Type string (p. 1438)

Read Only No

SolverSettings

The common set of solver settings.

Properties

No Properties.

SpecificHeatConstantPressure

The specific heat at constant pressure model provided via a specified expression.

Properties

Magnitude

The Magnitude is an expresion which evaluates to a Quantity with the correct units.

Type Expression (p. 1384)<Quantity (p. 1422)>

Read Only No

SphericalTriplet

Triplet represented in spherical coordinates

Properties

Phi

Angle in the XY plane counter-clockwise from the +X axis (a.k.a phi) phi = arctan(y / x)

Type Expression (p. 1384)<Quantity (p. 1422)>

Read Only No

R

Distance from the origin r = sqrt(x^2 + y^2 + z^2)

Type Expression (p. 1384)<Quantity (p. 1422)>

Read Only No

Theta

Angle between the +Z axis and the vector (a.k.a. polar angle, theta) theta = arccos(z / r)

Type Expression (p. 1384)<Quantity (p. 1422)>

Read Only No

StateOfMatterExtension

For the purpose of using generic in AddProperty need a base for the StateOfMatter

Properties

No Properties.

StatusGroup

Defines the behavior (suppression, ramping, factor, locked) for a collection of UserObjects

Properties

No Properties.

StepControlExtension

StepControlExtension is a class to represents controls for Simulation

Properties

NumberOfSteps

The number of simulation steps for the solution.

Type int (p. 1394)

Read Only Yes

StrainLifeParameters

The Strain Life Properties material model.

Properties

CyclicStrainHardeningExponent

Cyclic Strain Hardening Exponent

Type Expression (p. 1384)<Quantity (p. 1422)>

Read Only No

CyclicStrengthCoefficient

Cyclic Strength Coefficient

Type Expression (p. 1384)<Quantity (p. 1422)>

Read Only No

DuctilityCoefficient

Fatigue Ductility Coefficient

Type Expression (p. 1384)<Quantity (p. 1422)>

Read Only No

DuctilityExponent

Fatigue Ductility Exponent

Type Expression (p. 1384)<Quantity (p. 1422)>

Read Only No

StrainLifeCurve

Strain Life Curve

Type Expression (p. 1384)<Quantity (p. 1422)>

Read Only Yes

StrengthCoefficient

Fatigue Strength Coefficient

Type Expression (p. 1384)<Quantity (p. 1422)>

Read Only No

StrengthExponent

Fatigue Strength Exponent (Basquin's Exponent)

Type Expression (p. 1384)<Quantity (p. 1422)>

Read Only No

StressStrainCurve

Stress-Strain Curve

Type Expression (p. 1384)<Quantity (p. 1422)>

Read Only Yes

StreamLine

A result that displays flow paths within a vector field.

Properties

Direction

The integration direction to create streamlines.

Type DirectionType (p. 1376)

Read Only No

Distribution

The method used to distribute seed points.

Type SeedPointDistributionType (p. 1430)

Read Only No

EndTime

The maximum integration time of streamlines.

Type Quantity (p. 1422)

Read Only No

EndTimeFactor

The amount my which the maximum integration time is scaled when creating animated streamlines.

Type float (p. 1432)

Read Only No

EvaluateFullRange

Whether or not to evaluate all time points in the evaluation

Type bool (p. 1360)

Read Only No

FadeLineSegments

Controls if line segments fade towards the end.

Type bool (p. 1360)

Read Only No

FixedStepSize

The size of integration steps in streamline creation when using a fixed step size.

Type float (p. 1432)

Read Only No

LastAutomaticDisplayText

The last automatically-generated DisplayText assigned to this Result Object

Type string (p. 1438)

Read Only No

Location

The Location over which the Result is to be evaluated.

Type LocationSet (p. 1401)

Read Only No

MeshDistribution

Defines how the mesh is sampled to create seed points.

Type SeedPointMeshDistributionType (p. 1430)

Read Only No

NSteps

The maximum number of integration steps in streamline creation.

Type uint (p. 1446)

Read Only No

ResultName

Result name

Type string (p. 1438)

Read Only No

RibbonWidth

The width of streamline ribbon display.

Type double (p. 1378)

Read Only No

SegmentRatio

The length of segments when displaying dashed streamlines.

Type float (p. 1432)

Read Only No

SkipFactor

How many nodes or elements are skipped when sampling mesh to generate seed points.

Type uint (p. 1446)

Read Only No

StepSize

The method used to determine the size of integration steps in streamline creation.

Type StepSizeType (p. 1436)

Read Only No

StreamLineColorOption

How the streamline is colored.

Type StreamlineColorOption (p. 1437)

Read Only No

StreamLineDisplayStyle

Controls the display style of the streamlines.

Type StreamLineDisplayStyle (p. 1437)

Read Only No

StreamLineShape

The line display style for the streamlines.

Type StreamLineShapeDef (p. 1437)

Read Only No

StreamLineStyle

THe streamline line style.

Type StreamLineStyle (p. 1437)

Read Only No

SymbolScaleFactor

The size factor of streamline symbols.

Type double (p. 1378)

Read Only No

SymbolSegmentDensity

The number of symbols and segments per streamline

Type int (p. 1394)

Read Only No

SymbolShape

The symbol used on streamline display.

Type SymbolShape (p. 1440)

Read Only No

TubeDiameter

The diameter of streamline tube display.

Type double (p. 1378)

Read Only No

UniformDistributionSize

The target number of seed points.

Type uint (p. 1446)

Read Only No

Variable

The vector variable used to create the streamlines.

Type Expression (p. 1384)<Quantity (p. 1422)>

Read Only Yes

VolumetricGridSpacing

The spacing of uniform grid sampling.

Type Quantity (p. 1422)

Read Only No

WireThickness

The thickness of streamline wire display.

Type double (p. 1378)

Read Only No

SummaryParametersExtension

Summary of Result values

Properties

Average

The calculated average of the Quantity of interest.

Type Expression (p. 1384)<Quantity (p. 1422)>

Read Only Yes

Max

The calculated maximum value of the Quantity in this Result.

Type Expression (p. 1384)<Quantity (p. 1422)>

Read Only Yes

Min

The calculated minimum value of the Quantity in this Result.

Type Expression (p. 1384)<Quantity (p. 1422)>

Read Only Yes

Sum

The calculated Sum total of the Quantity being evaluated in this Result.

Type Expression (p. 1384)<Quantity (p. 1422)>

Read Only Yes

SumX

The calculated Sum total of the X-component of the Quantity evaluated in this Result.

Type Expression (p. 1384)<Quantity (p. 1422)>

Read Only Yes

SumY

The calculated Sum total of the Y-component of the Quantity evaluated in this Result.

Type Expression (p. 1384)<Quantity (p. 1422)>

Read Only Yes

SumZ

The calculated Sum total of the Z-component of the Quantity evaluated in this Result.

Type Expression (p. 1384)<Quantity (p. 1422)>

Read Only Yes

SuppressControl

Settings for the suppression of geometry or mesh in a Configuration.

Properties

PartSelections

Part selection for suppression

Type LocationSet (p. 1401)

Read Only No

Methods

GenerateConfigureControlCommand

Generates the output for a specific Configure Control.

SutherlandThreeCoefficientDefinition

The three coefficient model definition.

Properties

DisplayText

The general property that defines the user-visible name of an entity. This property is defined for all data entities but is used only in those entities that present a label in the user interface.

Type string (p. 1438)

Read Only No

EffectiveTemperature

Gets or sets the effective temperature.

Type Expression (p. 1384)<Quantity (p. 1422)>

Read Only No

ReferenceTemperature

Gets or sets the reference temperature.

Type Expression (p. 1384)<Quantity (p. 1422)>

Read Only No

ReferenceViscosity

Gets or sets the reference viscosity.

Type Expression (p. 1384)<Quantity (p. 1422)>

Read Only No

SutherlandTwoCoefficientDefinition

The two coefficient model definition

Properties

CoefficientC1

Gets or sets the coefficient c1.

Type Expression (p. 1384)<Quantity (p. 1422)>

Read Only No

CoefficientC2

Gets or sets the coefficient c2.

Type Expression (p. 1384)<Quantity (p. 1422)>

Read Only No

DisplayText

The general property that defines the user-visible name of an entity. This property is defined for all data entities but is used only in those entities that present a label in the user interface.

Type string (p. 1438)

Read Only No

TableResult

Table object that displays result data.

Properties

Data

Data to populate results table.

Type List (p. 1400)<List (p. 1400)<Quantity (p. 1422)>>

Read Only Yes

EvaluateFullRange

Whether or not to evaluate all time points in the evaluation

Type bool (p. 1360)

Read Only No

LastAutomaticDisplayText

The last automatically-generated DisplayText assigned to this Result Object.

Type string (p. 1438)

Read Only No

TensileUltimateStrength

The Tensile Ultimate Strength model.

Can be applied to Structural analyses

Properties

Magnitude

Gets or sets the magnitude.

Type Expression (p. 1384)<Quantity (p. 1422)>

Read Only No

TensileYieldStrength

The Tensile Yield Strength model.

Can be applied to Structural analyses

Properties

Magnitude

Gets or sets the magnitude.

Type Expression (p. 1384)<Quantity (p. 1422)>

Read Only No

ThreeCoefficientDefinition

The three coefficient definition

Properties

DisplayText

The general property that defines the user-visible name of an entity. This property is defined for all data entities but is used only in those entities that present a label in the user interface.

Type string (p. 1438)

Read Only No

ReferenceTemperature

Gets or sets the reference temperature.

Type Expression (p. 1384)<Quantity (p. 1422)>

Read Only No

ReferenceViscosity

Gets or sets the reference viscosity.

Type Expression (p. 1384)<Quantity (p. 1422)>

Read Only No

TemperatureExponent

Gets or sets the temperature exponent.

Type Expression (p. 1384)<Quantity (p. 1422)>

Read Only No

TimeDependentParametersExtension

Time dependent parameters for post objects

Properties

Time

The time for a Transient analysis.

Type Expression (p. 1384)<Quantity (p. 1422)>

Read Only No

Transcript

Controls the contents of a Transcript monitor.

Properties

Filename

The name of the file to be monitored in a Transcript.

Type string (p. 1438)

Read Only No

SourceType

The data source for the Transcript monitor.

Type TranscriptType (p. 1443)

Read Only No

TranslationTransform

Translates a reference frame by a vector.

Properties

RelativeToType

Define the translation vector relative to Local or Parent. This is a simplification of the TranslationVector RelativeTo property

Type RelativeToType (p. 1425)

Read Only No

Suppress

Suppress/Unsuppress the transform

Type bool (p. 1360)

Read Only No

TSection

This class represents T beam cross section

Properties

FlangeThickness

Flange Thickness

Type Quantity (p. 1422)

Read Only No

FlangeWidth

Flange Width

Type Quantity (p. 1422)

Read Only No

OverallHeight

Overall height

Type Quantity (p. 1422)

Read Only No

StemThickness

Stem Thickness

Type Quantity (p. 1422)

Read Only No

TwoCoefficientDefinition

The two coefficient definition.

Properties

DimensionalCoefficientB

Gets or sets the dimensional coefficient B.

Type Expression (p. 1384)<Quantity (p. 1422)>

Read Only No

DisplayText

The general property that defines the user-visible name of an entity. This property is defined for all data entities but is used only in those entities that present a label in the user interface.

Type string (p. 1438)

Read Only No

TemperatureExponent

Gets or sets the temperature exponent.

Type Expression (p. 1384)<Quantity (p. 1422)>

Read Only No

UniversalTriplet

Representation of a point or vector in cartesian, cylindrical, and spherical coordinates

Properties

CoordinateType

The type of coordinate: Cartesian, Cylindrical, or Spherical

Type TripletType (p. 1445)

Read Only No

UniversalVectorDefinition

Collection of properties to define a UniversalVector

Properties

CalculationMethod

Calculation method for defining the vector from geometry

Type VectorCalculationMethod (p. 1449)

Read Only No

DefineBy

Method for defining the vector

Type CoordinateInputMethod (p. 1369)

Read Only No

Location

Location to define the vector

Type LocationSet (p. 1401)

Read Only No

Magnitude

Magnitude of the vector

Type Expression (p. 1384)<Quantity (p. 1422)>

Read Only No

RelativeTo

Frame of reference for the vector

Type ReferenceFrame (p. 1424)

Read Only No

ReverseDirection

Flag to reverse the vector direction when defining from Geometry

Type bool (p. 1360)

Read Only No

VectorDefineBy

Define the vector from magnitude/direction or components

Type UniversalVectorDefineBy (p. 1446)

Read Only No

VectorResult

A result object that displays the magnitude and direction of a vector field.

Properties

CalculateAverage

Set to true if averages are to be calculated as part of the Result evaluation.

Type bool (p. 1360)

Read Only No

Distribution

The method for defining vector seed points.

Type SeedPointDistributionType (p. 1430)

Read Only No

EvaluateFullRange

Whether or not to evaluate all time points in the evaluation

Type bool (p. 1360)

Read Only No

LastAutomaticDisplayText

The last automatically-generated DisplayText assigned to this Result Object

Type string (p. 1438)

Read Only No

Location

The Location over which the Result is to be evaluated.

Type LocationSet (p. 1401)

Read Only No

MeshDistribution

Defines how the mesh is sampled to create seed points.

Type SeedPointMeshDistributionType (p. 1430)

Read Only No

RelativeTo

The Reference Frame which the result is relative to

Type ReferenceFrame (p. 1424)

Read Only No

ResultName

Name of the Result.

Type string (p. 1438)

Read Only No

SimulationStep

The Simulation Step object reference for a multi-step analysis.

Type SolutionStep (p. 1434)

Read Only No

SkipFactor

How many nodes or elements are skipped when sampling mesh to generate seed points.

Type uint (p. 1446)

Read Only No

SymbolLength

Controls how symbol length is determined.

Type SymbolLengthOptions (p. 1440)

Read Only No

SymbolScaleFactor

The scale factor for symbol size.

Type double (p. 1378)

Read Only No

SymbolShape

The symbol to be used to display the vectors.

Type SymbolShape (p. 1440)

Read Only No

UniformDistributionSize

The approximate number of symbols to be displayed.

Type uint (p. 1446)

Read Only No

Variable

The solution variable to be displayed

Type Expression (p. 1384)<Quantity (p. 1422)>

Read Only No

VolumetricGridSpacing

The spacing of uniform grid sampling.

Type Quantity (p. 1422)

Read Only No

Viscosity

The viscosity model provided via a specified expression.

Properties

Magnitude

The Magnitude is an expresion which evaluates to a Quantity with the correct units.

Type Expression (p. 1384)<Quantity (p. 1422)>

Read Only No

ViscosityCarreauShearAndTemperatureDependent

The Shear Rate and Temperature dependent model.

Properties

ActivationEnergy

Gets or sets the activation energy.

Type Expression (p. 1384)<Quantity (p. 1422)>

Read Only No

DisplayText

The general property that defines the user-visible name of an entity. This property is defined for all data entities but is used only in those entities that present a label in the user interface.

Type string (p. 1438)

Read Only No

InfiniteShearViscosity

Gets or sets the infinite shear viscosity.

Type Expression (p. 1384)<Quantity (p. 1422)>

Read Only No

PowerLawIndex

Gets or sets the index of the power law.

Type Expression (p. 1384)<Quantity (p. 1422)>

Read Only No

ReferenceTemperature

Gets or sets the reference temperature.

Type Expression (p. 1384)<Quantity (p. 1422)>

Read Only No

TimeConstant

Gets or sets the time constant.

Type Expression (p. 1384)<Quantity (p. 1422)>

Read Only No

ZeroShearViscosity

Gets or sets the zero shear viscosity.

Type Expression (p. 1384)<Quantity (p. 1422)>

Read Only No

ViscosityCarreauShearRateDependent

The Shear Rate dependent model.

Properties

DisplayText

The general property that defines the user-visible name of an entity. This property is defined for all data entities but is used only in those entities that present a label in the user interface.

Type string (p. 1438)

Read Only No

InfiniteShearViscosity

Gets or sets the infinite shear viscosity.

Type Expression (p. 1384)<Quantity (p. 1422)>

Read Only No

PowerLawIndex

Gets or sets the index of the power law.

Type Expression (p. 1384)<Quantity (p. 1422)>

Read Only No

TimeConstant

Gets or sets the time constant.

Type Expression (p. 1384)<Quantity (p. 1422)>

Read Only No

ZeroShearViscosity

Gets or sets the zero shear viscosity.

Type Expression (p. 1384)<Quantity (p. 1422)>

Read Only No

ViscosityCrossShearAndTemperatureDependent

The Shear Rate and Temperature Dependent model.

Properties

ActivationEnergy

Gets or sets the activation energy.

Type Expression (p. 1384)<Quantity (p. 1422)>

Read Only No

DisplayText

The general property that defines the user-visible name of an entity. This property is defined for all data entities but is used only in those entities that present a label in the user interface.

Type string (p. 1438)

Read Only No

PowerLawIndex

Gets or sets the index of the power law.

Type Expression (p. 1384)<Quantity (p. 1422)>

Read Only No

ReferenceTemperature

Gets or sets the reference temperature.

Type Expression (p. 1384)<Quantity (p. 1422)>

Read Only No

TimeConstant

Gets or sets the time constant.

Type Expression (p. 1384)<Quantity (p. 1422)>

Read Only No

ZeroShearViscosity

Gets or sets the zero shear viscosity.

Type Expression (p. 1384)<Quantity (p. 1422)>

Read Only No

ViscosityCrossShearRateDependent

The Shear Rate Dependent model.

Properties

DisplayText

The general property that defines the user-visible name of an entity. This property is defined for all data entities but is used only in those entities that present a label in the user interface.

Type string (p. 1438)

Read Only No

PowerLawIndex

Gets or sets the index of the power law.

Type Expression (p. 1384)<Quantity (p. 1422)>

Read Only No

TimeConstant

Gets or sets the time constant.

Type Expression (p. 1384)<Quantity (p. 1422)>

Read Only No

ZeroShearViscosity

Gets or sets the zero shear viscosity.

Type Expression (p. 1384)<Quantity (p. 1422)>

Read Only No

ViscosityHerschelBulkleyShearAndTemperatureDependent

The Shear Rate and Temperature dependent model.

Properties

ActivationEnergy

Gets or sets the activation energy.

Type Expression (p. 1384)<Quantity (p. 1422)>

Read Only No

ConsistencyIndex

Gets or sets the index of the consistency.

Type Expression (p. 1384)<Quantity (p. 1422)>

Read Only No

CriticalShearRate

Gets or sets the critical shear rate.

Type Expression (p. 1384)<Quantity (p. 1422)>

Read Only No

DisplayText

The general property that defines the user-visible name of an entity. This property is defined for all data entities but is used only in those entities that present a label in the user interface.

Type string (p. 1438)

Read Only No

PowerLawIndex

Gets or sets the index of the power law.

Type Expression (p. 1384)<Quantity (p. 1422)>

Read Only No

ReferenceTemperature

Gets or sets the reference temperature.

Type Expression (p. 1384)<Quantity (p. 1422)>

Read Only No

YieldStressThreshold

Gets or sets the yield stress threshold.

Type Expression (p. 1384)<Quantity (p. 1422)>

Read Only No

ViscosityHerschelBulkleyShearRateDependent

The Shear Rate dependent model.

Properties

ConsistencyIndex

Gets or sets the index of the consistency.

Type Expression (p. 1384)<Quantity (p. 1422)>

Read Only No

CriticalShearRate

Gets or sets the critical shear rate.

Type Expression (p. 1384)<Quantity (p. 1422)>

Read Only No

DisplayText

The general property that defines the user-visible name of an entity. This property is defined for all data entities but is used only in those entities that present a label in the user interface.

Type string (p. 1438)

Read Only No

PowerLawIndex

Gets or sets the index of the power law.

Type Expression (p. 1384)<Quantity (p. 1422)>

Read Only No

YieldStressThreshold

Gets or sets the yield stress threshold.

Type Expression (p. 1384)<Quantity (p. 1422)>

Read Only No

ViscosityNonNewtonianPowerLawShearAndTemperatureDependent

The Shear Rate and Temperature dependent model.

Properties

ActivationEnergy

Gets or sets the activation energy.

Type Expression (p. 1384)<Quantity (p. 1422)>

Read Only No

ConsistencyIndex

Gets or sets the index of the consistency.

Type Expression (p. 1384)<Quantity (p. 1422)>

Read Only No

DisplayText

The general property that defines the user-visible name of an entity. This property is defined for all data entities but is used only in those entities that present a label in the user interface.

Type string (p. 1438)

Read Only No

MaximumViscosityLimit

Gets or sets the maximum viscosity limit.

Type Expression (p. 1384)<Quantity (p. 1422)>

Read Only No

MinimumViscosityLimit

Gets or sets the minimum viscosity limit.

Type Expression (p. 1384)<Quantity (p. 1422)>

Read Only No

PowerLawIndex

Gets or sets the index of the power law.

Type Expression (p. 1384)<Quantity (p. 1422)>

Read Only No

ReferenceTemperature

Gets or sets the reference temperature.

Type Expression (p. 1384)<Quantity (p. 1422)>

Read Only No

ViscosityNonNewtonianPowerLawShearRateDependent

The Shear Rate dependent model

Properties

ConsistencyIndex

Gets or sets the index of the consistency.

Type Expression (p. 1384)<Quantity (p. 1422)>

Read Only No

DisplayText

The general property that defines the user-visible name of an entity. This property is defined for all data entities but is used only in those entities that present a label in the user interface.

Type string (p. 1438)

Read Only No

MaximumViscosityLimit

Gets or sets the maximum viscosity limit.

Type Expression (p. 1384)<Quantity (p. 1422)>

Read Only No

MinimumViscosityLimit

Gets or sets the minimum viscosity limit.

Type Expression (p. 1384)<Quantity (p. 1422)>

Read Only No

PowerLawIndex

Gets or sets the index of the power law.

Type Expression (p. 1384)<Quantity (p. 1422)>

Read Only No

VolumeExtractionRegion

Volume Extraction Region specifies the settings used when extracting a volume region for a wrap task.

Available options:

GeometryLocation	Specifies the surounding bodies around the volume to be extracted.
Point	Specifies a reference point where the volume is located to extact.
ExtraRefineLevel	Specifies number of extra refinement levels.
ImprintIterations	Specifies number of imprinting iterations for feature capturing.
AggressiveImprintIterations	Specifies additional iterations for aggressive imprinting to improve feature capture.

Properties

FeatureResolution

Sets the Feature Resolution factor to be used for all Mesh Regions.

Type double (p. 1378)

Read Only No

Location

A common property for all sub meshing objects with a defined Geometry Filter

Type LocationSet (p. 1401)

Read Only No

Point

Specifies a reference point for where the volume to extract is located.

Type Point (p. 1417)

Read Only No

RegionBehavior

Specifies the meshing region behavior.

Available options:

Wrap	Simplifies unclean geometry before sewing.
PassThroughToSew	Bypasses simplification of unclean geometry before sewing.

Type RegionBehavior (p. 1425)

Read Only No

WrapMethod

Specifies wrap method for the meshing region.

Available options:

ShrinkWrap
CutWrap

Type WrapMethod (p. 1451)

Read Only No

VolumeSimplificationRegion

Volume Simplification specifies the settings used when simplifying a model within a wrap task.

Available options:

GeometryLocation	Specifies the bodies and parts to be wrapped.
GroupingOption	Specifies grouping option for the bodies being wrapped.
ExtraRefineLevel	Specifies number of extra refinement levels.
ImprintIterations	Specifies number of imprinting iterations for feature capturing.
AggressiveImprintIterations	Specifies additional iterations for aggressive imprinting to improve feature capture.

Properties

FeatureResolution

Sets the Feature Resolution factor to be used for all Mesh Regions.

Type double (p. 1378)

Read Only No

Location

A common property for all sub meshing objects with a defined Geometry Filter

Type LocationSet (p. 1401)

Read Only No

MergeBodies

If checked, geometry is simplified by unifying multiple solid surfaces into one and ignoring any interior voids and faces. Otherwise, a conformal, well-connected surface mesh is created for each of the selected bodies.

Type bool (p. 1360)

Read Only No

RegionBehavior

Specifies the meshing region behavior.

Available options:

Wrap	Simplifies unclean geometry before sewing.
PassThroughToSew	Bypasses simplification of unclean geometry before sewing.

Type RegionBehavior (p. 1425)

Read Only No

WrapMethod

Specifies wrap method for the meshing region.

Available options:

ShrinkWrap
CutWrap

Type WrapMethod (p. 1451)

Read Only No

Wrap

Volume creation task uses specialized boundary wrapping to extract a well-connected surface mesh for the selected geometry.

Properties

MeshResolution

Defines the resolution for automatically calculating global meshing inputs. Selecting a higher resolution generates a better quality mesh.

Type int (p. 1394)

Read Only No

Reassociate

Specifies whether the current output model should be used for reassociating so that reference Ids are persistent or whether new references should be created.

Type bool (p. 1360)

Read Only No

UsePredefinedSettings

Automatically calculates the global sizing values based on the resolution value.

Type bool (p. 1360)

Read Only No

ZSection

This class represents Z beam cross section

Properties

LowerFlangeThickness

Lower Flange Thickness

Type Quantity (p. 1422)

Read Only No

LowerFlangeWidth

Lower Flange Width

Type Quantity (p. 1422)

Read Only No

OverallHeight

Overall Height

Type Quantity (p. 1422)

Read Only No

StemThickness

Stem Thickness

Type Quantity (p. 1422)

Read Only No

UpperFlangeThickness

Upper Flange Thickness

Type Quantity (p. 1422)

Read Only No

UpperFlangeWidth

Upper Flange Width

Type Quantity (p. 1422)

Read Only No

Fluids Data Entities

ContourResultOptions

No details are provided for this entry.

Properties

UseNodeValues

No details are provided for this entry.

Type bool (p. 1360)

Read Only No

Duration

No details are provided for this entry.

Properties

EndTime

No details are provided for this entry.

Type Expression (p. 1384)<Quantity (p. 1422)>

Read Only No

NumberOfTimeSteps

No details are provided for this entry.

Type int (p. 1394)

Read Only No

Option

No details are provided for this entry.

Type string (p. 1438)

Read Only No

HeatSrc

No details are provided for this entry.

Properties

Location

No details are provided for this entry.

Type LocationSet (p. 1401)

Read Only No

LaunchControls

No details are provided for this entry.

Properties

AfdRootPath

No details are provided for this entry.

Type string (p. 1438)

Read Only No

HostFilePath

No details are provided for this entry.

Type string (p. 1438)

Read Only No

NumberOfProcessors

No details are provided for this entry.

Type int (p. 1394)

Read Only No

SolverFilesDirectory

No details are provided for this entry.

Type string (p. 1438)

Read Only Yes

MaterialDistribution

No details are provided for this entry.

Properties

Distribution

No details are provided for this entry.

Type string (p. 1438)

Read Only No

MaterialModels

No details are provided for this entry.

Properties

MaterialModel

No details are provided for this entry.

Type string (p. 1438)

Read Only No

MomentumSrc

No details are provided for this entry.

Properties

Location

No details are provided for this entry.

Type LocationSet (p. 1401)

Read Only No

ParticleInjectionCnd

No details are provided for this entry.

Properties

Location

No details are provided for this entry.

Type LocationSet (p. 1401)

Read Only No

ParticleTrack

No details are provided for this entry.

Properties

EndTime

No details are provided for this entry.

Type Quantity (p. 1422)

Read Only No

EndTimeFactor

No details are provided for this entry.

Type float (p. 1432)

Read Only No

EvaluateFullRange

Whether or not to evaluate all time points in the evaluation

Type bool (p. 1360)

Read Only No

FadeLineSegments

No details are provided for this entry.

Type bool (p. 1360)

Read Only No

LastAutomaticDisplayText

The last automatically-generated DisplayText assigned to this Result Object

Type string (p. 1438)

Read Only No

Location

The Location over which the Result is to be evaluated.

Type LocationSet (p. 1401)

Read Only No

ParticlesMaterialAssignmentName

No details are provided for this entry.

Type string (p. 1438)

Read Only No

ParticleTrackColorOption

No details are provided for this entry.

Type ParticleTrackColorOption (p. 1414)

Read Only No

ParticleTrackDisplayStyle

No details are provided for this entry.

Type ParticleTrackDisplayStyle (p. 1414)

Read Only No

ParticleTrackShape

No details are provided for this entry.

Type ParticleTrackShapeDef (p. 1414)

Read Only No

ParticleTrackStyle

No details are provided for this entry.

Type ParticleTrackStyle (p. 1415)

Read Only No

ResultName

Result name

Type string (p. 1438)

Read Only No

SegmentRatio

No details are provided for this entry.

Type float (p. 1432)

Read Only No

SymbolScaleFactor

No details are provided for this entry.

Type double (p. 1378)

Read Only No

SymbolSegmentDensity

No details are provided for this entry.

Type int (p. 1394)

Read Only No

SymbolShape

No details are provided for this entry.

Type SymbolShape (p. 1440)

Read Only No

TubeDiameter

No details are provided for this entry.

Type double (p. 1378)

Read Only No

WireThickness

No details are provided for this entry.

Type double (p. 1378)

Read Only No

PersistablePosition

No details are provided for this entry.

Properties

coordX

No details are provided for this entry.

Type double (p. 1378)

Read Only No

coordY

No details are provided for this entry.

Type double (p. 1378)

Read Only No

coordZ

No details are provided for this entry.

Type double (p. 1378)

Read Only No

unit

No details are provided for this entry.

Type string (p. 1438)

Read Only No

PersistableStateMessage

No details are provided for this entry.

Properties

Location

No details are provided for this entry.

Type LocationSet (p. 1401)

Read Only No

Popup

No details are provided for this entry.

Type bool (p. 1360)

Read Only No

PropertyName

No details are provided for this entry.

Type string (p. 1438)

Read Only No

StateMessageLevel

No details are provided for this entry.

Type PersistableStateMessageLevel (p. 1415)

Read Only No

Text

No details are provided for this entry.

Type string (p. 1438)

Read Only No

PersistableSupplierItem

No details are provided for this entry.

Properties

No Properties.

PersistableVariableMinMaxPositions

No details are provided for this entry.

Properties

maxPositions

No details are provided for this entry.

Type List (p. 1400)<PersistablePosition (p. 1415)>

Read Only No

minPositions

No details are provided for this entry.

Type List (p. 1400)<PersistablePosition (p. 1415)>

Read Only No

PorositySrc

No details are provided for this entry.

Properties

Location

No details are provided for this entry.

Type LocationSet (p. 1401)

Read Only No

PressureDrop

No details are provided for this entry.

Properties

CalculateAverage

Set to true if averages are to be calculated as part of the Result evaluation.

Type bool (p. 1360)

Read Only No

EvaluateFullRange

Whether or not to evaluate all time points in the evaluation

Type bool (p. 1360)

Read Only No

LastAutomaticDisplayText

The last automatically-generated DisplayText assigned to this Result Object

Type string (p. 1438)

Read Only No

Location

The Location over which the Result is to be evaluated.

Type LocationSet (p. 1401)

Read Only No

Location2

No details are provided for this entry.

Type LocationSet (p. 1401)

Read Only No

RelativeTo

The Reference Frame which the result is relative to

Type ReferenceFrame (p. 1424)

Read Only No

ResultName

Name of the Result.

Type string (p. 1438)

Read Only No

SimulationStep

The Simulation Step object reference for a multi-step analysis.

Type SolutionStep (p. 1434)

Read Only No

Value

No details are provided for this entry.

Type Expression (p. 1384)<Quantity (p. 1422)>

Read Only No

Variable

The solution variable to be displayed

Type Expression (p. 1384)<Quantity (p. 1422)>

Read Only No

RegionInterface

No details are provided for this entry.

Properties

DefinitionMethod

No details are provided for this entry.

Type CreationMode (p. 1370)

Read Only No

Location1

No details are provided for this entry.

Type LocationSet (p. 1401)

Read Only No

Location2

No details are provided for this entry.

Type LocationSet (p. 1401)

Read Only No

PhysicsRegions1

No details are provided for this entry.

Type List (p. 1400)<PhysicsRegion (p. 1415)>

Read Only No

PhysicsRegions2

No details are provided for this entry.

Type List (p. 1400)<PhysicsRegion (p. 1415)>

Read Only No

RegionInterfaceGenerator

No details are provided for this entry.

Properties

Location

No details are provided for this entry.

Type LocationSet (p. 1401)

Read Only No

Tolerance

No details are provided for this entry.

Type Quantity (p. 1422)

Read Only No

Customization Data Entities

UserCommands

UserCommands is a way to send user commands directly to the solver.

Properties

No Properties.

Structural Data Entities

BeamEndRelease

This is a class that is meant to define a beam end release.

Properties

EdgesLocation

Location for the edges selection.

Type LocationSet (p. 1401)

Read Only No

IndependentEdges

Switch to set the edges to be independent or not.

Type bool (p. 1360)

Read Only No

ReferenceFrame

Select a reference frame for the end release.

Type ReferenceFrame (p. 1424)

Read Only No

RotationX

Switch to set Rotation X as a free or fixed DOF.

Type DOFBehavior (p. 1378)

Read Only No

RotationY

Switch to set Rotation Y as a free or fixed DOF.

Type DOFBehavior (p. 1378)

Read Only No

RotationZ

Switch to set Rotation Z as a free or fixed DOF.

Type DOFBehavior (p. 1378)

Read Only No

TranslationX

Switch to set Tranlation X as a free or fixed DOF.

Type DOFBehavior (p. 1378)

Read Only No

TranslationY

Switch to set Tranlation Y as a free or fixed DOF

Type DOFBehavior (p. 1378)

Read Only No

TranslationZ

Switch to set Tranlation Z as a free or fixed DOF.

Type DOFBehavior (p. 1378)

Read Only No

VertexLocation

Location for the vertex selection.

Type LocationSet (p. 1401)

Read Only No

BearingLoad

No details are provided for this entry.

Properties

Location

Location of the boundary condition

Type LocationSet (p. 1401)

Read Only No

BoltPretension

A bolt pretension can be used to model a bolt which is holding together two bodies. It can be used to apply a known value of axial load or adjustment (the length which has reduced by screw tightening) to the bolt.

Properties

AxialDirection

Select the separation surface normal : X, Y or Z axis of the selected reference frame.

Type SeparationSurfaceNormal (p. 1431)

Read Only No

AxialDirectionDefinition

Choose how the separation surface is selected : Program controlled or user defined It can be user defined only if a volume is selected. Otherwise it has to be program controlled.

Type SeparationSurfaceSpecification (p. 1431)

Read Only No

AxialDirectionLocation

Select a face or an edge to interpret the axial direction of the bolt.

Type LocationSet (p. 1401)

Read Only No

BoltDiameter

No details are provided for this entry.

Type Quantity (p. 1422)

Read Only No

DefineBy

Choose how the bolt is defined : Axial force or Adjustment

Type PretensionDefineBy (p. 1420)

Read Only No

Location

Location of the boundary condition

Type LocationSet (p. 1401)

Read Only No

Magnitude

Applied value of load

Type Expression (p. 1384)<Quantity (p. 1422)>

Read Only No

ModelAs

Choose how many bolts should be created if multiple volumes are selected : One for all selected volumes or one per selected volume.

Type PretensionModelAs (p. 1420)

Read Only No

ReferenceFrame

Select the separation surface frame.

Type ReferenceFrame (p. 1424)

Read Only No

TorqueCoefficient

No details are provided for this entry.

Type double (p. 1378)

Read Only No

BoundaryCondition

BoundaryCondition is the abstraction over load and constraints.

Properties

Location

Location of the boundary condition

Type LocationSet (p. 1401)

Read Only No

BushingJoint

This class represents the properties of a bushing joint connection between two locations. It is added to a JointBehavior if joint type is set as bushing.

Properties

No Properties.

CalculationTime

A class that holds calculation time of transient analysis.

Properties

Duration

Duration of the analysis

Type Quantity (p. 1422)

Read Only No

Charge

Current is a Scalar Load to precribe a given value of current at a given location in a model in electric solutions.

Properties

Location

Location of the boundary condition

Type LocationSet (p. 1401)

Read Only No

Magnitude

Applied value of load

Type Expression (p. 1384)<Quantity (p. 1422)>

Read Only No

Connection

This is a class that is meant to define different kinds of connections like Contact, Joint, Spring, Beam etc. via one of its behaviors when set Non creatable by a user.

Properties

Behavior

It displays the contact behavior referenced by this connection.

Type InterfaceBehavior (p. 1395)

Read Only No

CreationMode

It tells if this connection was created automatically by the program or created manually by the user

Type CreationMode (p. 1370)

Read Only No

Location1

This is the first location of the connection. For contact, this normally refers to the source of the contact.

Type LocationSet (p. 1401)

Read Only No

Location1Bodies

It displays the bodies of the entities referenced in Location 1 of the connection.

Type string (p. 1438)

Read Only Yes

Location2

This is the second location of the connection. For contact, this normally refers to the target of the contact.

Type LocationSet (p. 1401)

Read Only No

Location2Bodies

It displays the bodies of the entities referenced in Location 2 of the connection.

Type string (p. 1438)

Read Only Yes

ConnectionBehavior

ConnectionBehavior is an abstraction over all kinds of connection behaviors such as JointBehavior, BeamBehavior, LinkBehavior, SpringBehavior and ContactBehavior.

Properties

No Properties.

ConnectionStatisticalMeasures

This class represents the connection confidence percentage data. It get run-time added to an automatic generated contacts whne the generate confidence setting is turned on.

Properties

PercentageConfidenceLevel

Show the confidence level for the automatic generated connections

Type double (p. 1378)

Read Only Yes

Constraint

Constraint is an abstraction over all the boundary condtions that provide known values and does not require solution for a given location in the model such as Supports, Displacements, Rotations etc.

Properties

Location

Location of the boundary condition

Type LocationSet (p. 1401)

Read Only No

Contact

This is a class that is meant to define Contact

Properties

AllowSelfContact

No details are provided for this entry.

Type bool (p. 1360)

Read Only No

Behavior

It displays the contact behavior referenced by this connection.

Type InterfaceBehavior (p. 1395)

Read Only No

CreationMode

It tells if this connection was created automatically by the program or created manually by the user

Type CreationMode (p. 1370)

Read Only No

Location1

This is the first location of the connection. For contact, this normally refers to the source of the contact.

Type LocationSet (p. 1401)

Read Only No

Location1Bodies

It displays the bodies of the entities referenced in Location 1 of the connection.

Type string (p. 1438)

Read Only Yes

Location2

This is the second location of the connection. For contact, this normally refers to the target of the contact.

Type LocationSet (p. 1401)

Read Only No

Location2Bodies

It displays the bodies of the entities referenced in Location 2 of the connection.

Type string (p. 1438)

Read Only Yes

ContactBehavior

ContactBehavior is ConnectionBehavior to model various kinds of contact behaviors available from Mechanical Solvers such as contact types and their formulations.

Properties

ContactType

Type of contact for eg. Bonded, Frictional, Frictionless, etc.

Type ContactType (p. 1369)

Read Only No

ContactDamping

This class represents the contact damping settings. It get run-time added to a non linear structural contact.

Properties

NormalStabilizationDamping

Damping scaling factor along the normal direction to reduce the risk of rigid body motion

Type Expression (p. 1384)<Quantity (p. 1422)>

Read Only No

TangentStabilizationDamping

Damping scaling factor along the tangential direction to reduce the risk of rigid body motion

Type Expression (p. 1384)<Quantity (p. 1422)>

Read Only No

ContactDetection

This is a class resposible for applying the rules of contact detection

Properties

Algorithm

Property for selecting algorithm for contact detection

Type AlgorithmType (p. 1354)

Read Only No

FaceEdgeDetectionPreference

Property for setting the preference for face to edge or edge to face detection between bodies or parts

Type FaceEdgePreference (p. 1384)

Read Only No

FaceFaceDetectionPreference

Property for setting the preference for face to face detection between bodies or parts

Type FaceFacePreference (p. 1384)

Read Only No

Location

It is the location where it should detect contacts

Type LocationSet (p. 1401)

Read Only No

Preference

Property for setting the preference for detecting contact between bodies, or between bodies of different parts

Type PreferenceType (p. 1419)

Read Only No

Priority

Property for setting the priority for contact detection. Face override means priority is set for Face to Face -> Face to Edge -> Edge to Edge Edge override means priority is set for Edge to Edge -> Edge to Face -> Face to Face

Type PriorityType (p. 1420)

Read Only No

Tolerance

Property for minimum gap between bodies, to register a contact detection

Type Quantity (p. 1422)

Read Only No

ContactDetectionDirection

ContactDetectionDirection is a class to change the normals on shell bodies for contact connections as available from MAPDL solver

Properties

DetectionDirectionMethod

Set the detection direction method

Type SearchDirectionMethod (p. 1429)

Read Only No

ReverseDirectionLoaction1

Switch to turn on to reverse the contact search direction for location 1

Type bool (p. 1360)

Read Only No

ReverseDirectionLoaction2

Switch to turn on to reverse the contact search direction for location 2

Type bool (p. 1360)

Read Only No

ContactElasticSlipTolerance

This is a class resposible for setting elastic slip tolerance at the contact interface.

Properties

DefineBy

This propertry decides whether it should use program controlled value or a manually entered factor or value.

Type ValueDefineBy (p. 1448)

Read Only No

Value

Enter the factor or value

Type Expression (p. 1384)<Quantity (p. 1422)>

Read Only No

ContactHeatGeneration

This class represents the heat generation due to friction or current

Properties

FrictionHeatFactor

It represents the fraction of frictional dissipated energy converted into heat. It varies from 0 to 1. For an input of true 0, you must enter a very small value (for example, 1E-8).

Type Expression (p. 1384)<Quantity (p. 1422)>

Read Only No

HeatDistributionFactor

It represents the weight factor for the distribution of heat between the contact and target surfaces.

Type Expression (p. 1384)<Quantity (p. 1422)>

Read Only No

ContactInterfaceGapAdjustment

This class represents the interface gap adjustment settings. It gets run-time added to Contact and enabled as the contact type becomes frictional, rough, frictionless

Properties

InitialInterfaceTreatment

Defines how the initial interface of a contact pair is treated.

Type InitialInterfaceTreatmentType (p. 1393)

Read Only No

Offset

Defines the contact offset value. A positive value moves the contact closer together and a negative value moves the contact further apart.

Type Expression (p. 1384)<Quantity (p. 1422)>

Read Only No

ContactPenetrationTolerance

This is a class resposible for setting penetration tolerance at the contact interface.

Properties

DefineBy

This propertry decides whether it should use program controlled value or a manually entered factor or value.

Type ValueDefineBy (p. 1448)

Read Only No

Value

Enter the factor or value

Type Expression (p. 1384)<Quantity (p. 1422)>

Read Only No

ContactPinball

This is a class resposible for setting pinball value for contact detection.

Properties

DefineBy

This propertry decides whether it should use program controlled value or a manually entered factor or value.

Type ValueDefineBy (p. 1448)

Read Only No

Value

Enter the factor or value

Type Expression (p. 1384)<Quantity (p. 1422)>

Read Only No

ContactProperties

This class defines contact properties that can change per contact pair such as Symmetry Behavior etc. as exposed by Mechanical solvers.

Properties

ApplySymmetry

Switch to turn on symmetric contact - Additional contact pair will be generated by swapping Location 1 and Location 2

Type bool (p. 1360)

Read Only No

ContactRadius

This is a class resposible for radii settings for an edge-edge contact.

Properties

DefineBy

Switch to choose between program controlled radii values or use manually entered values.

Type RadiusDefineBy (p. 1422)

Read Only No

SourceRadius

Radius of source beam

Type Quantity (p. 1422)

Read Only No

TargetRadius

Radius of target beam

Type Quantity (p. 1422)

Read Only No

ContactStiffness

This class represents the various behavior of contact stiffness

Properties

UpdateStiffnessControl

Option to set the frequency of updating the stifness.

Type UpdateStiffnessControlType (p. 1447)

Read Only No

ContactSubObject

This is the base class for pinball, penetration tolerance and the slip tolerance.

Properties

DefineBy

This propertry decides whether it should use program controlled value or a manually entered factor or value.

Type ValueDefineBy (p. 1448)

Read Only No

Value

Enter the factor or value

Type Expression (p. 1384)<Quantity (p. 1422)>

Read Only No

Convection

Convection is a Scalar Load to precribe a given value of heat flux at a given location in a model in thermal solutions.

Properties

AmbientTemperature

Value of Ambient Temperature of Radiation.

Type Expression (p. 1384)<Quantity (p. 1422)>

Read Only No

Location

Location of the boundary condition

Type LocationSet (p. 1401)

Read Only No

Magnitude

Applied value of load

Type Expression (p. 1384)<Quantity (p. 1422)>

Read Only No

ReferenceFrame

Field variables of the spatial variation expression are defined with respect to this reference frame.

Type ReferenceFrame (p. 1424)

Read Only No

Current

Current is a Scalar Load to precribe a given value of current at a given location in a model in electric solutions.

Properties

Location

Location of the boundary condition

Type LocationSet (p. 1401)

Read Only No

Magnitude

Applied value of load

Type Expression (p. 1384)<Quantity (p. 1422)>

Read Only No

ReferenceFrame

Field variables of the spatial variation expression are defined with respect to this reference frame.

Type ReferenceFrame (p. 1424)

Read Only No

Specification

No details are provided for this entry.

Type CurrentSpecificationMethod (p. 1370)

Read Only No

CylindricalJoint

This class represents the properties of a cylindrical joint connection between two locations. It is added to a JointBehavior if joint type is set as cylindrical.

Properties

No Properties.

Displacement

Displacement is a Vector Constraint used to prescribe a given value of displacement at a given location in the model

Properties

AppliedRemotely

This property sets the type of displacement whether it is applied directly or remotely via an originating point

Type bool (p. 1360)

Read Only No

Location

Location of the boundary condition

Type LocationSet (p. 1401)

Read Only No

Specify

This property sets the type of displacement whether it is translation only or a combo of translation and rotation.

Type DisplacementType (p. 1377)

Read Only No

DisplacementByVector

Extension object for prescribing displacement by translation only or rotation only.

Properties

DefineBy

This property sets the type of vector specification which could be by Components or MagnitudeAndDirection or NormalToSurface.

Type VectorDefineBy (p. 1449)

Read Only No

RelativeTo

Vector components or direction has been specified with respect to this reference frame.

Type ReferenceFrame (p. 1424)

Read Only No

DistributedMass

No details are provided for this entry.

Properties

Location

Location of the boundary condition

Type LocationSet (p. 1401)

Read Only No

Mass

No details are provided for this entry.

Type Expression (p. 1384)<Quantity (p. 1422)>

Read Only No

MassType

No details are provided for this entry.

Type DistributedMassType (p. 1377)

Read Only No

EdgeEdgeContactSettings

This is a class resposible for edge-edge contact settings.

Properties

Treatment

Option to select if only parallel contact pair or both parallel and perpendicular contact pairs are created for an edge-edge contact.

Type EdgeEdgeTreatment (p. 1379)

Read Only No

EigenvalueBucklingExtension

No details are provided for this entry.

Properties

LoadMultiplier

No details are provided for this entry.

Type double (p. 1378)

Read Only Yes

ElectricConductance

This class represents the eletric conductance at the contact interface. It gets added when selected physics includes electric and formulation is not MPC.

Properties

Conductance

Contact conductance value

Type Expression (p. 1384)<Quantity (p. 1422)>

Read Only No

ConductanceControl

Switch to use program controlled conductance value or enter value.

Type ConductanceControlType (p. 1366)

Read Only No

Equipotential

This is a class that creates an equal voltage on the selection applied.

Properties

Location

Location of the boundary condition

Type LocationSet (p. 1401)

Read Only No

FatigueSettings

This is a class that contains the various settings pertaining to Fatigue analysis

Properties

AccumulateDamage

Specify whether you want to accumulate damage or not

Type bool (p. 1360)

Read Only No

AnalysisType

Select the type of approach used for fatigue analysis i.e. Stress life or Strain life

Type AnalysisType (p. 1354)

Read Only No

BlocksPerCycle

Specify the Blocks per cycle

Type double (p. 1378)

Read Only No

DefineBy

Specify the S-N curve data source

Type DefineBy (p. 1372)

Read Only No

DesignLife

Specify design life for damage and safety factor calculations

Type double (p. 1378)

Read Only No

DesignLifeUnit

Displays the life units chosen

Type string (p. 1438)

Read Only Yes

ENMeanStressTheory

Specify how mean stress effects are handled for Strain life fatigue

Type ENMeanStressTheory (p. 1381)

Read Only No

InfiniteLife

Specify the maximum life for strain life to avoid skewed contour plots showing very high lives

Type double (p. 1378)

Read Only No

LoadingType

Select the type of applied constant amplitude loading based on the loading ratio

Type LoadingType (p. 1400)

Read Only No

MeanStressTheory

Specify how mean stress effects are handled for Stress life fatigue

Type MeanStressTheory (p. 1402)

Read Only No

MultiAxisType

Specify the stress component type

Type MultiAxisType (p. 1407)

Read Only No

Ratio

Specify the loading ratio

Type double (p. 1378)

Read Only No

ScaleFactor

Specify the Scale factor to scale the load magnitude.

Type double (p. 1378)

Read Only No

TimePerCycle

Specify the Time per cycle

Type double (p. 1378)

Read Only No

UltimateStrength

Specify the Ultimate Strength for all materials in the model

Type Expression (p. 1384)<Quantity (p. 1422)>

Read Only No

UnitType

Specify the life units used to display fatigue life results

Type FatigueUnitType (p. 1385)

Read Only No

YieldStrength

Specify the Yield Strength for all materials in the model

Type Expression (p. 1384)<Quantity (p. 1422)>

Read Only No

FatigueUnitsExtension

/// Displays the units chosen for Fatigue life and design life.

Properties

UnitName

/// The units chosen for Fatigue life and design life.

Type string (p. 1438)

Read Only Yes

FixedJoint

This class represents the properties of a fixed joint connection between two locations. It is added to a JointBehavior if joint type is set as fixed.

Properties

No Properties.

Force

Force is a Vector Load to prescribe external force to a given model at a given location.

Properties

AppliedRemotely

Property to control if the force is applied remotely via an originating point or directly.

Type bool (p. 1360)

Read Only No

Location

Location of the boundary condition

Type LocationSet (p. 1401)

Read Only No

Specification

Property to specify if the applied force value is total force or the force per unit area of the selected surface location.

Type ApplicationMethod (p. 1355)

Read Only No

GeneralJoint

This class represents only general joint specific settings. It get run-time added to Joint as the joint type is general

Properties

No Properties.

GeneralJointDOFs

This class represents the properties of a general joint connection between two locations. It is added to a JointBehavior if joint type is set as general.

Properties

Rotation

It controls the behavior of rotational DOFs of a general joint

Type RotationalDOFBehavior (p. 1427)

Read Only No

TranslationX

Switch to set Tranlation X as a free DOF

Type TranslationalDOFBehavior (p. 1444)

Read Only No

TranslationY

Switch to set Tranlation Y as a free DOF

Type TranslationalDOFBehavior (p. 1444)

Read Only No

TranslationZ

Switch to set Tranlation Z as a free DOF

Type TranslationalDOFBehavior (p. 1444)

Read Only No

GroundLocation

This is a class that is meant to define ground location

Properties

Location

The selected location: evaluates to a set of reference ids. See LocationSet for more detail.

Type LocationSet (p. 1401)

Read Only Yes

HeatFlow

Heat Flow is a Scalar Load to precribe a given value of heat flow at a given location in a model in thermal solutions.

Properties

Location

Location of the boundary condition

Type LocationSet (p. 1401)

Read Only No

Magnitude

Applied value of load

Type Expression (p. 1384)<Quantity (p. 1422)>

Read Only No

ReferenceFrame

Field variables of the spatial variation expression are defined with respect to this reference frame.

Type ReferenceFrame (p. 1424)

Read Only No

HeatFlux

Heat Flux is a Scalar Load to precribe a given value of heat flux at a given location in a model in thermal solutions.

Properties

Location

Location of the boundary condition

Type LocationSet (p. 1401)

Read Only No

Magnitude

Applied value of load

Type Expression (p. 1384)<Quantity (p. 1422)>

Read Only No

ReferenceFrame

Field variables of the spatial variation expression are defined with respect to this reference frame.

Type ReferenceFrame (p. 1424)

Read Only No

HeatGeneration

Heat Generation is a Scalar Load to prescribe a uniform generation rate internal to a body.

Properties

DefineBy

Property to specify if the applied heat generation value is total heat generated or the heat generated per unit volume.

Type ApplicationMethod (p. 1355)

Read Only No

Location

Location of the boundary condition

Type LocationSet (p. 1401)

Read Only No

Magnitude

Applied value of load

Type Expression (p. 1384)<Quantity (p. 1422)>

Read Only No

ReferenceFrame

Field variables of the spatial variation expression are defined with respect to this reference frame.

Type ReferenceFrame (p. 1424)

Read Only No

HingeJoint

This class represents the properties of a hinge joint connection between two locations. It is added to a JointBehavior if joint type is set as hinge.

Properties

No Properties.

InertiaLoad

InertiaLoad is a Vector Load experienced by a body by the virtue of being in an accelerated reference frame due to its mass properties. It can be prescribed via an Acceleration, Angular Velocity or Angular Acceleration at a given location in the model.

Properties

LoadType

Load Type of Inertia Load such as Acceleration, Angular Velocity or Angular Acceleration.

Type VectorType (p. 1450)

Read Only No

Location

Location of the boundary condition

Type LocationSet (p. 1401)

Read Only No

InitialCondition

Initial conditions is the abstraction over all kinds of initial conditions.

Properties

Location

Location for InitialCondition

Type LocationSet (p. 1401)

Read Only No

InitialReferenceFrames

This class shows the two reference frames selected for two locations of a joint connection. It is added to a connection if joint behavior is selected.

Properties

Base

Reference frame for Location 1 of connection

Type ReferenceFrame (p. 1424)

Read Only No

Mobile

Reference frame for Location 2 of connection

Type ReferenceFrame (p. 1424)

Read Only No

InitialTemperature

This class specifes the initial values of the temperature

Properties

Location

Location for InitialCondition

Type LocationSet (p. 1401)

Read Only No

Magnitude

Value of scalar initial condition

Type Expression (p. 1384)<Quantity (p. 1422)>

Read Only No

InterfaceGenerator

InterfaceGenerator is class to define various settings for detection of Interface Conditions such a contacts.

Properties

BehaviorCreation

It allows you to create one contact behavior and share it or create separate ones for each one of searched interfaces.

Type InterfaceBehaviorCreation (p. 1395)

Read Only No

DetectionTypes

Set the detection types for generating the interfaces

Type InterfaceDetectionTypes (p. 1395)

Read Only No

GenerateFixedJoints

Switch to activate generation of fixed joints

Type bool (p. 1360)

Read Only No

GenerateHingeJoints

Switch to activate generation of hinge joints

Type bool (p. 1360)

Read Only No

InterfaceToleranceSpecification

It allows to let either program decides the tolerance value for you or you choose to input your own tolerance value

Type InterfaceToleranceSpecification (p. 1396)

Read Only No

InterfaceType

Type of inerfaces to be generated. Select from Contact, Joint and Mesh Interface.

Type InterfaceType (p. 1396)

Read Only No

Location

Location considered to search for interfaces.

Type LocationSet (p. 1401)

Read Only No

Tolerance

Tolerance value used to search for interfaces

Type Quantity (p. 1422)

Read Only No

Joint

This is a class that is meant to define different kinds of Joint

Properties

Behavior

It displays the contact behavior referenced by this connection.

Type InterfaceBehavior (p. 1395)

Read Only No

CreationMode

It tells if this connection was created automatically by the program or created manually by the user

Type CreationMode (p. 1370)

Read Only No

Location1

This is the first location of the connection. For contact, this normally refers to the source of the contact.

Type LocationSet (p. 1401)

Read Only No

Location1Bodies

It displays the bodies of the entities referenced in Location 1 of the connection.

Type string (p. 1438)

Read Only Yes

Location2

This is the second location of the connection. For contact, this normally refers to the target of the contact.

Type LocationSet (p. 1401)

Read Only No

Location2Bodies

It displays the bodies of the entities referenced in Location 2 of the connection.

Type string (p. 1438)

Read Only Yes

TrimOptimization

JointFormulation types used by Mapdl Fixed Joint Connection Behavior

Type JointTrimOptimization (p. 1398)

Read Only No

JointBehavior

JointBehavior is a ConnectionBehavior that models joint connection between two locations.

Properties

JointType

Joint type such as fixed, hinge, slot, translational, cylindrical, planar, etc.

Type JointType (p. 1398)

Read Only No

JointTypeBase

This is the base class for all joint types (fixed, hinge, cylindrical, general, etc.)

Properties

No Properties.

LaunchSettings

LaunchSettings is a class that allows defining solver launch controls such as Solver file directory, various solver input file names as exposed by MAPDL solver.

Properties

ComponentDataFileName

A name for auxiliary solver input file which contains all nodal and elemental components created in MAPDL solution.

Type string (p. 1438)

Read Only No

ConnectionElementsDataFileName

A name for auxiliary solver input file which contains all connection elements in MAPDL solution.

Type string (p. 1438)

Read Only No

ContactPairOutput

Activate Contact Pair Output in MAPDL solution.

Type ContactPairOutputYesNo (p. 1368)

Read Only No

DeleteWorkFiles

Override to keep all solver generated files in MAPDL solution.

Type bool (p. 1360)

Read Only No

DistributedSolve

Turn Off or On Distributed solution for MAPDL solver.

Type bool (p. 1360)

Read Only No

InputDataFileName

A name for master solver input file which assembles all other auxiliary input files (MeshDataFileName, ComponentDataFileName, etc.) for MAPDL Solver.

Type string (p. 1438)

Read Only No

JobName

A name for the job file in MAPDL solution.

Type string (p. 1438)

Read Only No

LoadElementsDataFileName

A name for auxiliary solver input file which contains all load elements in MAPDL solution.

Type string (p. 1438)

Read Only No

MeshDataFileName

A name for auxiliary solver input file which contains all mesh information(nodes and elements) for all participating bodies in MAPDL solution.

Type string (p. 1438)

Read Only No

NumberOfProcessors

Number of Processors to use for distributed solution in MAPDL solver.

Type uint (p. 1446)

Read Only No

OutputListFileName

A name for output file in MAPDL solution.

Type string (p. 1438)

Read Only No

SolverFilesDirectory

Solver Files Directory where solution happens for MAPDL solver.

Type string (p. 1438)

Read Only Yes

LinearizedResults

Extension object that presents linearized stress parameters

Properties

BendingStressBeginning

The bending stress component at the beginning of the line path.

Type Quantity (p. 1422)

Read Only Yes

BendingStressEnd

The bending stress component at the end of the line path.

Type Quantity (p. 1422)

Read Only Yes

MembranePlusBendingStressBeginning

The membrane + bending stress at the beginning of the line path.

Type Quantity (p. 1422)

Read Only Yes

MembranePlusBendingStressEnd

The membrane + bending stress at the end of the line path.

Type Quantity (p. 1422)

Read Only Yes

MembranePlusBendingStressMidLength

The membrane + bending stress at the mid-length of the line path.

Type Quantity (p. 1422)

Read Only Yes

MembraneStress

The membrane stress component.

Type Quantity (p. 1422)

Read Only Yes

PeakStressBeginning

The peak stress component at the beginning of the line path.

Type Quantity (p. 1422)

Read Only Yes

PeakStressEnd

The peak stress component at the end of the line path.

Type Quantity (p. 1422)

Read Only Yes

PeakStressMidLength

The peak stress component at the mid-length of the line path.

Type Quantity (p. 1422)

Read Only Yes

TotalStressBeginning

The total stress component at the beginning of the line path.

Type Quantity (p. 1422)

Read Only Yes

TotalStressEnd

The total stress component at the end of the line path.

Type Quantity (p. 1422)

Read Only Yes

TotalStressMidLength

The total stress component at the mid-length of the line path.

Type Quantity (p. 1422)

Read Only Yes

LinearizedStressChart

No details are provided for this entry.

Properties

EvaluateFullRange

Whether or not to evaluate all time points in the evaluation

Type bool (p. 1360)

Read Only No

LastAutomaticDisplayText

The last automatically-generated DisplayText assigned to this Result Object

Type string (p. 1438)

Read Only No

Location

The Location over which the Result is to be evaluated.

Type LocationSet (p. 1401)

Read Only No

NormalizeData

Normalize data so they are displayed as percentages

Type bool (p. 1360)

Read Only No

ReverseDirection

Reverse the direction of the data when it's sent to the graph

Type bool (p. 1360)

Read Only No

SimulationStep

The Simulation Step object reference for a multi-step analysis.

Type SolutionStep (p. 1434)

Read Only No

Variable

Data Model support for variable selector.

Type Expression (p. 1384)<Quantity (p. 1422)>

Read Only No

Variables

List of variables to be plotted

Type List (p. 1400)<Expression (p. 1384)<Quantity (p. 1422)>>

Read Only No

LinearizeOption

Extension object that provides the option to linearize stress results

Properties

Linearize

The option to linearize stress results

Type bool (p. 1360)

Read Only No

Load

Load is Boudary Condition to prescribe external load to a given location in a model. It is an abstraction over scalar and vector loads.

Properties

Location

Location of the boundary condition

Type LocationSet (p. 1401)

Read Only No

MechanicalPhysicsOptions

MechanicalPhysicsOptions is a class that represents various ways through which a physics option can be applied to bodies. For example various stress states to be analyzed such as 3DStress State, Plane Stress, Plane Strain, Axisymmetric, etc.

Properties

Location

The region participating in the solution.

Type LocationSet (p. 1401)

Read Only No

UseInertialReferenceFrame

For Rigid bodies, control to model them using a single Inertial Frame or individual ones

Type bool (p. 1360)

Read Only No

Moment

Moment is a vector load to prescribe an external moment load to a given location in a model.

Properties

Location

Location of the boundary condition

Type LocationSet (p. 1401)

Read Only No

OptimizationConstraint

OptimizationConstraint is a base class for all optimization constraint classes

Properties

No Properties.

OptimizationOptions

OptimizationOptions is an extension object for MechanicalPhysicsOptions to offer the objective options for the optimization process of a given location

Properties

ConstrainModeNumber

Sets the mode number which frequency to be constrained.

Type int (p. 1394)

Read Only No

ConstrainNaturalFrequency

Switch to constrain the natural frequency.

Type bool (p. 1360)

Read Only No

DesignTypes

Sets the design goal for the optimization analysis.

Type DesignTypes (p. 1375)

Read Only No

Location

The region participating in the optimization process.

Type LocationSet (p. 1401)

Read Only No

MaximumFrequency

upper bound of the frequency constraint range.

Type Quantity (p. 1422)

Read Only No

MaxModesToFind

Maximum modes to find for modal solutions as available from MAPDL solver.

Type int (p. 1394)

Read Only No

MinimumFrequency

Lower bound of the frequency constraint range.

Type Quantity (p. 1422)

Read Only No

ModeNumber

Sets the mode number needed for natural frequency optimal design.

Type int (p. 1394)

Read Only No

PreserveBondedContacts

Switch to preserve the contact regions during optimization analysis.

Type bool (p. 1360)

Read Only No

ReductionType

Sets the material removal to be based on for a model with multiple materials.

Type ReductionType (p. 1423)

Read Only No

SpecifyTargetReduction

Switch to limit the material removal.

Type bool (p. 1360)

Read Only No

TargetReduction

Sets a target for maximum material removal.

Type Expression (p. 1384)<Quantity (p. 1422)>

Read Only No

OutputControls

OutputControls is a class that specifes the list of output specifications(such as Deformation, Stress, Strain) to control the "what" goes in result file. It also controls "how frequently" results are stored.

Properties

No Properties.

PerimeterWeld

This class represents a perimeter weld connection between two locations.

Properties

Behavior

It displays the contact behavior referenced by this connection.

Type InterfaceBehavior (p. 1395)

Read Only No

CreationMode

It tells if this connection was created automatically by the program or created manually by the user

Type CreationMode (p. 1370)

Read Only No

Location1

This is the first location of the connection. For contact, this normally refers to the source of the contact.

Type LocationSet (p. 1401)

Read Only No

Location1Bodies

It displays the bodies of the entities referenced in Location 1 of the connection.

Type string (p. 1438)

Read Only Yes

Location2

This is the second location of the connection. For contact, this normally refers to the target of the contact.

Type LocationSet (p. 1401)

Read Only No

Location2Bodies

It displays the bodies of the entities referenced in Location 2 of the connection.

Type string (p. 1438)

Read Only Yes

PlanarJoint

This class represents the properties of a planar joint connection between two locations. It is added to a JointBehavior if joint type is set as planar.

Properties

No Properties.

Pressure

Pressure is a scalar load to prescribe an external pressure to a given location(face or element faces) in a model always in a direction normal to the location surface.

Properties

Location

Location of the boundary condition

Type LocationSet (p. 1401)

Read Only No

Magnitude

Applied value of load

Type Expression (p. 1384)<Quantity (p. 1422)>

Read Only No

ReferenceFrame

Field variables of the pressure spatial variation expression are defined with respect to this reference frame.

Type ReferenceFrame (p. 1424)

Read Only No

PretensionTolerance

It denotes the tolerance upto which nodes below the separation surface are considered to be above the surface. It has an affect of shifting separation surface downwards by this distance.

Properties

Tolerance

Value of pretension tolerance

Type Expression (p. 1384)<Quantity (p. 1422)>

Read Only No

ToleranceSpecification

Choose how pretension tolerance is defined : Program controlled or Manual

Type DefineBy (p. 1372)

Read Only No

PsdAcceleration

PsdAcceleration is a PSD acceleration random vibration excitation applied to supports in global X, Y or Z direction.

Properties

Location

Location of the boundary condition

Type LocationSet (p. 1401)

Read Only No

ReferenceFrame

Select a reference frame for the excitation direction.

Type ReferenceFrame (p. 1424)

Read Only No

PsdDisplacement

PsdDisplacement is a PSD displacement random vibration excitation applied to supports in global X, Y or Z direction.

Properties

Location

Location of the boundary condition

Type LocationSet (p. 1401)

Read Only No

ReferenceFrame

Select a reference frame for the excitation direction.

Type ReferenceFrame (p. 1424)

Read Only No

PsdGAcceleration

PsdGAcceleration is a G acceleration random vibration excitation applied to supports in global X, Y or Z direction.

Properties

Location

Location of the boundary condition

Type LocationSet (p. 1401)

Read Only No

ReferenceFrame

Select a reference frame for the excitation direction.

Type ReferenceFrame (p. 1424)

Read Only No

PsdLoad

PsdLoad is a base class for all PSD loads in a random vibration analysis.

Properties

Location

Location of the boundary condition

Type LocationSet (p. 1401)

Read Only No

ReferenceFrame

Select a reference frame for the excitation direction.

Type ReferenceFrame (p. 1424)

Read Only No

PsdUserDefinedTypeExtension

No details are provided for this entry.

Properties

ResultType

No details are provided for this entry.

Type ResultType (p. 1426)

Read Only No

PsdVelocity

PsdVelocity is a PSD velocidy random vibration excitation applied to supports in global X, Y or Z direction.

Properties

Location

Location of the boundary condition

Type LocationSet (p. 1401)

Read Only No

ReferenceFrame

Select a reference frame for the excitation direction.

Type ReferenceFrame (p. 1424)

Read Only No

Radiation

Radiation is a scalar load to prescribe an external Radiation to a given location in a model.

Properties

Emissivity

Value of Emissivity of Radiation.

Type Expression (p. 1384)<Quantity (p. 1422)>

Read Only No

Location

Location of the boundary condition

Type LocationSet (p. 1401)

Read Only No

Magnitude

Applied value of load

Type Expression (p. 1384)<Quantity (p. 1422)>

Read Only No

ReferenceFrame

Field variables of the spatial variation expression are defined with respect to this reference frame.

Type ReferenceFrame (p. 1424)

Read Only No

ReadOnlyShape

Read only shape is created to display intermediate shape to the user while solving an optimization analysis.

Properties

No Properties.

ReferenceTemperature

No details are provided for this entry.

Properties

Value

No details are provided for this entry.

Type Expression (p. 1384)<Quantity (p. 1422)>

Read Only No

RemoteLocation

This extension is added to boundary condition when it is applied remotely via an originating point.

Properties

Formulation

Formulation for the internally created remote point connection.

Type ConnectionEndBehavior (p. 1366)

Read Only No

OriginatingPoint

Originating point for the remote boundary condition.

Type LocationSet (p. 1401)

Read Only No

ResponsePsdResult

No details are provided for this entry.

Properties

ExpectedFrequency

The current value of the result.

Type Quantity (p. 1422)

Read Only No

LastAutomaticDisplayText

The last automatically-generated DisplayText assigned to this Result Object

Type string (p. 1438)

Read Only No

Location

The Location over which the result is to be evaluated.

Type LocationSet (p. 1401)

Read Only No

RelativeTo

The Reference Frame which the result is relative to

Type ReferenceFrame (p. 1424)

Read Only Yes

RmsValue

The current value of the result.

Type Quantity (p. 1422)

Read Only No

Variable

The result variable to be evaluated.

Type Expression (p. 1384)<Quantity (p. 1422)>

Read Only No

ResultReferenceExtension

No details are provided for this entry.

Properties

ResultReference

No details are provided for this entry.

Type ResultReference (p. 1426)

Read Only No

ScalarConstraint

ScalarConstraint is abstraction over constraints that are scalar by nature such as temperature in thermal solutions or voltage in electric solutions

Properties

Location

Location of the boundary condition

Type LocationSet (p. 1401)

Read Only No

Magnitude

Applied value of constraint

Type Expression (p. 1384)<Quantity (p. 1422)>

Read Only No

ScalarInitialCondition

This is used by scalar initial conditions.

Properties

Location

Location for InitialCondition

Type LocationSet (p. 1401)

Read Only No

Magnitude

Value of scalar initial condition

Type Expression (p. 1384)<Quantity (p. 1422)>

Read Only No

ScalarLoad

ScalarLoad is abstraction over loads that are scalar by nature such as pressure Radiation etc.

Properties

Location

Location of the boundary condition

Type LocationSet (p. 1401)

Read Only No

Magnitude

Applied value of load

Type Expression (p. 1384)<Quantity (p. 1422)>

Read Only No

ScaleFactorExtension

No details are provided for this entry.

Properties

Probability

No details are provided for this entry.

Type string (p. 1438)

Read Only Yes

ScaleFactor

No details are provided for this entry.

Type ScaleFactor (p. 1429)

Read Only No

ScaleFactorValue

No details are provided for this entry.

Type double (p. 1378)

Read Only No

ShearMomentDiagram

No details are provided for this entry.

Properties

Location

The Location over which the Result is to be evaluated.

Type LocationSet (p. 1401)

Read Only No

NormalizeData

Normalize data so they are displayed as percentages

Type bool (p. 1360)

Read Only No

RelativeTo

The Reference Frame which the result is relative to

Type ReferenceFrame (p. 1424)

Read Only Yes

ReverseDirection

Reverse the direction of the data when it's sent to the graph

Type bool (p. 1360)

Read Only No

SimulationStep

The Simulation Step object reference for a multi-step analysis.

Type SolutionStep (p. 1434)

Read Only No

Variable

Data Model support for variable selector.

Type Expression (p. 1384)<Quantity (p. 1422)>

Read Only No

SlotJoint

This class represents the properties of a slot joint connection between two locations. It is added to a JointBehavior if joint type is set as slot.

Properties

No Properties.

SmdOutputExtension

No details are provided for this entry.

Properties

Maximum

No details are provided for this entry.

Type Quantity (p. 1422)

Read Only Yes

Minimum

No details are provided for this entry.

Type Quantity (p. 1422)

Read Only Yes

SolidModelingOptions

SolidModelingOptions is a class that encapsulates various advanced solver element formulations available to users. Normally an end user never deals with them as solver defaults are good enough, but we do offer a scheme to override it.

Properties

BrickIntegrationType

IntegrationType of elements used by solvers such as Full or Reduced.

Type IntegrationType (p. 1395)

Read Only No

SphericalJoint

This class represents the properties of a spherical joint connection between two locations. It is added to a JointBehavior if joint type is set as spherical.

Properties

No Properties.

SpotWeld

This class represents a spot weld connection between two locations.

Properties

Behavior

It displays the contact behavior referenced by this connection.

Type InterfaceBehavior (p. 1395)

Read Only No

CreationMode

It tells if this connection was created automatically by the program or created manually by the user

Type CreationMode (p. 1370)

Read Only No

Location1

This is the first location of the connection. For contact, this normally refers to the source of the contact.

Type LocationSet (p. 1401)

Read Only No

Location1Bodies

It displays the bodies of the entities referenced in Location 1 of the connection.

Type string (p. 1438)

Read Only Yes

Location2

This is the second location of the connection. For contact, this normally refers to the target of the contact.

Type LocationSet (p. 1401)

Read Only No

Location2Bodies

It displays the bodies of the entities referenced in Location 2 of the connection.

Type string (p. 1438)

Read Only Yes

Spring

This is a class for spring connection

Properties

Behavior

It displays the contact behavior referenced by this connection.

Type InterfaceBehavior (p. 1395)

Read Only No

CreationMode

It tells if this connection was created automatically by the program or created manually by the user

Type CreationMode (p. 1370)

Read Only No

GroundedLocation

Sets a location to be grounded

Type GroundedLocation (p. 1389)

Read Only No

Location1

This is the first location of the connection. For contact, this normally refers to the source of the contact.

Type LocationSet (p. 1401)

Read Only No

Location1Bodies

It displays the bodies of the entities referenced in Location 1 of the connection.

Type string (p. 1438)

Read Only Yes

Location2

This is the second location of the connection. For contact, this normally refers to the target of the contact.

Type LocationSet (p. 1401)

Read Only No

Location2Bodies

It displays the bodies of the entities referenced in Location 2 of the connection.

Type string (p. 1438)

Read Only Yes

RelativeTo

No details are provided for this entry.

Type ReferenceFrame (p. 1424)

Read Only No

SpringLength

The length of the spring

Type Quantity (p. 1422)

Read Only Yes

SpringBehavior

SpringBehavior is a ConnectionBehavior that models linear and nolinear spring connection between two locations.

Properties

BehaviorType

No details are provided for this entry.

Type SpringBehaviorType (p. 1435)

Read Only No

Damping

Damping value for the spring

Type Expression (p. 1384)<Quantity (p. 1422)>

Read Only No

DampingX

No details are provided for this entry.

Type Expression (p. 1384)<Quantity (p. 1422)>

Read Only No

DampingY

No details are provided for this entry.

Type Expression (p. 1384)<Quantity (p. 1422)>

Read Only No

DampingZ

No details are provided for this entry.

Type Expression (p. 1384)<Quantity (p. 1422)>

Read Only No

FixedTranslations

No details are provided for this entry.

Type DoF (p. 1378)

Read Only No

FreeRotations

No details are provided for this entry.

Type DoF (p. 1378)

Read Only No

SpringType

Spring type such as Longitudinal or Torsional

Type SpringType (p. 1435)

Read Only No

Stiffness

Stiffness coefficient value for the spring

Type Expression (p. 1384)<Quantity (p. 1422)>

Read Only No

StiffnessX

No details are provided for this entry.

Type Expression (p. 1384)<Quantity (p. 1422)>

Read Only No

StiffnessY

No details are provided for this entry.

Type Expression (p. 1384)<Quantity (p. 1422)>

Read Only No

StiffnessZ

No details are provided for this entry.

Type Expression (p. 1384)<Quantity (p. 1422)>

Read Only No

SpringPreloadSettings

This class represents preload settings for a spring connection between two locations.

Properties

InitialCondition

Enter preload value

Type Quantity (p. 1422)

Read Only No

PreloadType

Set the preload type to None, Load, Length, Torque or Rotation load and Length are available for longitudical springs only. Torque and Rotation are available for torsional springs only.

Type SpringPreloadType (p. 1435)

Read Only No

StructuralMass

Structural mass to model one or more selected bodies as one rigid body with known mass and mass moments of inertia.

Properties

ConnectingLocation

This is the location by which point mass is connected to the rest of the model

Type LocationSet (p. 1401)

Read Only No

Formulation

Formulation for the internally created remote point connection.

Type ConnectionEndBehavior (p. 1366)

Read Only No

Location

This is the location where point mass is located.

Type LocationSet (p. 1401)

Read Only No

Mass

Mass value of the point mass

Type Expression (p. 1384)<Quantity (p. 1422)>

Read Only No

MassMomentOfInertiaX

Mass moment of interia of point mass about X axis

Type Expression (p. 1384)<Quantity (p. 1422)>

Read Only No

MassMomentOfInertiaY

Mass moment of interia of point mass about Y axis

Type Expression (p. 1384)<Quantity (p. 1422)>

Read Only No

MassMomentOfInertiaZ

Mass moment of interia of point mass about Z axis

Type Expression (p. 1384)<Quantity (p. 1422)>

Read Only No

Support

Support is a constraint to precribe zero values to active degrees of freedom for a given model at a given location.

Properties

FixedDOFs

Collection of degrees of freedom which are fixed.

Type String[] (p. 1438)

Read Only No

Location

Location of the boundary condition

Type LocationSet (p. 1401)

Read Only No

RelativeTo

Degrees of freedom are defined with respect to this reference frame.

Type ReferenceFrame (p. 1424)

Read Only No

SupportType

Type of support : Fixed, Free or User defined.

Type SupportType (p. 1439)

Read Only No

TemperatureCondition

Temperature Condition is scalar load to prescribe temperature to a given location(bodies or elements) in a model in pure structural solutions.

Properties

Location

Location of the boundary condition

Type LocationSet (p. 1401)

Read Only No

Magnitude

Applied value of load

Type Expression (p. 1384)<Quantity (p. 1422)>

Read Only No

ReferenceFrame

Field variables of the spatial variation expression are defined with respect to this reference frame.

Type ReferenceFrame (p. 1424)

Read Only No

TemperatureConstraint

Temperature Constraint is a scalar constraint to precribe known value of temperature in solutions that suppport thermal degrees of freedom.

Properties

Location

Location of the boundary condition

Type LocationSet (p. 1401)

Read Only No

Magnitude

Applied value of constraint

Type Expression (p. 1384)<Quantity (p. 1422)>

Read Only No

ThermalConductance

This class represents the thermal conductance at the contact interface. It gets added when selected physics includes thermal and formulation is not MPC.

Properties

Conductance

Contact conductance value

Type Expression (p. 1384)<Quantity (p. 1422)>

Read Only No

ConductanceControl

Switch to use program controlled conductance value or enter value.

Type ConductanceControlType (p. 1366)

Read Only No

ThermalMass

Thermal mass to model one or more selected bodies with known thermal capacitance

Properties

Capacitance

Thermal capacitance of the point mass

Type Expression (p. 1384)<Quantity (p. 1422)>

Read Only No

ConnectingLocation

This is the location by which thermal mass is connected to the rest of the model

Type LocationSet (p. 1401)

Read Only No

Formulation

Formulation for the internally created remote point connection.

Type EndBehavior (p. 1380)

Read Only No

Location

This is the location where thermal mass is located.

Type LocationSet (p. 1401)

Read Only No

TranslationalJoint

This class represents the properties of a translational joint connection between two locations. It is added to a JointBehavior if joint type is set as translational.

Properties

No Properties.

TranslationRotation

Extension object for prescribing displacement by both translation and rotation components.

Properties

RelativeTo

Degrees of freedom are defined with respect to this reference frame.

Type ReferenceFrame (p. 1424)

Read Only No

RotationX

The X rotation component

Type Expression (p. 1384)<Quantity (p. 1422)>

Read Only No

RotationY

The Y rotation component

Type Expression (p. 1384)<Quantity (p. 1422)>

Read Only No

RotationZ

The Z rotation component

Type Expression (p. 1384)<Quantity (p. 1422)>

Read Only No

TranslationX

The X translation component

Type Expression (p. 1384)<Quantity (p. 1422)>

Read Only No

TranslationY

The Y translation component

Type Expression (p. 1384)<Quantity (p. 1422)>

Read Only No

TranslationZ

The Z translation component

Type Expression (p. 1384)<Quantity (p. 1422)>

Read Only No

UniversalJoint

This class represents the properties of a universal joint connection between two locations. It is added to a JointBehavior if joint type is set as universal.

Properties

No Properties.

UnknownJoint

This class is used only for initialization purposes if joint type is unknown.

Properties

No Properties.

Vector

Vector is class used by all Boundary Conditions and Connection that support a Vector Behavior.

Properties

DefineBy

This property sets the type of vector specification which could be by Components or MagnitudeAndDirection or NormalToSurface.

Type VectorDefineBy (p. 1449)

Read Only No

RelativeTo

Vector components or direction has been specified with respect to this reference frame.

Type ReferenceFrame (p. 1424)

Read Only No

VectorByComponents

VectorByComponents is class to define a vector by Components.

Properties

Component1

The x component

Type Expression (p. 1384)<Quantity (p. 1422)>

Read Only No

Component2

The y component

Type Expression (p. 1384)<Quantity (p. 1422)>

Read Only No

Component3

The z component

Type Expression (p. 1384)<Quantity (p. 1422)>

Read Only No

CoordinateType

The type of coordinate system to be used which could be Cartesian or Cylindrical or Spherical.

Type TripletType (p. 1445)

Read Only Yes

RelativeTo

Vector components has been specified with respect to this reference frame.

Type ReferenceFrame (p. 1424)

Read Only No

VectorByGeometry

ByGeometry is class to define a vector using geometry.

Properties

Magnitude

Magnitude of the vector

Type Expression (p. 1384)<Quantity (p. 1422)>

Read Only No

VectorByMagnitudeAndDirection

VectorByMagnitudeAndDirection is class to define a vector by magnitude and direction.

Properties

Direction

User can select a geometric entity for this property which will be used to infer a direction for the vector.

Type LocationSet (p. 1401)

Read Only No

Magnitude

Magnitude of the vector

Type Expression (p. 1384)<Quantity (p. 1422)>

Read Only No

ReverseDirection

This property can be used to reverse the direction which is inferred from the geometric selection.

Type bool (p. 1360)

Read Only No

VectorConstraint

A Vector Constraint is constraint that has an underneath vectors like Displacement, Velocity etc

Properties

Location

Location of the boundary condition

Type LocationSet (p. 1401)

Read Only No

VectorInitialCondition

Initial conditions is the abstraction over all kinds of vector initial conditions.

Properties

Location

Location for InitialCondition

Type LocationSet (p. 1401)

Read Only No

VectorLoad

VectorLoad is an abstraction over loads that are vector by nature such as Force, Moment, Traction etc.

Properties

Location

Location of the boundary condition

Type LocationSet (p. 1401)

Read Only No

Voltage

Voltage is a scalar constraint to precribe known value of voltage in solutions that suppport electric degrees of freedom.

Properties

Location

Location of the boundary condition

Type LocationSet (p. 1401)

Read Only No

Magnitude

Applied value of constraint

Type Expression (p. 1384)<Quantity (p. 1422)>

Read Only No

Mapdl Data Entities

AdditionalConvergenceControls

AdditionalConvergenceControls is a Solver Settings class that allows defining nonlinear controls such as line search, predictor etc. as exposed by MAPDL solver.

Properties

EquilibriumIterations

Choose number of equilibrium iterations for non-linear solution in MAPDL solver.

Type int (p. 1394)

Read Only No

EquilibriumIterationsKey

Choose equilibrium iterations keys for non-linear solution in MAPDL solver.

Type IterationKey (p. 1397)

Read Only No

LineSearch

Activate Line Search for non-linear solution in MAPDL solver.

Type ProgramControlType (p. 1421)

Read Only No

PredictorCorrector

Activate Predictor for non-linear solution in MAPDL solver.

Type PredictorKey (p. 1419)

Read Only No

BucklingControls

No details are provided for this entry.

Properties

IncludeNegativeLoadMultiplier

No details are provided for this entry.

Type NegativeBucklingLoadMultiplier (p. 1408)

Read Only No

ContactFormulation

ContactFormulation is a class which represents various contact formulations as available from MAPDL solver.

Properties

ConstraintType

Specify the constraint type for contacts.

Type ContactConstraintType (p. 1368)

Read Only No

DetectAsymmetryAutomatically

A flag to notify the program if auto-asymmetry should be used in the physics solution.

Type bool (p. 1360)

Read Only No

DetectionType

Specify the location of contact detection to be used in the analysis in order to obtain a good convergence.

Type ContactDetectionPoint (p. 1368)

Read Only No

DynamicFrictionCoefficient

Dynamic coefficient of friction.

Type Expression (p. 1384)<Quantity (p. 1422)>

Read Only No

Formulation

Set the contact formulation to be used by the solver for a particular contact pair.

Type FormulationType (p. 1387)

Read Only No

IncludeShellThickness

Switch to include shell thickness or not in the contact analysis.

Type bool (p. 1360)

Read Only No

PhysicsToTransfer

Method for selecting the physics type: Program controlled or Manual.

Type ContactPhysicsToTransfer (p. 1368)

Read Only No

SmallSliding

Possible options for a small sliding to occur in bonded or no separation contacts.

Type SmallSlidingSetting (p. 1433)

Read Only No

TransferredPhysics

Specified physics type(s) for the contact formulation.

Type PhysicsType (p. 1415)

Read Only No

ConvergenceControl

ConvergenceControl is Solver Settings class that allows defining various non-linear convergence techniques as exposed by MAPDL solver such as CNVTOL command.

Properties

Convergence

An enumeration to turn on, off Convergence Controls as available from MAPDL solver.

Type ProgramControlType (p. 1421)

Read Only No

MinimumReferenceValue

Convergence minimum reference value defined for a given Convergence Control as available from MAPDL solver.

Type Quantity (p. 1422)

Read Only No

ReferenceValue

Convergence value defined for a given Convergence Control as available from MAPDL solver.

Type Quantity (p. 1422)

Read Only No

ReferenceValueSpecification

Method for selecting the specification type: Program controlled or Manual

Type ReferenceValueSpecification (p. 1424)

Read Only No

Tolerance

Convergence tolerance value defined for a given Convergence Control as available from MAPDL solver.

Type double (p. 1378)

Read Only No

ConvergenceControls

ConvergenceCriterias is a Solution Progression class that allows defining convergence controls as exposed by MAPDL solver.

Properties

No Properties.

CyclicSymmetryConstraint

CyclicSymmetryConstraint is a manufacturing constraint to apply a cyclic symmetry constraint in an optimization analysis.

Properties

AxialDirection

Select an axis direction of the selected reference frame for the cyclic symmetry.

Type AxisType (p. 1357)

Read Only No

AxialDirectionDefinition

Select a method for defining the axial direction: based on a geometry selection or a reference frame.

Type AxialDirectionDefinedBy (p. 1356)

Read Only No

AxialDirectionLocation

Select a face and use its normal as the axial direction for the cyclic symmetry.

Type LocationSet (p. 1401)

Read Only No

NumberOfSectors

Specified number of sectors for cyclic symmetry.

Type int (p. 1394)

Read Only No

ReferenceFrame

Select a reference frame for the axial direction.

Type ReferenceFrame (p. 1424)

Read Only No

GeneralNonLinearControls

GeneralNonLinearControls is Solver Settings class that allows defining some general nonlinear techniques such as large deflection or Newton Raphson Methods as exposed by MAPDL solver.

Properties

LargeDeflection

A switch to activate geometric nonlinearity.

Type ProgramControlType (p. 1421)

Read Only No

NRSetting

An enumeration to select various Newton Raphson Methods as available from MAPDL solver.

Type NewtonRaphsonOption (p. 1408)

Read Only No

GeneralSolutionControls

GeneralSolutionControls is Solver Settings class that allows defining some general solution controls such as unit sytems, Equation solver types etc. as exposed by MAPDL solver.

Properties

InertiaRelief

Inertia Relief as available from MAPDL solver.

Type bool (p. 1360)

Read Only No

InitialContactTreatment

Set the initial contact treatment option.

Type InitialContactTreatment (p. 1393)

Read Only No

LimitSearchToRange

Activate search to a given range of frequencies for frequency domain solutions as available from MAPDL solver.

Type YesOrNo (p. 1451)

Read Only No

MatrixSolver

Equation Solver Types as available from MAPDL solver.

Type EquationSolverType (p. 1382)

Read Only No

MaxModesToFind

Maximum modes to find for modal solutions as available from MAPDL solver.

Type int (p. 1394)

Read Only No

RangeMaximum

A value of maximum frequency for frequency domain solutions with a given range as available from MAPDL solver.

Type Quantity (p. 1422)

Read Only No

RangeMinimum

A value of minimum frequency for frequency domain solutions with a given range as available from MAPDL solver.

Type Quantity (p. 1422)

Read Only No

SolverDamping

An enumeration for damped modal solver types as available from MAPDL solver.

Type DampedSolverType (p. 1371)

Read Only No

SolverUnitSystem

Solver unit systems as available from MAPDL solver.

Type string (p. 1438)

Read Only No

MapdlFixedJoint

MapdlFixedJoint is a class to represents Fixed Joint Connection Behavior as available from MAPDL solver

Properties

Formulation

JointFormulation types used by Mapdl Fixed Joint Connection Behavior

Type JointFormulation (p. 1397)

Read Only No

ReductionMethod

ReductionMethod used by Mapdl Fixed Joint Connection Behavior

Type JointReductionMethod (p. 1398)

Read Only No

MAPDLFunctionProvider

This is a class that implements MAPDL expressions like AllContacts(), AllJoints() etc.

Properties

DisplayText

The general property that defines the user-visible name of an entity. This property is defined for all data entities but is used only in those entities that present a label in the user interface.

Type string (p. 1438)

Read Only No

MemberSizeConstraint

MemberSizeConstraint is a manufacturing constraint to apply the minimum and maximum constraints on the memeber size in an optimization analysis

Properties

MaximumConstraintType

Maximum constraint type of the member size constraint: Program Controlled or Manual

Type ConstraintType (p. 1367)

Read Only No

MaximumSize

Maximum size of the member size constraint

Type Expression (p. 1384)<Quantity (p. 1422)>

Read Only No

MinimumConstraintType

Minimum constraint type of the member size constraint: Program Controlled or Manual

Type ConstraintType (p. 1367)

Read Only No

MinimumSize

Minimum size of the member size constraint

Type Expression (p. 1384)<Quantity (p. 1422)>

Read Only No

OptimizationControls

This is an extension object to SolverSettings for optimization controls

Properties

ConvergenceAccuracy

Sets a percentage of convergence accuracy for the optimization process

Type Expression (p. 1384)<Quantity (p. 1422)>

Read Only No

MaximumNumberOfIterations

Sets the maximum number of iteration for the optimization process

Type int (p. 1394)

Read Only No

MaxNumberOfIntermediateFiles

Sets the maximum number of intermediate files for the optimization process

Type uint (p. 1446)

Read Only No

MinimumNormalizedDensity

Sets the minimum normalized density for the optimization process

Type Expression (p. 1384)<Quantity (p. 1422)>

Read Only No

OptimizationSolver

Sets a solver type for the optimization process

Type OptimizationSolverType (p. 1410)

Read Only No

OutputSpecification

OutputSpecification is a class that defines each individual item that can be controlled to be stored in a result file such as Deformation, Stress or Strain.

Properties

EquallySpacedTimePoints

Time Points frequency of OutputSpecification

Type int (p. 1394)

Read Only No

Frequency

Frequency of OutputSpecification, how often a given output type is written to result file

Type CalculationFrequency (p. 1361)

Read Only No

Location

Location for OutputSpecification

Type LocationSet (p. 1401)

Read Only No

OutputType

OutputType for OutputSpecification such as Deformation, Stress or Strain

Type OutputType (p. 1412)

Read Only No

RecurrenceRate

RecurrenceRate of OutputSpecification

Type int (p. 1394)

Read Only No

OutputSpecifications

OutputSpecifications is a class that is repository of individual OutputSpecification such as Deformation, Stress or Strain.

Properties

No Properties.

PullOutDirection

PullOutDirection is a manufacturing constraint used in optimization analysis

Properties

Axis

Axis used as a reference for the pull out direction

Type AxisType (p. 1357)

Read Only No

Direction

Direction indicates the direction of the pull out

Type DirectionType (p. 1376)

Read Only No

ReferenceFrame

Field variables of the spatial variation expression are defined with respect to this reference frame.

Type ReferenceFrame (p. 1424)

Read Only No

RandomVibrationControls

RandomVibrationControls, an extension object to Solver Settings class, allows to define various analysis settings for random vibration as exposed by MAPDL solver.

Properties

ConstantDampingFactor

A value for constant damping factor used in a random vibration analysis.

Type double (p. 1378)

Read Only No

ExcludeInsignificantModes

Switch to exclude the insignificant modes from a random vibration analysis.

Type bool (p. 1360)

Read Only No

ModeSignificanceLevel

A value for mode significance level to exclude insignificant modes in a random vibration analysis.

Type double (p. 1378)

Read Only No

Stabilization

Stabilization is a class that allows defining Stablization parameters for nonlinear solutions as exposed by MAPDL solver.

Properties

DampingFactor

Value for Damping Factor.

Type double (p. 1378)

Read Only No

EnergyDissipationRatio

Value for Energy Dissipation Ratio.

Type double (p. 1378)

Read Only No

ForceLimit

Value of Force Limit.

Type double (p. 1378)

Read Only No

StabilizationKey

Activate various Stabilization Keys in MAPDL Solution.

Type StabilizationKey (p. 1435)

Read Only No

SubStepOption

Activate various substep options.

Type SubStepOption (p. 1438)

Read Only No

StressConstraint

StressConstraint is a stress constraint used in an optimization analysis

Properties

Location

Location of the stress constraint

Type LocationSet (p. 1401)

Read Only No

MaximumStress

Maximum value of the stress constraint

Type Expression (p. 1384)<Quantity (p. 1422)>

Read Only No

Type

Type displays the type of the stress constraint

Type StressType (p. 1438)

Read Only Yes

SymmetryConstraint

SymmetryConstraint is a manufacturing constraint to apply a symmetry constraint in an optimization analysis.

Properties

PlaneOfSymmetry

Plane selection for the plane of symmetry.

Type Plane (p. 1416)

Read Only No

TimeStepOptions

TimeStepOptions is a Solver Settings class that help define the time step controls available in MAPDL solver.

Properties

DefinedBy

Define Auto Time Stepping by Time or Substeps as available in MAPDL solver.

Type AutoTimeSteppingDefinedBy (p. 1356)

Read Only No

InitialNumberOfSubsteps

When defining auto time stepping by substeps, define Initial Substeps as available in MAPDL solver.

Type int (p. 1394)

Read Only No

InitialTimeStep

When defining auto time stepping by time steps, define Initial Time Step as available in MAPDL solver.

Type Quantity (p. 1422)

Read Only No

MaximumNumberOfSubsteps

When defining auto time stepping by substeps, define Maximum Substeps as available in MAPDL solver.

Type int (p. 1394)

Read Only No

MaximumTimeStep

When defining auto time stepping by time steps, define Maximum Time Step as available in MAPDL solver.

Type Quantity (p. 1422)

Read Only No

MinimumNumberOfSubsteps

When defining auto time stepping by substeps, define Minimum Substeps as available in MAPDL solver.

Type int (p. 1394)

Read Only No

MinimumTimeStep

When defining auto time stepping by time steps, define Minimum Time Step as available in MAPDL solver.

Type Quantity (p. 1422)

Read Only No

NumberOfSubsteps

When auto time stepping is off define number of substeps as available in MAPDL solver.

Type int (p. 1394)

Read Only No

Substepping

Define Substepping options such as specified range, fixed number or program controlled as available in MAPDL solver.

Type Substepping (p. 1438)

Read Only No

TimeIntegration

Turn on or off Time Integration for inertia effects as available in MAPDL solver.

Type OnOffSwitch (p. 1409)

Read Only No

TimeStep

When auto time stepping is off define time steps as available in MAPDL solver.

Type Quantity (p. 1422)

Read Only No

TrimOptimization

TrimOptimization is a class to represents various trimming options available for contact connection.

Properties

Tolerance

Tolerance to be used when trimming is on.

Type Quantity (p. 1422)

Read Only No

Type

Trim Type to control trimming such as keeping it on, off or Program Controllled.

Type TrimType (p. 1445)

Read Only No

ZeroThermalStrainTemperature

ZeroThermalStrainTemperature is a class that hold temperature at zero thermal strain.

Properties

Magnitude

Magnitude of ZeroThermalStrainTemperature

Type Quantity (p. 1422)

Read Only No

Fluent Data Entities

BoundaryMonitor

No details are provided for this entry.

Properties

AllowSubsetLocation

No details are provided for this entry.

Type bool (p. 1360)

Read Only No

Boundary

No details are provided for this entry.

Type FluentBoundary (p. 1386)

Read Only No

Location

No details are provided for this entry.

Type LocationSet (p. 1401)

Read Only No

MonitoredOperation

No details are provided for this entry.

Type SurfaceMonitorOperation (p. 1439)

Read Only No

MonitoredVariable

No details are provided for this entry.

Type SurfaceMonitorVariable (p. 1439)

Read Only No

SurfaceMonitorConvergenceCriteria

No details are provided for this entry.

Type double (p. 1378)

Read Only No

Flow

No details are provided for this entry.

Properties

AngularSpeed

No details are provided for this entry.

Type Expression (p. 1384)<Quantity (p. 1422)>

Read Only No

AxialSpeed

No details are provided for this entry.

Type Expression (p. 1384)<Quantity (p. 1422)>

Read Only No

FanPressureRise

No details are provided for this entry.

Type Expression (p. 1384)<Quantity (p. 1422)>

Read Only No

FlowDirection

No details are provided for this entry.

Type FlowSpecification (p. 1386)

Read Only No

IncludeFan

No details are provided for this entry.

Type bool (p. 1360)

Read Only No

InletSpec

No details are provided for this entry.

Type InletType (p. 1394)

Read Only No

Location

No details are provided for this entry.

Type LocationSet (p. 1401)

Read Only No

MassFlowRate

No details are provided for this entry.

Type Expression (p. 1384)<Quantity (p. 1422)>

Read Only No

MassFractionComponent1

No details are provided for this entry.

Type Expression (p. 1384)<Quantity (p. 1422)>

Read Only No

NormalSpeed

No details are provided for this entry.

Type Expression (p. 1384)<Quantity (p. 1422)>

Read Only No

OutletSpec

No details are provided for this entry.

Type OutletType (p. 1411)

Read Only No

ReferenceFrame

No details are provided for this entry.

Type ReferenceFrame (p. 1424)

Read Only No

SolvedSpecie

No details are provided for this entry.

Type MaterialAssignment (p. 1402)

Read Only No

StaticPressure

No details are provided for this entry.

Type Expression (p. 1384)<Quantity (p. 1422)>

Read Only No

TotalPressure

No details are provided for this entry.

Type Expression (p. 1384)<Quantity (p. 1422)>

Read Only No

FlowTemperature

No details are provided for this entry.

Properties

Boundary

No details are provided for this entry.

Type FluentBoundary (p. 1386)

Read Only No

Temperature

No details are provided for this entry.

Type Expression (p. 1384)<Quantity (p. 1422)>

Read Only No

FluentBoundary

No details are provided for this entry.

Properties

Location

No details are provided for this entry.

Type LocationSet (p. 1401)

Read Only No

FluentLaunchControls

No details are provided for this entry.

Properties

MpiType

No details are provided for this entry.

Type MpiTypeValue (p. 1407)

Read Only No

NumberOfProcesses

No details are provided for this entry.

Type int (p. 1394)

Read Only No

ProcessDistribution

No details are provided for this entry.

Type ProcessDistributionOption (p. 1421)

Read Only No

FluentLocationInfo

No details are provided for this entry.

Properties

No Properties.

FluentPhysicsOptions

No details are provided for this entry.

Properties

Gravity

No details are provided for this entry.

Type GravityDefinition (p. 1389)

Read Only No

IncludeBuoyancy

No details are provided for this entry.

Type bool (p. 1360)

Read Only No

IncludeGravity

No details are provided for this entry.

Type bool (p. 1360)

Read Only No

OperatingPressure

No details are provided for this entry.

Type Expression (p. 1384)<Quantity (p. 1422)>

Read Only No

OperatingTemperature

No details are provided for this entry.

Type Expression (p. 1384)<Quantity (p. 1422)>

Read Only No

RadiationModel

No details are provided for this entry.

Type RadiationSpecification (p. 1422)

Read Only No

TurbulenceModel

No details are provided for this entry.

Type TurbulenceSpecification (p. 1445)

Read Only No

FluentSolveError

No details are provided for this entry.

Properties

No Properties.

FluentSource

No details are provided for this entry.

Properties

Location

No details are provided for this entry.

Type LocationSet (p. 1401)

Read Only No

FluidEnergySource

No details are provided for this entry.

Properties

EnergySource

No details are provided for this entry.

Type Expression (p. 1384)<Quantity (p. 1422)>

Read Only No

Location

No details are provided for this entry.

Type LocationSet (p. 1401)

Read Only No

HeatTransfer

No details are provided for this entry.

Properties

ConvectionTemperature

No details are provided for this entry.

Type Expression (p. 1384)<Quantity (p. 1422)>

Read Only No

Emissivity

No details are provided for this entry.

Type Expression (p. 1384)<double (p. 1378)>

Read Only No

HeatFlow

No details are provided for this entry.

Type Expression (p. 1384)<Quantity (p. 1422)>

Read Only No

HeatFlux

No details are provided for this entry.

Type Expression (p. 1384)<Quantity (p. 1422)>

Read Only No

HeatTransferCoefficient

No details are provided for this entry.

Type Expression (p. 1384)<Quantity (p. 1422)>

Read Only No

Location

No details are provided for this entry.

Type LocationSet (p. 1401)

Read Only No

RadiativeTemperature

No details are provided for this entry.

Type Expression (p. 1384)<Quantity (p. 1422)>

Read Only No

Temperature

No details are provided for this entry.

Type Expression (p. 1384)<Quantity (p. 1422)>

Read Only No

ThermalSpec

No details are provided for this entry.

Type ThermalWallSpecification (p. 1441)

Read Only No

InterfaceModel

No details are provided for this entry.

Properties

FilmMaterial

No details are provided for this entry.

Type Material (p. 1402)

Read Only No

FilmThickness

No details are provided for this entry.

Type Expression (p. 1384)<Quantity (p. 1422)>

Read Only No

HeatGenerationRate

No details are provided for this entry.

Type Expression (p. 1384)<Quantity (p. 1422)>

Read Only No

Interface

No details are provided for this entry.

Type RegionInterface2 (p. 1425)

Read Only No

InterfaceModelSpec

No details are provided for this entry.

Type InterfaceModelSpecification (p. 1396)

Read Only No

InterfaceMonitor

No details are provided for this entry.

Properties

AllowSubsetLocation

No details are provided for this entry.

Type bool (p. 1360)

Read Only No

Boundary

No details are provided for this entry.

Type FluentBoundary (p. 1386)

Read Only No

Location

No details are provided for this entry.

Type LocationSet (p. 1401)

Read Only No

MonitoredOperation

No details are provided for this entry.

Type SurfaceMonitorOperation (p. 1439)

Read Only No

MonitoredVariable

No details are provided for this entry.

Type SurfaceMonitorVariable (p. 1439)

Read Only No

RegionInterface

No details are provided for this entry.

Type RegionInterface2 (p. 1425)

Read Only No

SideSelection

No details are provided for this entry.

Type InterfaceSideSelection (p. 1396)

Read Only No

SurfaceMonitorConvergenceCriteria

No details are provided for this entry.

Type double (p. 1378)

Read Only No

LocationInfo

No details are provided for this entry.

Properties

No Properties.

MeshSurfaceZone

No details are provided for this entry.

Properties

No Properties.

MeshVolumeZone

No details are provided for this entry.

Properties

No Properties.

MomentumSource

No details are provided for this entry.

Properties

Location

No details are provided for this entry.

Type LocationSet (p. 1401)

Read Only No

MomentumXSource

No details are provided for this entry.

Type Expression (p. 1384)<Quantity (p. 1422)>

Read Only No

MomentumYSource

No details are provided for this entry.

Type Expression (p. 1384)<Quantity (p. 1422)>

Read Only No

MomentumZSource

No details are provided for this entry.

Type Expression (p. 1384)<Quantity (p. 1422)>

Read Only No

PhysicsRegionLocationInfo

No details are provided for this entry.

Properties

No Properties.

PointMonitor

No details are provided for this entry.

Properties

AllowSubsetLocation

No details are provided for this entry.

Type bool (p. 1360)

Read Only No

Boundary

No details are provided for this entry.

Type FluentBoundary (p. 1386)

Read Only No

Location

No details are provided for this entry.

Type LocationSet (p. 1401)

Read Only No

MonitoredOperation

No details are provided for this entry.

Type SurfaceMonitorOperation (p. 1439)

Read Only No

MonitoredVariable

No details are provided for this entry.

Type SurfaceMonitorVariable (p. 1439)

Read Only No

PointX

No details are provided for this entry.

Type double (p. 1378)

Read Only No

PointY

No details are provided for this entry.

Type double (p. 1378)

Read Only No

PointZ

No details are provided for this entry.

Type double (p. 1378)

Read Only No

SurfaceMonitorConvergenceCriteria

No details are provided for this entry.

Type double (p. 1378)

Read Only No

PorositySource

No details are provided for this entry.

Properties

ConeHalfAngle

No details are provided for this entry.

Type Expression (p. 1384)<Quantity (p. 1422)>

Read Only No

Direction1X

No details are provided for this entry.

Type Expression (p. 1384)<double (p. 1378)>

Read Only No

Direction1Y

No details are provided for this entry.

Type Expression (p. 1384)<double (p. 1378)>

Read Only No

Direction1Z

No details are provided for this entry.

Type Expression (p. 1384)<double (p. 1378)>

Read Only No

Direction2X

No details are provided for this entry.

Type Expression (p. 1384)<double (p. 1378)>

Read Only No

Direction2Y

No details are provided for this entry.

Type Expression (p. 1384)<double (p. 1378)>

Read Only No

Direction2Z

No details are provided for this entry.

Type Expression (p. 1384)<double (p. 1378)>

Read Only No

InertialResistance

No details are provided for this entry.

Type Expression (p. 1384)<Quantity (p. 1422)>

Read Only No

InertialResistanceX

No details are provided for this entry.

Type Expression (p. 1384)<Quantity (p. 1422)>

Read Only No

InertialResistanceY

No details are provided for this entry.

Type Expression (p. 1384)<Quantity (p. 1422)>

Read Only No

InertialResistanceZ

No details are provided for this entry.

Type Expression (p. 1384)<Quantity (p. 1422)>

Read Only No

IsConical

No details are provided for this entry.

Type bool (p. 1360)

Read Only No

Location

No details are provided for this entry.

Type LocationSet (p. 1401)

Read Only No

Permeability

No details are provided for this entry.

Type Expression (p. 1384)<Quantity (p. 1422)>

Read Only No

PermeabilityX

No details are provided for this entry.

Type Expression (p. 1384)<Quantity (p. 1422)>

Read Only No

PermeabilityY

No details are provided for this entry.

Type Expression (p. 1384)<Quantity (p. 1422)>

Read Only No

PermeabilityZ

No details are provided for this entry.

Type Expression (p. 1384)<Quantity (p. 1422)>

Read Only No

PointOnConeAxisX

No details are provided for this entry.

Type Expression (p. 1384)<Quantity (p. 1422)>

Read Only No

PointOnConeAxisY

No details are provided for this entry.

Type Expression (p. 1384)<Quantity (p. 1422)>

Read Only No

PointOnConeAxisZ

No details are provided for this entry.

Type Expression (p. 1384)<Quantity (p. 1422)>

Read Only No

Porosity

No details are provided for this entry.

Type Expression (p. 1384)<double (p. 1378)>

Read Only No

PorosityMediaMaterial

No details are provided for this entry.

Type PorosityMediaMaterial (p. 1417)

Read Only No

PorosityType

No details are provided for this entry.

Type PorosityType (p. 1418)

Read Only No

PorousMedia

No details are provided for this entry.

Type MaterialAssignment (p. 1402)

Read Only No

ReferenceBoundary

No details are provided for this entry.

Properties

Boundary

No details are provided for this entry.

Type FluentBoundary (p. 1386)

Read Only No

RegionInterface2

No details are provided for this entry.

Properties

DefinitionMethod

No details are provided for this entry.

Type CreationMode (p. 1370)

Read Only No

Location1

No details are provided for this entry.

Type LocationSet (p. 1401)

Read Only No

Location2

No details are provided for this entry.

Type LocationSet (p. 1401)

Read Only No

PhysicsRegions1

No details are provided for this entry.

Type List (p. 1400)<PhysicsRegion (p. 1415)>

Read Only No

PhysicsRegions2

No details are provided for this entry.

Type List (p. 1400)<PhysicsRegion (p. 1415)>

Read Only No

RegionInterfaceGenerator2

No details are provided for this entry.

Properties

Location

No details are provided for this entry.

Type LocationSet (p. 1401)

Read Only No

Tolerance

No details are provided for this entry.

Type Quantity (p. 1422)

Read Only No

SolidEnergySource

No details are provided for this entry.

Properties

EnergySource

No details are provided for this entry.

Type Expression (p. 1384)<Quantity (p. 1422)>

Read Only No

Location

No details are provided for this entry.

Type LocationSet (p. 1401)

Read Only No

SolutionControls

No details are provided for this entry.

Properties

AdaptiveTimestepping

No details are provided for this entry.

Type bool (p. 1360)

Read Only No

ComputeLocalScale

No details are provided for this entry.

Type bool (p. 1360)

Read Only No

ConvergenceCriteria

No details are provided for this entry.

Type double (p. 1378)

Read Only No

ConvergenceOption

No details are provided for this entry.

Type ConvergenceOption (p. 1369)

Read Only No

ConvertToPoly

No details are provided for this entry.

Type bool (p. 1360)

Read Only No

EnergyConvergenceCriteria

No details are provided for this entry.

Type double (p. 1378)

Read Only No

ImproveMeshIfRequired

No details are provided for this entry.

Type bool (p. 1360)

Read Only No

IncludePeriodicity

No details are provided for this entry.

Type bool (p. 1360)

Read Only No

NumberOfIterations

No details are provided for this entry.

Type int (p. 1394)

Read Only No

NumberOfTimeSteps

No details are provided for this entry.

Type int (p. 1394)

Read Only No

PhysicalTimeStep

No details are provided for this entry.

Type Quantity (p. 1422)

Read Only No

SurfaceMonitorChainingOperation

No details are provided for this entry.

Type SurfaceMonitorChainingOperation (p. 1439)

Read Only No

TransientConvergenceCriteria

No details are provided for this entry.

Type double (p. 1378)

Read Only No

TurnAllBodiesIntoZones

No details are provided for this entry.

Type bool (p. 1360)

Read Only No

SurfaceLocationInfo

No details are provided for this entry.

Properties

No Properties.

Symmetry

No details are provided for this entry.

Properties

Location

No details are provided for this entry.

Type LocationSet (p. 1401)

Read Only No

ThermalApplication

No details are provided for this entry.

Properties

Boundary

No details are provided for this entry.

Type FluentBoundary (p. 1386)

Read Only No

VolumeMonitor

No details are provided for this entry.

Properties

AllowSubsetLocation

No details are provided for this entry.

Type bool (p. 1360)

Read Only No

Boundary

No details are provided for this entry.

Type FluentBoundary (p. 1386)

Read Only No

Location

No details are provided for this entry.

Type LocationSet (p. 1401)

Read Only No

MonitoredOperation

No details are provided for this entry.

Type SurfaceMonitorOperation (p. 1439)

Read Only No

MonitoredVariable

No details are provided for this entry.

Type SurfaceMonitorVariable (p. 1439)

Read Only No

SurfaceMonitorConvergenceCriteria

No details are provided for this entry.

Type double (p. 1378)

Read Only No

Volume

No details are provided for this entry.

Type UserObject (p. 1448)

Read Only No

Wall

No details are provided for this entry.

Properties

AngularSpeed

No details are provided for this entry.

Type Expression (p. 1384)<Quantity (p. 1422)>

Read Only No

Location

No details are provided for this entry.

Type LocationSet (p. 1401)

Read Only No

MotionSpec

No details are provided for this entry.

Type WallMotionSpecification (p. 1450)

Read Only No

ReferenceFrame

No details are provided for this entry.

Type ReferenceFrame (p. 1424)

Read Only No

SlipSpec

No details are provided for this entry.

Type SlipSpecification (p. 1433)

Read Only No

WallHeatTransfer

No details are provided for this entry.

Properties

Boundary

No details are provided for this entry.

Type FluentBoundary (p. 1386)

Read Only No

ConvectionTemperature

No details are provided for this entry.

Type Expression (p. 1384)<Quantity (p. 1422)>

Read Only No

Emissivity

No details are provided for this entry.

Type Expression (p. 1384)<double (p. 1378)>

Read Only No

HeatFlow

No details are provided for this entry.

Type Expression (p. 1384)<Quantity (p. 1422)>

Read Only No

HeatFlux

No details are provided for this entry.

Type Expression (p. 1384)<Quantity (p. 1422)>

Read Only No

HeatTransferCoefficient

No details are provided for this entry.

Type Expression (p. 1384)<Quantity (p. 1422)>

Read Only No

RadiativeTemperature

No details are provided for this entry.

Type Expression (p. 1384)<Quantity (p. 1422)>

Read Only No

Temperature

No details are provided for this entry.

Type Expression (p. 1384)<Quantity (p. 1422)>

Read Only No

ThermalSpec

No details are provided for this entry.

Type ThermalWallSpecification (p. 1441)

Read Only No

WallRadiation

No details are provided for this entry.

Properties

Boundary

No details are provided for this entry.

Type FluentBoundary (p. 1386)

Read Only No

DiffuseFraction

No details are provided for this entry.

Type Expression (p. 1384)<double (p. 1378)>

Read Only No

Emissivity

No details are provided for this entry.

Type Expression (p. 1384)<double (p. 1378)>

Read Only No

PhysicsCoupling Data Entities

CouplingInterface

Configure a coupling interface.

Properties

MappingControl

Configure controls for mapping.

Type IDictionary (p. 1375)<string (p. 1438), Object (p. 1409)>

Read Only No

Side

Configure one side of a coupling interface.

Type IDictionary (p. 1375)<string (p. 1438), Object (p. 1409)>

Read Only No

Suppress

context to be documented

Type String[] (p. 1438)

Read Only No

TransferToSideTwo

Configure transfers to side two of a coupling interface.

Type IDictionary (p. 1375)<string (p. 1438), Object (p. 1409)>

Read Only No

CouplingInterface

Configure a coupling interface.

Properties

MappingControl

Configure controls for mapping.

Type IDictionary (p. 1375)<string (p. 1438), Object (p. 1409)>

Read Only No

Side

Configure one side of a coupling interface.

Type IDictionary (p. 1375)<string (p. 1438), Object (p. 1409)>

Read Only No

Suppress

context to be documented

Type String[] (p. 1438)

Read Only No

TransferToSideTwo

Configure transfers to side two of a coupling interface.

Type IDictionary (p. 1375)<string (p. 1438), Object (p. 1409)>

Read Only No

PhysicsCoupling

No details are provided for this entry.

Properties

DisplayText

No details are provided for this entry.

Type string (p. 1438)

Read Only No

PhysicsRegion1

No details are provided for this entry.

Type DataObject (p. 1371)

Read Only No

PhysicsRegion2

No details are provided for this entry.

Type DataObject (p. 1371)

Read Only No

Side1Location

No details are provided for this entry.

Type string (p. 1438)

Read Only No

Side2Locations

No details are provided for this entry.

Type Object (p. 1409)

Read Only No

SourceFrequency

No details are provided for this entry.

Type string (p. 1438)

Read Only No

System Coupling

System Coupling Setup

This container holds Set Up data for an instance of System Coupling.

Methods

CreateDataTransfer

Creates a data transfer.

Return The new data transfer that was created.

Type DataReference (p. 1371)

Required Arguments

Name The name for this data transfer.

Type string (p. 1438)

Optional Arguments

SourceParticipant The source of this data transfer.

Type DataReference (p. 1371)

SourceRegion The source Region for this data transfer.

Type DataReference (p. 1371)

SourceVariable The source Variable for this data transfer.

Type DataReference (p. 1371)

TargetParticipant The destination for this data transfer.

Type DataReference (p. 1371)

TargetRegion The destination region for this data transfer.

Type DataReference (p. 1371)

TargetVariable The destination variable for this data transfer.

Type DataReference (p. 1371)

Example

The following example demonstrates creation of a data transfer. It is assumed that user had created two participants (e.g. Transient Structural and Fluid Flow(Fluent) and connected to System Coupling System. System Coupling User's Guide in help documentation provides information on "Tutorial: Oscillating Plate with Two-Way Fluid-Structure Interaction". The same information can be referred to see how two participants can be setup and connect to System Coupling System.

```
# Get System Couplng Setup
system1 = GetSystem(Name="SC")
setup1 = system1.GetContainer(ComponentName="Setup")
# Get source participant (e.g. Transient Structural) , source region and source variable
participant1 = setup1.GetParticipant(Name="Solution")
region1 = participant1.GetRegion(Name="Fluid Solid Interface")
variable1 = region1.GetVariable(Name="Incremental Displacement")
# Get target participant (e.g. Fluid Flow(Fluent), target region and target variable
participant2 = setup1.GetParticipant(Name="Solution 1")
region2 = participant2.GetRegion(Name="wall_deforming")
variable2 = region2.GetVariable(Name="displacement")
# Create a data transfer
 dataTransfer1 = setup1.CreateDataTransfer(
   Name="Data Transfer",
   SourceParticipant=participant1,
   SourceRegion=region1,
   SourceVariable=variable1,
   TargetParticipant=participant2,
   TargetRegion=region2,
   TargetVariable=variable2)
```

ExportSCIFile

Writes the system coupling input file at the specified path.

Required Arguments

Path Fully qualified path of the SCI File.

Type string (p. 1438)

GetAnalysisSettings

Returns the analysis settings entity in a container.

Return Data reference of the analysis settings in the container.

Type DataReference (p. 1371)

GetDataTransfer

Returns the data transfer of a given name in a container.

Return The data transfer that matches the specified name.

Type DataReference (p. 1371)

Required Arguments

Name The name or display name of the data transfer.

Type string (p. 1438)

GetDataTransfers

Returns the collection of data transfers in a container. If no data transfers are in the container, the collection is empty.

Return Collection of the data transfers in the container.

Type DataReferenceSet (p. 1371)

GetExpertSettings

Returns the expert settings entity from a system coupling setup container.

Return Data reference of the expert settings in the container.

Type DataReference (p. 1371)

GetIntermediateRestartDataOutputControls

Returns the intermediate result files output controls from a system coupling setup container. Renamed from ResultFiles to RestartData to avoid confusion for MAPDL users.

Return Data reference of the intermediate result files output controls in the container.

Type DataReference (p. 1371)

GetParticipant

Returns the participant with a given name from a system coupling setup container.

Return The participant that matches the specified name.

Type DataReference (p. 1371)

Required Arguments

Name The internal name or display name of the participant.

Type string (p. 1438)

GetParticipants

Returns the collection of participants in a container. If no participants are in the container, the collection is empty.

Return Collection of the participants in the container.

Type DataReferenceSet (p. 1371)

GetSequenceControls

Returns the sequence controls from a system coupling setup container.

Return Data reference of the sequence controls in the container.

Type DataReference (p. 1371)

ReadRestartPoints

Generates a list of restart points in analysis settings

Data Entities

AnalysisSettings

The entity to store the analysis settings for a coupling run.

Properties

AnalysisType

The coupled analysis type.

Type CoupledAnalysisType (p. 1370)

Read Only No

DisableSolutionUpdate

This flag disables updates if restarts are not supported and solution data exists

Type bool (p. 1360)

Read Only No

DisplayText

The general property that defines the user-visible name of an entity. This property is defined for all data entities but is used only in those entities that present a label in the user interface.

Type string (p. 1438)

Read Only No

DurationDefinedBy

This property specifies how we should determine the end of the coupling run.

Type DurationType (p. 1379)

Read Only No

EndTime

The end time for the coupling run

Type Quantity (p. 1422)

Read Only No

Initialization

The initialziation setting.

Type InitializationType (p. 1394)

Read Only No

MaximumIteration

The maximum number of iterations per coupling step for the coupling run.

Type int (p. 1394)

Read Only No

MinimumIteration

The minimum number of iterations per coupling step for the coupling run.

Type int (p. 1394)

Read Only No

NumberOfSteps

The number of time steps for the coupling run.

Type int (p. 1394)

Read Only No

RestartStep

The restart step for the coupling run.

Type int (p. 1394)

Read Only No

RestartTime

The restart time for the coupling run.

Type Quantity (p. 1422)

Read Only No

StepSize

The step size for the coupling run.

Type Quantity (p. 1422)

Read Only No

DataTransfer

Entity to store a data transfer information.

Properties

DisplayText

The general property that defines the user-visible name of an entity. This property is defined for all data entities but is used only in those entities that present a label in the user interface.

Type string (p. 1438)

Read Only No

IsSuppressed

Suppression state of the entity.

Type bool (p. 1360)

Read Only No

SourceParticipant

Participant system providing the data.

Type DataReference (p. 1371)

Read Only No

SourceRegion

Participant region providing the data.

Type DataReference (p. 1371)

Read Only No

SourceVariable

Variable provided by the source participant.

Type DataReference (p. 1371)

Read Only No

TargetParticipant

Participant consuming the data.

Type DataReference (p. 1371)

Read Only No

TargetRegion

Participant region consuming the data.

Type DataReference (p. 1371)

Read Only No

TargetVariable

Variable consumed by the target participant.

Type DataReference (p. 1371)

Read Only No

TransferSettings

Settings to specify how the data transfers are executed.

Type DataReference (p. 1371)

Read Only No

Methods

Delete

Deletes a specified data transfer.

Duplicate

Duplicates the data transfer.

Return Data reference to the duplicate data transfer.

Type DataReference (p. 1371)

GetSettings

Returns the settings for a specified data transfer.

Return The data transfer settings.

Type DataReference (p. 1371)

SetSuppression

Suppresses or unsuppresses a data transfer

Required Arguments

Suppressed The boolean value to specify if the item should be suppressed or unsuppressed

Type bool (p. 1360)

DataTransferSettings

Entity to store the settings for the data transfer control.

Properties

ConvergenceTarget

The target value used when evaluating convergence of the data transfer within a coupling iteration.

Type double (p. 1378)

Read Only No

DisplayText

The general property that defines the user-visible name of an entity. This property is defined for all data entities but is used only in those entities that present a label in the user interface.

Type string (p. 1438)

Read Only No

Ramping

Setting that defines the ramping.

Type RampingType (p. 1422)

Read Only No

TransferAt

Setting that defines when the transfers should happen.

Type TransferAtType (p. 1443)

Read Only No

UnderRelaxationFactor

Convergence stability factor for highly non-linear couplings.

Type double (p. 1378)

Read Only No

ExecutionControlSettings

No details are provided for this entry.

Properties

DisplayText

The general property that defines the user-visible name of an entity. This property is defined for all data entities but is used only in those entities that present a label in the user interface.

Type string (p. 1438)

Read Only No

EngineVersion

No details are provided for this entry.

Type int (p. 1394)

Read Only No

ExpertSettings

The entity to store advanced options for data mapping.

Properties

DisplayText

The general property that defines the user-visible name of an entity. This property is defined for all data entities but is used only in those entities that present a label in the user interface.

Type string (p. 1438)

Read Only No

Settings

The expert setting parameters dictionary.

Type Dictionary (p. 1375)<string (p. 1438), string (p. 1438)>

Read Only No

Methods

AddProperty

Adds a property to expert settings.

Required Arguments

Property Property to be added.

Type string (p. 1438)

GetProperty

Returns the property value for a specified property.

Return The value of the property.

Type string (p. 1438)

Required Arguments

Property The name of the property.

Type string (p. 1438)

RemoveProperty

Removes an existing property from expert settings.

Required Arguments

Property Property to be removed.

Type string (p. 1438)

SetProperty

Sets the value of a property in expert settings.

Required Arguments

Property The name of the property.

Type string (p. 1438)

Value The value of the property.

Type string (p. 1438)

IntermediateResultFilesOutputControls

The entity to store the settings for result files creation during a coupling run.

Properties

DisplayText

The general property that defines the user-visible name of an entity. This property is defined for all data entities but is used only in those entities that present a label in the user interface.

Type string (p. 1438)

Read Only No

OutputFrequency

This property specifies the frequency at which the result files are generated.

Type OutputFrequencyType (p. 1412)

Read Only No

StepInterval

The step interval at which the result files are generated.

Type int (p. 1394)

Read Only No

SequenceControl

The entity to store solver sequence information for a coupling run.

Properties

DisplayText

The general property that defines the user-visible name of an entity. This property is defined for all data entities but is used only in those entities that present a label in the user interface.

Type string (p. 1438)

Read Only No

Sequence

Dictionary containing sequence values for each participant.

Type Dictionary (p. 1375)<DataReference (p. 1371), int (p. 1394)>

Read Only No

Methods

GetSequence

Returns the sequence value for a participant.

Return Sequence value for the given participant.

Type int (p. 1394)

Required Arguments

Participant The participant for which to get the sequence.

Type DataReference (p. 1371)

SetSequence

This command will set the sequence number for the specified participant.

Required Arguments

Participant The participant for which to set sequence value.

Type DataReference (p. 1371)

Value Sequence value for the given participant.

Type int (p. 1394)

SystemCouplingCoSimulationParticipant

This is a base class for the entity which represents System Coupling Co-simulation Participant information.

Properties

DisplayText

The general property that defines the user-visible name of an entity. This property is defined for all data entities but is used only in those entities that present a label in the user interface.

Type string (p. 1438)

Read Only No

InternalName

The internal name of the multiphysics participant (usually a solver) that is providing this data

Type string (p. 1438)

Read Only No

SystemCouplingParticipant

This is a base class for the entity which represents System Coupling Participant information. System Coupling Participant information comprises of System and solver-level information related to coupling. Note- Coupling participants are systems that will provide and/or consume data in a coupled analysis. Example systems in Workbench include: Analysis Systems – Steady-State Thermal, Transient Thermal, Static Structural, Transient Structural, Fluid Flow (Fluent) Component Systems – Fluent, External Data

Properties

DisplayText

The general property that defines the user-visible name of an entity. This property is defined for all data entities but is used only in those entities that present a label in the user interface.

Type string (p. 1438)

Read Only No

InternalName

The internal name of the multiphysics participant (usually a solver) that is providing this data

Type string (p. 1438)

Read Only No

SystemCouplingRegion

This entity represents a System Coupling Region. A region is most often a point, line, surface or volume that is part (or all) of the geometry or topology of a coupling participant.

Properties

DisplayText

The general property that defines the user-visible name of an entity. This property is defined for all data entities but is used only in those entities that present a label in the user interface.

Type string (p. 1438)

Read Only No

InternalName

The name of the region as understood by the solver

Type string (p. 1438)

Read Only No

Topology

The topology of the region

Type TopologyType (p. 1442)

Read Only No

SystemCouplingStaticDataParticipant

This is a base class/entity for the entity which represents System Coupling Static Data Participant information.

Properties

DisplayText

The general property that defines the user-visible name of an entity. This property is defined for all data entities but is used only in those entities that present a label in the user interface.

Type string (p. 1438)

Read Only No

InternalName

The internal name of the multiphysics participant (usually a solver) that is providing this data

Type string (p. 1438)

Read Only No

SystemCouplingSteadyCoSimulationParticipant

The entity represents System Coupling Steady Co-simulation Participant information e.g. information related to Static Structural Participant System for coupling purpose.

Properties

DisplayText

The general property that defines the user-visible name of an entity. This property is defined for all data entities but is used only in those entities that present a label in the user interface.

Type string (p. 1438)

Read Only No

InternalName

The internal name of the multiphysics participant (usually a solver) that is providing this data

Type string (p. 1438)

Read Only No

SystemCouplingSteadyStaticDataParticipant

The entity represents System Coupling Steady Static Data Participant information e.g. information related to External Data Participant System for coupling purposes.

Properties

DisplayText

The general property that defines the user-visible name of an entity. This property is defined for all data entities but is used only in those entities that present a label in the user interface.

Type string (p. 1438)

Read Only No

InternalName

The internal name of the multiphysics participant (usually a solver) that is providing this data

Type string (p. 1438)

Read Only No

SystemCouplingTransientCoSimulationParticipant

The entity represents System Coupling Transient Co-simulation Participant information e.g. information related to Transient Structural Participant System for coupling purposes.

Properties

DisplayText

The general property that defines the user-visible name of an entity. This property is defined for all data entities but is used only in those entities that present a label in the user interface.

Type string (p. 1438)

Read Only No

InternalName

The internal name of the multiphysics participant (usually a solver) that is providing this data

Type string (p. 1438)

Read Only No

SystemCouplingTransientStaticDataParticipant

The entity represents System Coupling Transient Static Data Participant information.

Properties

DisplayText

The general property that defines the user-visible name of an entity. This property is defined for all data entities but is used only in those entities that present a label in the user interface.

Type string (p. 1438)

Read Only No

InternalName

The internal name of the multiphysics participant (usually a solver) that is providing this data

Type string (p. 1438)

Read Only No

SystemCouplingVariable

This entity represents a System Coupling Variable. A variable is a physical quantity such as force, length, or temperature that can be transferred between regions of participant systems.

Properties

DataType

The tensor type of the variable

Type DataTypeEnumeration (p. 1371)

Read Only No

DisplayText

The general property that defines the user-visible name of an entity. This property is defined for all data entities but is used only in those entities that present a label in the user interface.

Type string (p. 1438)

Read Only No

Exposure

The type of transfer that will be done with this variable

Type VariableExposure (p. 1448)

Read Only No

InternalName

The name of the variable

Type string (p. 1438)

Read Only No

PhysicalType

The quantity type of the variable

Type string (p. 1438)

Read Only No

System Coupling Solution

This container holds Solution data for an instance of System Coupling.

Methods

CreateConvergenceChart

Creates a new convergence chart.

Return The data reference to the new convergence chart.

Type DataReference (p. 1371)

Required Arguments

Name The name for this Convergence Chart.

Type string (p. 1438)

CreateSolutionInformation

Creates a new solution information entity and adds it to the specified system coupling solution container.

Return The new solution information that was created.

Type DataReference (p. 1371)

Required Arguments

Name The name for this Solution Information.

Type string (p. 1438)

SolutionInformationFilePath The path of the file to read solution information.

Type string (p. 1438)

GetAllSolutionInformation

Returns the collection of solution information entities in a container. If no solution information entities are in the container, the collection is empty.

Return Collection of the solution informations in the container.

Type DataReferenceSet (p. 1371)

GetChartVariableNames

Returns a dictionary of fully qualified chart variable names and display names.

During System Coupling Solution cell Update or after Solution cell Update, these names can be used to create chart variable using "CreateVariable" data entity method of "ConvergenceChart".

Data transfer chart variable

Qualified Name Format - "Target Participant Internal Name":"Data Transfer Internal Name":"Variable Name": "Operator Name"
Qualfied Name Example- "Solution 1:Data Transfer 1:Change:Maximum"
Display Name Example- "Fluid Flow (FLUENT):Data Transfer 1:Change:Maximum"

Solver chart variable

Qualified Name Format - "Participant Internal Name":"Variable Name"
Qualfied Name Example- "Solution 1:Continuity Convergence"
Display Name Example- "Fluid Flow (FLUENT):Continuity Convergence"

Return A dictionary of fully qualified chart variable names and display names

Please see summary documentation of this (GetChartVariableNames) query on details of format and example of fully qualified name.

Type Dictionary (p. 1375)<string (p. 1438), string (p. 1438)>

GetConvergenceChart

Returns the convergence chart of a given name in a container.

Return The convergence chart that matches the specified name.

Type DataReference (p. 1371)

Required Arguments

Name The name or display name of the convergence chart.

Type string (p. 1438)

GetConvergenceCharts

Returns the collection of convergence charts in a container. If no convergence charts are in the container, the collection is empty.

Return Collection of the convergence charts in the container.

Type DataReferenceSet (p. 1371)

GetSolutionComponentProperties

Returns the solution component properties for a system coupling solution container.

Return Data reference to the properties objects.

Type DataReference (p. 1371)

GetSolutionInformation

Returns the solution information of a given name in a container.

Return The solution information that matches the specified name.

Type DataReference (p. 1371)

Required Arguments

Name The name or display name of the solution information.

Type string (p. 1438)

Data Entities

AxisContinuous

A chart axis that spans a set of continuous values. An example is an axis of an XY plot

Properties

AutomaticRange

The property to define whether or not automatic scaling should be applied to the axis, or whether the RangeMin and RangeMax should be used.

Type bool (p. 1360)

Read Only No

DisplayText

The general property that defines the user-visible name of an entity. This property is defined for all data entities but is used only in those entities that present a label in the user interface.

Type string (p. 1438)

Read Only No

QuantityName

The name of the quantity associated with axis data.

E.g. Coupling Iteration, Coupling Step, Coupling Time

Type string (p. 1438)

Read Only No

RangeMaximum

The maximum range of the values in this axis.

Type double (p. 1378)

Read Only No

RangeMinimum

The minimum range of the values in this axis.

Type double (p. 1378)

Read Only No

Scale

The scale of the axis. Scale can be defined as Linear/CommonLog (Log base 10)/Natural Log.

Type Scale (p. 1429)

Read Only No

Title

The title of the axis.

Type string (p. 1438)

Read Only No

ChartVariable

Entity representing a variable in Convergence Chart

Properties

Color

The line color of this chart variable in a plot.

This property is valid only for the chart displayed in Scene pane.

Type Color (p. 1365)

Read Only No

DisplayText

The general property that defines the user-visible name of an entity. This property is defined for all data entities but is used only in those entities that present a label in the user interface.

Type string (p. 1438)

Read Only No

LineWidth

The width of the line drawn for this chart variable in pixels.

This property is valid only for the chart displayed in Scene pane.

Type float (p. 1432)

Read Only No

QualifiedName

The variable quantity to display.

Type string (p. 1438)

Read Only No

RefinementLevel

Refinement level for the data to be plotted.

Type string (p. 1438)

Read Only No

SymbolSize

The size of a symbol in pixels when a symbol is drawn for this variable. The rendered symbol size may be slightly smaller or larger than expected if symbol does not correctly fit into the specified number of pixels.

This property is valid only for the chart displayed in Scene pane.

Type uint (p. 1446)

Read Only No

Methods

Delete

Deletes a specified chart variable.

ConvergenceChart

Entity to store a convergence chart information.

Properties

AxisX

Associated X Axis

Type DataReference (p. 1371)

Read Only No

AxisY

Associated Y Axis

Type DataReference (p. 1371)

Read Only No

DisplayText

The general property that defines the user-visible name of an entity. This property is defined for all data entities but is used only in those entities that present a label in the user interface.

Type string (p. 1438)

Read Only No

Variables

Collection of variables to be plotted.

Type DataReferenceSet (p. 1371)

Read Only No

XAxis

X Axis Quantity Name

Type string (p. 1438)

Read Only No

Methods

CreateVariable

Creates a chart variable based on specified quailified name and adds it to the specified convergence chart.

During System Coupling Solution cell Update or after Solution cell Update, user can create convergence chart and add chart variables. A chart variable is based on specified fully qualified name. GetChartVariableNames query returns a dictionary of fully qualified chart variable names and display names. This can be used to create charts.

Data transfer chart variable format

Qualified Name Format - "Target Participant Internal Name":"Data Transfer Internal Name":"Variable Name": "Operator Name"
Qualfied Name Example- "Solution 1:Data Transfer 1:Change:Maximum"
Display Name Example- "Fluid Flow (FLUENT):Data Transfer 1:Change:Maximum"

Solver chart variable format

Qualified Name Format - "Participant Internal Name":"Variable Name"
Qualfied Name Example- "Solution 1:Continuity Convergence"
Display Name Example- "Fluid Flow (FLUENT):Continuity Convergence"

Return The new created chart variable.

Type DataReference (p. 1371)

Required Arguments

QualifiedName The fully qualified name for this chart variable.

Please see summary documentation of this (CreateVariable) data entity method on details of format and example of fully qualified name.

Type string (p. 1438)

Optional Arguments

DisplayName The display name for this chart variable.

Type string (p. 1438)

Example

The following example demonstrates creation of chart variables. It is assumed that user has setup partcipants (e.g. Transient Structural and Fluid Flow(Fluent) and System Coupling system, and also solved coupled analysis.

```
# Get System Couplng Solution
system1 = GetSystem(Name="SC")
solution1 = system1.GetContainer(ComponentName="Solution")
# Create Convergence Chart
ConvergenceChart1 = solution1.CreateConvergenceChart(Name="Chart")
```

```
    # Create a Data Transfer chart variable
    ChartVariable1 = ConvergenceChart1.CreateVariable(QualifiedName="Solution 1:Data Transfer 1:Change:Maximum
    # Create a solver chart variable
    ChartVariable2 = ConvergenceChart1.CreateVariable(QualifiedName="Solution 1:Continuity Convergence",Displa
```

Delete

Delete's a specified convergence chart.

GetAxis

Returns the axis for a specified convergence chart

Return The axis

Type DataReference (p. 1371)

Required Arguments

Name Name of the Axis

Type string (p. 1438)

GetChartVariable

Returns the chart variable of a given name from a convergence chart

Return The chart variable that matches the specified name.

Type DataReference (p. 1371)

Required Arguments

QualifiedName The qualified name of the chart variable.

Type string (p. 1438)

GetChartVariables

Returns the collection of chart variables for a given convergence chart

Return A collection of the variables in the chart.

Type DataReferenceSet (p. 1371)

SolutionComponentProperties

The entity to store additional command line options which are passed to the coupling service on update.

Properties

CommandLineOptions

Additional command line options which are passed to the coupling service on update.

Type string (p. 1438)

Read Only No

DisplayText

The general property that defines the user-visible name of an entity. This property is defined for all data entities but is used only in those entities that present a label in the user interface.

Type string (p. 1438)

Read Only No

UpdateOption

Solution Process Update Option

Type SolutionUpdateOption (p. 1434)

Read Only No

SolutionInformation

Entity to store the solution information provided by the coupled participants.

Properties

DisplayText

The general property that defines the user-visible name of an entity. This property is defined for all data entities but is used only in those entities that present a label in the user interface.

Type string (p. 1438)

Read Only No

FilePath

The path of the solution information file.

Type string (p. 1438)

Read Only No

Methods

Delete

Deletes a specified solution information entity.

TurboSystems

Turbo Geometry

This container holds Geometry data for an instance of BladeGen.

Methods

CreateBladeMesh

Creates a new Mesh system and automatically meshes the fluid zone for the blade geometry from the Blade Design component.

CreateGeometry

This command class creates a new BladeEditor model. An up-to-date Geometry cell appears on the project schematic containing the new model.

DeleteBGDFiles

Deletes the BGD files in the Filenames list from the BladeGen component directory.

Required Arguments

Filenames Command argument containing the files to be deleted.

Type List (p. 1400)<string (p. 1438)>

Edit

Opens the BladeGen editor to allow modification of Blade Design data.

This command will open the editor only if one is not already open on this component. If this component's editor is already open, then it will be raised to the front.

Exit

Exits the BladeGen editor.

Any changes made in this editor will be retained on exit. These changes are made permanent by a Project Save, and will be discarded in the event of closing the project without saving. If no editor is open on the component in question, this command will have no effect.

Optional Arguments

ExitApp This parameter is deprecated and will be ignored.

Type bool (p. 1360)

Default Value True

GetTurboGeometryProperties

Returns the Data Entity which contains user settings and properties for the Blade Design container.

Return A reference to the requested data entity.

Type DataReference (p. 1371)

Import

Imports blade geometry data into the BladeGen editor from an existing BladeGen file.

Return Returned array of blade row names.

Type String[] (p. 1438)

Required Arguments

FilePath Command argument containing the file name to be opened.

Type string (p. 1438)

Example

```
template1 = GetTemplate(TemplateName="BladeGen")
system1 = template1.CreateSystem()
bladeDesign1 = system1.GetContainer(ComponentName="Blade Design")
bladeDesign1.Import(FilePath="myfilepath/pump.bgd")
```

ImportSelectedBladerow

Imports blade geometry data into the BladeGen editor from the specified bladerow

Required Arguments

Filename Command argument containing the file name to be opened.

Type string (p. 1438)

Data Entities

TurboGeometryProperties

This data entity provides access to the import properties that are used to determine how the blade geometry is handled when it is transferred to a downstream Geometry cell.

Properties

BladeExt

Import Option that specifies the blade surface extension length (as a percentage of the average hub to shroud distance) when the Blade Design data is transferred to a downstream Geometry. These surfaces are extended and then trimmed to the MasterProfile sketch to ensure that the blade solid correctly matches the hub and shroud contours.

Type double (p. 1378)

Read Only No

BladeLoftOption

Import Option that specifies how to loft the blade surfaces when the Blade Design data is transferred to a downstream Geometry.

Available options:

Streamwise	Loft the blade surfaces in the streamwise direction through curves that run from hub to shroud.
Spanwise	Loft the blade surfaces in the spanwise direction through the blade profile curves.

Type BladeLoftType (p. 1358)

Read Only No

CreateAllBlades

Import Option that specifies whether to create one or all blades when the Blade Design data is transferred to a downstream Geometry.

Available options:

True	BladeEditor will create all the blades using the number of blades specified in the BladeGen model.
False	Only one blade will be created.

Type bool (p. 1360)

Read Only No

CreateFluidZone

Import Option that specifies whether to create the fluid zone body when the Blade Design data is transferred to a downstream Geometry. The resulting Enclosure can be used for a CFD analysis of the blade passage.

Available options:

True	Create the fluid zone Enclosure.
False	Don't create the fluid zone Enclosure.

Type bool (p. 1360)

Read Only No

CreateHub

Import Option that specifies whether a hub body will be created when the Blade Design data is transferred to a downstream Geometry.

Available options:

True	BladeEditor will create a HubProfile sketch for the non-flow path hub geometry, and will create a revolved body feature called HubBody.
False	BladeEditor will not create the hub body.

Type bool (p. 1360)

Read Only No

CreateNamedSelections

Import Option that specifies whether to create the Named Selections for the fluid zone when the Blade Design data is transferred to a downstream Geometry. If this property is selected, then BladeEditor will create Named Selections (regions) for the typical faces of the blade passage, i.e., Blade, Hub, Shroud, Inflow, Outflow, PeriodicA and PeriodicB. These Named Selections can be used as selection groups in other ANSYS Workbench applications. Note that this property is available only if Create Fluid Zone is selected.

Available options:

True	Create the Named Selections.
False	Don't create the Named Selections.

Type bool (p. 1360)

Read Only No

DisplayText

The general property that defines the user-visible name of an entity. This property is defined for all data entities but is used only in those entities that present a label in the user interface.

Type string (p. 1438)

Read Only No

GeometryTransferOption

Option that specifies which method to use to transfer to a downstream BladeEditor geometry.

Available options:

Import BGD	Transfer the geometry to BladeEditor using the native BladeGen BGD format.
Load NDF	Transfer the geometry to BladeEditor using the Neutral Data Format, generating native BladeEditor features.

Type DataTransferType (p. 1371)

Read Only No

LayerNumber

Import Option that specifies the integer value of the Layer Number to use for the shroud clearance when the Blade Design data is transferred to a downstream Geometry. This property only applies when the Shroud Clearance property is set to Relative Layer or Absolute Layer.

Type int (p. 1394)

Read Only No

MachineType

Specification of the machine type, used by the downstream VistaTF setup

Available options:

Pump
AxialCompressor
CentrifugalCompressor
Fan
AxialTurbine
RadialTurbine
HydraulicTurbine
Other
Unknown

Type MachineType (p. 1401)

Read Only No

MergeBladeTopology

Import Option that specifies whether to merge the blade faces when the Blade Design data is transferred to a downstream Geometry. If not merged, there will be four faces corresponding to the leading edge,

trailing edge, pressure and suction surfaces of the blade. If merged, blade faces that are tangent to one another will be merged into a single face.

Available options:

True	Merge the blade faces where they are tangent to one another.
False	Don't merge the blade faces.

Type bool (p. 1360)

Read Only No

ModelUnits

THIS PROPERTY IS DEPRECATED IN WB16.0 AND IS RETAINED PURELY FOR BACKWARDS COMPATIBILITY.

The length scale units the BladeGen model was created in. Used when transfering the BladeGen model to VistaTF.

Available options:

m
cm
mm
inches
ft

Type BMunitsType (p. 1359)

Read Only No

PeriodicSurfExt

Import Option that defines the periodic surface extension length (as a percentage of the average hub to shroud distance) when the Blade Design data is transferred to a downstream Geometry.

Type double (p. 1378)

Read Only No

PeriodicSurfOption

Import Option that specifies the style of the periodic interface surfaces for the fluid zone when the Blade Design data is transferred to a downstream Geometry. Note that this property is available only if Create Fluid Zone is selected.

Available options:

OnePiece	The periodic surface is created as a single surface.
ThreePieces	The periodic surface is created in three connected pieces: one upstream of the blade, one within the passage, and one downstream of

	the blade. This style can better accommodate highly curved or twisted blades, and is similar to the ANSYS TurboGrid style of periodic surface.

Type PeriodicSurfType (p. 1415)

Read Only No

ShroudClearance

Import Option that specifies whether to include the shroud clearance when the Blade Design data is transferred to a downstream Geometry.

Available options:

None	No shroud clearance is created.
RelativeLayer	The selected Layer Number is relative to the shroud layer, e.g., 1 implies the first layer closest to the shroud layer, 2 implies the second closest layer to the shroud, etc.
AbsoluteLayer	The selected layer index counts up from the hub layer, which is zero.

Type ClearanceType (p. 1364)

Read Only No

SpanwiseCount

The number of spanwise gridlines used in the downstream VistaTF calculation. Default = 4

Type int (p. 1394)

Read Only No

StreamwiseCount

The number of streamwise gridlines used in the downstream VistaTF calculation. Default = 20

Type int (p. 1394)

Read Only No

Turbo Mesh

This container holds Mesh data for an instance of TurboGrid.

Methods

Edit

Opens the TurboGrid editor to allow modification of Turbo Mesh data.

Optional Arguments

Interactive Run the editor in interactive mode if True, or in no GUI mode if False.

If not specified, the editor runs in interactive mode.

Type bool (p. 1360)

Default Value True

TopologySuspended If True, open the editor with the topology in a suspended state. Otherwise, open the editor with the suspended state of topology the same as when the editor was last closed.

If not specified, it defaults to false.

Type bool (p. 1360)

Default Value False

Exit

Exits the editor.

Any changes made in this editor will be retained on exit. These changes are made permanent by a Project Save, and will be discarded in the event of closing the project without saving.

If no editor is open on the component in question, this command will have no effect.

GetTurboMeshProperties

Returns the Data Entity which contains user settings and properties for the Turbo Mesh container.

Return A reference to the requested data entity.

Type DataReference (p. 1371)

LoadBladeGen

This command imports the specified BladeGen INF file data into the Turbo Mesh cell. If an upstream geometry link to the Turbo Mesh cell exists an error message is generated.

```
template1 = GetTemplate(TemplateName="TurboGrid")
system1 = template1.CreateSystem()
turboMesh1 = system1.GetContainer(ComponentName="Turbo Mesh")
turboMesh1.LoadBladeGen(FileName="myfilepath/BladeGen.inf")
```

Required Arguments

FileName Name, and path, of the BladeGen INF file to be imported.

Type string (p. 1438)

ReinitializeContainer

This command reinitialises the selected container, resetting the local data and using the cached initialisation CCL commands

SendCommand

Sends commands to the editor for this component using CFX Command Language (CCL) syntax. If the editor for this component is not open, it will be launched before the commands are sent and subsequently closed. In this mode, component data is loaded and saved as if calling Edit(Interactive=False) and Exit around the SendCommand invocation.

The instructions must be CFX Command Language session commands that are valid for the editor in question.

Required Arguments

Command Valid CFX Command Language (CCL) commands

Type string (p. 1438)

SetInitCCL

Sets the initialisation commands for this component using CFX Command Language (CCL) syntax. On setting the commands they may be applied to the editor in question using ApplyInitCCLCommand.

The instructions must be CFX Command Language session commands that are valid for the editor in question.

Required Arguments

Command Valid CFX Command Language (CCL) commands

Type string (p. 1438)

Data Entities

TurboMeshProperties

This data entity provides access to the properties that are used to determine which blade geometry to mesh and how to handle the inlet and outlet positions.

Properties

AvailableFlowpaths

Displays a list of the available flowpaths and bladerows. Use this information as a guide when specifying the Flowpath and Bladerow properties (described below). Use the Refresh command in the context menu to update the list after linking.

Type string (p. 1438)

Read Only Yes

CADSelectedBladeRow

Specifies which Flowpath feature in BladeEditor contains the bladerow that is to be loaded in ANSYS TurboGrid. On initial refresh, it will default to the first flowpath available.

Type string (p. 1438)

Read Only No

DisplayText

The general property that defines the user-visible name of an entity. This property is defined for all data entities but is used only in those entities that present a label in the user interface.

Type string (p. 1438)

Read Only No

DownstreamBladerowNumber

Specifies the bladerow number for the bladerow that is immediately downstream of the current bladerow. This property is available only when Outlet Position Method is set to Adjacent Blade.

Type int (p. 1394)

Read Only No

IgnoreBladeNamesInINF

If this property is set to true than blade names in the upstream geometry are ignored for labelling regions in the Turbo mesh.

Type bool (p. 1360)

Read Only No

InletBlock

If this is checked then an inlet block will be generated, if possible, when the mesh is created.

Type bool (p. 1360)

Read Only No

InletPositionOption

Specifies how the inlet points are positioned in TurboGrid. This property is only available when multiple bladerows in the same flowpath have been exported from BladeEditor.

Available options:

Manual	The user will specify the inlet points in TurboGrid.
AdjacentBlade	TurboGrid will calculate the inlet points to be halfway between the selected upstream bladerow and the current bladerow.

Type OpeningPositionMethod (p. 1410)

Read Only No

MaximumFaceAngle

This is the maximum face angle in the mesh if the mesh has been generated.

Type Quantity (p. 1422)

Read Only No

MeshNamePrefix

If specified, this string will be prepended to all region names in the mesh when transferred to a CFX system. This option is only available when Beta features are enabled.

Type string (p. 1438)

Read Only No

MinimumFaceAngle

This is the minimum face angle in the mesh if the mesh has been generated.

Type Quantity (p. 1422)

Read Only No

OutletBlock

If this is checked then an outlet block will be generated, if possible, when the mesh is created.

Type bool (p. 1360)

Read Only No

OutletPositionOption

Specifies how the outlet points are positioned in TurboGrid. This property is only available when multiple bladerows in the same flowpath have been exported from BladeEditor.

Available options:

Manual	The user will specify the outlet points in TurboGrid.
AdjacentBlade	TurboGrid will calculate the outlet points to be halfway between the selected downstream bladerow and the current bladerow.

Type OpeningPositionMethod (p. 1410)

Read Only No

SelectedBladerowNum

Specifies which bladerow (within the specified Flowpath feature) is to be loaded in ANSYS TurboGrid. The bladerows are number sequentially, starting from 1 for native Blade features.

Type int (p. 1394)

Read Only No

SelectedFlowpathName

Specifies which Flowpath feature in BladeEditor contains the bladerow that is to be loaded in ANSYS TurboGrid. On initial refresh, it will default to the first flowpath available.

Type string (p. 1438)

Read Only No

UpstreamBladerowNumber

Specifies the bladerow number for the bladerow that is immediately upstream of the current bladerow. This property is available only when Inlet Position Method is set to Adjacent Blade.

Type int (p. 1394)

Read Only No

Turbo Performance Map

This container holds data for an instance of Turbo Performance Map.

Methods

Edit

This command class launches the Turbo Setup popup GUI.

GetPerformanceMapProperties

Returns the Data Entity which contains user settings and properties for the Performance Map container.

Return Reference to the requested Data Entity

Type DataReference (p. 1371)

Data Entities

PerformanceMapProperties

This data entity provides access to the properties which define the Performance Map component.

Properties

BaseDutyPoint

Reference to the base duty operating point

Type DataReference (p. 1371)

Read Only Yes

BaseDutySpeed

This property specifies the base duty rotational speed of the impeller.

Type Quantity (p. 1422)

Read Only No

DesignPoints

References to the design points in the parameter manager generated by the performance map

Type List (p. 1400)<DataReference (p. 1371)>

Read Only No

DisplayText

The general property that defines the user-visible name of an entity. This property is defined for all data entities but is used only in those entities that present a label in the user interface.

Type string (p. 1438)

Read Only No

ExitCorrected

Option to use exit corrrected mass flow outlet

Type bool (p. 1360)

Read Only No

FlowSysResults

persistent flow system list for the performance maps plot

Type List (p. 1400)<string (p. 1438)>

Read Only Yes

FlowSysSelected

persistent flow system selection on the Operating Conditions tab

Type string (p. 1438)

Read Only Yes

ImperialUnits

Imperial units option

Type bool (p. 1360)

Read Only No

IncludeOutOfDate

persistent flag to include out-of-date points on the performance map plot

Type bool (p. 1360)

Read Only Yes

InputParams

Dictionary with references to the connected flow systems and their input parameter references The key is the reference to the fluid flow system. The value is a list of references to the speed and mass flow parameters (in that order).

Type Dictionary (p. 1375)<DataReference (p. 1371), List (p. 1400)<DataReference (p. 1371)>>

Read Only No

MassParam

Reference to the mass flow rate parameter in the performance map

Type DataReference (p. 1371)

Read Only Yes

SIunits

SI units option

Type bool (p. 1360)

Read Only No

SpeedParam

Reference to the percentage speed parameter in the performance map

Type DataReference (p. 1371)

Read Only Yes

XField

persistent x axis field for the performance maps plot

Type string (p. 1438)

Read Only Yes

YField

persistent y axis field for the performance maps plot

Type string (p. 1438)

Read Only Yes

Turbo Setup

This container holds data for an instance of Turbo Setup.

Methods

CreatePerformanceMap

This command class creates a new Turbo Fluid Flow system on the project schematic. Appropriate input and output parameters are exposed from the system.

Return Data reference to the Turbo Fluid Flow system.

Type DataReference (p. 1371)

CreateThroughflow

This command class creates a new VistaTF system. The system, comprising Setup, Solution and Results cells, is created on the project schematic and the operating conditions set accordingly.

Return Container representing the VistaTF Setup cell.

Type DataContainerReference (p. 1371)

CreateTurboflow

This command class creates a new Turbomachinery Fluid Flow system. The new system appears on the project schematic containing Turbo Mesh, Setup, Solution and Results cells. The Turbo Mesh and Setup cells are initialised leaving the system ready to update.

Return Data reference to the Turbo Fluid Flow system.

Type DataReference (p. 1371)

DisplayTurboSetupResults

This command class display the performance map results for the Turbo Setup.

Required Arguments

TFFsystems List of Turbo Fluid Flow Systems

Type List (p. 1400)<DataReference (p. 1371)>

Edit

This command class launches the Turbo Setup popup GUI.

GetTurboFluidFlowSystems

Returns the Data References to the Turbo Fluid Flow systems used by the Turbo Setup for Performance Maps.

Return Data References to the associated Turbo Fluid Flow systems

Type List (p. 1400)<DataReference (p. 1371)>

GetTurboSetupProperties

Returns the Data Entity which contains user settings and properties for the Turbo Setup container.

Return Reference to the requested Data Entity

Type DataReference (p. 1371)

ReadOperatingPoints

This command class updates the Turbo Setup performance map with changes made to the operating points in the parameter manager.

ReadPerformanceMapResults

This command class reads the results parameters from the Turbo Fluid Flow systems referenced by the Turbo Setup and populates the corresponding Turbo Setup results properties.

Return List of Turbo Fluid Flow Systems

Type List (p. 1400)<DataReference (p. 1371)>

WriteOperatingPoints

This command class creates/updates the parameter manager with the design points defined by the Turbo Setup performance map.

Data Entities

TurboSetupProperties

This data entity provides access to the properties which define the Turbo Setup component.

Properties

Acentric

This property specifies the acentric factor for the working fluid. Note that this is only available when MaterialPropsOption is set to 'User' and the GasModelOption is set to 'Real'.

Type float (p. 1432)

Read Only No

Alpha3

This property specifies the flow angle at the inlet to the machine in the absolute reference frame.

Type Quantity (p. 1422)

Read Only No

BaseDutyPoint

Reference to the base duty operating point

Type DataReference (p. 1371)

Read Only Yes

BaseDutyResults

Dictionary of results for the base duty point. The dictionary Key is the Turbo Fluid Flow system. The dictionary Value is itself a dictionary holding the results parameter references and corresponding Quantities

Type Dictionary (p. 1375)<DataReference (p. 1371), Dictionary (p. 1375)<DataReference (p. 1371), Quantity (p. 1422)>>

Read Only Yes

BaseSpeedParam

Reference to the base duty speed parameter

Type DataReference (p. 1371)

Read Only Yes

ComponentConfig

This property specifies the configuration of the components which the machine is comprised of.

Available options:

Impeller and Vaneless Diffuser

Type ComponentConfigType (p. 1366)

Read Only No

Cp_A0

For a user-defined real gas, the specific heat capacity is specified as a polynomial function of temperature over two temperature ranges. This property specifies the constant component (coefficient of T^0) of the lower temperature range polynomial. Note that this is only available when MaterialPropsOption is set to 'User' and the GasModelOption is set to 'Real'.

Type double (p. 1378)

Read Only No

Cp_A1

For a user-defined real gas, the specific heat capacity is specified as a polynomial function of temperature over two temperature ranges. This property specifies the coefficient of T^1 of the lower temperature range polynomial. Note that this is only available when MaterialPropsOption is set to 'User' and the GasModelOption is set to 'Real'.

Type double (p. 1378)

Read Only No

Cp_A2

For a user-defined real gas, the specific heat capacity is specified as a polynomial function of temperature over two temperature ranges. This property specifies the coefficient of T^2 of the lower temperature range polynomial. Note that this is only available when MaterialPropsOption is set to 'User' and the GasModelOption is set to 'Real'.

Type double (p. 1378)

Read Only No

Cp_A3

For a user-defined real gas, the specific heat capacity is specified as a polynomial function of temperature over two temperature ranges. This property specifies the coefficient of T^3 of the lower temperature range polynomial. Note that this is only available when MaterialPropsOption is set to 'User' and the GasModelOption is set to 'Real'.

Type double (p. 1378)

Read Only No

Cp_A4

For a user-defined real gas, the specific heat capacity is specified as a polynomial function of temperature over two temperature ranges. This property specifies the coefficient of T^4 of the lower temperature range polynomial. Note that this is only available when MaterialPropsOption is set to 'User' and the GasModelOption is set to 'Real'.

Type double (p. 1378)

Read Only No

Cp_Amax

For a user-defined real gas, the specific heat capacity is specified as a polynomial function of temperature over two temperature ranges. This property specifies the maximum temperature limit for which the lower temperature range polynomial is applicable. Note that this is only available when MaterialPropsOption is set to 'User' and the GasModelOption is set to 'Real'.

Type float (p. 1432)

Read Only No

Cp_Amin

For a user-defined real gas, the specific heat capacity is specified as a polynomial function of temperature over two temperature ranges. This property specifies the minimum temperature limit for which the lower temperature range polynomial is applicable. Note that this is only available when MaterialPropsOption is set to 'User' and the GasModelOption is set to 'Real'.

Type float (p. 1432)

Read Only No

CriticalPressure

This property specifies the critical pressure for the working fluid. Note that this is only available when MaterialPropsOption is set to 'User' and the GasModelOption is set to 'Real'.

Type Quantity (p. 1422)

Read Only No

CriticalTemp

This property specifies the critical temperature for the working fluid. Note that this is only available when MaterialPropsOption is set to 'User' and the GasModelOption is set to 'Real'.

Type Quantity (p. 1422)

Read Only No

CriticalVol

This property specifies the critical volume for the working fluid. Note that this is only available when MaterialPropsOption is set to 'User' and the GasModelOption is set to 'Real'.

Type Quantity (p. 1422)

Read Only No

DisplayText

The general property that defines the user-visible name of an entity. This property is defined for all data entities but is used only in those entities that present a label in the user interface.

Type string (p. 1438)

Read Only No

GammaUser

This property specifies the ratio of specific heats for the working fluid. Note that this is only available when MaterialPropsOption is set to 'User' and the GasModelOption is set to 'Ideal'.

Type float (p. 1432)

Read Only No

GasModel

This property specifies whether to treat the working fluid as an Ideal or a Real gas.

Available options:

Ideal
Real

Type GasModelType (p. 1388)

Read Only No

ImperialUnits

Imperial units option

Type bool (p. 1360)

Read Only No

IncludeOutOfDate

persistent flag to include out-of-date points on the performance map plot

Type bool (p. 1360)

Read Only Yes

MassFlow

This property specifies the design point mass flow rate passing through the machine.

Type Quantity (p. 1422)

Read Only No

MassFlowRates

Mass flow rates used to define the performance map

Type List (p. 1400)<List (p. 1400)<Quantity (p. 1422)>>

Read Only Yes

MassParam

Reference to the mass flow rate parameter in the performance map

Type DataReference (p. 1371)

Read Only Yes

MaterialNameSelection

This property specifies the name of the working fluid, as selected from the database. Note that this is only available when the MaterialPropsOption is set to 'Database'.

Available options:

air
carbon_dioxide
hydrogen
methane
nitrogen
oxygen
parahydrogen
propylene
R123
R125
R134a
R141b
R142b
R245fa
water

Type MaterialNamesList (p. 1402)

Read Only No

MaterialPropsOption

This property specifies whether the working fluid properties are chosen from the materials database or specified directly by the user.

Available options:

Database
User

Type MaterialPropsType (p. 1402)

Read Only No

MaxFlow1

Maximum flow rate on speedline 1.

Type Quantity (p. 1422)

Read Only No

MaxFlow10

Maximum flow rate on speedline 10.

Type Quantity (p. 1422)

Read Only No

MaxFlow11

Maximum flow rate on speedline 11.

Type Quantity (p. 1422)

Read Only No

MaxFlow12

Maximum flow rate on speedline 12.

Type Quantity (p. 1422)

Read Only No

MaxFlow13

Maximum flow rate on speedline 13.

Type Quantity (p. 1422)

Read Only No

MaxFlow14

Maximum flow rate on speedline 14.

Type Quantity (p. 1422)

Read Only No

MaxFlow15

Maximum flow rate on speedline 15.

Type Quantity (p. 1422)

Read Only No

MaxFlow2

Maximum flow rate on speedline 2.

Type Quantity (p. 1422)

Read Only No

MaxFlow3

Maximum flow rate on speedline 3.

Type Quantity (p. 1422)

Read Only No

MaxFlow4

Maximum flow rate on speedline 4.

Type Quantity (p. 1422)

Read Only No

MaxFlow5

Maximum flow rate on speedline 5.

Type Quantity (p. 1422)

Read Only No

MaxFlow6

Maximum flow rate on speedline 6.

Type Quantity (p. 1422)

Read Only No

MaxFlow7

Maximum flow rate on speedline 7.

Type Quantity (p. 1422)

Read Only No

MaxFlow8

Maximum flow rate on speedline 8.

Type Quantity (p. 1422)

Read Only No

MaxFlow9

Maximum flow rate on speedline 9.

Type Quantity (p. 1422)

Read Only No

MaxFlowrates

Maximum number of flow rates per speedline permitted

Type int (p. 1394)

Read Only Yes

MaxIterParam

Reference to the maximum iteration parameter in the performance map

Type DataReference (p. 1371)

Read Only Yes

MaxSpeedlines

Maximum number of speedlines permitted

Type int (p. 1394)

Read Only Yes

MeshNodeCount

This property specifies the target passage node count used with the ATM meshing scheme in TurboGrid The minimum value for this property is 50,000 nodes

Type int (p. 1394)

Read Only No

MinFlow1

Minimum flow rate on speedline 1.

Type Quantity (p. 1422)

Read Only No

MinFlow10

Minimum flow rate on speedline 10.

Type Quantity (p. 1422)

Read Only No

MinFlow11

Minimum flow rate on speedline 11.

Type Quantity (p. 1422)

Read Only No

MinFlow12

Minimum flow rate on speedline 12.

Type Quantity (p. 1422)

Read Only No

MinFlow13

Minimum flow rate on speedline 13.

Type Quantity (p. 1422)

Read Only No

MinFlow14

Minimum flow rate on speedline 14.

Type Quantity (p. 1422)

Read Only No

MinFlow15

Minimum flow rate on speedline 15.

Type Quantity (p. 1422)

Read Only No

MinFlow2

Minimum flow rate on speedline 2.

Type Quantity (p. 1422)

Read Only No

MinFlow3

Minimum flow rate on speedline 3.

Type Quantity (p. 1422)

Read Only No

MinFlow4

Minimum flow rate on speedline 4.

Type Quantity (p. 1422)

Read Only No

MinFlow5

Minimum flow rate on speedline 5.

Type Quantity (p. 1422)

Read Only No

MinFlow6

Minimum flow rate on speedline 6.

Type Quantity (p. 1422)

Read Only No

MinFlow7

Minimum flow rate on speedline 7.

Type Quantity (p. 1422)

Read Only No

MinFlow8

Minimum flow rate on speedline 8.

Type Quantity (p. 1422)

Read Only No

MinFlow9

Minimum flow rate on speedline 9.

Type Quantity (p. 1422)

Read Only No

MuUser

This property specifies the dynamic viscosity of the working fluid.

Type Quantity (p. 1422)

Read Only No

NFlow1

The number of mass flow rates specified on speedline 1.

Type int (p. 1394)

Read Only No

NFlow10

The number of mass flow rates specified on speedline 10.

Type int (p. 1394)

Read Only No

NFlow11

The number of mass flow rates specified on speedline 11.

Type int (p. 1394)

Read Only No

NFlow12

The number of mass flow rates specified on speedline 12.

Type int (p. 1394)

Read Only No

NFlow13

The number of mass flow rates specified on speedline 13.

Type int (p. 1394)

Read Only No

NFlow14

The number of mass flow rates specified on speedline 14.

Type int (p. 1394)

Read Only No

NFlow15

The number of mass flow rates specified on speedline 15.

Type int (p. 1394)

Read Only No

NFlow2

The number of mass flow rates specified on speedline 2.

Type int (p. 1394)

Read Only No

NFlow3

The number of mass flow rates specified on speedline 3.

Type int (p. 1394)

Read Only No

NFlow4

The number of mass flow rates specified on speedline 4.

Type int (p. 1394)

Read Only No

NFlow5

The number of mass flow rates specified on speedline 5.

Type int (p. 1394)

Read Only No

NFlow6

The number of mass flow rates specified on speedline 6.

Type int (p. 1394)

Read Only No

NFlow7

The number of mass flow rates specified on speedline 7.

Type int (p. 1394)

Read Only No

NFlow8

The number of mass flow rates specified on speedline 8.

Type int (p. 1394)

Read Only No

NFlow9

The number of mass flow rates specified on speedline 9.

Type int (p. 1394)

Read Only No

NFlows

Number of flow rates per speedline in the performance map

Type List (p. 1400)<int (p. 1394)>

Read Only Yes

NSpeeds

The number of speedlines to be calculated by the Turbo Fluid Flow system.

Type int (p. 1394)

Read Only No

Omega

This property specifies the design point rotational speed of the impeller.

Type Quantity (p. 1422)

Read Only No

Omega1

The percentage rotational speed of the impeller for speedline 1.

Type int (p. 1394)

Read Only No

Omega10

The percentage rotational speed of the impeller for speedline 10.

Type int (p. 1394)

Read Only No

Omega11

The percentage rotational speed of the impeller for speedline 11.

Type int (p. 1394)

Read Only No

Omega12

The percentage rotational speed of the impeller for speedline 12.

Type int (p. 1394)

Read Only No

Omega13

The percentage rotational speed of the impeller for speedline 13.

Type int (p. 1394)

Read Only No

Omega14

The percentage rotational speed of the impeller for speedline 14.

Type int (p. 1394)

Read Only No

Omega15

The percentage rotational speed of the impeller for speedline 15.

Type int (p. 1394)

Read Only No

Omega2

The percentage rotational speed of the impeller for speedline 2.

Type int (p. 1394)

Read Only No

Omega3

The percentage rotational speed of the impeller for speedline 3.

Type int (p. 1394)

Read Only No

Omega4

The percentage rotational speed of the impeller for speedline 4.

Type int (p. 1394)

Read Only No

Omega5

The percentage rotational speed of the impeller for speedline 5.

Type int (p. 1394)

Read Only No

Omega6

The percentage rotational speed of the impeller for speedline 6.

Type int (p. 1394)

Read Only No

Omega7

The percentage rotational speed of the impeller for speedline 7.

Type int (p. 1394)

Read Only No

Omega8

The percentage rotational speed of the impeller for speedline 8.

Type int (p. 1394)

Read Only No

Omega9

The percentage rotational speed of the impeller for speedline 9.

Type int (p. 1394)

Read Only No

OperatingPts

References to the operating points in the performance map

Type List (p. 1400)<List (p. 1400)<DataReference (p. 1371)>>

Read Only Yes

Preswirl

This property specifies how the radial distribution of the preswirl angle is calculated.

Available options:

constant
free
forced
linear

Type PreswirlType (p. 1419)

Read Only No

R160_NFlow1

Obsolete property. Maintained for backwards compatibility only

Type int (p. 1394)

Read Only Yes

R160_NFlow10

Obsolete property. Maintained for backwards compatibility only

Type int (p. 1394)

Read Only Yes

R160_NFlow2

Obsolete property. Maintained for backwards compatibility only

Type int (p. 1394)

Read Only Yes

R160_NFlow3

Obsolete property. Maintained for backwards compatibility only

Type int (p. 1394)

Read Only Yes

R160_NFlow4

Obsolete property. Maintained for backwards compatibility only

Type int (p. 1394)

Read Only Yes

R160_NFlow5

Obsolete property. Maintained for backwards compatibility only

Type int (p. 1394)

Read Only Yes

R160_NFlow6

Obsolete property. Maintained for backwards compatibility only

Type int (p. 1394)

Read Only Yes

R160_NFlow7

Obsolete property. Maintained for backwards compatibility only

Type int (p. 1394)

Read Only Yes

R160_NFlow8

Obsolete property. Maintained for backwards compatibility only

Type int (p. 1394)

Read Only Yes

R160_NFlow9

Obsolete property. Maintained for backwards compatibility only

Type int (p. 1394)

Read Only Yes

R160_NSpeeds

Obsolete property. Maintained for backwards compatibility only

Type int (p. 1394)

Read Only Yes

Results

Dictionary of results for the performance map. The dictionary Key is the Turbo Fluid Flow system. The dictionary Value comprises a nested List of dictionaries as follows: The dictionary holds the results parameter references and corresponding Quantities for a flow rate The inner List of dictionaries contains the results for all the flow rates on a speedline The outer List contains the results for all the speedlines in the map

Type Dictionary (p. 1375)<DataReference (p. 1371), List (p. 1400)<List (p. 1400)<Dictionary (p. 1375)<DataReference (p. 1371), Quantity (p. 1422)>>>>

Read Only Yes

RotationalDirection

Specifies the positive direction of machine rotation about the Z-axis.

Available options:

RightHanded	Positive rotation is clockwise about the Z-axis.
LeftHanded	Positive rotation is counterclockwise about the Z-axis.

Type RotationType (p. 1427)

Read Only No

RUser

This property specifies the specific gas constant for the working fluid. Note that this is only available when MaterialPropsOption is set to 'User'.

Type Quantity (p. 1422)

Read Only No

SIunits

SI units option

Type bool (p. 1360)

Read Only No

Speed10MassFlow1

Mass flow rate 1 on speedline 10.

Type Quantity (p. 1422)

Read Only No

Speed10MassFlow10

Mass flow rate 10 on speedline 10.

Type Quantity (p. 1422)

Read Only No

Speed10MassFlow11

Mass flow rate 11 on speedline 10.

Type Quantity (p. 1422)

Read Only No

Speed10MassFlow12

Mass flow rate 12 on speedline 10.

Type Quantity (p. 1422)

Read Only No

Speed10MassFlow13

Mass flow rate 13 on speedline 10.

Type Quantity (p. 1422)

Read Only No

Speed10MassFlow14

Mass flow rate 14 on speedline 10.

Type Quantity (p. 1422)

Read Only No

Speed10MassFlow15

Mass flow rate 15 on speedline 10.

Type Quantity (p. 1422)

Read Only No

Speed10MassFlow2

Mass flow rate 2 on speedline 10.

Type Quantity (p. 1422)

Read Only No

Speed10MassFlow3

Mass flow rate 3 on speedline 10.

Type Quantity (p. 1422)

Read Only No

Speed10MassFlow4

Mass flow rate 4 on speedline 10.

Type Quantity (p. 1422)

Read Only No

Speed10MassFlow5

Mass flow rate 5 on speedline 10.

Type Quantity (p. 1422)

Read Only No

Speed10MassFlow6

Mass flow rate 6 on speedline 10.

Type Quantity (p. 1422)

Read Only No

Speed10MassFlow7

Mass flow rate 7 on speedline 10.

Type Quantity (p. 1422)

Read Only No

Speed10MassFlow8

Mass flow rate 8 on speedline 10.

Type Quantity (p. 1422)

Read Only No

Speed10MassFlow9

Mass flow rate 9 on speedline 10.

Type Quantity (p. 1422)

Read Only No

Speed11MassFlow1

Mass flow rate 1 on speedline 11.

Type Quantity (p. 1422)

Read Only No

Speed11MassFlow10

Mass flow rate 10 on speedline 11.

Type Quantity (p. 1422)

Read Only No

Speed11MassFlow11

Mass flow rate 11 on speedline 11.

Type Quantity (p. 1422)

Read Only No

Speed11MassFlow12

Mass flow rate 12 on speedline 11.

Type Quantity (p. 1422)

Read Only No

Speed11MassFlow13

Mass flow rate 13 on speedline 11.

Type Quantity (p. 1422)

Read Only No

Speed11MassFlow14

Mass flow rate 14 on speedline 11.

Type Quantity (p. 1422)

Read Only No

Speed11MassFlow15

Mass flow rate 15 on speedline 11.

Type Quantity (p. 1422)

Read Only No

Speed11MassFlow2

Mass flow rate 2 on speedline 11.

Type Quantity (p. 1422)

Read Only No

Speed11MassFlow3

Mass flow rate 3 on speedline 11.

Type Quantity (p. 1422)

Read Only No

Speed11MassFlow4

Mass flow rate 4 on speedline 11.

Type Quantity (p. 1422)

Read Only No

Speed11MassFlow5

Mass flow rate 5 on speedline 11.

Type Quantity (p. 1422)

Read Only No

Speed11MassFlow6

Mass flow rate 6 on speedline 11.

Type Quantity (p. 1422)

Read Only No

Speed11MassFlow7

Mass flow rate 7 on speedline 11.

Type Quantity (p. 1422)

Read Only No

Speed11MassFlow8

Mass flow rate 8 on speedline 11.

Type Quantity (p. 1422)

Read Only No

Speed11MassFlow9

Mass flow rate 9 on speedline 11.

Type Quantity (p. 1422)

Read Only No

Speed12MassFlow1

Mass flow rate 1 on speedline 12.

Type Quantity (p. 1422)

Read Only No

Speed12MassFlow10

Mass flow rate 10 on speedline 12.

Type Quantity (p. 1422)

Read Only No

Speed12MassFlow11

Mass flow rate 11 on speedline 12.

Type Quantity (p. 1422)

Read Only No

Speed12MassFlow12

Mass flow rate 12 on speedline 12.

Type Quantity (p. 1422)

Read Only No

Speed12MassFlow13

Mass flow rate 13 on speedline 12.

Type Quantity (p. 1422)

Read Only No

Speed12MassFlow14

Mass flow rate 14 on speedline 12.

Type Quantity (p. 1422)

Read Only No

Speed12MassFlow15

Mass flow rate 15 on speedline 12.

Type Quantity (p. 1422)

Read Only No

Speed12MassFlow2

Mass flow rate 2 on speedline 12.

Type Quantity (p. 1422)

Read Only No

Speed12MassFlow3

Mass flow rate 3 on speedline 12.

Type Quantity (p. 1422)

Read Only No

Speed12MassFlow4

Mass flow rate 4 on speedline 12.

Type Quantity (p. 1422)

Read Only No

Speed12MassFlow5

Mass flow rate 5 on speedline 12.

Type Quantity (p. 1422)

Read Only No

Speed12MassFlow6

Mass flow rate 6 on speedline 12.

Type Quantity (p. 1422)

Read Only No

Speed12MassFlow7

Mass flow rate 7 on speedline 12.

Type Quantity (p. 1422)

Read Only No

Speed12MassFlow8

Mass flow rate 8 on speedline 12.

Type Quantity (p. 1422)

Read Only No

Speed12MassFlow9

Mass flow rate 9 on speedline 12.

Type Quantity (p. 1422)

Read Only No

Speed13MassFlow1

Mass flow rate 1 on speedline 13.

Type Quantity (p. 1422)

Read Only No

Speed13MassFlow10

Mass flow rate 10 on speedline 13.

Type Quantity (p. 1422)

Read Only No

Speed13MassFlow11

Mass flow rate 11 on speedline 13.

Type Quantity (p. 1422)

Read Only No

Speed13MassFlow12

Mass flow rate 12 on speedline 13.

Type Quantity (p. 1422)

Read Only No

Speed13MassFlow13

Mass flow rate 13 on speedline 13.

Type Quantity (p. 1422)

Read Only No

Speed13MassFlow14

Mass flow rate 14 on speedline 13.

Type Quantity (p. 1422)

Read Only No

Speed13MassFlow15

Mass flow rate 15 on speedline 13.

Type Quantity (p. 1422)

Read Only No

Speed13MassFlow2

Mass flow rate 2 on speedline 13.

Type Quantity (p. 1422)

Read Only No

Speed13MassFlow3

Mass flow rate 3 on speedline 13.

Type Quantity (p. 1422)

Read Only No

Speed13MassFlow4

Mass flow rate 4 on speedline 13.

Type Quantity (p. 1422)

Read Only No

Speed13MassFlow5

Mass flow rate 5 on speedline 13.

Type Quantity (p. 1422)

Read Only No

Speed13MassFlow6

Mass flow rate 6 on speedline 13.

Type Quantity (p. 1422)

Read Only No

Speed13MassFlow7

Mass flow rate 7 on speedline 13.

Type Quantity (p. 1422)

Read Only No

Speed13MassFlow8

Mass flow rate 8 on speedline 13.

Type Quantity (p. 1422)

Read Only No

Speed13MassFlow9

Mass flow rate 9 on speedline 13.

Type Quantity (p. 1422)

Read Only No

Speed14MassFlow1

Mass flow rate 1 on speedline 14.

Type Quantity (p. 1422)

Read Only No

Speed14MassFlow10

Mass flow rate 10 on speedline 14.

Type Quantity (p. 1422)

Read Only No

Speed14MassFlow11

Mass flow rate 11 on speedline 14.

Type Quantity (p. 1422)

Read Only No

Speed14MassFlow12

Mass flow rate 12 on speedline 14.

Type Quantity (p. 1422)

Read Only No

Speed14MassFlow13

Mass flow rate 13 on speedline 14.

Type Quantity (p. 1422)

Read Only No

Speed14MassFlow14

Mass flow rate 14 on speedline 14.

Type Quantity (p. 1422)

Read Only No

Speed14MassFlow15

Mass flow rate 15 on speedline 14.

Type Quantity (p. 1422)

Read Only No

Speed14MassFlow2

Mass flow rate 2 on speedline 14.

Type Quantity (p. 1422)

Read Only No

Speed14MassFlow3

Mass flow rate 3 on speedline 14.

Type Quantity (p. 1422)

Read Only No

Speed14MassFlow4

Mass flow rate 4 on speedline 14.

Type Quantity (p. 1422)

Read Only No

Speed14MassFlow5

Mass flow rate 5 on speedline 14.

Type Quantity (p. 1422)

Read Only No

Speed14MassFlow6

Mass flow rate 6 on speedline 14.

Type Quantity (p. 1422)

Read Only No

Speed14MassFlow7

Mass flow rate 7 on speedline 14.

Type Quantity (p. 1422)

Read Only No

Speed14MassFlow8

Mass flow rate 8 on speedline 14.

Type Quantity (p. 1422)

Read Only No

Speed14MassFlow9

Mass flow rate 9 on speedline 14.

Type Quantity (p. 1422)

Read Only No

Speed15MassFlow1

Mass flow rate 1 on speedline 15.

Type Quantity (p. 1422)

Read Only No

Speed15MassFlow10

Mass flow rate 10 on speedline 15.

Type Quantity (p. 1422)

Read Only No

Speed15MassFlow11

Mass flow rate 11 on speedline 15.

Type Quantity (p. 1422)

Read Only No

Speed15MassFlow12

Mass flow rate 12 on speedline 15.

Type Quantity (p. 1422)

Read Only No

Speed15MassFlow13

Mass flow rate 13 on speedline 15.

Type Quantity (p. 1422)

Read Only No

Speed15MassFlow14

Mass flow rate 14 on speedline 15.

Type Quantity (p. 1422)

Read Only No

Speed15MassFlow15

Mass flow rate 15 on speedline 15.

Type Quantity (p. 1422)

Read Only No

Speed15MassFlow2

Mass flow rate 2 on speedline 15.

Type Quantity (p. 1422)

Read Only No

Speed15MassFlow3

Mass flow rate 3 on speedline 15.

Type Quantity (p. 1422)

Read Only No

Speed15MassFlow4

Mass flow rate 4 on speedline 15.

Type Quantity (p. 1422)

Read Only No

Speed15MassFlow5

Mass flow rate 5 on speedline 15.

Type Quantity (p. 1422)

Read Only No

Speed15MassFlow6

Mass flow rate 6 on speedline 15.

Type Quantity (p. 1422)

Read Only No

Speed15MassFlow7

Mass flow rate 7 on speedline 15.

Type Quantity (p. 1422)

Read Only No

Speed15MassFlow8

Mass flow rate 8 on speedline 15.

Type Quantity (p. 1422)

Read Only No

Speed15MassFlow9

Mass flow rate 9 on speedline 15.

Type Quantity (p. 1422)

Read Only No

Speed1MassFlow1

Mass flow rate 1 on speedline 1.

Type Quantity (p. 1422)

Read Only No

Speed1MassFlow10

Mass flow rate 10 on speedline 1.

Type Quantity (p. 1422)

Read Only No

Speed1MassFlow11

Mass flow rate 11 on speedline 1.

Type Quantity (p. 1422)

Read Only No

Speed1MassFlow12

Mass flow rate 12 on speedline 1.

Type Quantity (p. 1422)

Read Only No

Speed1MassFlow13

Mass flow rate 13 on speedline 1.

Type Quantity (p. 1422)

Read Only No

Speed1MassFlow14

Mass flow rate 14 on speedline 1.

Type Quantity (p. 1422)

Read Only No

Speed1MassFlow15

Mass flow rate 15 on speedline 1.

Type Quantity (p. 1422)

Read Only No

Speed1MassFlow2

Mass flow rate 2 on speedline 1.

Type Quantity (p. 1422)

Read Only No

Speed1MassFlow3

Mass flow rate 3 on speedline 1.

Type Quantity (p. 1422)

Read Only No

Speed1MassFlow4

Mass flow rate 4 on speedline 1.

Type Quantity (p. 1422)

Read Only No

Speed1MassFlow5

Mass flow rate 5 on speedline 1.

Type Quantity (p. 1422)

Read Only No

Speed1MassFlow6

Mass flow rate 6 on speedline 1.

Type Quantity (p. 1422)

Read Only No

Speed1MassFlow7

Mass flow rate 7 on speedline 1.

Type Quantity (p. 1422)

Read Only No

Speed1MassFlow8

Mass flow rate 8 on speedline 1.

Type Quantity (p. 1422)

Read Only No

Speed1MassFlow9

Mass flow rate 9 on speedline 1.

Type Quantity (p. 1422)

Read Only No

Speed2MassFlow1

Mass flow rate 1 on speedline 2.

Type Quantity (p. 1422)

Read Only No

Speed2MassFlow10

Mass flow rate 10 on speedline 2.

Type Quantity (p. 1422)

Read Only No

Speed2MassFlow11

Mass flow rate 11 on speedline 2.

Type Quantity (p. 1422)

Read Only No

Speed2MassFlow12

Mass flow rate 12 on speedline 2.

Type Quantity (p. 1422)

Read Only No

Speed2MassFlow13

Mass flow rate 13 on speedline 2.

Type Quantity (p. 1422)

Read Only No

Speed2MassFlow14

Mass flow rate 14 on speedline 2.

Type Quantity (p. 1422)

Read Only No

Speed2MassFlow15

Mass flow rate 15 on speedline 2.

Type Quantity (p. 1422)

Read Only No

Speed2MassFlow2

Mass flow rate 2 on speedline 2.

Type Quantity (p. 1422)

Read Only No

Speed2MassFlow3

Mass flow rate 3 on speedline 2.

Type Quantity (p. 1422)

Read Only No

Speed2MassFlow4

Mass flow rate 4 on speedline 2.

Type Quantity (p. 1422)

Read Only No

Speed2MassFlow5

Mass flow rate 5 on speedline 2.

Type Quantity (p. 1422)

Read Only No

Speed2MassFlow6

Mass flow rate 6 on speedline 2.

Type Quantity (p. 1422)

Read Only No

Speed2MassFlow7

Mass flow rate 7 on speedline 2.

Type Quantity (p. 1422)

Read Only No

Speed2MassFlow8

Mass flow rate 8 on speedline 2.

Type Quantity (p. 1422)

Read Only No

Speed2MassFlow9

Mass flow rate 9 on speedline 2.

Type Quantity (p. 1422)

Read Only No

Speed3MassFlow1

Mass flow rate 1 on speedline 3.

Type Quantity (p. 1422)

Read Only No

Speed3MassFlow10

Mass flow rate 10 on speedline 3.

Type Quantity (p. 1422)

Read Only No

Speed3MassFlow11

Mass flow rate 11 on speedline 3.

Type Quantity (p. 1422)

Read Only No

Speed3MassFlow12

Mass flow rate 12 on speedline 3.

Type Quantity (p. 1422)

Read Only No

Speed3MassFlow13

Mass flow rate 13 on speedline 3.

Type Quantity (p. 1422)

Read Only No

Speed3MassFlow14

Mass flow rate 14 on speedline 3.

Type Quantity (p. 1422)

Read Only No

Speed3MassFlow15

Mass flow rate 15 on speedline 3.

Type Quantity (p. 1422)

Read Only No

Speed3MassFlow2

Mass flow rate 2 on speedline 3.

Type Quantity (p. 1422)

Read Only No

Speed3MassFlow3

Mass flow rate 3 on speedline 3.

Type Quantity (p. 1422)

Read Only No

Speed3MassFlow4

Mass flow rate 4 on speedline 3.

Type Quantity (p. 1422)

Read Only No

Speed3MassFlow5

Mass flow rate 5 on speedline 3.

Type Quantity (p. 1422)

Read Only No

Speed3MassFlow6

Mass flow rate 6 on speedline 3.

Type Quantity (p. 1422)

Read Only No

Speed3MassFlow7

Mass flow rate 7 on speedline 3.

Type Quantity (p. 1422)

Read Only No

Speed3MassFlow8

Mass flow rate 8 on speedline 3.

Type Quantity (p. 1422)

Read Only No

Speed3MassFlow9

Mass flow rate 9 on speedline 3.

Type Quantity (p. 1422)

Read Only No

Speed4MassFlow1

Mass flow rate 1 on speedline 4.

Type Quantity (p. 1422)

Read Only No

Speed4MassFlow10

Mass flow rate 10 on speedline 4.

Type Quantity (p. 1422)

Read Only No

Speed4MassFlow11

Mass flow rate 11 on speedline 4.

Type Quantity (p. 1422)

Read Only No

Speed4MassFlow12

Mass flow rate 12 on speedline 4.

Type Quantity (p. 1422)

Read Only No

Speed4MassFlow13

Mass flow rate 13 on speedline 4.

Type Quantity (p. 1422)

Read Only No

Speed4MassFlow14

Mass flow rate 14 on speedline 4.

Type Quantity (p. 1422)

Read Only No

Speed4MassFlow15

Mass flow rate 15 on speedline 4.

Type Quantity (p. 1422)

Read Only No

Speed4MassFlow2

Mass flow rate 2 on speedline 4.

Type Quantity (p. 1422)

Read Only No

Speed4MassFlow3

Mass flow rate 3 on speedline 4.

Type Quantity (p. 1422)

Read Only No

Speed4MassFlow4

Mass flow rate 4 on speedline 4.

Type Quantity (p. 1422)

Read Only No

Speed4MassFlow5

Mass flow rate 5 on speedline 4.

Type Quantity (p. 1422)

Read Only No

Speed4MassFlow6

Mass flow rate 6 on speedline 4.

Type Quantity (p. 1422)

Read Only No

Speed4MassFlow7

Mass flow rate 7 on speedline 4.

Type Quantity (p. 1422)

Read Only No

Speed4MassFlow8

Mass flow rate 8 on speedline 4.

Type Quantity (p. 1422)

Read Only No

Speed4MassFlow9

Mass flow rate 9 on speedline 4.

Type Quantity (p. 1422)

Read Only No

Speed5MassFlow1

Mass flow rate 1 on speedline 5.

Type Quantity (p. 1422)

Read Only No

Speed5MassFlow10

Mass flow rate 10 on speedline 5.

Type Quantity (p. 1422)

Read Only No

Speed5MassFlow11

Mass flow rate 11 on speedline 5.

Type Quantity (p. 1422)

Read Only No

Speed5MassFlow12

Mass flow rate 12 on speedline 5.

Type Quantity (p. 1422)

Read Only No

Speed5MassFlow13

Mass flow rate 13 on speedline 5.

Type Quantity (p. 1422)

Read Only No

Speed5MassFlow14

Mass flow rate 14 on speedline 5.

Type Quantity (p. 1422)

Read Only No

Speed5MassFlow15

Mass flow rate 15 on speedline 5.

Type Quantity (p. 1422)

Read Only No

Speed5MassFlow2

Mass flow rate 2 on speedline 5.

Type Quantity (p. 1422)

Read Only No

Speed5MassFlow3

Mass flow rate 3 on speedline 5.

Type Quantity (p. 1422)

Read Only No

Speed5MassFlow4

Mass flow rate 4 on speedline 5.

Type Quantity (p. 1422)

Read Only No

Speed5MassFlow5

Mass flow rate 5 on speedline 5.

Type Quantity (p. 1422)

Read Only No

Speed5MassFlow6

Mass flow rate 6 on speedline 5.

Type Quantity (p. 1422)

Read Only No

Speed5MassFlow7

Mass flow rate 7 on speedline 5.

Type Quantity (p. 1422)

Read Only No

Speed5MassFlow8

Mass flow rate 8 on speedline 5.

Type Quantity (p. 1422)

Read Only No

Speed5MassFlow9

Mass flow rate 9 on speedline 5.

Type Quantity (p. 1422)

Read Only No

Speed6MassFlow1

Mass flow rate 1 on speedline 6.

Type Quantity (p. 1422)

Read Only No

Speed6MassFlow10

Mass flow rate 10 on speedline 6.

Type Quantity (p. 1422)

Read Only No

Speed6MassFlow11

Mass flow rate 11 on speedline 6.

Type Quantity (p. 1422)

Read Only No

Speed6MassFlow12

Mass flow rate 12 on speedline 6.

Type Quantity (p. 1422)

Read Only No

Speed6MassFlow13

Mass flow rate 13 on speedline 6.

Type Quantity (p. 1422)

Read Only No

Speed6MassFlow14

Mass flow rate 14 on speedline 6.

Type Quantity (p. 1422)

Read Only No

Speed6MassFlow15

Mass flow rate 15 on speedline 6.

Type Quantity (p. 1422)

Read Only No

Speed6MassFlow2

Mass flow rate 2 on speedline 6.

Type Quantity (p. 1422)

Read Only No

Speed6MassFlow3

Mass flow rate 3 on speedline 6.

Type Quantity (p. 1422)

Read Only No

Speed6MassFlow4

Mass flow rate 4 on speedline 6.

Type Quantity (p. 1422)

Read Only No

Speed6MassFlow5

Mass flow rate 5 on speedline 6.

Type Quantity (p. 1422)

Read Only No

Speed6MassFlow6

Mass flow rate 6 on speedline 6.

Type Quantity (p. 1422)

Read Only No

Speed6MassFlow7

Mass flow rate 7 on speedline 6.

Type Quantity (p. 1422)

Read Only No

Speed6MassFlow8

Mass flow rate 8 on speedline 6.

Type Quantity (p. 1422)

Read Only No

Speed6MassFlow9

Mass flow rate 9 on speedline 6.

Type Quantity (p. 1422)

Read Only No

Speed7MassFlow1

Mass flow rate 1 on speedline 7.

Type Quantity (p. 1422)

Read Only No

Speed7MassFlow10

Mass flow rate 10 on speedline 7.

Type Quantity (p. 1422)

Read Only No

Speed7MassFlow11

Mass flow rate 11 on speedline 7.

Type Quantity (p. 1422)

Read Only No

Speed7MassFlow12

Mass flow rate 12 on speedline 7.

Type Quantity (p. 1422)

Read Only No

Speed7MassFlow13

Mass flow rate 13 on speedline 7.

Type Quantity (p. 1422)

Read Only No

Speed7MassFlow14

Mass flow rate 14 on speedline 7.

Type Quantity (p. 1422)

Read Only No

Speed7MassFlow15

Mass flow rate 15 on speedline 7.

Type Quantity (p. 1422)

Read Only No

Speed7MassFlow2

Mass flow rate 2 on speedline 7.

Type Quantity (p. 1422)

Read Only No

Speed7MassFlow3

Mass flow rate 3 on speedline 7.

Type Quantity (p. 1422)

Read Only No

Speed7MassFlow4

Mass flow rate 4 on speedline 7.

Type Quantity (p. 1422)

Read Only No

Speed7MassFlow5

Mass flow rate 5 on speedline 7.

Type Quantity (p. 1422)

Read Only No

Speed7MassFlow6

Mass flow rate 6 on speedline 7.

Type Quantity (p. 1422)

Read Only No

Speed7MassFlow7

Mass flow rate 7 on speedline 7.

Type Quantity (p. 1422)

Read Only No

Speed7MassFlow8

Mass flow rate 8 on speedline 7.

Type Quantity (p. 1422)

Read Only No

Speed7MassFlow9

Mass flow rate 9 on speedline 7.

Type Quantity (p. 1422)

Read Only No

Speed8MassFlow1

Mass flow rate 1 on speedline 8.

Type Quantity (p. 1422)

Read Only No

Speed8MassFlow10

Mass flow rate 10 on speedline 8.

Type Quantity (p. 1422)

Read Only No

Speed8MassFlow11

Mass flow rate 11 on speedline 8.

Type Quantity (p. 1422)

Read Only No

Speed8MassFlow12

Mass flow rate 12 on speedline 8.

Type Quantity (p. 1422)

Read Only No

Speed8MassFlow13

Mass flow rate 13 on speedline 8.

Type Quantity (p. 1422)

Read Only No

Speed8MassFlow14

Mass flow rate 14 on speedline 8.

Type Quantity (p. 1422)

Read Only No

Speed8MassFlow15

Mass flow rate 15 on speedline 8.

Type Quantity (p. 1422)

Read Only No

Speed8MassFlow2

Mass flow rate 2 on speedline 8.

Type Quantity (p. 1422)

Read Only No

Speed8MassFlow3

Mass flow rate 3 on speedline 8.

Type Quantity (p. 1422)

Read Only No

Speed8MassFlow4

Mass flow rate 4 on speedline 8.

Type Quantity (p. 1422)

Read Only No

Speed8MassFlow5

Mass flow rate 5 on speedline 8.

Type Quantity (p. 1422)

Read Only No

Speed8MassFlow6

Mass flow rate 6 on speedline 8.

Type Quantity (p. 1422)

Read Only No

Speed8MassFlow7

Mass flow rate 7 on speedline 8.

Type Quantity (p. 1422)

Read Only No

Speed8MassFlow8

Mass flow rate 8 on speedline 8.

Type Quantity (p. 1422)

Read Only No

Speed8MassFlow9

Mass flow rate 9 on speedline 8.

Type Quantity (p. 1422)

Read Only No

Speed9MassFlow1

Mass flow rate 1 on speedline 9.

Type Quantity (p. 1422)

Read Only No

Speed9MassFlow10

Mass flow rate 10 on speedline 9.

Type Quantity (p. 1422)

Read Only No

Speed9MassFlow11

Mass flow rate 11 on speedline 9.

Type Quantity (p. 1422)

Read Only No

Speed9MassFlow12

Mass flow rate 12 on speedline 9.

Type Quantity (p. 1422)

Read Only No

Speed9MassFlow13

Mass flow rate 13 on speedline 9.

Type Quantity (p. 1422)

Read Only No

Speed9MassFlow14

Mass flow rate 14 on speedline 9.

Type Quantity (p. 1422)

Read Only No

Speed9MassFlow15

Mass flow rate 15 on speedline 9.

Type Quantity (p. 1422)

Read Only No

Speed9MassFlow2

Mass flow rate 2 on speedline 9.

Type Quantity (p. 1422)

Read Only No

Speed9MassFlow3

Mass flow rate 3 on speedline 9.

Type Quantity (p. 1422)

Read Only No

Speed9MassFlow4

Mass flow rate 4 on speedline 9.

Type Quantity (p. 1422)

Read Only No

Speed9MassFlow5

Mass flow rate 5 on speedline 9.

Type Quantity (p. 1422)

Read Only No

Speed9MassFlow6

Mass flow rate 6 on speedline 9.

Type Quantity (p. 1422)

Read Only No

Speed9MassFlow7

Mass flow rate 7 on speedline 9.

Type Quantity (p. 1422)

Read Only No

Speed9MassFlow8

Mass flow rate 8 on speedline 9.

Type Quantity (p. 1422)

Read Only No

Speed9MassFlow9

Mass flow rate 9 on speedline 9.

Type Quantity (p. 1422)

Read Only No

SpeedParam

Reference to the percentage speed parameter in the performance map

Type DataReference (p. 1371)

Read Only Yes

Speeds

Percentage speeds used to define the speedlines in the performance map

Type List (p. 1400)<int (p. 1394)>

Read Only Yes

StagPressure

This property specifies the design point stagnation pressure, P01, at the inlet to the machine.

Type Quantity (p. 1422)

Read Only No

StagTemp

This property specifies the design point stagnation temperature, T01, at the inlet to the machine.

Type Quantity (p. 1422)

Read Only No

TipConfig

This property specifies the configuration of the impeller shroud.

Available options:

Unshrouded impeller
Shrouded impeller

Type ImpellerType (p. 1391)

Read Only No

UseTipClearance

This property specifies whether the impeller uses a tip clearance. Note that this is only available for unshrouded impellers.

Type bool (p. 1360)

Read Only No

XField

persistent x axis field for the performance maps plot

Type string (p. 1438)

Read Only Yes

YField

persistent y axis field for the performance maps plot

Type string (p. 1438)

Read Only Yes

Vista AFD Analysis

This container holds Analysis data for an instance of Vista AFD.

Methods

CreateBladeDesign

This command class creates a new BladeGen model. An up-to-date BladeGen cell appears on the project schematic containing the new model.

Return Container representing the BladeGen cell.

Type DataContainerReference (p. 1371)

CreateGeometry

This command class creates a new BladeEditor model. An up-to-date Geometry cell appears on the project schematic containing the new model.

Return Container representing the Geometry cell.

Type DataContainerReference (p. 1371)

Edit

This command class launches the Vista popup GUI.

GetVistaAFDAnalysisProperties

This query takes a container reference and returns the Data Entity which contains user settings and properties for the VistaAFD Analysis container.

Return Reference to the requested Data Entity

Type DataReference (p. 1371)

Data Entities

VistaAFDAnalysis

Analysis represents a VistaAFD throughflow calculation used to analyse the design performance

Properties

Alpha1

User input, IGV exit angle

Type Quantity (p. 1422)

Read Only No

Alpha3

User input, OGV exit angle

Type Quantity (p. 1422)

Read Only No

AlphaOGVHub

OGV hub gas exit angle

Type Quantity (p. 1422)

Read Only No

AlphaOGVMean

OGV mean gas exit angle

Type Quantity (p. 1422)

Read Only No

AlphaRotHub

Rotor hub gas exit angle

Type Quantity (p. 1422)

Read Only No

AlphaRotMean

Rotor mean gas exit angle

Type Quantity (p. 1422)

Read Only No

Blade

Blade number to export to Bladegen (0=IGV, 1=Rotor, 2=OGV)

Type int (p. 1394)

Read Only No

BladeBetaExit

Blade exit angles

Type List (p. 1400)<List (p. 1400)<float (p. 1432)>>

Read Only No

BladeBetaInlet

Blade inlet angles

Type List (p. 1400)<List (p. 1400)<float (p. 1432)>>

Read Only No

BladeOption

Blade option for export to BladeGen

Type BladeType (p. 1358)

Read Only No

BMunits

BladeGen/BladeEditor units

Type string (p. 1438)

Read Only No

BMunitsOption

BladeGen/BladeEditor units option

Type BMunitsType (p. 1359)

Read Only No

DeHallerOGVHub

OGV hub DeHaller number

Type float (p. 1432)

Read Only No

DeHallerOGVMean

OGV mean DeHaller number

Type float (p. 1432)

Read Only No

DeHallerRotHub

Rotor hub DeHaller number

Type float (p. 1432)

Read Only No

DeHallerRotMean

Rotor mean DeHaller number

Type float (p. 1432)

Read Only No

DevIGVHub

IGV hub deviation

Type Quantity (p. 1422)

Read Only No

DevIGVMean

IGV mean deviation

Type Quantity (p. 1422)

Read Only No

DevOGVHub

OGV hub deviation

Type Quantity (p. 1422)

Read Only No

DevOGVMean

OGV mean deviation

Type Quantity (p. 1422)

Read Only No

DevRotHub

Rotor hub deviation

Type Quantity (p. 1422)

Read Only No

DevRotMean

Rotor mean deviation

Type Quantity (p. 1422)

Read Only No

DfOGVHub

OGV hub diffusion factor

Type float (p. 1432)

Read Only No

DfOGVMean

OGV mean diffusion factor

Type float (p. 1432)

Read Only No

DfRotHub

Rotor hub diffusion factor

Type float (p. 1432)

Read Only No

DfRotMean

Rotor mean diffusion factor

Type float (p. 1432)

Read Only No

Diameter

User input, outer diameter

Type Quantity (p. 1422)

Read Only No

DisplayText

The general property that defines the user-visible name of an entity. This property is defined for all data entities but is used only in those entities that present a label in the user interface.

Type string (p. 1438)

Read Only No

Eta

Aerodynamic efficiency

Type float (p. 1432)

Read Only No

EtaInput

User input, efficiency estimate

Type float (p. 1432)

Read Only No

EtaTS

System efficiency (t-s)

Type float (p. 1432)

Read Only No

EtaTSPipe

Downstream system efficiency (t-s)

Type float (p. 1432)

Read Only No

EtaTT

System efficiency (t-t)

Type float (p. 1432)

Read Only No

HeadRise

User input, total head rise

Type Quantity (p. 1422)

Read Only No

HtrIn

User input, hub/tip rotor inlet

Type float (p. 1432)

Read Only No

HtrOut

User input, hub/tip rotor outlet

Type float (p. 1432)

Read Only No

HubLoadParam

User input, hub loading parameter

Type float (p. 1432)

Read Only No

HubVelFactor

User input, hub velocity deficit factor

Type float (p. 1432)

Read Only No

HubX

Hub annulus X-coords

Type List (p. 1400)<Quantity (p. 1422)>

Read Only No

HubY

Hub annulus Y-coords

Type List (p. 1400)<Quantity (p. 1422)>

Read Only No

IGV

User input, IGV option

Type bool (p. 1360)

Read Only No

IGVhubThickX

IGV hub thickness X-coord

Type List (p. 1400)<float (p. 1432)>

Read Only No

IGVhubThickY

IGV hub thickness Y-coord

Type List (p. 1400)<Quantity (p. 1422)>

Read Only No

IGVleadingX

IGV leading edge X-coords

Type List (p. 1400)<Quantity (p. 1422)>

Read Only No

IGVleadingY

IGV leading edge Y-coords

Type List (p. 1400)<Quantity (p. 1422)>

Read Only No

IGVshrThickX

IGV shroud thickness X-coord

Type List (p. 1400)<float (p. 1432)>

Read Only No

IGVshrThickY

IGV shroud thickness Y-coord

Type List (p. 1400)<Quantity (p. 1422)>

Read Only No

IGVthetaLE

IGV leading edge theta

Type List (p. 1400)<float (p. 1432)>

Read Only No

IGVtrailingX

IGV trailing edge X-coords

Type List (p. 1400)<Quantity (p. 1422)>

Read Only No

IGVtrailingY

IGV trailing edge Y-coords

Type List (p. 1400)<Quantity (p. 1422)>

Read Only No

ImperialUnits

User input, Imperial units option

Type bool (p. 1360)

Read Only No

InnerIter

User input, number of inner loop design calculation iterations

Type int (p. 1394)

Read Only No

Layer1

Intermediate spanwise layer1 for Export

Type bool (p. 1360)

Read Only No

Layer2

Intermediate spanwise layer2 for Export

Type bool (p. 1360)

Read Only No

Layer3

Intermediate spanwise layer3 for Export

Type bool (p. 1360)

Read Only No

LoadHub

Rotor hub loading

Type float (p. 1432)

Read Only No

LoadMean

Rotor mean loading

Type float (p. 1432)

Read Only No

MassFlow

User input, mass flow rate

Type Quantity (p. 1422)

Read Only No

MaxLoadHub

Rotor maximum hub loading

Type float (p. 1432)

Read Only No

MaxLoadMean

Rotor maximum mean loading

Type float (p. 1432)

Read Only No

MixLoss

User input, downstream mixing losses

Type float (p. 1432)

Read Only No

NMain

Number of blades in each row

Type List (p. 1400)<int (p. 1394)>

Read Only No

OGV

User input, OGV option

Type bool (p. 1360)

Read Only No

OGVhubThickX

OGV hub thickness X-coord

Type List (p. 1400)<float (p. 1432)>

Read Only No

OGVhubThickY

OGV hub thickness Y-coord

Type List (p. 1400)<Quantity (p. 1422)>

Read Only No

OGVleadingX

OGV leading edge X-coords

Type List (p. 1400)<Quantity (p. 1422)>

Read Only No

OGVleadingY

OGV leading edge Y-coords

Type List (p. 1400)<Quantity (p. 1422)>

Read Only No

OGVshrThickX

OGV shroud thickness X-coord

Type List (p. 1400)<float (p. 1432)>

Read Only No

OGVshrThickY

OGV shroud thickness Y-coord

Type List (p. 1400)<Quantity (p. 1422)>

Read Only No

OGVthetaLE

OGV leading edge theta

Type List (p. 1400)<float (p. 1432)>

Read Only No

OGVtrailingX

OGV trailing edge X-coords

Type List (p. 1400)<Quantity (p. 1422)>

Read Only No

OGVtrailingY

OGV trailing edge Y-coords

Type List (p. 1400)<Quantity (p. 1422)>

Read Only No

Omega

User input, rotational speed

Type Quantity (p. 1422)

Read Only No

OuterIter

User input, number of outer loop design calculation iterations

Type int (p. 1394)

Read Only No

Pdyn

Outlet dynamic pressure

Type Quantity (p. 1422)

Read Only No

PdynPipe

Downstream dynamic pressure

Type Quantity (p. 1422)

Read Only No

PhiHub

Rotor hub flow coefficient

Type float (p. 1432)

Read Only No

PhiMean

Rotor mean flow coefficient

Type float (p. 1432)

Read Only No

Power

Power

Type Quantity (p. 1422)

Read Only No

RatioIGV

User input, IGV aspect ratio

Type float (p. 1432)

Read Only No

RatioOGV

User input, OGV aspect ratio

Type float (p. 1432)

Read Only No

RatioRotor

User input, Rotor aspect ratio

Type float (p. 1432)

Read Only No

RotorHubThickX

Rotor hub thickness X-coord

Type List (p. 1400)<float (p. 1432)>

Read Only No

RotorHubThickY

Rotor hub thickness Y-coord

Type List (p. 1400)<Quantity (p. 1422)>

Read Only No

RotorLeadingX

Rotor leading edge X-coords

Type List (p. 1400)<Quantity (p. 1422)>

Read Only No

RotorLeadingY

Rotor leading edge Y-coords

Type List (p. 1400)<Quantity (p. 1422)>

Read Only No

RotorShrThickX

Rotor shroud thickness X-coord

Type List (p. 1400)<float (p. 1432)>

Read Only No

RotorShrThickY

Rotor shroud thickness Y-coord

Type List (p. 1400)<Quantity (p. 1422)>

Read Only No

RotorThetaLE

Rotor leading edge theta

Type List (p. 1400)<float (p. 1432)>

Read Only No

RotorTrailingX

Rotor trailing edge X-coords

Type List (p. 1400)<Quantity (p. 1422)>

Read Only No

RotorTrailingY

Rotor trailing edge Y-coords

Type List (p. 1400)<Quantity (p. 1422)>

Read Only No

Sc90MaxIter

User input, maximum number of solver iterations

Type int (p. 1394)

Read Only No

Sc90Relax

User input, solver relaxation factor

Type float (p. 1432)

Read Only No

Sc90Tol

User input, solver tolerance

Type float (p. 1432)

Read Only No

ShrX

Shroud annulus X-coords

Type List (p. 1400)<Quantity (p. 1422)>

Read Only No

ShrY

Shroud annulus Y-coords

Type List (p. 1400)<Quantity (p. 1422)>

Read Only No

SIunits

User input, SI units option

Type bool (p. 1360)

Read Only No

Span

Spanwise fractions

Type List (p. 1400)<float (p. 1432)>

Read Only No

StagPressure

User input, inlet stagnation pressure

Type Quantity (p. 1422)

Read Only No

StagTemp

User input, inlet stagnation temperature

Type Quantity (p. 1422)

Read Only No

Torque

Torque

Type Quantity (p. 1422)

Read Only No

TrimIGV

User input, IGV profile trim

Type float (p. 1432)

Read Only No

TrimOGV

User input, OGV profile trim

Type float (p. 1432)

Read Only No

TrimRotor

User input, Rotor profile trim

Type float (p. 1432)

Read Only No

VanesIGV

User input, IGV number of vanes

Type int (p. 1394)

Read Only No

VanesOGV

User input, OGV number of vanes

Type int (p. 1394)

Read Only No

VanesRotor

User input, Rotor number of vanes

Type int (p. 1394)

Read Only No

VistaAFDTitle

Editor Title

Type string (p. 1438)

Read Only No

Vista AFD Design

This container holds Design data for an instance of Vista AFD.

Methods

CreateBladeDesign

This command class creates a new BladeGen model. An up-to-date BladeGen cell appears on the project schematic containing the new model.

Return Container representing the BladeGen cell.

Type DataContainerReference (p. 1371)

CreateGeometry

This command class creates a new BladeEditor model. An up-to-date Geometry cell appears on the project schematic containing the new model.

Return Container representing the Geometry cell.

Type DataContainerReference (p. 1371)

Edit

This command class launches the Vista popup GUI.

GetVistaAFDDesignProperties

This query takes a container reference and returns the Data Entity which contains user settings and properties for the VistaAFD Design container.

Return Reference to the requested Data Entity

Type DataReference (p. 1371)

Data Entities

VistaAFDDesign

Design represents a VistaAFD throughflow calculation used to define the blade profiles

Properties

Alpha1

User input, IGV exit angle

Type Quantity (p. 1422)

Read Only No

Alpha3

User input, OGV exit angle

Type Quantity (p. 1422)

Read Only No

AlphaOGVHub

OGV hub gas exit angle

Type Quantity (p. 1422)

Read Only No

AlphaOGVMean

OGV mean gas exit angle

Type Quantity (p. 1422)

Read Only No

AlphaRotHub

Rotor hub gas exit angle

Type Quantity (p. 1422)

Read Only No

AlphaRotMean

Rotor mean gas exit angle

Type Quantity (p. 1422)

Read Only No

Blade

Blade number to export to Bladegen (0=IGV, 1=Rotor, 2=OGV)

Type int (p. 1394)

Read Only No

BladeBetaExit

Blade exit angles

Type List (p. 1400)<List (p. 1400)<float (p. 1432)>>

Read Only No

BladeBetaInlet

Blade inlet angles

Type List (p. 1400)<List (p. 1400)<float (p. 1432)>>

Read Only No

BladeOption

Blade option for export to BladeGen

Type BladeType (p. 1358)

Read Only No

BMunits

BladeGen/BladeEditor units

Type string (p. 1438)

Read Only No

BMunitsOption

BladeGen/BladeEditor units option

Type BMunitsType (p. 1359)

Read Only No

DeHallerOGVHub

OGV hub DeHaller number

Type float (p. 1432)

Read Only No

DeHallerOGVMean

OGV mean DeHaller number

Type float (p. 1432)

Read Only No

DeHallerRotHub

Rotor hub DeHaller number

Type float (p. 1432)

Read Only No

DeHallerRotMean

Rotor mean DeHaller number

Type float (p. 1432)

Read Only No

DevIGVHub

IGV hub deviation

Type Quantity (p. 1422)

Read Only No

DevIGVMean

IGV mean deviation

Type Quantity (p. 1422)

Read Only No

DevOGVHub

OGV hub deviation

Type Quantity (p. 1422)

Read Only No

DevOGVMean

OGV mean deviation

Type Quantity (p. 1422)

Read Only No

DevRotHub

Rotor hub deviation

Type Quantity (p. 1422)

Read Only No

DevRotMean

Rotor mean deviation

Type Quantity (p. 1422)

Read Only No

DfOGVHub

OGV hub diffusion factor

Type float (p. 1432)

Read Only No

DfOGVMean

OGV mean diffusion factor

Type float (p. 1432)

Read Only No

DfRotHub

Rotor hub diffusion factor

Type float (p. 1432)

Read Only No

DfRotMean

Rotor mean diffusion factor

Type float (p. 1432)

Read Only No

Diameter

User input, outer diameter

Type Quantity (p. 1422)

Read Only No

DisplayText

The general property that defines the user-visible name of an entity. This property is defined for all data entities but is used only in those entities that present a label in the user interface.

Type string (p. 1438)

Read Only No

Eta

Aerodynamic efficiency

Type float (p. 1432)

Read Only No

EtaInput

User input, efficiency estimate

Type float (p. 1432)

Read Only No

EtaTS

System efficiency (t-s)

Type float (p. 1432)

Read Only No

EtaTSPipe

Downstream system efficiency (t-s)

Type float (p. 1432)

Read Only No

EtaTT

System efficiency (t-t)

Type float (p. 1432)

Read Only No

HeadRise

User input, total head rise

Type Quantity (p. 1422)

Read Only No

HtrIn

User input, hub/tip rotor inlet

Type float (p. 1432)

Read Only No

HtrOut

User input, hub/tip rotor outlet

Type float (p. 1432)

Read Only No

HubLoadParam

User input, hub loading parameter

Type float (p. 1432)

Read Only No

HubVelFactor

User input, hub velocity deficit factor

Type float (p. 1432)

Read Only No

HubX

Hub annulus X-coords

Type List (p. 1400)<Quantity (p. 1422)>

Read Only No

HubY

Hub annulus Y-coords

Type List (p. 1400)<Quantity (p. 1422)>

Read Only No

IGV

User input, IGV option

Type bool (p. 1360)

Read Only No

IGVhubThickX

IGV hub thickness X-coord

Type List (p. 1400)<float (p. 1432)>

Read Only No

IGVhubThickY

IGV hub thickness Y-coord

Type List (p. 1400)<Quantity (p. 1422)>

Read Only No

IGVleadingX

IGV leading edge X-coords

Type List (p. 1400)<Quantity (p. 1422)>

Read Only No

IGVleadingY

IGV leading edge Y-coords

Type List (p. 1400)<Quantity (p. 1422)>

Read Only No

IGVshrThickX

IGV shroud thickness X-coord

Type List (p. 1400)<float (p. 1432)>

Read Only No

IGVshrThickY

IGV shroud thickness Y-coord

Type List (p. 1400)<Quantity (p. 1422)>

Read Only No

IGVthetaLE

IGV leading edge theta

Type List (p. 1400)<float (p. 1432)>

Read Only No

IGVtrailingX

IGV trailing edge X-coords

Type List (p. 1400)<Quantity (p. 1422)>

Read Only No

IGVtrailingY

IGV trailing edge Y-coords

Type List (p. 1400)<Quantity (p. 1422)>

Read Only No

ImperialUnits

User input, Imperial units option

Type bool (p. 1360)

Read Only No

InnerIter

User input, number of inner loop design calculation iterations

Type int (p. 1394)

Read Only No

Layer1

Intermediate spanwise layer1 for Export

Type bool (p. 1360)

Read Only No

Layer2

Intermediate spanwise layer2 for Export

Type bool (p. 1360)

Read Only No

Layer3

Intermediate spanwise layer3 for Export

Type bool (p. 1360)

Read Only No

LoadHub

Rotor hub loading

Type float (p. 1432)

Read Only No

LoadMean

Rotor mean loading

Type float (p. 1432)

Read Only No

MassFlow

User input, mass flow rate

Type Quantity (p. 1422)

Read Only No

MaxLoadHub

Rotor maximum hub loading

Type float (p. 1432)

Read Only No

MaxLoadMean

Rotor maximum mean loading

Type float (p. 1432)

Read Only No

MixLoss

User input, downstream mixing losses

Type float (p. 1432)

Read Only No

NMain

Number of blades in each row

Type List (p. 1400)<int (p. 1394)>

Read Only No

OGV

User input, OGV option

Type bool (p. 1360)

Read Only No

OGVhubThickX

OGV hub thickness X-coord

Type List (p. 1400)<float (p. 1432)>

Read Only No

OGVhubThickY

OGV hub thickness Y-coord

Type List (p. 1400)<Quantity (p. 1422)>

Read Only No

OGVleadingX

OGV leading edge X-coords

Type List (p. 1400)<Quantity (p. 1422)>

Read Only No

OGVleadingY

OGV leading edge Y-coords

Type List (p. 1400)<Quantity (p. 1422)>

Read Only No

OGVshrThickX

OGV shroud thickness X-coord

Type List (p. 1400)<float (p. 1432)>

Read Only No

OGVshrThickY

OGV shroud thickness Y-coord

Type List (p. 1400)<Quantity (p. 1422)>

Read Only No

OGVthetaLE

OGV leading edge theta

Type List (p. 1400)<float (p. 1432)>

Read Only No

OGVtrailingX

OGV trailing edge X-coords

Type List (p. 1400)<Quantity (p. 1422)>

Read Only No

OGVtrailingY

OGV trailing edge Y-coords

Type List (p. 1400)<Quantity (p. 1422)>

Read Only No

Omega

User input, rotational speed

Type Quantity (p. 1422)

Read Only No

OuterIter

User input, number of outer loop design calculation iterations

Type int (p. 1394)

Read Only No

Pdyn

Outlet dynamic pressure

Type Quantity (p. 1422)

Read Only No

PdynPipe

Downstream dynamic pressure

Type Quantity (p. 1422)

Read Only No

PhiHub

Rotor hub flow coefficient

Type float (p. 1432)

Read Only No

PhiMean

Rotor mean flow coefficient

Type float (p. 1432)

Read Only No

Power

Power

Type Quantity (p. 1422)

Read Only No

RatioIGV

User input, IGV aspect ratio

Type float (p. 1432)

Read Only No

RatioOGV

User input, OGV aspect ratio

Type float (p. 1432)

Read Only No

RatioRotor

User input, Rotor aspect ratio

Type float (p. 1432)

Read Only No

RotorHubThickX

Rotor hub thickness X-coord

Type List (p. 1400)<float (p. 1432)>

Read Only No

RotorHubThickY

Rotor hub thickness Y-coord

Type List (p. 1400)<Quantity (p. 1422)>

Read Only No

RotorLeadingX

Rotor leading edge X-coords

Type List (p. 1400)<Quantity (p. 1422)>

Read Only No

RotorLeadingY

Rotor leading edge Y-coords

Type List (p. 1400)<Quantity (p. 1422)>

Read Only No

RotorShrThickX

Rotor shroud thickness X-coord

Type List (p. 1400)<float (p. 1432)>

Read Only No

RotorShrThickY

Rotor shroud thickness Y-coord

Type List (p. 1400)<Quantity (p. 1422)>

Read Only No

RotorThetaLE

Rotor leading edge theta

Type List (p. 1400)<float (p. 1432)>

Read Only No

RotorTrailingX

Rotor trailing edge X-coords

Type List (p. 1400)<Quantity (p. 1422)>

Read Only No

RotorTrailingY

Rotor trailing edge Y-coords

Type List (p. 1400)<Quantity (p. 1422)>

Read Only No

Sc90MaxIter

User input, maximum number of solver iterations

Type int (p. 1394)

Read Only No

Sc90Relax

User input, solver relaxation factor

Type float (p. 1432)

Read Only No

Sc90Tol

User input, solver tolerance

Type float (p. 1432)

Read Only No

ShrX

Shroud annulus X-coords

Type List (p. 1400)<Quantity (p. 1422)>

Read Only No

ShrY

Shroud annulus Y-coords

Type List (p. 1400)<Quantity (p. 1422)>

Read Only No

SIunits

User input, SI units option

Type bool (p. 1360)

Read Only No

Span

Spanwise fractions

Type List (p. 1400)<float (p. 1432)>

Read Only No

StagPressure

User input, inlet stagnation pressure

Type Quantity (p. 1422)

Read Only No

StagTemp

User input, inlet stagnation temperature

Type Quantity (p. 1422)

Read Only No

Torque

Torque

Type Quantity (p. 1422)

Read Only No

TrimIGV

User input, IGV profile trim

Type float (p. 1432)

Read Only No

TrimOGV

User input, OGV profile trim

Type float (p. 1432)

Read Only No

TrimRotor

User input, Rotor profile trim

Type float (p. 1432)

Read Only No

VanesIGV

User input, IGV number of vanes

Type int (p. 1394)

Read Only No

VanesOGV

User input, OGV number of vanes

Type int (p. 1394)

Read Only No

VanesRotor

User input, Rotor number of vanes

Type int (p. 1394)

Read Only No

VistaAFDTitle

Editor Title

Type string (p. 1438)

Read Only No

Vista AFD Meanline

This container holds Meanline data for an instance of Vista AFD.

Methods

Edit

This command class launches the Vista popup GUI.

GetVistaAFDMeanlineProperties

This query takes a container reference and returns the Data Entity which contains user settings and properties for the VistaAFD Meanline container.

Return Reference to the requested Data Entity

Type DataReference (p. 1371)

Import

This command imports Vista data into the Blade Design cell from an existing BladeGen file. If no appropriate Vista data is found in the specified BladeGen file, an error message is generated.

```
template1 = GetTemplate(TemplateName="VistaCPD")
system1 = template1.CreateSystem()
bladeDesign1 = system1.GetContainer(ComponentName="Blade Design")
bladeDesign1.Import(FilePath="myfilepath/pump.bgd")
```

Required Arguments

FileName Name, and path, of the BladeGen file to be imported.

Type string (p. 1438)

Data Entities

VistaAFDMeanline

Meanline represents a VistaAFD meanline calculation as an initial 1D design

Properties

Alpha1

User input, IGV exit angle

Type Quantity (p. 1422)

Read Only No

Alpha3

User input, OGV exit angle

Type Quantity (p. 1422)

Read Only No

AlphaOGVHub

OGV hub gas exit angle

Type Quantity (p. 1422)

Read Only No

AlphaOGVMean

OGV mean gas exit angle

Type Quantity (p. 1422)

Read Only No

AlphaRotHub

Rotor hub gas exit angle

Type Quantity (p. 1422)

Read Only No

AlphaRotMean

Rotor mean gas exit angle

Type Quantity (p. 1422)

Read Only No

DeHallerOGVHub

OGV hub DeHaller number

Type float (p. 1432)

Read Only No

DeHallerOGVMean

OGV mean DeHaller number

Type float (p. 1432)

Read Only No

DeHallerRotHub

Rotor hub DeHaller number

Type float (p. 1432)

Read Only No

DeHallerRotMean

Rotor mean DeHaller number

Type float (p. 1432)

Read Only No

Diameter

User input, outer diameter

Type Quantity (p. 1422)

Read Only No

DisplayText

The general property that defines the user-visible name of an entity. This property is defined for all data entities but is used only in those entities that present a label in the user interface.

Type string (p. 1438)

Read Only No

Eta

Aerodynamic efficiency

Type float (p. 1432)

Read Only No

EtaInput

User input, efficiency estimate

Type float (p. 1432)

Read Only No

EtaTS

System efficiency (t-s)

Type float (p. 1432)

Read Only No

EtaTSPipe

Downstream system efficiency (t-s)

Type float (p. 1432)

Read Only No

EtaTT

System efficiency (t-t)

Type float (p. 1432)

Read Only No

HeadRise

User input, total head rise

Type Quantity (p. 1422)

Read Only No

HtrIn

User input, hub/tip rotor inlet

Type float (p. 1432)

Read Only No

HtrOut

User input, hub/tip rotor outlet

Type float (p. 1432)

Read Only No

HubLoadParam

User input, hub loading parameter

Type float (p. 1432)

Read Only No

HubVelFactor

User input, hub velocity deficit factor

Type float (p. 1432)

Read Only No

HubX

Hub annulus X-coords

Type List (p. 1400)<Quantity (p. 1422)>

Read Only No

HubY

Hub annulus Y-coords

Type List (p. 1400)<Quantity (p. 1422)>

Read Only No

IGV

User input, IGV option

Type bool (p. 1360)

Read Only No

ImperialUnits

User input, Imperial units option

Type bool (p. 1360)

Read Only No

LoadHub

Rotor hub loading

Type float (p. 1432)

Read Only No

LoadMean

Rotor mean loading

Type float (p. 1432)

Read Only No

MassFlow

User input, mass flow rate

Type Quantity (p. 1422)

Read Only No

MaxLoadHub

Rotor maximum hub loading

Type float (p. 1432)

Read Only No

MaxLoadMean

Rotor maximum mean loading

Type float (p. 1432)

Read Only No

MixLoss

User input, downstream mixing losses

Type float (p. 1432)

Read Only No

OGV

User input, OGV option

Type bool (p. 1360)

Read Only No

Omega

User input, rotational speed

Type Quantity (p. 1422)

Read Only No

Pdyn

Outlet dynamic pressure

Type Quantity (p. 1422)

Read Only No

PdynPipe

Downstream dynamic pressure

Type Quantity (p. 1422)

Read Only No

PhiHub

Rotor hub flow coefficient

Type float (p. 1432)

Read Only No

PhiMean

Rotor mean flow coefficient

Type float (p. 1432)

Read Only No

Power

Power

Type Quantity (p. 1422)

Read Only No

RatioIGV

User input, IGV aspect ratio

Type float (p. 1432)

Read Only No

RatioOGV

User input, OGV aspect ratio

Type float (p. 1432)

Read Only No

RatioRotor

User input, Rotor aspect ratio

Type float (p. 1432)

Read Only No

ShrX

Shroud annulus X-coords

Type List (p. 1400)<Quantity (p. 1422)>

Read Only No

ShrY

Shroud annulus Y-coords

Type List (p. 1400)<Quantity (p. 1422)>

Read Only No

SIunits

User input, SI units option

Type bool (p. 1360)

Read Only No

StagPressure

User input, inlet stagnation pressure

Type Quantity (p. 1422)

Read Only No

StagTemp

User input, inlet stagnation temperature

Type Quantity (p. 1422)

Read Only No

Torque

Torque

Type Quantity (p. 1422)

Read Only No

TrimIGV

User input, IGV profile trim

Type float (p. 1432)

Read Only No

TrimOGV

User input, OGV profile trim

Type float (p. 1432)

Read Only No

TrimRotor

User input, Rotor profile trim

Type float (p. 1432)

Read Only No

VanesIGV

User input, IGV number of vanes

Type int (p. 1394)

Read Only No

VanesOGV

User input, OGV number of vanes

Type int (p. 1394)

Read Only No

VanesRotor

User input, Rotor number of vanes

Type int (p. 1394)

Read Only No

VistaAFDTitle

Editor Title

Type string (p. 1438)

Read Only No

Vista CCD

This container holds Analysis data for an instance of Vista CCD.

Methods

CreateBladeDesign

This command class creates a new BladeGen model. An up-to-date BladeGen cell appears on the project schematic containing the new model.

Return Container representing the BladeGen cell.

Type DataContainerReference (p. 1371)

CreateGeometry

This command class creates a new BladeEditor model. An up-to-date Geometry cell appears on the project schematic containing the new model.

Return Container representing the Geometry cell.

Type DataContainerReference (p. 1371)

CreateTF

This command class creates a new Turbo Setup system, a new geometry component (BladeGen or BladeEditor) and a Vista TF system. The geometry and Vista TF systems are linked appropriately and inputs populated appropriately ready for update.

Optional Arguments

UseBladegen Indicates whether to use a Bladegen cell or a BladeEditor(Geometry) cell

Type bool (p. 1360)

Default Value False

CreateThroughflow

This command class creates a new throughflow system. The system, comprising Geometry, Setup, Solution and Results cells, is created on the project schematic and is updated performing the throughflow analysis automatically.

Optional Arguments

UseBladegen Indicates whether to use a Bladegen cell or a BladeEditor(Geometry) cell

Type bool (p. 1360)

Default Value False

CreateTurboflow

This command class creates new Turbo Setup, Geometry (BladeGen or BladeEditor) and Turbomachinery Fluid Flow systems. The new systems appear on the project schematic. The Turbo Setup and Geometry components are updated. The Turbo Fluid Flow system components are initialised and the Turbo Mesh component is connected to the upstream Geometry.

Optional Arguments

UseBladegen Indicates whether to use a Bladegen cell or a BladeEditor(Geometry) cell

Type bool (p. 1360)

Default Value False

CreateTurboSetup

This command class creates a new Turbo Setup system. The Turbo Setup is populated with data from the Vista Tools component it's launched from

Return Container representing the Turbo Setup cell.

Type DataContainerReference (p. 1371)

Edit

This command class launches the Vista popup GUI.

GetVistaCCDBladeDesignProperties

This query takes a container reference and returns the Data Entity which contains user settings and properties for the VistaCCD Blade Design container.

Return Reference to the requested Data Entity

Type DataReference (p. 1371)

Import

This command imports Vista data into the Blade Design cell from an existing BladeGen file. If no appropriate Vista data is found in the specified BladeGen file, an error message is generated.

```
template1 = GetTemplate(TemplateName="VistaCPD")
system1 = template1.CreateSystem()
bladeDesign1 = system1.GetContainer(ComponentName="Blade Design")
bladeDesign1.Import(FilePath="myfilepath/pump.bgd")
```

Required Arguments

FileName Name, and path, of the BladeGen file to be imported.

Type string (p. 1438)

Data Entities

VistaCCDBladeDesign

This data entity provides access to the properties which define the VistaCCD project. This includes both the input and the results properties.

Properties

Acentric

This property specifies the acentric factor for the working fluid. Note that this is only available when MaterialPropsOption is set to 'User' and the GasModelOption is set to 'Real'.

Type float (p. 1432)

Read Only No

Alpha3

This property specifies the flow angle at the impeller inlet in the absolute reference frame.

Type Quantity (p. 1422)

Read Only No

Alpha5rms

This property reports the absolute flow angle at the impeller trailing edge.

Type Quantity (p. 1422)

Read Only No

AnChkRatio

This property reports the annulus choke ratio.

Type float (p. 1432)

Read Only No

B5

This property reports the axial distance between hub and shroud at the impeller trailing edge (tip width).

Type Quantity (p. 1422)

Read Only No

Beta5rms

This property reports the relative flow angle at the impeller trailing edge.

Type Quantity (p. 1422)

Read Only No

BetaBlade3HubUser

This property specifies the hub leading edge blade angle. Note that this is NOT available when the StackingOption is set to 'Radial'.

Type Quantity (p. 1422)

Read Only No

BetaBlade3ShrUser

This property specifies the impeller shroud leading edge blade angle. Note that this is only available when the ShroudDiameterOption is set to 'Angle'.

Type Quantity (p. 1422)

Read Only No

BetaBlade5

This property specifies the impeller backsweep angle.

Type Quantity (p. 1422)

Read Only No

BetaBladeLEhub

This property reports the blade angle at the impeller leading edge hub location.

Type Quantity (p. 1422)

Read Only No

BetaBladeLErms

This property reports the blade angle at the impeller leading edge meanline location.

Type Quantity (p. 1422)

Read Only No

BetaBladeLEshr

This property reports the blade angle at the impeller leading edge shroud location.

Type Quantity (p. 1422)

Read Only No

BetaLEhub

This property reports the relative flow angle at the impeller leading edge hub location.

Type Quantity (p. 1422)

Read Only No

BetaLErms

This property reports the relative flow angle at the impeller leading edge meanline location.

Type Quantity (p. 1422)

Read Only No

BetaLEshr

This property reports the relative flow angle at the impeller leading edge shroud location.

Type Quantity (p. 1422)

Read Only No

BMunitsOption

This property specifies the units used when creating a new BladeGen/BladeEditor model. Note that this is independent of the units used in the VistaCCD popup GUI.

Available options:

mm
cm
inches
ft
m

Type BMunitsType (p. 1359)

Read Only No

ChkRatio

This property reports the impeller choke ratio.

Type float (p. 1432)

Read Only No

ChokeUser

This property specifies the impeller choke ratio. Note that this is only available when the IncidenceOption is set to 'choke'.

Type float (p. 1432)

Read Only No

ClearanceOption

This property specifies impeller tip clearance is specified. 'Ratio' indicates that the tip clearance is specified as a fraction of the tip width 'User' specifies that the clearance will be defined directly by the user.

Available options:

Ratio
User

Type ClearanceType (p. 1364)

Read Only No

ClearRatio

This property reports the axial tip clearance ratio of the impeller.

Type float (p. 1432)

Read Only No

ClearRatioUser

This property specifies the ratio of the impeller tip clearance to the tip width. Note that this is only available when ClearanceOption is set to 'Ratio'.

Type float (p. 1432)

Read Only No

ClearUser

This property specifies the value of the impeller tip clearance. Note that this is only available when ClearanceOption is set to 'User'.

Type Quantity (p. 1422)

Read Only No

CorrelationOption

This property specifies the correlation used to calculate the stage efficiency. Note that this is only available when the EfficiencyOption is set to 'Correlation'.

Available options:

CaseyRobinson
CaseyMarty
Rodgers

Type EtaCorrelType (p. 1382)

Read Only No

Cp_A0

For a user-defined real gas, the specific heat capacity is specified as a polynomial function of temperature over two temperature ranges. This property specifies the constant component (coefficient of T^0) of the lower temperature range polynomial. Note that this is only available when MaterialPropsOption is set to 'User' and the GasModelOption is set to 'Real'.

Type double (p. 1378)

Read Only No

Cp_A1

For a user-defined real gas, the specific heat capacity is specified as a polynomial function of temperature over two temperature ranges. This property specifies the coefficient of T^1 of the lower temperature range polynomial. Note that this is only available when MaterialPropsOption is set to 'User' and the GasModelOption is set to 'Real'.

Type double (p. 1378)

Read Only No

Cp_A2

For a user-defined real gas, the specific heat capacity is specified as a polynomial function of temperature over two temperature ranges. This property specifies the coefficient of T^2 of the lower temperature range polynomial. Note that this is only available when MaterialPropsOption is set to 'User' and the GasModelOption is set to 'Real'.

Type double (p. 1378)

Read Only No

Cp_A3

For a user-defined real gas, the specific heat capacity is specified as a polynomial function of temperature over two temperature ranges. This property specifies the coefficient of T^3 of the lower temperature range polynomial. Note that this is only available when MaterialPropsOption is set to 'User' and the GasModelOption is set to 'Real'.

Type double (p. 1378)

Read Only No

Cp_A4

For a user-defined real gas, the specific heat capacity is specified as a polynomial function of temperature over two temperature ranges. This property specifies the coefficient of T^4 of the lower temperature range polynomial. Note that this is only available when MaterialPropsOption is set to 'User' and the GasModelOption is set to 'Real'.

Type double (p. 1378)

Read Only No

Cp_A5

For a user-defined real gas, the specific heat capacity is specified as a polynomial function of temperature over two temperature ranges. This property specifies the coefficient of T^5 of the lower temperature range polynomial. Note that this is only available when MaterialPropsOption is set to 'User' and the GasModelOption is set to 'Real'.

Type double (p. 1378)

Read Only No

Cp_A6

For a user-defined real gas, the specific heat capacity is specified as a polynomial function of temperature over two temperature ranges. This property specifies the coefficient of T^6 of the lower temperature range polynomial. Note that this is only available when MaterialPropsOption is set to 'User' and the GasModelOption is set to 'Real'.

Type double (p. 1378)

Read Only No

Cp_A7

For a user-defined real gas, the specific heat capacity is specified as a polynomial function of temperature over two temperature ranges. This property specifies the coefficient of T^7 of the lower temperature range polynomial. Note that this is only available when MaterialPropsOption is set to 'User' and the GasModelOption is set to 'Real'.

Type double (p. 1378)

Read Only No

Cp_Amax

For a user-defined real gas, the specific heat capacity is specified as a polynomial function of temperature over two temperature ranges. This property specifies the maximum temperature limit for which the lower temperature range polynomial is applicable. Note that this is only available when MaterialPropsOption is set to 'User' and the GasModelOption is set to 'Real'.

Type float (p. 1432)

Read Only No

Cp_Amin

For a user-defined real gas, the specific heat capacity is specified as a polynomial function of temperature over two temperature ranges. This property specifies the minimum temperature limit for which the lower temperature range polynomial is applicable. Note that this is only available when MaterialPropsOption is set to 'User' and the GasModelOption is set to 'Real'.

Type float (p. 1432)

Read Only No

Cp_B0

For a user-defined real gas, the specific heat capacity is specified as a polynomial function of temperature over two temperature ranges. This property specifies the constant component (coefficient of T^0) of the upper temperature range polynomial. Note that this is only available when MaterialPropsOption is set to 'User' and the GasModelOption is set to 'Real'.

Type double (p. 1378)

Read Only No

Cp_B1

For a user-defined real gas, the specific heat capacity is specified as a polynomial function of temperature over two temperature ranges. This property specifies the coefficient of T^1 of the upper temperature range polynomial. Note that this is only available when MaterialPropsOption is set to 'User' and the GasModelOption is set to 'Real'.

Type double (p. 1378)

Read Only No

Cp_B2

For a user-defined real gas, the specific heat capacity is specified as a polynomial function of temperature over two temperature ranges. This property specifies the coefficient of T^2 of the upper temperature range polynomial. Note that this is only available when MaterialPropsOption is set to 'User' and the GasModelOption is set to 'Real'.

Type double (p. 1378)

Read Only No

Cp_B3

For a user-defined real gas, the specific heat capacity is specified as a polynomial function of temperature over two temperature ranges. This property specifies the coefficient of T^3 of the upper temperature range polynomial. Note that this is only available when MaterialPropsOption is set to 'User' and the GasModelOption is set to 'Real'.

Type double (p. 1378)

Read Only No

Cp_B4

For a user-defined real gas, the specific heat capacity is specified as a polynomial function of temperature over two temperature ranges. This property specifies the coefficient of T^4 of the upper temperature range polynomial. Note that this is only available when MaterialPropsOption is set to 'User' and the GasModelOption is set to 'Real'.

Type double (p. 1378)

Read Only No

Cp_B5

For a user-defined real gas, the specific heat capacity is specified as a polynomial function of temperature over two temperature ranges. This property specifies the coefficient of T^5 of the upper temperature range polynomial. Note that this is only available when MaterialPropsOption is set to 'User' and the GasModelOption is set to 'Real'.

Type double (p. 1378)

Read Only No

Cp_B6

For a user-defined real gas, the specific heat capacity is specified as a polynomial function of temperature over two temperature ranges. This property specifies the coefficient of T^6 of the upper temperature range polynomial. Note that this is only available when MaterialPropsOption is set to 'User' and the GasModelOption is set to 'Real'.

Type double (p. 1378)

Read Only No

Cp_B7

For a user-defined real gas, the specific heat capacity is specified as a polynomial function of temperature over two temperature ranges. This property specifies the coefficient of T^7 of the upper temperature range polynomial. Note that this is only available when MaterialPropsOption is set to 'User' and the GasModelOption is set to 'Real'.

Type double (p. 1378)

Read Only No

Cp_Bmax

For a user-defined real gas, the specific heat capacity is specified as a polynomial function of temperature over two temperature ranges. This property specifies the maximum temperature limit for which the upper temperature range polynomial is applicable. Note that this is only available when MaterialPropsOption is set to 'User' and the GasModelOption is set to 'Real'.

Type float (p. 1432)

Read Only No

CriticalPressure

This property specifies the critical pressure for the working fluid. Note that this is only available when MaterialPropsOption is set to 'User' and the GasModelOption is set to 'Real'.

Type Quantity (p. 1422)

Read Only No

CriticalTemp

This property specifies the critical temperature for the working fluid. Note that this is only available when MaterialPropsOption is set to 'User' and the GasModelOption is set to 'Real'.

Type Quantity (p. 1422)

Read Only No

CriticalVol

This property specifies the critical volume for the working fluid. Note that this is only available when MaterialPropsOption is set to 'User' and the GasModelOption is set to 'Real'.

Type Quantity (p. 1422)

Read Only No

D3Hub

This property specifies the hub inlet diameter for the impeller.

Type Quantity (p. 1422)

Read Only No

D3ShrUser

This property specifies the impeller shroud diameter at the leading edge. Note that this is only available when the ShroudDiameterOption is set to 'Diameter'.

Type Quantity (p. 1422)

Read Only No

D5

This property reports the diameter at the impeller trailing edge (tip diameter).

Type Quantity (p. 1422)

Read Only No

Diffuser

This property specifies whether the diffuser section is vaned or vaneless.

Available options:

Vaned
Vaneless

Type DiffuserType (p. 1376)

Read Only No

DisplayText

The general property that defines the user-visible name of an entity. This property is defined for all data entities but is used only in those entities that present a label in the user interface.

Type string (p. 1438)

Read Only No

DLEhub

This property reports the diameter at the impeller leading edge hub location.

Type Quantity (p. 1422)

Read Only No

DLErms

This property reports the diameter at the impeller leading edge meanline location.

Type Quantity (p. 1422)

Read Only No

DLEshr

This property reports the diameter at the impeller leading edge shroud location.

Type Quantity (p. 1422)

Read Only No

EfficiencyOption

This property specifies whether to use a correlation to automatically calculate the compressor stage efficiency, or to use a user-defined efficiency.

Available options:

Correlation
User

```
template1 = GetTemplate(TemplateName="VistaCCD")
system1 = template1.CreateSystem()
bladeDesign1 = system1.GetContainer(ComponentName="Blade Design")
vistaCCDProperties1 = bladeDesign1.GetVistaCCDBladeDesignProperties()
vistaCCDProperties1.EfficiencyOption = "Correlation"
```

Type EtaType (p. 1383)

Read Only No

EtaIsen

This property reports the isentropic efficiency for the compressor stage.

Type float (p. 1432)

Read Only No

EtaIsenImp

This property reports the isentropic efficiency for the compressor impeller.

Type float (p. 1432)

Read Only No

EtaIsenImpUser

This property specifies the user defined impeller isentropic efficiency. Note that this is only available when the ImpellerEfficiencyOption is set to 'User'.

Type float (p. 1432)

Read Only No

EtaIsenUser

This property specifies the user defined stage isentropic efficiency. Note that this is only available when the UserEfficiencyOption is set to 'Isentropic'.

Type float (p. 1432)

Read Only No

EtaPoly

This property reports the polytropic efficiency for the compressor stage.

Type float (p. 1432)

Read Only No

EtaPolyImp

This property reports the polytropic efficiency for the compressor impeller.

Type float (p. 1432)

Read Only No

EtaPolyUser

This property specifies the user defined stage polytropic efficiency. Note that this is only available when the UserEfficiencyOption is set to 'Polytropic'.

Type float (p. 1432)

Read Only No

Gamma

This property reports the ratio of specific heats of the working fluid.

Type float (p. 1432)

Read Only No

GammaUser

This property specifies the ratio of specific heats for the working fluid. Note that this is only available when MaterialPropsOption is set to 'User' and the GasModelOption is set to 'Ideal'.

Type float (p. 1432)

Read Only No

GasModelOption

This property specifies whether to treat the working fluid as an Ideal or a Real gas.

Available options:

Ideal
Real

Type GasModelType (p. 1388)

Read Only No

GeometryStyle

This property specifies the approach taken when creating a new Geometry model from a successful VistaCCD calculation.

Available options:

Interactive
Parametric

Type GeometryStyleType (p. 1388)

Read Only No

H05

This property reports the total enthalpy at the impeller trailing edge.

Type Quantity (p. 1422)

Read Only No

H0LE

This property reports the stagnation enthalpy at the impeller leading edge.

Type Quantity (p. 1422)

Read Only No

Impeller

This property specifies whether the impeller is unshrouded or shrouded.

Available options:

Unshrouded
Shrouded

Type ImpellerType (p. 1391)

Read Only No

ImpellerEfficiencyOption

This property specifies whether to automatically calculate the impeller efficiency by linking this to the stage efficiency, or to use a user-defined efficiency.

Available options:

LinkToStage
User

Type EtaImpType (p. 1382)

Read Only No

ImpellerLength

This property specifies the impeller axial length to tip diameter ratio.

Type float (p. 1432)

Read Only No

ImpellerLengthOption

This property specifies whether the impeller length ratio is calculated automatically, or specified by the user.

Available options:

Automatic
User

Type ImpellerLengthType (p. 1391)

Read Only No

ImpellerLengthUserOpt

This property specifies whether the impeller axial length ratio will be defined by the user.

Type bool (p. 1360)

Read Only No

IncidenceOption

This property specifies the method used to calculate the incidence at the impeller shroud. The incidence may be either specified directly or calculated using the specified choke ratio.

Available options:

incidence
choke

Type IncidenceType (p. 1393)

Read Only No

IncLEhub

This property reports the impeller incidence at the hub location.

Type Quantity (p. 1422)

Read Only No

IncLErms

This property reports the impeller incidence at the meanline location.

Type Quantity (p. 1422)

Read Only No

IncLEshr

This property reports the impeller incidence at the shroud location.

Type Quantity (p. 1422)

Read Only No

IncShrUser

This property specifies the incidence at the impeller shroud. Note that this is only available when the IncidenceOption is set to 'incidence'.

Type Quantity (p. 1422)

Read Only No

LEInclination

This property reports the leading edge angle of inclination in the meridional plane.

Type Quantity (p. 1422)

Read Only No

LEInclinationUser

This property specifies the inclination of the leading edge relative to a radial line in the meridional view.

Type Quantity (p. 1422)

Read Only No

Loading

This property reports the impeller loading parameter (delH/U^2).

Type float (p. 1432)

Read Only No

M5rms

This property reports the absolute Mach number at the impeller trailing edge.

Type float (p. 1432)

Read Only No

MachU5

This property reports the blade Mach number at the impeller trailing edge (tip Mach number).

Type float (p. 1432)

Read Only No

MassFlow

This property specifies the design point mass flow rate passing through the compressor stage.

Type Quantity (p. 1422)

Read Only No

MaterialNameSelection

This property specifies the name of the working fluid, as selected from the database. Note that this is only available when the MaterialPropsOption is set to 'Database'.

Available options:

air
carbon_dioxide
hydrogen
methane
nitrogen
oxygen
parahydrogen
propylene
R123
R125
R134a
R141b
R142b
R245fa
water

```
template1 = GetTemplate(TemplateName="VistaCCD")
system1 = template1.CreateSystem()
bladeDesign1 = system1.GetContainer(ComponentName="Blade Design")
```

```
vistaCCDProperties1 = bladeDesign1.GetVistaCCDBladeDesignProperties()
vistaCCDProperties1.MaterialPropsOption = "Database"
vistaCCDProperties1.MaterialNameSelection = "nitrogen"
```

Type MaterialNamesList (p. 1402)

Read Only No

MaterialPropsOption

This property specifies whether the working fluid properties are chosen from the materials database or specified directly by the user.

Available options:

Database
User

Type MaterialPropsType (p. 1402)

Read Only No

MerVelGrad

This property specifies the gradient of the velocity profile from hub to shroud at the impeller leading edge. The gradient is set using the ratio of the meridional velocity at the shroud leading edge radius to that at the average leading edge radius.

Type float (p. 1432)

Read Only No

MrelLEhub

This property reports the relative Mach number at the impeller leading edge hub location.

Type float (p. 1432)

Read Only No

MrelLErms

This property reports the relative Mach number at the impeller leading edge meanline location.

Type float (p. 1432)

Read Only No

MrelLEshr

This property reports the relative Mach number at the impeller leading edge shroud location.

Type float (p. 1432)

Read Only No

MrmsLE

This property reports the absolute Mach number at the impeller leading edge meanline location.

Type float (p. 1432)

Read Only No

Mu

This property reports the dynamic viscosity of the working fluid.

Type Quantity (p. 1422)

Read Only No

MuUser

This property specifies the dynamic viscosity of the working fluid.

Type Quantity (p. 1422)

Read Only No

NMain

This property specifies the number of impeller main vanes.

Type int (p. 1394)

Read Only No

NormalToHubLE

This property specifies that the main impeller blade leading is normal to the hub curve

Type bool (p. 1360)

Read Only No

Ns

This property reports the specific speed of the impeller.

Type float (p. 1432)

Read Only No

NSplit

This property specifies the number of impeller splitter vanes. Note that this MUST be a multiple of the number of impeller main vanes.

Type int (p. 1394)

Read Only No

Nu

This property reports the kinematic viscosity of the working fluid.

Type Quantity (p. 1422)

Read Only No

NuUser

This property specifies the kinematic viscosity of the working fluid.

Type Quantity (p. 1422)

Read Only No

Omega

This property specifies the design point rotational speed of the impeller.

Type Quantity (p. 1422)

Read Only No

P05rms

This property reports the total pressure at the impeller trailing edge.

Type Quantity (p. 1422)

Read Only No

P5rms

This property reports the static pressure at the impeller trailing edge.

Type Quantity (p. 1422)

Read Only No

PIF

This property reports the power input factor of the compressor.

Type float (p. 1432)

Read Only No

PIFOption

This property specifies whether the power input factor is calculated using a correlation, or specified by the user.

Available options:

Correlation
User

Type PIFType (p. 1416)

Read Only No

PIFUser

This property specifies the user defined power input factor. Note that this is only available when the PIFOption is set to 'User'.

Type float (p. 1432)

Read Only No

Power

This property reports the impeller power.

Type Quantity (p. 1422)

Read Only No

PressureRatio

This property specifies the design point total-to-total pressure ratio for the compressor stage.

Type float (p. 1432)

Read Only No

PreswirlOption

This property specifies how the radial distribution of the preswirl angle is calculated.

Available options:

constant
free
forced
linear

Type PreswirlType (p. 1419)

Read Only No

RakeAngle

This property specifies the impeller rake angle (trailing edge lean angle).

Type Quantity (p. 1422)

Read Only No

Reb5

This property reports the Reynolds number based on the impeller tip width dimension.

Type float (p. 1432)

Read Only No

ReCorrectOpt

This property specifies if the Reynolds number correction is to be made to the stage efficiency correlation. Note that this is only available when the EfficiencyOption is set to 'Correlation'.

Type bool (p. 1360)

Read Only No

Red5

This property reports the Reynolds number based on the impeller tip diameter dimension.

Type float (p. 1432)

Read Only No

RelVelRatio

This property specifies the ratio of the relative velocity at the trailing edge to that at the leading edge shroud location.

Type float (p. 1432)

Read Only No

RelVelRatioMod

This property reports the ratio of the rms relative velocity at the trailing edge to the shroud relative velocity at the leading edge

Type float (p. 1432)

Read Only No

RGas

This property reports the specific gas constant of the working fluid.

Type Quantity (p. 1422)

Read Only No

RUser

This property specifies the specific gas constant for the working fluid. Note that this is only available when MaterialPropsOption is set to 'User'.

Type Quantity (p. 1422)

Read Only No

S5

This property reports the specific entropy at the impeller trailing edge.

Type Quantity (p. 1422)

Read Only No

ShrLELoc

This property specifies the main impeller blade leading edge location on the shroud

Type float (p. 1432)

Read Only No

ShroudDiameterOption

This property specifies the method used to calculate the impeller shroud diameter at the leading edge. 'Diameter' allows the diameter to be directly specified. 'Angle' indicates that the diameter will be calculated from the shroud leading edge blade angle. 'Optimum' calculates the diameter such that the relative Mach number at the shroud leading edge is minimised.

Available options:

Diameter
Angle
Optimum

Type ShroudDiameterType (p. 1432)

Read Only No

SLE

This property reports the specific entropy at the impeller leading edge.

Type Quantity (p. 1422)

Read Only No

StackingOption

This property specifies the method used to calculate the leading edge blade angles. Using the radial approach both hub and meanline leading edge blade angles are calculated from the shroud leading edge blade angle. Using either tangential or sine based approaches, the hub leading edge blade angle

is user defined and the meanline leading edge blade angle is interpolated from the hub and shroud blade angles.

Available options:

Radial
Tan
Sin

Type StackingType (p. 1436)

Read Only No

StagPressure

This property specifies the design point stagnation pressure at the inlet to the compressor stage.

Type Quantity (p. 1422)

Read Only No

StagTemp

This property specifies the design point stagnation temperature at the inlet to the compressor stage.

Type Quantity (p. 1422)

Read Only No

SurfaceFinish

This property specifies the surface finish of the impeller. The surface roughness has a secondary effect on the calculated efficiency.

Available options:

Machined
Cast

Type RoughnessType (p. 1427)

Read Only No

SzrFlowCoeff

This property reports the impeller flow coefficient.

Type float (p. 1432)

Read Only No

T05rms

This property reports the total temperature at the impeller trailing edge.

Type Quantity (p. 1422)

Read Only No

ThkHub

This property specifies the hub vane normal thickness.

Type Quantity (p. 1422)

Read Only No

ThkShr

This property specifies the shroud vane normal thickness.

Type Quantity (p. 1422)

Read Only No

ThroatAreaLE

This property reports the throat area at the impeller leading edge.

Type Quantity (p. 1422)

Read Only No

TipCorrectOpt

This property specifies if the tip clearance and shroud correction is to be made to the stage efficiency correlation. Note that this is only available when the EfficiencyOption is set to 'Correlation'.

Type bool (p. 1360)

Read Only No

U5

This property reports the blade speed at the impeller trailing edge (tip speed).

Type Quantity (p. 1422)

Read Only No

UserEfficiencyOption

This property specifies whether the user defined stage efficiency is isentropic or polytropic. Note that this is only available when the EfficiencyOption is set to 'User'.

Available options:

Isentropic
Polytropic

Type EtaUserType (p. 1383)

Read Only No

V5rms

This property reports the absolute velocity at the impeller trailing edge.

Type Quantity (p. 1422)

Read Only No

ViscosityOption

This property specifies the method used to set the viscosity of the working fluid. The viscosity may be calculated using Sutherland's law, specified as a constant dynamic viscosity or as a constant kinematic viscosity. Note that this is only available when MaterialPropsOption is set to 'User'.

Available options:

Sutherland
Dynamic
Kinematic

Type ViscosityType (p. 1450)

Read Only No

ViscosityOptionR145

OBSOLETE PROPERTY RETAINED FOR BACKWARDS COMPATIBILITY IN POST MIGRATION STEP

This property specifies the method used to set the kinematic viscosity of the working fluid. The kinematic viscosity may either be calculated using Sutherland's law for Air, or defined as a constant value by the user. Note that this is only available when MaterialPropsOption is set to 'User'.

Available options:

Sutherland
User

Type NuUserType (p. 1409)

Read Only No

VmLEhub

This property reports the meridional velocity at the impeller leading edge hub location.

Type Quantity (p. 1422)

Read Only No

VmLErms

This property reports the meridional velocity at the impeller leading edge meanline location.

Type Quantity (p. 1422)

Read Only No

VmLEshr

This property reports the meridional velocity at the impeller leading edge shroud location.

Type Quantity (p. 1422)

Read Only No

VmRatioLE

This property reports the ratio of the shroud meridional velocity to the RMS meridional velocity at the leading edge.

Type float (p. 1432)

Read Only No

VrmsLE

This property reports the absolute velocity at the impeller leading edge meanline location.

Type Quantity (p. 1422)

Read Only No

VwLEhub

This property reports the tangential velocity at the impeller leading edge hub location.

Type Quantity (p. 1422)

Read Only No

VwLErms

This property reports the tangential velocity at the impeller leading edge meanline location.

Type Quantity (p. 1422)

Read Only No

VwLEshr

This property reports the tangential velocity at the impeller leading edge shroud location.

Type Quantity (p. 1422)

Read Only No

VwRatioLE

This property reports the ratio of the shroud swirl velocity to the RMS swirl velocity at the leading edge.

Type float (p. 1432)

Read Only No

VwRatioUser

This property specifies the ratio of the inlet tangential velocity at the shroud to that at the meanline. Note that this property is only valid when the PreswirlOption is set to linear.

Type float (p. 1432)

Read Only No

W5rms

This property reports the relative velocity at the impeller trailing edge.

Type Quantity (p. 1422)

Read Only No

Vista CCM

This container holds Analysis data for an instance of Vista CCM.

Methods

Edit

This command class launches the Vista popup GUI.

GetVistaCCMBladeDesignProperties

This query takes a container reference and returns the Data Entity which contains user settings and properties for the VistaCCD Performance Map container.

Return Reference to the requested Data Entity

Type DataReference (p. 1371)

Data Entities

VistaCCMBladeDesign

Setup represents a VistaCCM project definition.

Properties

Alpha3

Inlet angle

Type Quantity (p. 1422)

Read Only No

B5

User input: impeller tip width

Type Quantity (p. 1422)

Read Only No

BetaBlade5

User input: impeller blade exit angle

Type Quantity (p. 1422)

Read Only No

BetaBladeLEhub

User input: hub vane angle at the leading edge

Type Quantity (p. 1422)

Read Only No

BetaBladeLEshr

User input: shroud vane angle at the leading edge

Type Quantity (p. 1422)

Read Only No

ConditionsFromUpstream

Flag to choose whether to update operating conditions from upstream cell or not

Type bool (p. 1360)

Read Only No

D5

User input: impeller exit diameter

Type Quantity (p. 1422)

Read Only No

DisplayText

The general property that defines the user-visible name of an entity. This property is defined for all data entities but is used only in those entities that present a label in the user interface.

Type string (p. 1438)

Read Only No

DLEhub

User input: Hub diameter at the leading edge

Type Quantity (p. 1422)

Read Only No

DLEshr

User input: shroud diameter at the leading edge

Type Quantity (p. 1422)

Read Only No

EffPolyData

Results data: Range of polytropic efficiencies

Type List (p. 1400)<List (p. 1400)<float (p. 1432)>>

Read Only No

EtaPoly

Polytropic efficiency

Type float (p. 1432)

Read Only No

GammaUser

Ratio of specific heats

Type float (p. 1432)

Read Only No

GasModel

Gas model

Type int (p. 1394)

Read Only No

ImperialUnits

User input, Imperial units option

Type bool (p. 1360)

Read Only No

KDiff

User input: Diffuser type (-1.0 = vaneless, 1.0 = vaned)

Type float (p. 1432)

Read Only No

KType

User input: Machine type (-1.0 = process, 1.0 = turbocharger)

Type float (p. 1432)

Read Only No

ListSpeed

Results data: Range of speeds

Type List (p. 1400)<float (p. 1432)>

Read Only No

Loading

Work factor

Type float (p. 1432)

Read Only No

MachU5

Tip Mach number

Type float (p. 1432)

Read Only No

MassFlow

Design point mass flow rate

Type Quantity (p. 1422)

Read Only No

MassFlowData

Results data: Range of mass flow rates

Type List (p. 1400)<List (p. 1400)<float (p. 1432)>>

Read Only No

NMain

User input: number of main vanes

Type int (p. 1394)

Read Only No

NSpeeds

User input, number of speeds

Type int (p. 1394)

Read Only No

NSplit

User input: number of splitter vanes

Type int (p. 1394)

Read Only No

PIF

Power input factor

Type float (p. 1432)

Read Only No

PresRatioData

Results data: Range of pressure ratios

Type List (p. 1400)<List (p. 1400)<float (p. 1432)>>

Read Only No

PressureRatio

Design point pressure ratio

Type float (p. 1432)

Read Only No

RUser

Gas constant

Type Quantity (p. 1422)

Read Only No

SIunits

User input, SI units option

Type bool (p. 1360)

Read Only No

SpeedMax

User input, maximum speed

Type Quantity (p. 1422)

Read Only No

SpeedMaxFixed

User input, maximum speed

Type Quantity (p. 1422)

Read Only No

SpeedMin

User input, minimum speed

Type Quantity (p. 1422)

Read Only No

SpeedMinFixed

User input, minimum speed

Type Quantity (p. 1422)

Read Only No

StagPressure

Inlet stagnation pressure

Type Quantity (p. 1422)

Read Only No

StagPressureFixed

Inlet stagnation pressure

Type Quantity (p. 1422)

Read Only No

StagTemp

Inlet stagnation temperature

Type Quantity (p. 1422)

Read Only No

StagTempFixed

Inlet stagnation temperature

Type Quantity (p. 1422)

Read Only No

SzrFlowCoeff

Flow coefficient

Type float (p. 1432)

Read Only No

ThkHub

User input: Hub vane normal thickness

Type Quantity (p. 1422)

Read Only No

ThkShr

User input: shroud vane normal thickness

Type Quantity (p. 1422)

Read Only No

ThroatArea

User input: throat area

Type Quantity (p. 1422)

Read Only No

TipMachInData

Results data: Range of inlet tip Mach numbers

Type List (p. 1400)<List (p. 1400)<float (p. 1432)>>

Read Only No

TipMachOutData

Results data: Range of outlet tip Mach numbers

Type List (p. 1400)<List (p. 1400)<float (p. 1432)>>

Read Only No

VistaCCMTitle

Editor Title

Type string (p. 1438)

Read Only No

Vista CPD

This container holds Analysis data for an instance of Vista CPD.

Methods

CreateBladeDesign

This command class creates a new BladeGen model. An up-to-date BladeGen cell appears on the project schematic containing the new model.

Return Container representing the BladeGen cell.

Type DataContainerReference (p. 1371)

CreateGeometry

This command class creates a new BladeEditor model. An up-to-date Geometry cell appears on the project schematic containing the new model.

Return Container representing the Geometry cell.

Type DataContainerReference (p. 1371)

CreateThroughflow

This command class creates a new throughflow system. The system, comprising Geometry, Setup, Solution and Results cells, is created on the project schematic and is updated performing the throughflow analysis automatically.

Optional Arguments

UseBladegen Indicates whether to use a Bladegen cell or a BladeEditor(Geometry) cell

Type bool (p. 1360)

Default Value False

CreateVoluteMesh

This command class creates a new pump volute geometry and mesh. An up-to-date Mesh system appears on the project schematic containing the new geometry and mesh cells.

Edit

This command class launches the Vista popup GUI.

GetVistaCPDBladeDesignProperties

This query takes a container reference and returns the Data Entity which contains user settings and properties for the VistaCPD Blade Design container.

Return Reference to the requested Data Entity

Type DataReference (p. 1371)

Import

This command imports Vista data into the Blade Design cell from an existing BladeGen file. If no appropriate Vista data is found in the specified BladeGen file, an error message is generated.

```
template1 = GetTemplate(TemplateName="VistaCPD")
system1 = template1.CreateSystem()
bladeDesign1 = system1.GetContainer(ComponentName="Blade Design")
bladeDesign1.Import(FilePath="myfilepath/pump.bgd")
```

Required Arguments

FileName Name, and path, of the BladeGen file to be imported.

Type string (p. 1438)

Data Entities

VistaCPDBladeDesign

This data entity provides access to the properties which define the VistaCPD project. This includes both the input and the results properties.

Properties

Alpha2

This property reports the absolute flow angle at the impeller trailing edge.

Type Quantity (p. 1422)

Read Only Yes

AlphaIn

This property specifies the absolute flow angle, measured with respect to the tangential direction, at the leading edge of the pump impeller.

Type Quantity (p. 1422)

Read Only No

AreaDiff

This property reports the volute diffuser exit area.

Type Quantity (p. 1422)

Read Only Yes

AspectRatio

This property specifies the aspect ratio (height/width) of the rectangular volute cross section. This is valid when VoluteStyleOpt is 'Rectangular'.

Type float (p. 1432)

Read Only No

B2

This property reports the hub to shroud distance at the impeller trailing edge (tip width).

Type Quantity (p. 1422)

Read Only Yes

B3

This property reports the width of the volute inlet.

Type Quantity (p. 1422)

Read Only Yes

Beta1

This property reports the relative flow angle at the impeller leading edge meanline section.

Type Quantity (p. 1422)

Read Only Yes

Beta1Blade

This property reports the impeller leading edge blade angle at the meanline section.

Type Quantity (p. 1422)

Read Only Yes

Beta1BladeHub

This property reports the impeller leading edge blade angle at the hub section.

Type Quantity (p. 1422)

Read Only Yes

Beta1BladeHubUser

This property specifies the impeller leading edge blade angle at the hub, measured with respect to the tangential direction. This is valid when the HubBeta1Opt is set to 'User'

Type Quantity (p. 1422)

Read Only No

Beta1BladeShr

This property reports the impeller leading edge blade angle at the shroud section.

Type Quantity (p. 1422)

Read Only Yes

Beta1BladeShrUser

This property specifies the impeller leading edge blade angle at the shroud, measured with respect to the tangential direction. This is valid when the ShrBeta1Opt is set to 'User'

Type Quantity (p. 1422)

Read Only No

Beta1BladeUser

This property specifies the impeller leading edge blade angle at the meanline, measured with respect to the tangential direction. This is valid when the HubBeta1Opt is set to 'User'

Type Quantity (p. 1422)

Read Only No

Beta1Hub

This property reports the relative flow angle at the impeller leading edge hub section.

Type Quantity (p. 1422)

Read Only Yes

Beta1Shr

This property reports the relative flow angle at the impeller leading edge shroud section.

Type Quantity (p. 1422)

Read Only Yes

Beta2

This property reports the relative flow angle at the impeller trailing edge.

Type Quantity (p. 1422)

Read Only Yes

Beta2Blade

This property specifies the blade angle at the impeller trailing edge, measured with respect to the tangential direction.

Type Quantity (p. 1422)

Read Only No

BMExportOption

This property specifies whether the impeller is to be exported as an isolated impeller or coupled to a volute. The isolated impeller option provides for an extended exit diffuser to assist the analysis process. If the impeller is coupled to the volute, the exit diffuser is short to match with the volute inlet.

Available options:

Isolated
Coupled

Type ImpellerExportType (p. 1390)

Read Only No

BMunitsOption

This property specifies the units used when creating a new BladeGen/BladeEditor model. Note that this is independent of the units used in the VistaCPD popup GUI.

Available options:

mm
inches

Type BMunitsType (p. 1359)

Read Only No

C2

This property reports the absolute flow velocity at the impeller trailing edge.

Type Quantity (p. 1422)

Read Only Yes

Cm1

This property reports the meridional flow velocity at the impeller leading edge meanline section.

Type Quantity (p. 1422)

Read Only Yes

Cm1Hub

This property reports the meridional flow velocity at the impeller leading edge hub section.

Type Quantity (p. 1422)

Read Only Yes

Cm1Shr

This property reports the meridional flow velocity at the impeller leading edge shroud section.

Type Quantity (p. 1422)

Read Only Yes

Cu1

This property reports the tangential flow velocity at the impeller leading edge meanline section.

Type Quantity (p. 1422)

Read Only Yes

Cu1Hub

This property reports the tangential flow velocity at the impeller leading edge hub section.

Type Quantity (p. 1422)

Read Only Yes

Cu1Shr

This property reports the tangential flow velocity at the impeller leading edge shroud section.

Type Quantity (p. 1422)

Read Only Yes

Cu2

This property reports the tangential flow velocity at the impeller trailing edge.

Type Quantity (p. 1422)

Read Only Yes

Cus

This property reports the slip velocity at the impeller trailing edge. This is defined as the difference between the theoretical 'no-slip' tangential flow velocity and the true tangential flow velocity.

Type Quantity (p. 1422)

Read Only Yes

D1

This property reports the diameter of the impeller leading edge at the meanline section.

Type Quantity (p. 1422)

Read Only Yes

D1Hub

This property reports the diameter of the impeller leading edge at the hub section.

Type Quantity (p. 1422)

Read Only Yes

D1Shr

This property reports the diameter of the impeller leading edge at the shroud section.

Type Quantity (p. 1422)

Read Only Yes

D2

This property reports the diameter at the impeller trailing edge meanline section (tip diameter).

Type Quantity (p. 1422)

Read Only Yes

D2Opt

This property specifies the method used to set the impeller tip diameter. It may be calculated automatically, from a specified head coefficient or the value may be user defined.

Available options:

Automatic
HeadCoeff
User

Type TipDiamType (p. 1442)

Read Only No

D2User

This property specifies the impeller tip diameter. This is valid when the D2Opt is set to 'User'.

Type Quantity (p. 1422)

Read Only No

DEye

This property reports the impeller shroud diameter at the eye of the impeller.

Type Quantity (p. 1422)

Read Only Yes

DEyeHub

This property reports the impeller hub diameter at the eye of the impeller.

Type Quantity (p. 1422)

Read Only Yes

DiamDiff

This property reports the hydraulic diameter of the volute diffuser exit.

Type Quantity (p. 1422)

Read Only Yes

DiamDiffUser

This property specifies the value of the volute diffuser exit diameter. This is valid when UserDiamDiff is set to true.

Type Quantity (p. 1422)

Read Only No

DiffRatio

This property reports the diffusion ratio of the impeller.

Type float (p. 1432)

Read Only Yes

DisplayText

The general property that defines the user-visible name of an entity. This property is defined for all data entities but is used only in those entities that present a label in the user interface.

Type string (p. 1438)

Read Only No

DShaft

This property reports the impeller shaft diameter.

Type Quantity (p. 1422)

Read Only Yes

EfficiencyOption

This property specifies how the impeller efficiencies are calculated.

With this set to 'Automatic' the efficiencies are calculated using empirical correlations. All other options require the specification of three of the efficiencies.The remaining value is calculated from those specified. For example, specifying 'Hydraulic' indicates that the hydraulic efficiency will be calculated and the volumetric, mechanical and overall pump efficiencies must be specified.

Available options:

Automatic
Hydraulic
Volumetric
Mechanical
Pump

Type EffType (p. 1379)

Read Only No

FlowCoeff

This property reports the flow coefficient of the impeller.

Type float (p. 1432)

Read Only Yes

Head

This property specifies the design point head rise for the impeller.

Type Quantity (p. 1422)

Read Only No

HeadCoeff

This property reports the head coefficient of the impeller.

Type float (p. 1432)

Read Only Yes

HeadCoeffUser

This property specifies the head coefficient, used to calculate the impeller tip diameter. This is valid when the D2Opt is set to 'HeadCoeff'.

Type float (p. 1432)

Read Only No

HeightDiff

This property reports the volute diffuser exit height. In the case of a circular outlet, (elliptic volute cross section), this is the same as the hydraulic diameter.

Type Quantity (p. 1422)

Read Only Yes

HubBeta1Opt

This property specifies the method used to set the impeller leading edge blade angles at the hub and meanline sections. These blade angles may be calculated relative to the leading edge blade angle at the shroud using either cosine or cotangent relationships, or they may be defined directly by the user.

Available options:

Cos
Cot
User

Type HubLEBetaType (p. 1390)

Read Only No

HubInletDraft

This property specifies the impeller hub inlet draft angle. This is defined as the angle between the hub and the horizontal line at the hub inlet.

Type Quantity (p. 1422)

Read Only No

HydEff

This property reports the hydraulic efficiency of the impeller.

Type float (p. 1432)

Read Only Yes

HydEffUser

This property specifies the impeller hydraulic efficiency. This is valid when the EfficiencyOption is set to 'Volumetric', 'Mechanical' or 'Pump'.

Type float (p. 1432)

Read Only No

Inc

This property reports the incidence at the impeller leading edge meanline section.

Type Quantity (p. 1422)

Read Only Yes

IncHub

This property reports the incidence at the impeller leading edge hub section.

Type Quantity (p. 1422)

Read Only Yes

IncShr

This property reports the incidence at the impeller leading edge shroud section.

Type Quantity (p. 1422)

Read Only Yes

IncShrUser

This property specifies the angle of incidence for the impeller at the shroud. This is valid when the ShrBeta1Opt is set to 'Incidence'.

Type Quantity (p. 1422)

Read Only No

Ks

This property reports the stability factor of the impeller.

Type float (p. 1432)

Read Only Yes

LengthDiff

This property reports the volute diffuser axial length.

Type Quantity (p. 1422)

Read Only Yes

LengthDiffUser

This property specifies the value of the volute diffuser axial length. This is valid when UserLengthDiff is set to true.

Type Quantity (p. 1422)

Read Only No

MechEff

This property reports the mechanical efficiency of the impeller.

Type float (p. 1432)

Read Only Yes

MechEffUser

This property specifies the impeller mechanical efficiency. This is valid when the EfficiencyOption is set to 'Hydraulic', 'Volumetric' or 'Pump'.

Type float (p. 1432)

Read Only No

MerVelRatio

This property specifies the gradient of the velocity profile from hub to shroud at the impeller leading edge. The gradient is set using the ratio of the meridional velocity at the shroud leading edge radius to that at the average leading edge radius.

Type float (p. 1432)

Read Only No

MinDiamFactor

This property specifies the shaft minimum diameter factor. This is a 'factor of safety' applied to the shaft minimum diameter as calculated from the maximum allowable shear stress of the shaft.

Type float (p. 1432)

Read Only No

NPSHr

This property reports the net positive suction head required (NPSHr) of the impeller.

Type Quantity (p. 1422)

Read Only Yes

Nq

This property reports the specific speed of the impeller using the European units system.

Type float (p. 1432)

Read Only Yes

Ns

This property reports the specific speed of the impeller using the US units system.

Type float (p. 1432)

Read Only Yes

Nss

This property reports the non-dimensional suction specific speed of the impeller.

Type float (p. 1432)

Read Only Yes

NumVanes

This property specifies the number of impeller vanes.

Type int (p. 1394)

Read Only No

OmegaS

This property reports the non-dimensional specific speed of the impeller.

Type float (p. 1432)

Read Only Yes

PowShaft

This property reports the shaft power of the impeller.

Type Quantity (p. 1422)

Read Only Yes

PumpEff

This property reports the overall efficiency of the impeller. This is the product of the hydraulic, volumetric and mechanical efficiencies.

Type float (p. 1432)

Read Only Yes

PumpEffUser

This property specifies the impeller overall efficiency. This is valid when the EfficiencyOption is set to 'Hydraulic', 'Volumetric' or 'Mechanical'.

Type float (p. 1432)

Read Only No

R3

This property reports the volute base-circle radius. This is defined as the distance from the centreline to the volute tongue.

Type Quantity (p. 1422)

Read Only Yes

Rake

This property reports the lean angle at the impeller trailing edge (rake angle). Note that although this is also specified as an input property, the process to achieve the rake angle is iterative and may not always be achievable. In this situation there will be a difference between this value and that specified by the input property.

Type Quantity (p. 1422)

Read Only Yes

RakeUser

This property specifies the blade lean angle at the impeller trailing edge (rake angle)

Type Quantity (p. 1422)

Read Only No

Rho

This property specifies the density of the working fluid.

Type Quantity (p. 1422)

Read Only No

RMajor

This property reports a list of the major radii of the elliptic volute cross sections. This is valid when VoluteStyleOpt is 'Elliptic'.

Type List (p. 1400)<Quantity (p. 1422)>

Read Only Yes

RMinor

This property reports a list of the minor radii of the elliptic volute cross sections. This is valid when VoluteStyleOpt is 'Elliptic'.

Type List (p. 1400)<Quantity (p. 1422)>

Read Only Yes

ShaftDiamRatio

This property specifies the shaft diameter ratio. This is defined as the ratio of the hub diameter to the shaft diameter. It is used to determine the hub diameter from the shaft diameter and size of the impeller fittings used to fix the impeller to the shaft.

Type float (p. 1432)

Read Only No

ShrBeta1Opt

This property specifies how the impeller leadng edge blade angle at the shroud is set. With this option set to 'Incidence' the blade angle is calculated from the specified incidence, otherwise the blade angle is specified directly.

Available options:

Incidence

User

Type ShrLEBetaType (p. 1431)

Read Only No

SlipRatio

This property reports the ratio of the slip velocity to the blade speed at the impeller trailing edge.

Type float (p. 1432)

Read Only Yes

Speed

This property specifies the rotational speed of the pump impeller.

Type Quantity (p. 1422)

Read Only No

Theta2

This property reports the angle of inclination to the horizontal of the impeller trailing edge, when viewed in the meridional plane.

Type Quantity (p. 1422)

Read Only Yes

ThetaCR

This property specifies the volute casing rotation angle. This is defined as the angle between the vertical line and the tongue location when viewing the central section through the volute.

Type Quantity (p. 1422)

Read Only No

ThetaDiff

This property reports the volute diffuser cone angle.

This is defined as the angle between the sloping sides of a circular based conic frustum, with the same inlet area, exit area and axial length as the volute diffuser.

Type Quantity (p. 1422)

Read Only Yes

Thk

This property reports the impeller vane thickness.

Type Quantity (p. 1422)

Read Only Yes

ThkRatio

This property specifies the ratio of the impeller vane thickness to the tip diameter. It is used in order to specify the impeller vane thickness in a non-dimensional manner.

Type float (p. 1432)

Read Only No

TongueClear

This property reports the volute tongue clearance. This is defined as the volute base-circle radius minus the impeller tip radius.

Type Quantity (p. 1422)

Read Only Yes

TongueThk

This property reports the thickness of the volute tongue ie. the tongue diameter at the cutwater.

Type Quantity (p. 1422)

Read Only Yes

U1

This property reports the blade speed at the impeller leading edge meanline section.

Type Quantity (p. 1422)

Read Only Yes

U1Hub

This property reports the blade speed at the impeller leading edge hub section.

Type Quantity (p. 1422)

Read Only Yes

U1Shr

This property reports the blade speed at the impeller leading edge shroud section.

Type Quantity (p. 1422)

Read Only Yes

U2

This property reports the blade speed at the impeller trailing edge (tip speed).

Type Quantity (p. 1422)

Read Only Yes

UserDiamDiff

This property specifies that the volute diffuser exit diameter is to be defined by the user, rather than calculated automatically.

Type bool (p. 1360)

Read Only No

UserLengthDiff

This property specifies that the volute diffuser axial length is to be defined by the user, rather than calculated automatically.

Type bool (p. 1360)

Read Only No

VolA

This property reports a list of the volute cross sectional areas from cutwater to throat.

Type List (p. 1400)<Quantity (p. 1422)>

Read Only Yes

VolEff

This property reports the volumetric efficiency of the impeller.

Type float (p. 1432)

Read Only Yes

VolEffUser

This property specifies the impeller volumetric efficiency. This is valid when the EfficiencyOption is set to 'Hydraulic', 'Mechanical' or 'Pump'.

Type float (p. 1432)

Read Only No

VolFlow

This property specifies the design point volume flow rate delivered by the pump.

Type Quantity (p. 1422)

Read Only No

VolHeight

This property reports a list of the height of the rectangular volute cross sections. This is valid when VoluteStyleOpt is 'Rectangular'.

Type List (p. 1400)<Quantity (p. 1422)>

Read Only Yes

VolR

This property reports a list of the radii of the centroids of the volute cross sections.

Type List (p. 1400)<Quantity (p. 1422)>

Read Only Yes

VolRouter

This property reports a list of the outer radii of the volute cross sections.

Type List (p. 1400)<Quantity (p. 1422)>

Read Only Yes

VoluteStyleOpt

This property specifies the volute cross section shape.

Available options:

Elliptic
Rectangular

Type VoluteType (p. 1450)

Read Only No

VolWidth

This property reports a list of the width of the rectangular volute cross sections. This is valid when VoluteStyleOpt is 'Rectangular'.

Type List (p. 1400)<Quantity (p. 1422)>

Read Only Yes

W1

This property reports the relative flow velocity at the impeller leading edge meanline section.

Type Quantity (p. 1422)

Read Only Yes

W1Hub

This property reports the relative flow velocity at the impeller leading edge hub section.

Type Quantity (p. 1422)

Read Only Yes

W1Shr

This property reports the relative flow velocity at the impeller leading edge shroud section.

Type Quantity (p. 1422)

Read Only Yes

W2

This property reports the relative flow velocity at the impeller trailing edge.

Type Quantity (p. 1422)

Read Only Yes

Vista RTD

This container holds Analysis data for an instance of Vista RTD.

Methods

CreateBladeDesign

This command class creates a new BladeGen model. An up-to-date BladeGen cell appears on the project schematic containing the new model.

Return Container representing the BladeGen cell.

Type DataContainerReference (p. 1371)

CreateGeometry

This command class creates a new BladeEditor model. An up-to-date Geometry cell appears on the project schematic containing the new model.

Return Container representing the Geometry cell.

Type DataContainerReference (p. 1371)

CreateThroughflow

This command class creates a new throughflow system. The system, comprising Geometry, Setup, Solution and Results cells, is created on the project schematic and is updated performing the throughflow analysis automatically.

Optional Arguments

UseBladegen Indicates whether to use a Bladegen cell or a BladeEditor(Geometry) cell

Type bool (p. 1360)

Default Value False

Edit

This command class launches the Vista popup GUI.

GetVistaRTDBladeDesignProperties

This query takes a container reference and returns the Data Entity which contains user settings and properties for the VistaRTD Blade Design container.

Return Reference to the requested Data Entity

Type DataReference (p. 1371)

ImportBladeGen

This command imports Vista data into the Blade Design cell from an existing BladeGen file. If no appropriate Vista data is found in the specified BladeGen file, an error message is generated.

```
template1 = GetTemplate(TemplateName="VistaCPD")
system1 = template1.CreateSystem()
bladeDesign1 = system1.GetContainer(ComponentName="Blade Design")
bladeDesign1.Import(FilePath="myfilepath/pump.bgd")
```

Required Arguments

FileName Name, and path, of the BladeGen file to be imported.

Type string (p. 1438)

Data Entities

VistaRTDBladeDesign

This data entity provides access to the properties which define the VistaRTD project. This includes both the input and the results properties.

Properties

AFR

This property specifies the air/fuel ratio of the working fluid. This is only available when using the AFR option for GasProps.

Type float (p. 1432)

Read Only No

Alpha2

This property reports the absolute flow angle at the impeller inlet.

Type Quantity (p. 1422)

Read Only Yes

Alpha2User

This property specifies the absolute flow angle at the impeller leading edge. Note that this is not available when the InletOption is set to 'Calculated'.

Type Quantity (p. 1422)

Read Only No

Alpha3

This property reports the absolute flow angle at the impeller exit station.

Type Quantity (p. 1422)

Read Only Yes

Alpha3User

This property specifies the absolute flow angle at the impeller trailing edge.

Type Quantity (p. 1422)

Read Only No

Beta2

This property reports the relative flow angle at the impeller inlet.

Type Quantity (p. 1422)

Read Only Yes

Beta2User

This property specifies the relative flow angle at the impeller leading edge. Note that this is not available when the InletOption is set to 'Calculated'.

Type Quantity (p. 1422)

Read Only No

Beta3

This property reports the meanline section relative flow angle at the impeller exit station.

Type Quantity (p. 1422)

Read Only Yes

Beta3hub

This property reports the hub section relative flow angle at the impeller exit station.

Type Quantity (p. 1422)

Read Only Yes

Beta3shroud

This property reports the shroud section relative flow angle at the impeller exit station.

Type Quantity (p. 1422)

Read Only Yes

Beta3User

This property specifies the relative flow angle at the impeller trailing edge.

Type Quantity (p. 1422)

Read Only No

BMunitsOption

This property specifies the units used when creating a new BladeGen/BladeEditor model. Note that this is independent of the units used in the VistaRTD popup GUI.

Available options:

mm
cm
inches
ft
m

Type BMunitsType (p. 1359)

Read Only No

ClearanceOption

This property specifies impeller tip clearance is specified. 'Ratio' indicates that the tip clearance is specified as a fraction of the tip width 'User' specifies that the clearance will be defined directly by the user.

Available options:

Ratio
User

Type ClearanceType (p. 1364)

Read Only No

ClearLoss

This property reports the proportion of energy loss attributed to the clearances between rotating and stationary components. Note this is only available when using the 'Correlation' for the efficiency calculation method, EtaOpt.

Type float (p. 1432)

Read Only Yes

ClearRatioUser

This property specifies the ratio of the impeller tip clearance to the tip width. Note that this is only available when ClearanceOption is set to 'Ratio'.

Type float (p. 1432)

Read Only No

ClearUser

This property specifies the value of the impeller tip clearance. Note that this is only available when ClearanceOption is set to 'User'.

Type Quantity (p. 1422)

Read Only No

CorrelationOption

This property specifies the correlation used to calculate the stage efficiency. Note that this is only available when the EtaOption is set to 'Correlation'.

Available options:

Suhrmann
Baines

Type EtaCorrelType (p. 1382)

Read Only No

CpMean

This property reports the average specific heat capacity at constant pressure for the turbine stage.

Type Quantity (p. 1422)

Read Only Yes

CpUser

This property specifies the specific heat capacity at constant pressure, Cp, of the working fluid. This is only available when using the Fixed option for GasProps.

Type Quantity (p. 1422)

Read Only No

D2

This property reports the impeller inlet diameter.

Type Quantity (p. 1422)

Read Only Yes

D3hub

This property reports the impeller exit hub diameter.

Type Quantity (p. 1422)

Read Only Yes

D3shroud

This property reports the impeller exit shroud diameter.

Type Quantity (p. 1422)

Read Only Yes

DiameterRatio

This property reports the ratio of the impeller inlet diameter to the meanline exit diameter.

Type float (p. 1432)

Read Only Yes

DisplayText

The general property that defines the user-visible name of an entity. This property is defined for all data entities but is used only in those entities that present a label in the user interface.

Type string (p. 1438)

Read Only No

EtaImpTS

This property reports the total-to-static impeller efficiency.

Type float (p. 1432)

Read Only Yes

EtaImpTT

This property reports the total-to-total impeller efficiency.

Type float (p. 1432)

Read Only Yes

EtaNozzle

This property specifies the value of the total-to-static nozzle efficiency. The term 'nozzle' here refers to the geometry upstream of the impeller used to control the inlet flow angle. This may be either bladed or unbladed, eg. a volute. A specified nozzle efficiency of 1.0 implies no pressure loss across the nozzle and consequently the nozzle is neglected from the calculation.

Type float (p. 1432)

Read Only No

EtaOption

This property specifies whether to use a correlation to automatically calculate the turbine stage efficiency, or to use a user-defined efficiency.

Available options:

User
Correlation

```
template1 = GetTemplate(TemplateName="VistaRTD")
system1 = template1.CreateSystem()
bladeDesign1 = system1.GetContainer(ComponentName="Blade Design")
vistaRTDProperties1 = bladeDesign1.GetVistaRTDBladeDesignProperties()
vistaRTDProperties1.EtaOption = "Correlation"
```

Type EtaType (p. 1382)

Read Only No

EtaStageTS

This property reports the total-to-static stage efficiency.

Type float (p. 1432)

Read Only Yes

EtaStageTT

This property reports the total-to-total stage efficiency.

Type float (p. 1432)

Read Only Yes

EtaUser

This property specifies the value of the total-to-total turbine stage efficiency. Note that this entry is only available when the EtaOption is set to 'User'.

Type float (p. 1432)

Read Only No

ExitAngle

This property specifies the flow angle at the impeller exit. This will either be an absolute or relative value depending on the 'ExitOption'.

Type Quantity (p. 1422)

Read Only No

ExitLoss

This property reports the proportion of energy loss attributed to the exhaust. Note this is only available when using the 'Correlation' for the efficiency calculation method, EtaOpt.

Type float (p. 1432)

Read Only Yes

ExitOption

This property specifies the ExitAngle type.

Available options:

Absolute
Relative

Type ExitAngleType (p. 1384)

Read Only No

ExpRatio

This property specifies the design point total-to-total expansion ratio, P01/P03, for the turbine stage.

Type float (p. 1432)

Read Only No

ExpRatioTs

This property reports the total to static expansion ratio for the turbine stage (P01/P3).

Type float (p. 1432)

Read Only Yes

FricLoss

This property reports the proportion of energy loss attributed to friction. Note this is only available when using the 'Correlation' for the efficiency calculation method, EtaOpt.

Type float (p. 1432)

Read Only Yes

GammaMean

This property reports the average ratio of specific heats (gamma) for the turbine stage.

Type float (p. 1432)

Read Only Yes

GasProps

This property specifies the method used to calculate the properties of the working fluid.

Available options:

Air
AFR
Fixed

```
template1 = GetTemplate(TemplateName="VistaRTD")
system1 = template1.CreateSystem()
bladeDesign1 = system1.GetContainer(ComponentName="Blade Design")
vistaRTDProperties1 = bladeDesign1.GetVistaRTDBladeDesignProperties()
vistaRTDProperties1.GasProps = "AFR"
```

Type GasPropType (p. 1388)

Read Only No

HubRatio

This property specifies the ratio of the impeller exit radius at the hub to the impeller inlet radius (tip radius). This enables the hub exit radius to be controlled in a non-dimensional way.

Type float (p. 1432)

Read Only No

ImpellerLength

This property reports the ratio of the impeller axial length to the impeller tip diameter.

Type float (p. 1432)

Read Only Yes

ImpellerLengthOption

This property specifies whether the impeller length ratio is calculated automatically, or specified by the user.

Available options:

Automatic
User

Type ImpellerLengthType (p. 1391)

Read Only No

ImpellerLengthUser

This property specifies the ratio of the impeller axial length to the impeller tip diameter. This enables the impeller axial length to be controlled in a non-dimensional way.

Type float (p. 1432)

Read Only No

ImpellerLengthUserOpt

This property specifies whether the impeller axial length ratio will be defined by the user.

Type bool (p. 1360)

Read Only No

ImpellerNumber

This property specifies the number of impeller vanes.

Type int (p. 1394)

Read Only No

ImpellerThickness

This property specifies the average vane thickesses at the impeller exit.

Type Quantity (p. 1422)

Read Only No

IncLoss

This property reports the proportion of energy loss attributed to the incidence at the leading edge. Note this is only available when using the 'Correlation' for the efficiency calculation method, EtaOpt.

Type float (p. 1432)

Read Only Yes

InletAngle

OBSOLETE PROPERTY RETAINED FOR BACKWARDS COMPATIBILITY IN THE POST MIGRATION STEP

This property specifies the flow angle at the impeller inlet. This will either be specified as an absolute or relative value depending on the 'InletOption'.

Type Quantity (p. 1422)

Read Only No

InletOption

This property specifies the InletAngle type.

Available options:

Absolute
Relative
Calculated. ****** OBSOLETE OPTION PLEASE USE ABSOLUTE OR RELATIVE ********

Type InletAngleType (p. 1394)

Read Only No

Loading

This property reports the basic loading for the turbine stage (delH/U^2).

Type float (p. 1432)

Read Only Yes

LoadLoss

This property reports the proportion of energy loss attributed to loading. Note this is only available when using the 'Correlation' for the efficiency calculation method, EtaOpt.

Type float (p. 1432)

Read Only Yes

Mach2

This property reports the absolute Mach number at the impeller inlet.

Type float (p. 1432)

Read Only Yes

Mach3rms

This property reports the meanline section absolute Mach number at the impeller exit station.

Type float (p. 1432)

Read Only Yes

MachRel2

This property reports the relative Mach number at the impeller inlet.

Type float (p. 1432)

Read Only Yes

MachRel3shroud

This property reports the shroud section relative Mach number at the impeller exit station.

Type float (p. 1432)

Read Only Yes

MassFlow

This property specifies the design point mass flow rate passing through the turbine stage.

Type Quantity (p. 1422)

Read Only No

MrootToverP

This property reports the characteristic flow function for the design point (M x SQRT(T) / P).

Type float (p. 1432)

Read Only Yes

NozChkRatio

This property reports the choke ratio for the nozzle.

Type float (p. 1432)

Read Only Yes

NozExitDiameter

This property reports the nozzle exit diameter.

Type Quantity (p. 1422)

Read Only Yes

NozThtArea

This property reports the throat area of the nozzle.

Type Quantity (p. 1422)

Read Only Yes

NozVlessRatio

This property reports the ratio of the 'vaneless' nozzle area to the impeller inlet radius. Note that the nozzle area used here neglects the vane thickness, hence the 'vaneless' prefix.

Type Quantity (p. 1422)

Read Only Yes

Ns

This property reports the specific speed of the impeller.

Type float (p. 1432)

Read Only Yes

Omega

This property specifies the design point rotational speed of the impeller.

Type Quantity (p. 1422)

Read Only No

OvrChkRatio

This property reports the overall choke ratio (mass flow/choke flow) for the turbine stage.

Type float (p. 1432)

Read Only Yes

P02

This property reports the total pressure at impeller inlet.

Type Quantity (p. 1422)

Read Only Yes

P03

This property reports the total pressure at impeller exit.

Type Quantity (p. 1422)

Read Only Yes

P2

This property reports the static pressure at impeller inlet.

Type Quantity (p. 1422)

Read Only Yes

P3

This property reports the static pressure at impeller exit.

Type Quantity (p. 1422)

Read Only Yes

Power

This property reports the aerodynamic power generated by the turbine, neglecting mechanical losses.

Type Quantity (p. 1422)

Read Only Yes

Reaction

This property reports the degree of reaction for the turbine stage, (T2 - T3)/(T01 - T03).

Type float (p. 1432)

Read Only Yes

RelVelRatShroud

This property reports the ratio of the impeller exit relative velocity at the shroud (W3shr) to the impeller inlet relative velocity (W2).

Type float (p. 1432)

Read Only Yes

RUser

This property specifies the specific gas constant, R, of the working fluid. This is only available when using the Fixed option for GasProps.

Type Quantity (p. 1422)

Read Only No

ShroudRatio

This property specifies the ratio of the impeller exit radius at the shroud to the impeller inlet radius (tip radius). This enables the shroud exit radius to be controlled in a non-dimensional way.

Type float (p. 1432)

Read Only No

SpanwiseDistributionOption

This property specifies the spanwise distribution used when exporting the impeller blade.

Available options:

General
Radial

Type SpanwiseDistributionType (p. 1435)

Read Only No

SpeedRatio

This property specifies the blade speed ratio, U2/C0, of the impeller, where U2 is the impeller tip speed and C0 is the spouting velocity. In this definition the spouting velocity is calculated from the total-to-total isentropic temperature drop.

Type float (p. 1432)

Read Only No

StagPressure

This property specifies the design point stagnation pressure, P01, at the inlet to the turbine stage. Note that this is defined upstream of the nozzle guide vane ahead of the impeller.

Type Quantity (p. 1422)

Read Only No

StagTemp

This property specifies the design point stagnation temperature, T01, at the inlet to the turbine stage. Note that this is defined upstream of the nozzle guide vane ahead of the impeller.

Type Quantity (p. 1422)

Read Only No

SurfaceFinish

This property specifies the surface finish of the impeller. The surface roughness has a secondary effect on the calculated efficiency.

Available options:

Machined
Cast

Type RoughnessType (p. 1427)

Read Only No

T02

This property reports the total temperature at impeller inlet.

Type Quantity (p. 1422)

Read Only Yes

T03

This property reports the total temperature at impeller exit.

Type Quantity (p. 1422)

Read Only Yes

T2

This property reports the static temperature at impeller inlet.

Type Quantity (p. 1422)

Read Only Yes

T3

This property reports the static temperature at impeller exit.

Type Quantity (p. 1422)

Read Only Yes

TipWidth

This property reports the impeller inlet tip width.

Type Quantity (p. 1422)

Read Only Yes

TotalLoss

This property reports the sum of the friction, loading, clearance and exit losses. It is equal to one minus the total-to-total stage efficiency. Note this is only available when using the 'Correlation' for the efficiency calculation method, EtaOpt.

Total losses

Type float (p. 1432)

Read Only Yes

U2

This property reports the blade speed at the impeller inlet (tip speed)

Type Quantity (p. 1422)

Read Only Yes

U3shroud

This property reports the shroud section blade speed (tip speed) at the impeller exit station.

Type Quantity (p. 1422)

Read Only Yes

V2

This property reports the absolute velocity at the impeller inlet.

Type Quantity (p. 1422)

Read Only Yes

V3rms

This property reports the meanline section absolute velocity at the impeller exit station.

Type Quantity (p. 1422)

Read Only Yes

Vax3rms

This property reports the meanline section axial velocity at the impeller exit station.

Type Quantity (p. 1422)

Read Only Yes

VelRatio1

This property reports the flow coefficient at impeller exit. This is the ratio of the average axial velocity at the impeller exit (Vax3) to the impeller tip speed (U2).

Type float (p. 1432)

Read Only Yes

VelRatio2

This property reports the blade speed ratio, U2/C0, of the impeller, where U2 is the impeller tip speed and C0 is the spouting velocity. In this definition the spouting velocity is calculated from the total-to-static isentropic temperature drop.

Type float (p. 1432)

Read Only Yes

Vr2

This property reports the radial velocity at the impeller inlet.

Type Quantity (p. 1422)

Read Only Yes

Vw2

This property reports the swirl (tangential) velocity at the impeller inlet

Type Quantity (p. 1422)

Read Only Yes

Vw3rms

This property reports the meanline section swirl velocity at the impeller exit station.

Type Quantity (p. 1422)

Read Only Yes

W2

This property reports the relative velocity at the impeller inlet.

Type Quantity (p. 1422)

Read Only Yes

W3shroud

This property reports the shroud section relative velocity at the impeller exit station.

Type Quantity (p. 1422)

Read Only Yes

ZeroBetaInlet

Specifies whether or not the relative flow angle at the impeller inlet is zero. Note that a zero inlet angle is a 'special case' where the exit angles are calculated rather than specified.

Type bool (p. 1360)

Read Only No

Vista TF Setup

This container holds Setup data for an instance of Vista TF.

Methods

GetSetupEntity

This query is used to retrieve the Vista TF Setup data entity for the specified container.

Return A reference to the requested data entity.

Type DataReference (p. 1371)

ImportAeroTemplate

Imports throughflow aerodynamic data into the Vista TF Setup from an existing throughflow aero template (.aert) file.

Required Arguments

FilePath Path of the file to be imported.

Type string (p. 1438)

ImportControlTemplate

Imports throughflow control data into the Vista TF Setup from an existing throughflow control template (.cont) file.

Required Arguments

FilePath Path of the file to be imported.

Type string (p. 1438)

ImportCorrelationsTemplate

Imports throughflow correlations data into the Vista TF Setup from an existing throughflow correlations template (.cort) file.

Required Arguments

FilePath Path of the file to be imported.

Type string (p. 1438)

ImportCustomRealGas

Imports custom real gas data into the Vista TF Setup from a real gas file (.rgp)

Required Arguments

FilePath Path of the file to be imported.

Type string (p. 1438)

ImportGeometry

Imports throughflow geometry data into the Vista TF Setup from an existing throughflow geometry (.geo) file.

Required Arguments

FilePath Path of the file to be imported.

Type string (p. 1438)

Data Entities

VistaTFSetup

This data entity represents a Vista TF project definition on a geometry.

Properties

Acentric

User input, acentric factor

Type float (p. 1432)

Read Only No

ConditionsFromUpstream

Flag to choose whether to update operating conditions from upstream cell or not

Type bool (p. 1360)

Read Only No

CpCoeff

User input: Cp expression coefficients

Type List (p. 1400)<double (p. 1378)>

Read Only No

CpGas

Specifies the gas specific heat at constant pressure when FluidOption = IdealGas.

Type Quantity (p. 1422)

Read Only No

CpMax

User input: Cp expression, max temperature

Type float (p. 1432)

Read Only No

CpMin

User input: Cp expression, min temperature

Type float (p. 1432)

Read Only No

CriticalPressure

User input, critical pressure

Type Quantity (p. 1422)

Read Only No

CriticalTemp

User input, critical temperature

Type Quantity (p. 1422)

Read Only No

CriticalVol

User input, critical volume

Type Quantity (p. 1422)

Read Only No

CustomRealGasFilename

The filename for the specified custom real gas file. Only used where the custom real gas option is specified.

Type string (p. 1438)

Read Only No

CwFluid

Specifies the liquid specific heat when FluidOption = Liquid.

Type Quantity (p. 1422)

Read Only No

DisplayText

The general property that defines the user-visible name of an entity. This property is defined for all data entities but is used only in those entities that present a label in the user interface.

Type string (p. 1438)

Read Only No

DynamicViscosity

Specifies the fluid dynamic viscosity or machine Reynolds number.

Available options:

A value less than 1 [N s m^-2] (or equivalent value in other units) is interpreted as the dynamic viscosity. Note that the value must be greater than 0.0000001 [N s m^-2].
A value of 0 causes Vista TF to calculate the dynamic viscosity from an inbuilt equation for dynamic viscosity based on Sutherland's law and the Inlet Total Temperature. This works only for an ideal gas.
A value greater than 1 [N s m^-2] (or equivalent value in other units) is interpreted as the Reynolds number, in which case Vista TF calculates the dynamic viscosity using this Reynolds number, the Reference Diameter, the Machine Rotational Speed, and the fluid density.

Type Quantity (p. 1422)

Read Only No

FlowOption

Specifies the types of boundary conditions.

Available options:

MassFlow	Specify the mass flow rate and the inlet total pressure and total temperature conditions.
PressureRatio	Specify the estimated mass flow rate and the total to static pressure ratio on the mean streamline.
PressureDifference	Specify the estimated mass flow rate and the total to static pressure difference on the mean streamline.

Type FlowType (p. 1386)

Read Only No

FluidOption

Specifies the type of fluid.

Available options:

IdealGas	For compressible flows with ideal gas properties.
RealGas	For compressible flow with real gas properties
Liquid	For incompressible flows.

Type FluidType (p. 1386)

Read Only No

GammaGas

Specifies the gas specific heat ratio when FluidOption = IdealGas.

Type float (p. 1432)

Read Only No

InitialCmUref

Specifies the initial guess for the meridional velocity divided by a characteristic velocity, where the latter is half the Reference Diameter multiplied by the Machine Rotational Speed. For more information, see the description for cm_start in Specification of the Control Data File (*.con) in the Vista TF Reference Guide.

Type float (p. 1432)

Read Only No

InletPt

Specifies the inlet total pressure.

Type Quantity (p. 1422)

Read Only No

InletSwirlAngle

Specifies the inlet swirl angle, which is from the axial direction and positive in the direction of rotation.

Type Quantity (p. 1422)

Read Only No

InletTt

Specifies the inlet total temperature.

Type Quantity (p. 1422)

Read Only No

MachineTypeOption

Specifies the type of machine that will be analyzed. This property specifies which template files are used and which report is used for the results. This property is only available if the geometry data is imported rather than transferred from BladeEditor.

Available options:

Pump
AxialCompressor
CentrifugalCompressor
Fan
AxialTurbine
RadialTurbine
HydraulicTurbine
Unknown
Other (Obsolete use Unknown)

Type MachineType (p. 1401)

Read Only No

MassFlow

Specifies the mass flow rate when FlowOption = MassFlow or the estimated mass flow rate when FlowOption = PressureRatio or PressureDifference.

Type Quantity (p. 1422)

Read Only No

MaxIterations

Specifies the number of calculating streamlines used in the solver.

Type int (p. 1394)

Read Only No

NumBladeRows

Specifies the number of blade rows solved in the analysis. This property is only available if the geometry data is imported rather than transferred from BladeEditor, and it is only used by the report template for the Results.

Type int (p. 1394)

Read Only No

NumStreamlines

Specifies the number of calculating streamlines used in the solver.

Type int (p. 1394)

Read Only No

PolytropicEfficiency

Specifies the small scale polytropic efficiency for the machine, and is used to calculate the losses.

Type float (p. 1432)

Read Only No

PressureDifference

Specifies the total to static pressure difference on the mean streamline when FlowOption = PressureDifference.

Type Quantity (p. 1422)

Read Only No

PressureRatio

Specifies the total to static pressure ratio on the mean streamline when FlowOption = PressureRatio.

Type float (p. 1432)

Read Only No

RealGasOption

Specifies which real gas is used

Type RealGas (p. 1423)

Read Only No

RefDiameter

Specifies the reference diameter for the definition of the flow coefficient.

Type Quantity (p. 1422)

Read Only No

RefDiameterOption

Specifies whether to have Vista TF calculate the reference diameter automatically or to use a user defined value.

Available options:

Automatic	Reference diameter is calculated using the first rotating component.
User defined	Reference diameter is specified by the user.

Type RefDiameterType (p. 1424)

Read Only No

RGas

User input, specific gas constant

Type Quantity (p. 1422)

Read Only No

RhoFluid

Specifies the fluid density when FluidOption = Liquid.

Type Quantity (p. 1422)

Read Only No

RotationalDirection

Specifies the positive direction of machine rotation about the Z-axis.

Available options:

RightHanded	Positive rotation is clockwise about the Z-axis.
LeftHanded	Positive rotation is counterclockwise about the Z-axis.

Type RotationType (p. 1427)

Read Only No

RotationalSpeed

The specified machine rotational speed.

Type Quantity (p. 1422)

Read Only No

Vista TF Solution

This container holds Solution data for an instance of Vista TF.

Data Entities

SolutionOptionsDataEntity

This data entity represents the Vista TF solution options for a project definition.

Properties

DisplayText

The general property that defines the user-visible name of an entity. This property is defined for all data entities but is used only in those entities that present a label in the user interface.

Type string (p. 1438)

Read Only No

ExpressionPrefix

Specifies the name that CFD-Post will prepend all Vista TF system

This field will be populated either on the first solve of this container, or during the duplicate operation, where it will be copied over from the duplicated container.

Type string (p. 1438)

Read Only No

VistaTFSolution

This data entity has no user modifiable properties.

Properties

DisplayText

The general property that defines the user-visible name of an entity. This property is defined for all data entities but is used only in those entities that present a label in the user interface.

Type string (p. 1438)

Read Only No

Units

Units

This container holds the project Unit Systems and unit settings.

Data Entities

UnitSystem

Holds the definition of a unit system within the project.

Properties

DisplayText

The general property that defines the user-visible name of an entity. This property is defined for all data entities but is used only in those entities that present a label in the user interface.

Type string (p. 1438)

Read Only No

IsDefault

True if this is the current default unit system.

Type bool (p. 1360)

Read Only No

IsProjectUnitSystem

True if this is the current project unit system.

Type bool (p. 1360)

Read Only No

UnitSystemName

The internal name of the unit system.

Type string (p. 1438)

Read Only Yes

Methods

Export

Exports unit system information into a Units XML file.

Required Arguments

FilePath The full path and name of the file to be created.

Type string (p. 1438)

Namespaced Commands

AQWA

This namespace holds top-level commands and queries related to AQWA.

EditModel

Opens the AQWA editor to allow modification of AQWA Model data.

Required Arguments

Container Model container

Type DataContainerReference (p. 1371)

EditSetup

Opens the AQWA editor to allow modification of AQWA Setup data or viewing of the AQWA Results.

Required Arguments

Container Setup container

Type DataContainerReference (p. 1371)

EditSolution

Opens the AQWA editor to allow modification of AQWA Solution data.

Required Arguments

SolutionContainer Solution Container.

Type DataContainerReference (p. 1371)

ChemkinCommon

No details are provided for this entry.

CreateApplicationInputParameter

No details are provided for this entry.

Required Arguments

Container No details are provided for this entry.

Type DataContainerReference (p. 1371)

parameter No details are provided for this entry.

Type DesignPointParameter (p. 1374)

CreateApplicationOutputParameter

No details are provided for this entry.

Required Arguments

Container No details are provided for this entry.

Type DataContainerReference (p. 1371)

parameter No details are provided for this entry.

Type DesignPointParameter (p. 1374)

CreateInputParameter

No details are provided for this entry.

Required Arguments

Container No details are provided for this entry.

Type DataContainerReference (p. 1371)

parameter No details are provided for this entry.

Type DesignPointParameter (p. 1374)

CreateOutputParameter

No details are provided for this entry.

Required Arguments

Container No details are provided for this entry.

Type DataContainerReference (p. 1371)

parameter No details are provided for this entry.

Type DesignPointParameter (p. 1374)

DeleteApplicationInputParameter

No details are provided for this entry.

Required Arguments

Container No details are provided for this entry.

Type DataContainerReference (p. 1371)

parameter No details are provided for this entry.

Type DesignPointParameter (p. 1374)

DeleteApplicationOutputParameter

No details are provided for this entry.

Required Arguments

Container No details are provided for this entry.

Type DataContainerReference (p. 1371)

parameter No details are provided for this entry.

Type DesignPointParameter (p. 1374)

DeleteDesignPointParameter

Delete one input parameter from the given Container.

Required Arguments

Container DataObjectContainer of parameters

Type DataContainerReference (p. 1371)

Parameter Name of the parameter to be delected

Type DesignPointParameter (p. 1374)

ImportProject

Import a project into Chemkin

Required Arguments

Container Reference to the data container

Type DataContainerReference (p. 1371)

ProjectFilePath Path to the project file

Type string (p. 1438)

JournalOperation

No details are provided for this entry.

Required Arguments

Command Command to send to application

Type string (p. 1438)

Container Reference to a data container

Type DataContainerReference (p. 1371)

RebaseProject

Update the project path in Chemkin

Required Arguments

Container Reference to a data container

Type DataContainerReference (p. 1371)

SaveProject

Update the project path in Chemkin

Required Arguments

Container Reference to a data container

Type DataContainerReference (p. 1371)

SolveCommand

Update the project path in Chemkin

Required Arguments

Container Reference to a data container

Type DataContainerReference (p. 1371)

ValidateProject

Validate the project

Required Arguments

Container Reference to a data container

Type DataContainerReference (p. 1371)

Customization

This namespace holds top-level commands and queries related to ACT Customization in AIM

GetCustomTemplateState

No details are provided for this entry.

Return No details are provided for this entry.

Type List (p. 1400)<Object (p. 1409)>

Required Arguments

Template No details are provided for this entry.

Type DataReference (p. 1371)

EngData

This namespace holds top-level commands and queries related to Engineering Data.

CreateLibrary

Creates a new Engineering Data library.

Return The created Engineering Data library.

Type DataContainerReference (p. 1371)

Required Arguments

Name The name of a new library.

Type string (p. 1438)

Optional Arguments

FilePath The target path for a new library.

Type string (p. 1438)

OpenLibrary

Opens a library of engineering information so that it can be viewed and if permissions allow, edited.

Return The DataContainerReference for the library that was opened.

Type DataContainerReference (p. 1371)

Required Arguments

Source The source of the library.

Type string (p. 1438)

Example

This code shows how to open a library from the provided samples.

```
installDir = r"C:\Program Files\ANSYS Inc\v121"
library1 = EngData.OpenLibrary(
   Name="General Materials",
   Source=installDir+r"\Addins\EngineeringData\Samples\General_Materials.xml")
```

EngineeringData

This namespace holds top-level commands and queries related to Engineering Data

GetFieldVariablesQuery

Queries Engineering Data for those field variables which can be created on a PropertyData

```
material1 = engineeringData1.GetMaterial(Name="Structural Steel")
// Create FieldVariableDataObject to get/set the field data for all
// "Temperature" field variables in "Structural Steel"
fieldVariableData1 = material1.CreateFieldVariableDataObject(Name="Temperature")
// Set upper and lower limits for all "temperature field variables in "Structural Steel"
fieldVariableData1.LowerLimit = 100
fieldVariableData1.UpperLimit = 300
```

Return Dictionary of field variable collection

Type Dictionary (p. 1375)<string (p. 1438), string (p. 1438)>

Required Arguments

DataReturn The data information to return for each field variable. The string can be seen in the journal by looking at the Qualifiers and CustomData when creating a field variable in the UI

Type string (p. 1438)

Extensions

This namespace holds top-level commands and queries related to ACT.

InstallExtension

This command install an extension identified by its filename.

If this extension already exists, the user can force to install the extension by setting the variable "ForceInstall" to true.

Required Arguments

ExtensionFileName ExtensionFileName argument containing the file name of the extension.

Type string (p. 1438)

Optional Arguments

ForceInstall Indicates whether the extension will be installed even if this extension already exists.

Type bool (p. 1360)

Default Value False

Example

To install an extension:

```
Extensions.InstallExtension(ExtensionFileName=r"c:\my_extensions_repository\my_extension.wbex")
```

To install an extension and force installation if the extension already exists:

```
Extensions.InstallExtension(ExtensionFileName=r"c:\my_extensions_repository\my_extension.wbex",ForceInstal
```

LoadExtension

This command load an installed extension.

Required Arguments

Id Id argument containing the GUID of the extension.

Type string (p. 1438)

Optional Arguments

Format Format argument containing the format of the extension (Scripted or Binary).

Type string (p. 1438)

Version Version argument containing the version of the extension.

Type string (p. 1438)

Example

To load an extension:

```
Extensions.LoadExtension(Id="7FD4DE83-39D0-4252-859B-2393C67F8EC8")
Extensions.LoadExtension(Id="7FD4DE83-39D0-4252-859B-2393C67F8EC8", Version="1.0")
Extensions.LoadExtension(Id="7FD4DE83-39D0-4252-859B-2393C67F8EC8", Version="1.0", Format="Binary")
```

UninstallExtension

This command uninstall an extension.

Required Arguments

Id Id argument containing the GUID of the extension.

Type string (p. 1438)

Optional Arguments

ForceUninstall Indicates whether the extension will be uninstalled silently.

Type bool (p. 1360)

Default Value False

Format Format argument containing the format of the extension (Scripted or Binary).

Type string (p. 1438)

Version Version argument containing the version of the extension.

Type string (p. 1438)

Example

To uninstall an extension:

```
Extensions.UninstallExtension(ExtensionFileName=r"c:\my_extensions_repository\my_extension.wbex")
```

UnloadExtension

This command unload a loaded extension.

Required Arguments

Id Id argument containing the GUID of the extension.

Type string (p. 1438)

Optional Arguments

Format Format argument containing the format of the extension (Scripted or Binary).

Type string (p. 1438)

Version Version argument containing the version of the extension.

Type string (p. 1438)

Example

To unload an extension:

```
Extensions.UnloadExtension(Id="7FD4DE83-39D0-4252-859B-2393C67F8EC8")
Extensions.UnloadExtension(Id="7FD4DE83-39D0-4252-859B-2393C67F8EC8", Version="1.0")
Extensions.UnloadExtension(Id="7FD4DE83-39D0-4252-859B-2393C67F8EC8", Version="1.0", Format="Binary")
```

Fluent

This namespace holds top-level commands and queries related to Fluent.

CfxStartPolling

Starts polling during solution monitoring

FluentStartPolling

Starts polling during solution monitoring

ForteCommon

This namespace holds common top-level commands and queries related to Forte

ImportMesh

Import a Mesh into the setup component for Forte

Required Arguments

Container Reference to the data container

Type DataContainerReference (p. 1371)

MeshPath Path to the Mesh file

Type string (p. 1438)

ImportProject

Import a project into Forte

Required Arguments

Container Reference to the data container

Type DataContainerReference (p. 1371)

ProjectFilePath Path to the project file

Type string (p. 1438)

JournalOperation

No details are provided for this entry.

Required Arguments

Command Command to send to application

Type string (p. 1438)

Container Reference to a data container

Type DataContainerReference (p. 1371)

MonitorRun

Open the Forte monitor application

Required Arguments

Container Reference to a data container

Type DataContainerReference (p. 1371)

RebaseProject

Update the project path in Forte

Required Arguments

Container Reference to a data container

Type DataContainerReference (p. 1371)

ReplayJournal

Play a journal file in Forte

Required Arguments

Container Reference to a data container

Type DataContainerReference (p. 1371)

JournalPath Path to the journal file

Type string (p. 1438)

SendCommand

No details are provided for this entry.

Required Arguments

Command Command to send to application

Type string (p. 1438)

Container Reference to a data container

Type DataContainerReference (p. 1371)

Graphics

This namespace holds top-level commands and queries related to Graphics.

GetAxisContinuous

A query to return an AxisContinuous data reference.

Return A reference to the AxisContinuous data entity.

Type DataReference (p. 1371)

Required Arguments

Name The data entity name of the AxisContinuous object to be found.

Type string (p. 1438)

GetAxisDiscrete

A query to return an AxisDiscrete data reference.

Return A reference to the AxisDiscrete data entity.

Type DataReference (p. 1371)

Required Arguments

Name The data entity name of the AxisDiscrete object to be found.

Type string (p. 1438)

GetChartXY

A query to return a ChartXY data reference.

Return A reference to the ChartXY data entity.

Type DataReference (p. 1371)

Required Arguments

Name The data entity name of the ChartXY object to be found.

Type string (p. 1438)

GetChartXYZ

A query to return an ChartXYZ data reference.

Return A reference to the ChartXYZ data entity.

Type DataReference (p. 1371)

Required Arguments

Name The data entity name of the ChartXYZ object to be found.

Type string (p. 1438)

GetCorrelationMatrix

A query to return a CorrelationMatrix data reference.

Return A reference to the CorrelationMatrix data entity.

Type DataReference (p. 1371)

Required Arguments

Name The data entity name of the CorrelationMatrix object to be found.

Type string (p. 1438)

GetLegend

A query to return a Legend data reference.

Return A reference to the Legend data entity.

Type DataReference (p. 1371)

Required Arguments

Name The data entity name of the Legend object to be found.

Type string (p. 1438)

GetMultiAxisChart

A query to return a MultiAxisChart data reference.

Return A reference to the MultiAxisChart data entity.

Type DataReference (p. 1371)

Required Arguments

Name The data entity name of the MultiAxisChart object to be found.

Type string (p. 1438)

GetPieChart

A query to return a PieChart data reference.

Return A reference to the PieChart data entity.

Type DataReference (p. 1371)

Required Arguments

Name The data entity name of the PieChart object to be found.

Type string (p. 1438)

GetRenderStyle

A query to return a RenderStyle data reference.

Return A reference to the RenderStyle data entity.

Type DataReference (p. 1371)

Required Arguments

Name The data entity name of the RenderStyle object to be found.

Type string (p. 1438)

GetVariable

A query to return a Variable data reference.

Return A reference to the Variable data entity.

Type DataReference (p. 1371)

Required Arguments

Name The data entity name of the Variable object to be found.

Type string (p. 1438)

GetVariableXY

A query to return a VariableXY data reference.

Return A reference to the VariableXY data entity.

Type DataReference (p. 1371)

Required Arguments

Name The data entity name of the VariableXY object to be found.

Type string (p. 1438)

GetVariableXYZ

A query to return a VariableXYZ data reference.

Return A reference to the VariableXYZ data entity.

Type DataReference (p. 1371)

Required Arguments

Name The data entity name of the VariableXYZ object to be found.

Type string (p. 1438)

IcePak

This namespace holds top-level commands and queries related to IcePak

IcepakStartPolling

Starts polling during solution monitoring

Mechanical

This namespace holds top-level commands and queries related to Mechanical.

ImportLegacyDatabase

Imports a legacy database given the filepath

Return A reference to the LegacyDataBaseResumeData object that contains information about the imported database

Type DataReference (p. 1371)

Required Arguments

FilePath Path to legacy database to import.

Type string (p. 1438)

Meshing

This namespace holds top-level commands and queries related to Meshing.

ImportLegacyDatabase

Imports legacy Meshing editor files or CFX mesh files to the Meshing editor from an existing .cmdb file.

Return The data references created by the import process.

Type DataReference (p. 1371)

Required Arguments

FilePath The file path to the legacy database to be imported.

Type string (p. 1438)

MultiphysicsCoupling

This namespace holds top-level commands and queries related to System Coupling.

ExportSystemCouplingSetup

No details are provided for this entry.

Required Arguments

Container Setup container

Type DataContainerReference (p. 1371)

DestinationDirectory Fully qualified path of the directory to write to

Type string (p. 1438)

Parameters

This namespace holds top-level commands and queries related to Parameters, Design Points and DesignXploration.

ApproveAllRomGeneratedData

Approve all ROM data contained in ROM or DX addin.

ClearDesignPointsCache

Clears the Design Points Cache for Design Exploration features. DesignXplorer is using an internal cache of Design Points to reduce the number of Design Point update operations. This cache is available accross all of the design exploration systems of a project. This command clears all the data contained in the cache.

Example

The following example shows how to clear the cache in a project.

```
ClearDesignPointsCache()
```

CreateDesignPoint

A command to create a new design point that contains all the parameters, but with their values set to null.

The design point can be created initially as a "exported" design point, which will allow the files and associated databases to be reloaded at a later time to view the results of the design point update.

Return A DataReference to the new design point entity.

Type DataReference (p. 1371)

Optional Arguments

Exported Indicates that the newly created design point will be exported.

Type bool (p. 1360)

Default Value False

Name Name of new design point

Type string (p. 1438)

Retained Indicates that the newly created design point will be retained.

Type bool (p. 1360)

Default Value False

Example

The following example illustrates proper CreateDesignPoint command invocation:

```
newDP = Parameters.CreateDesignPoint(Exported=True, Retained=False)
```

CreateParameter

Creates a parameter, optionally associated with a data entity property. The expression and visible name for the parameters can be optionally specified.

Return A DataReference to the new parameter entity.

Type DataReference (p. 1371)

Optional Arguments

DisplayText The display text for the parameter.

Type string (p. 1438)

Entity The data model entity that holds the property to be parameterized.

Type DataReference (p. 1371)

Expression The initial expression for the parameter.

Type string (p. 1438)

IsDirectOutput Indicates if the parameter is a direct output parameter.

Type bool (p. 1360)

Default Value False

IsOutput Indicates if the parameter is an output parameter.

Type bool (p. 1360)

PropertyName The name of the data model property which the parameter is associated with.

Type string (p. 1438)

Example

The following two examples illustrate input and output parameter creation

```
myEntity = #Query to obtain your entity on to which parameters will be created.
inptParam1 = Parameters.CreateParameter(Entity=myEntity,
                                        PropertyName="Value",
                                        IsOutput=False,
                                        IsDirectOutput=False,
                                        Expression=None,
                                        DisplayText="My Input Parameter")

outpParam1 = Parameters.CreateParameter(Entity=myEntity,
                                        PropertyName="Value",
                                        IsOutput=True,
                                        IsDirectOutput=True,
                                        Expression=None,
                                        DisplayText="My Output Parameter")

exprParam1 = Parameters.CreateParameter(Entity=None,
                                        PropertyName=None,
                                        IsOutput=False,
                                        IsDirectOutput=False,
                                        Expression="cos(1)",
                                        DisplayText="My Input Expression Parameter")

exprParam2 = Parameters.CreateParameter(Entity=None,
                                        PropertyName=None,
                                        IsOutput=True,
                                        IsDirectOutput=False,
                                        Expression="sin(P1)",
                                        DisplayText="My Output Expression Parameter")
```

CreateParameterSummaryChart

Creates a multi-axis parallel coordinate chart based on the parameters supplied to the command.

```
chart1 = Parameters.CreateParameterSummaryChart(Parameters=[])
-or-
parameter1 = Parameters.GetParameter(Name="P1")
parameter2 = Parameters.GetParameter(Name="P2")
parameter3 = Parameters.GetParameter(Name="P3")
chart1 = Parameters.CreateParameterSummaryChart(Parameters=[parameter1, parameter2, parameter3])
```

Return A data reference that represents the created chart.

Type DataReference (p. 1371)

Required Arguments

Parameters The parameters to be included in the parallel coordinate plot.

Type DataReferenceSet (p. 1371)

CreateParameterVsParameterChart

Creates a 2-dimensional (x,y) chart that can be used to compare two parameters.

Return A data reference that represents the chart that is created.

Type DataReference (p. 1371)

Optional Arguments

XAxisBottom A data reference for the parameter that represents the bottom x-axis.

Type DataReference (p. 1371)

XAxisTop A data reference for the parameter that represents the top x-axis.

Type DataReference (p. 1371)

YAxisLeft A data reference for the parameter that represents the left y-axis.

Type DataReference (p. 1371)

YAxisRight A data reference for the parameter that represents the right y-axis.

Type DataReference (p. 1371)

Example

This example illustrates the creation of a chart comparing two parameters.

```
parameter1 = Parameters.GetParameter(Name="P1")
parameter2 = Parameters.GetParameter(Name="P2")
chart1 = Parameters.CreateParameterVsParameterChart(XAxisBottom=parameter1, YAxisLeft=parameter2)
```

ExportAllDesignPointsData

Export the parameter values for up-to-date parameters for all the design points to a csv file.

Required Arguments

FilePath The exported file name.

Type string (p. 1438)

Optional Arguments

AppendMode True to append to an existing csv file, False to overwrite it.

Type bool (p. 1360)

Default Value False

DesignPoints The set of design points to be exported in the same file. If this parameter is not given, all the design points are exported.

Type List (p. 1400)<DataReference (p. 1371)>

Example

The following example shows how the user can export up-to-date parameter values of all the design points of the project.

```
Parameters.ExportAllDesignPointsData(FileName="designPoints.csv")
```

ExportLink

Export the link to design set to a csv file.

Required Arguments

FileName The exported file name.

Type string (p. 1438)

Optional Arguments

AppendMode True to append to an existing csv file, False to overwrite it.

Type bool (p. 1360)

Default Value False

Example

The following example shows how the user can export the dictionary of linked point.

```
Parameters.ExportLink(FileName="Link.csv")
```

GetAllDesignPoints

Returns a set of all design points in the project.

Return A data reference set containing all the design points.

Type DataReferenceSet (p. 1371)

GetAllSortedDesignPoints

Returns a set of all design points, sorted by their update order in ascending, in the project.

Return A data reference set containing all the sorted design points.

Type DataReferenceSet (p. 1371)

GetAllDesignPointsWithFiles

Returns a set of all design points in the project that have files under their design point directories.

Return A data reference set containing all the design points that have files under their design point directories.

Type DataReferenceSet (p. 1371)

GetAllExportedDesignPoints

Retrieves a set of all exported design points.

Return A set of all design points.

Type DataReferenceSet (p. 1371)

Optional Arguments

IncludingBaseDesignPoint A flag to determine whether the query to also return the base design point, which is always retained.

Type bool (p. 1360)

Default Value True

GetAllParameters

Returns a set of all parameters in the project.

Return A data reference set containing all the parameters.

Type DataReferenceSet (p. 1371)

GetAllRetainedDesignPoints

Retrieves a set of all retained design points.

Return A set of all design points.

Type DataReferenceSet (p. 1371)

Optional Arguments

IncludingBaseDesignPoint A flag to determine whether the query to also return the base design point, which is always retained.

Type bool (p. 1360)

Default Value True

GetAllRetainedDesignPointsWithValidData

Retrieves a set of all retained design points with valid data

Return A set of all design points.

Type DataReferenceSet (p. 1371)

Optional Arguments

IncludingBaseDesignPoint A flag to determine whether the query to also return the base design point, which is always retained.

Type bool (p. 1360)

Default Value True

GetDesignPoint

Returns a design point based on the name of the design point.

Return The design point of interest.

Type DataReference (p. 1371)

Required Arguments

Name The name of the design point.

Type string (p. 1438)

GetNamedExpression

A query to return a reference for a named expression given a name and a scope

Return The named expression of interest.

Type DataReference (p. 1371)

Required Arguments

Name The name of the named expression.

Type string (p. 1438)

Optional Arguments

Scope The scope in which the named expression is located. Default is global scope.

Type DataReference (p. 1371)

GetParameter

A query to return a reference for a parameter given a name and a scope

Return The parameter of interest.

Type DataReference (p. 1371)

Required Arguments

Name The name of the parameter.

Type string (p. 1438)

Optional Arguments

Scope The scope in which the parameter is located. If this parameter is omitted or is null then the returned parameter will be global.

Type DataReference (p. 1371)

GetParameterSummaryChart

A query to return a reference for a parameter summary chart by name.

Return The parameter chart of interest.

Type DataReference (p. 1371)

Required Arguments

Name The name of the chart.

Type string (p. 1438)

GetParameterVsParameterChart

A query to return a reference for a parameter vs. parameter chart by name.

Return The parameter chart of interest.

Type DataReference (p. 1371)

Required Arguments

Name The name of the chart.

Type string (p. 1438)

GetScope

Returns a data reference to a named scope

Return the Scope in the Parameter Manager

Type DataReference (p. 1371)

Required Arguments

Name The name of the scope entity.

Type string (p. 1438)

IsParameterInDesignPointUpToDate

Checks whether a parameter is up to date in a design point.

Return True if the parameter is up to date in the design point.

Type bool (p. 1360)

Required Arguments

DesignPoint The design point data reference.

Type DataReference (p. 1371)

Parameter The parameter data reference.

Type DataReference (p. 1371)

IsParameterUpToDate

Indicates whether a parameter is up to date in a design point.

Return Return if the parameter is up to date in the design point.

Type bool (p. 1360)

Required Arguments

Parameter Parameter

Type DataReference (p. 1371)

Optional Arguments

DesignPoint Design point

Type DataReference (p. 1371)

OptimizeUpdateOrder

Optimizes Update Order of Design Points to minimize the number of modifications between two consecutives Design Points

RemoveUnusedRomSnapshots

Remove all the ROM snapshots that are not used by DX and not in any up to date point.

Required Arguments

SystemId No details are provided for this entry.

Type string (p. 1438)

SetParameter

Sets the expression of the specified parameter in the base design point.

Required Arguments

Expression The string with the expression for the parameter.

Type string (p. 1438)

Parameter The parameter data reference.

Type DataReference (p. 1371)

SetUpdateOrderByRow

Sets a value for the update order property of all design points using a sorting method.

Optional Arguments

SortBy The definition of the sorting.

Type List (p. 1400)<string (p. 1438)>

Project

This namespace holds top-level commands and queries related to the Project, File and Units.

AbandonBackgroundTasks

No details are provided for this entry.

AbandonTopLevelBackgroundDesignPointUpdate

No details are provided for this entry.

AddPropertyTable

This command is used to allow the expression of a property to reference data in the named table. This is a reference and not a copy, and so if the table is used in the expression, and a change is made on

the referenced table and the change is in the data being referenced the value of the property will be modified. The command creates table functions in the context of this expression which can be used with constants, field variables, or another named expression. If the property does not support an expression an error will occur.

Return An int which corresponds to the Index passed in to the command and indicates the comand successfully completed.

Type int (p. 1394)

Required Arguments

Dependents A List of the column index(es) to use from the Table which correspond to the dependent variable(s) of the Property. The indexes can be an Integer index or the String name of the column.

Type List (p. 1400)<Object (p. 1409)>

Independents A List of the column index(es) to use from the Table for use in the expression and correspond to the independent variables of the Property. The indexes can be an Integer index or the String name of the column.

Type List (p. 1400)<Object (p. 1409)>

Index An integer which uniquely identifies the Table for use in the expression of the Property. In the expression this table can be accessed via tableIndex(args). The table with an index of one can be accessed in the expression as table(args) or table1(args).

Type uint (p. 1446)

Parent The DataReference of the object which has the Property that the Table should be added to.

Type DataReference (p. 1371)

Property The String which provides the path of the property on the Parent.

Type string (p. 1438)

Table The DataReference of the table to add to the Property.

Type DataReference (p. 1371)

Optional Arguments

Bounds An enum which describes what will occur when the index is outside of the bounds of the table. The valid enum values are; Constant, Fit, Zero, and Error. The default is Error.

Type TableInterpolationBeyondBounds (p. 1441)

Default Value Error

Interpolation An enum of the type of interpolation to use when accessing data from the table at a given index. The valid enum values are; None, Linear, Cubic Spline. The default is Linear.

Type TableInterpolation (p. 1440)

Default Value Linear

Example

```
tbl1 = pressure1.CreateTable(Columns=[[“X”, “Length”], [“Y”, “Length”], [“Z”, “Length”], [“Pressure”, “Pre
#
# Use three of the columns from tbl1
bc1.AddPropertyTable(Property=“Magnitude”, Table=tbl1, Index=1, Independents=[“X”, “Y”], Dependents=[“Pres
# In the table function the x and y refer to field variables
bc1.SetPropertyExpression(Entity=bc1, Name=“Magnitude”, Expresssion=“table1(x, y)”)
#
# Use four of the columns from tbl1
bc1.AddPropertyTable(Property=“Temperature”, Table=tbl1, Index=1, Independents=[“X”, “Y”, “Z”], Dependents
# In the table function the x, y, and z refer to field variables
bc1.SetPropertyExpression(Entity=bc1, Name=“Temperature”, Expression=“table1(x, y, z)”)
#
# Use two of the columns from tbl1
bc1.AddPropertyTable(Property=“Magnitude”, Table=tbl1, Index=1, Independents=[“X”], Dependents=[“Pressure”
# In the table function the x refers to field variables
bc1.SetPropertyExpression(Entity=bc1, Name=“Magnitude”, Expression=“table1(x)”)
#
# Use of a constant
bc1.SetPropertyExpression(Entity=bc1, Name=“Magnitude”, Expression=“table1(20)”)
```

AddSourceToComponentInSystem

No details are provided for this entry.

Required Arguments

SourceComponent No details are provided for this entry.

Type DataReference (p. 1371)

TargetComponent No details are provided for this entry.

Type DataReference (p. 1371)

AddTableColumn

AddColumn will add a column at the specified index to the table.

Return The Integer which corresponds to the index of the column which was newly added to the table.

Type int (p. 1394)

Required Arguments

Name The String used to uniquely identify the column (case-sensitive) in this table. If the value is not unique an error will occur.

Type string (p. 1438)

Table The DataReference of the table to add this column of data.

Type DataReference (p. 1371)

Optional Arguments

Data A List of Strings which specifies the data for this column. The String must be a valid Quantity string in the form "value [units]".

If the size of the list is smaller than the current number of rows the data will be initialized to an undefined value. If the size of the List is greater than the current number of rows then an error will occur.

Type List (p. 1400)<Quantity (p. 1422)>

Index The Integer index of the location for the newly added column. An index of 0 (zero) will add to the left-most side of the table. The default is to add the column to the right-most side of the table.

Type int (p. 1394)

PhysicalQuantity The String for the physical quantity (e.g. Density, Pressure, etc.) used to verify the data has the correct units.

Type string (p. 1438)

Example

This example illustrates adding columns to a table.

```
# NameSpace is the Addin namespace in use (e.g. Study, EngData, etc.)  which may have its specific CreateT
# drParent is the DataReference of an object that has been previously created
table1 = NameSpace.CreateTable(Parent=drParent, Columns=[])
table1.AddRow()
table1.AddRow()
table1.AddRow()
table1.AddColumn("Displacement", "Length", Data=["0 [m]",".001 [m]"])
table1.AddColumn("Temperature","Temperature", Index=0)
```

AddTableRow

AddRow will add a row at the specified index to the table.

Return The Integer which corresponds to the index of the row which was newly added to the table.

Type int (p. 1394)

Required Arguments

Table The DataReference of the table to add this row of data.

Type DataReference (p. 1371)

Optional Arguments

Data A List of Strings which specifies the data for this row. The String must be a valid Quantity string in the form "value [units]".

Type List (p. 1400)<Quantity (p. 1422)>

Index The Integer index of the location for the newly added row. An index of 0 (zero) will add to the top of the table. The default is to add the row to the end of the table.

Type int (p. 1394)

Example

This example illustrates adding rows to a table.

```
# NameSpace is the Addin namespace in use (e.g. Study, EngData, etc.)  which may have its specific CreateT
# drParent is the DataReference of an object that has been previously created
table1 = NameSpace.CreateTable(Parent=drParent, Columns=[("Temperature","Temperature"),("Displacement","Le
table1.AddRow(Data=["100 [C]","1.0 [m]"])
table1.AddRow()
table1.AddRow(Data=[".06 [m]"])
table1.AddRow(Index=0, Data=["50 [C]",".001 [m]"])
```

Archive

Creates a project archive based on current project files and contents. The project must be saved before an archive can be created.

Required Arguments

FilePath The full file path of the project archive to be created.

Type string (p. 1438)

Optional Arguments

FailIfMissingFiles Set to true to force failure if there are any files to repair.

Type bool (p. 1360)

Default Value False

IncludeExternalImportedFiles Whether to include files imported to the project from external locations.

Type bool (p. 1360)

Default Value False

IncludeSkippedFiles Whether to include Results and Solution files in the archive.

Type bool (p. 1360)

Default Value True

IncludeUserFiles Whether to include files in the project user_files directory.

Type bool (p. 1360)

Default Value True

Example

This example creates a project archive from the currently loaded project. The full file path is specified and the argument will include items from the user_files folder, include results/solution files, and exclude external files imported into the project.

```
Archive(FilePath=r"C:\Users\AnsUser\simpleStructural.zip",
        IncludeUserFiles=True,
        IncludeSkippedFiles=True,
        IncludeExternalImportedFiles=False,
        FailIfFilesNeedRepair=False)
```

CleanSystem

No details are provided for this entry.

Required Arguments

Systems No details are provided for this entry.

Type List (p. 1400)<DataReference (p. 1371)>

ClearMessagesBySource

Clears all messages that have no Source from the message store.

Optional Arguments

Source The optional source of the message.

Type Object (p. 1409)

SourceUsage The optional message storage arguments.

Type ClearMessagesOption (p. 1364)

ClearMessages

Clears all messages from the message store.

CopyFile

Copies a file to a target directory. Either the SourceFile argument or the SourceFilePath argument must be specified.

Return A reference to the created file is returned in this argument.

Type DataReference (p. 1371)

Required Arguments

DestinationDirectoryPath The full path to the destination directory.

Type string (p. 1438)

Overwrite Specifies whether to overwrite an existing files of the same name. Note: A registered file may not be overwritten.

Type bool (p. 1360)

Optional Arguments

SourceFile A reference to the file to be copied. A file reference can be obtained from commands and queries like RegisterFileCommand and GetRegisteredFileQuery.

Type DataReference (p. 1371)

SourceFilePath A full path to the file to be copied.

Type string (p. 1438)

Example

This example illustrates a path-based file copy procedure. The specified file will be copied to the specified destination directory path. The destination file will not be registered and the return value will be null.

```
CopyFile(SourceFilePath=r"C:\Users\anyuser\myTextFile.txt",
          DestinationDirectoryPath=r"C:\Users\anyuser\newfolder",
          Overwrite=True)
```

This second example illustrates a reference-based file copy procedure. The specified file will be copied to the specified destination directory path. The destination file will also be registered and the new reference returned.

```
file1 = GetRegisteredFile(FilePath=r"C:\path_to_file\file.txt")
file2 = CopyFile(SourceFile=file1,
          DestinationDirectoryPath=r"C:\Users\anyuser\newfolder",
```

```
            Overwrite=True)
```

CopySystems

No details are provided for this entry.

Return No details are provided for this entry.

Type DataReferenceSet (p. 1371)

Required Arguments

KeepConnections No details are provided for this entry.

Type bool (p. 1360)

Systems No details are provided for this entry.

Type List (p. 1400)<DataReference (p. 1371)>

CreateCustomUnitSystem

Creates a custom unit system based on predefined unit system (e.g., MKS, US Customary) or other custom unit sytem already defined.

Return A reference to the created UnitSystem data entity.

Type DataReference (p. 1371)

Required Arguments

BaseUnitSystemName The internal name of the unit system that is the basis for the newly created system.

Type string (p. 1438)

UnitSystemDisplayName The displayed name of the new unit system.

Type string (p. 1438)

UnitSystemName The internal name of the new unit system.

Type string (p. 1438)

Example

This example illustrates the creation of a custom unit system.

```
myUnitSystem = CreateCustomUnitSystem(UnitSystemName="MyUnitSystem",
                                      UnitSystemDisplayName="My Custom Unit System",
                                      BaseUnitSystemName="SI")
```

CreateTemplateFromProject

No details are provided for this entry.

Return No details are provided for this entry.

Type DataReference (p. 1371)

Required Arguments

Name No details are provided for this entry.

Type string (p. 1438)

CreateUnitSystem

Creates the UnitSystem data entity for the named system.

Return A data reference to the created UnitSystem entity.

Type DataReference (p. 1371)

Required Arguments

UnitSystemName The internal name of the unit system to be created.

Type string (p. 1438)

DeleteFile

Deletes the specified file. Either the File argument or the FilePath argument must be specified.

Optional Arguments

BackUp Specifies whether to back up the file before deletion. This optional argument's default value is true.

Type bool (p. 1360)

Default Value True

DeleteIfShared Specifies whether a registered file should be deleted even if it is still in use. This optional argument's default value is true. If this argument's value is false, an exception will be thrown if a shared file is encountered. To avoid the exception, set ErrorIfShared to false.

Type bool (p. 1360)

Default Value False

ErrorIfShared Specifies whether to throw an exception when trying to delete a shared file if DeleteIfShared is set to false.

Type bool (p. 1360)

File A reference to the file to be deleted. A file reference can be obtained from commands and queries like RegisterFileCommand and GetRegisteredFileQuery.

Type DataReference (p. 1371)

Example

The following example illustrates a simple file deletion; file is not deleteed if it is registered.

```
DeleteFile(FilePath=r"C:\Users\anyuser\path-to-file.extension")
```

The next example illustrates forced file deletion.

```
DeleteFile(FilePath=r"C:\Users\anyuser\path-to-file.extension", DeleteIfShared=True)
```

The next example illustrates a deletion of a registered file via a file reference. The file will be deleted if even if it is still in use. Note that the file will be backed up.

```
fileRef = GetRegisteredFile(FilePath=r"C:\Users\anyuser\path-to-file.extension")
DeleteFile(File=fileRef, DeleteIfShared=True,
               BackUp=True)
```

The final example illustrates a deletion of a registered file via a file reference without a forced deletion. If the file is shared, an error will not occur, and the file reference count will be decremented.

```
fileRef = GetRegisteredFile(FilePath=r"C:\Users\anyuser\path-to-file.extension")
DeleteFile(File=fileRef, ErrorIfShared=False)
```

DeleteSystems

No details are provided for this entry.

Required Arguments

Systems No details are provided for this entry.

Type List (p. 1400)<DataReference (p. 1371)>

DeleteTableColumn

DeleteColumn will delete a column at the specified index from the table.

Required Arguments

Index An Integer index or the String name of the column (case-sensitive) to remove from the table. The first column has an index of 0 (zero).

Type Object (p. 1409)

Table The DataReference of the table to delete the column from.

Type DataReference (p. 1371)

Example

This example illustrates removing a column from a table.

```
# NameSpace is the Addin namespace in use (e.g. Study, EngData, etc.)  which may have its specific CreateT
# drParent is the DataReference of an object that has been previously created
table1 = NameSpace.CreateTable(Parent=drParent, Columns=[("Temperature","Temperature"),("Displacement","Le
table1.AddRow(Data=["100 [C]","1.0 [m]"])
table1.AddRow(Data=["200 [C]",".02 [m]"])
table1.AddRow(Data=["300 [C]",".06 [m]"])
table1.DeleteColumn(Index='Displacement')
```

DeleteTableRow

DeleteRow will delete a row at the specified index from the table.

Required Arguments

Index The Integer index of the row to remove from the table. The row indexes for the table begin at 0 (zero).

Type int (p. 1394)

Table The DataReference of the table to delete the row from.

Type DataReference (p. 1371)

Example

This example illustrates removing a row from a table.

```
# NameSpace is the Addin namespace in use (e.g. Study, EngData, etc.)  which may have its specific CreateT
# drParent is the DataReference of an object that has been previously created
table1 = NameSpace.CreateTable(Parent=drParent, Columns=[("Temperature","Temperature"),("Displacement","Le
table1.AddRow(Data=["100 [C]","1.0 [m]"])
table1.AddRow(Data=["200 [C]",".02 [m]"])
table1.AddRow(Data=["300 [C]",".06 [m]"])
table1.DeleteRow(Index=2)
```

DeleteTabularDataRow

Delete a row from a tabular data sheet.

Required Arguments

Index Index of the row to delete.

Type int (p. 1394)

TabularData Data Entity that can be bound to ITabularDataWrite

Type DataReference (p. 1371)

Optional Arguments

SheetName Name of the sheet to access.

Type string (p. 1438)

SheetQualifiers SheetQualifiers is used to pass in the qualifiers to select between multiple sheets witht the same name. This is a dictionary of the Qualifier and its Value.

Type Dictionary (p. 1375)<string (p. 1438), string (p. 1438)>

Example

The following example illustrates the deletion of a row from a tabular data sheet.

```
#
# Create a new Engineering Data System and access Structural Steel
#
template1 = GetTemplate(TemplateName="EngData")
system1 = template1.CreateSystem()
engineeringData1 = system1.GetContainer(ComponentName="Engineering Data")
matl1 = engineeringData1.GetMaterial(Name="Structural Steel")
#
# Delete the first row in the Density property
#
matlProp1 = matl1.GetProperty(Name="Density")
DeleteTabularDataRow(matlProp1, Index = 0)
#
# Delete the first row in the Coefficient of Thermal Expansion property with
# optional SheeetName and SheetQualifiers
#
matlProp2 = matl1.GetProperty(Name="Coefficient of Thermal Expansion")
DeleteTabularDataRow(matlProp2,
                  SheetName="Coefficient of Thermal Expansion",
                  SheetQualifiers={"Definition Method": "Secant", "Behavior": "Isotropic"},
                  Index = 0)
```

DeleteUnitSystem

Deletes the named unit system.

Required Arguments

UnitSystemName The name of the unit system to delete.

Type string (p. 1438)

ExportReport

Writes out project report containing both framework- and addin-supplied content. The report output is in XML format. Or if the file path extension is html, the XML will be converted to html output file.

Required Arguments

FilePath The full destination file path for the reporting. If the extension is .xml, the report will be produced in XML format. If the extension is .html, the report will be poduced in XML and then translated into HTML format.

Type string (p. 1438)

GetAbsoluteUserPathName

Gets the absolute user path name for the given relative path name, based on the current setting of the user path root. See also GetUserPathRoot and SetUserPathRoot.

Return The returned absolute user path name.

Type string (p. 1438)

Required Arguments

RelativePathName The relative path name.

Type string (p. 1438)

Example

This example uses the SetUserPathRoot and GetAbsoluteUserPathName to read a project from two different user directories.

```
SetUserPathRoot(DirectoryPath = "C:/Users/myUser1/Projects")
Open(FilePath=GetAbsoluteUserPathName("proj1.wbpj")) # Read project from first location
SetUserPathRoot(UserPathRoot = "C:/Users/myUser2/Projects")
Open(FilePath=GetAbsoluteUserPathName("proj1.wbpj")) # Read project from second location
```

GetAllFiles

Gets the list of all files that have been registered with the project (even if external to the project directory) or exist within the project files directory.

Return List of paths to all known project files.

Type List (p. 1400)<string (p. 1438)>

GetAllSystems

No details are provided for this entry.

Return No details are provided for this entry.

Type DataReferenceSet (p. 1371)

GetMiscellaneousSettings

No details are provided for this entry.

Return No details are provided for this entry.

Type DataReference (p. 1371)

GetComponentTemplate

No details are provided for this entry.

Return No details are provided for this entry.

Type DataReference (p. 1371)

Required Arguments

Name No details are provided for this entry.

Type string (p. 1438)

GetCurrentRegisteredFiles

Gets the files registered with the project.

Return The set of data references to registered files.

Type DataReferenceSet (p. 1371)

GetDesignPointUpdateSettings

No details are provided for this entry.

Return No details are provided for this entry.

Type DataReference (p. 1371)

GetFileReference

Gets the File Reference to a registered file based upon the file reference's name. Throws an Invalid Operation Exception if a matching File Reference is not found.

Return The data reference of the file whose name matches the supplied parameter.

Type DataReference (p. 1371)

Required Arguments

Name The file reference name.

Type string (p. 1438)

GetFilesForDirectory

Gets the set of data references to registered files within a directory.

Return The set of data references to files in the directory.

Type DataReferenceSet (p. 1371)

Required Arguments

DirectoryPath The full path to the directory.

Type string (p. 1438)

GetFileType

Returns a reference to a FileType data entity, given the data entity name of the FileType object.

Return The data reference to the FileType data entitity.

Type DataReference (p. 1371)

Required Arguments

Name The data entity name of the FileType object.

Type string (p. 1438)

GetFrameworkBuildVersion

Get the current framework build version.

Return Return the framework build version.

Type string (p. 1438)

GetFrameworkVersion

Get the currently executing framework version.

Return Return the currently executing framework version.

Type string (p. 1438)

GetLicenseNames

No details are provided for this entry.

Return No details are provided for this entry.

Type Dictionary (p. 1375)<string (p. 1438), string (p. 1438)>

GetMaxProjectPathLength

Gets the max project path length created in WorkBench based on the project directory and project name.

Return The max project path length

Type int (p. 1394)

Required Arguments

ProjectDirectory The full path to the project directory.

Type string (p. 1438)

ProjectName The project name

Type string (p. 1438)

GetMessages

Returns DataReferences for all stored messages, ordered by message publication date/time with most recent messages first.

Return The list of StoredMessage data entities for current messages.

Type DataReferenceSet (p. 1371)

GetNeededLicenses

No details are provided for this entry.

Return No details are provided for this entry.

Type Dictionary (p. 1375)<string (p. 1438), int (p. 1394)>

GetPersistedProjectVersion

Given the path to a WB project, return the last-saved-in version. Note: The command will return '00' if a project is not supplied and the current session project has not been saved.

Return The Project's persisted version (e.g, 120, 121, 130, 140, 145, ...).

Type string (p. 1438)

Optional Arguments

FilePath The Project File Path (wbpj).

Type string (p. 1438)

GetProjectDirectory

Gets the full path to the current project directory.

Return The project directory full path.

Type string (p. 1438)

GetProjectFile

Gets the full path of the current project file (*.wbpj).

Return The full path of the current project file.

Type string (p. 1438)

GetProjectTemplate

No details are provided for this entry.

Return No details are provided for this entry.

Type DataReference (p. 1371)

Required Arguments

Name No details are provided for this entry.

Type string (p. 1438)

GetProjectUnitSystem

Gets the current unit system for the project.

Return A reference to the UnitSystem data entity.

Type DataReference (p. 1371)

GetPropertyExpression

Gets the expression defining a data entity property's value.

Return Returns the expression that currently defines the property's value.

Type string (p. 1438)

Required Arguments

Entity The entity to query.

Type DataReference (p. 1371)

Name The property name or member path.

Type string (p. 1438)

GetQuantityUnitForUnitSystem

Gets the unit for a quantity in the specified unit system.

Return The current Unit for the quantity.

Type string (p. 1438)

Required Arguments

QuantityName The name of the Quantity of interest.

Type string (p. 1438)

UnitSystemName The internal name of the unit system.

Type string (p. 1438)

GetQuantityUnitsForUnitSystem

Gets the units for all quantities in the specified unit system.

Return The units for all quantities

Type Dictionary (p. 1375)<string (p. 1438), string (p. 1438)>

Required Arguments

UnitSystem The internal name of the unit system

Type DataReference (p. 1371)

GetRegisteredFile

Gets the File Reference to a registered file based upon the file's path.

Return The data reference of the file whose location points to the specified path. Null if no data reference is found.

Type DataReference (p. 1371)

Required Arguments

FilePath The full path of the file.

Type string (p. 1438)

GetRegisteredFilesForDirectory

Gets list of File References for a directory path.

Return The list of data reference of files whose location root is DirectoryPath. Empty list if no data reference is found.

Type DataReferenceSet (p. 1371)

Required Arguments

DirectoryPath The full path to the directory.

Type string (p. 1438)

GetRsmQueues

Query to return Rsm queue names from available configured queues when Rsm is running using the new architecture

Return List of Rsm queues names, or null if configured queues are not available

Type List (p. 1400)<string (p. 1438)>

GetSchematicSettings

No details are provided for this entry.

Return No details are provided for this entry.

Type DataReference (p. 1371)

GetScriptVersion

returns the software version that the current script that is being executed is compliant with

Return the software version that the current script that is being executed is compliant with

Type string (p. 1438)

GetSystem

No details are provided for this entry.

Return No details are provided for this entry.

Type DataReference (p. 1371)

Required Arguments

Name No details are provided for this entry.

Type string (p. 1438)

GetTableData

GetData will return the data in a table.

Return A List of Strings or a List of List of Strings for the requested range.

Type List (p. 1400)<Object (p. 1409)>

Required Arguments

Table The DataReference of the table to get data from.

Type DataReference (p. 1371)

Optional Arguments

ColumnRange A Tuple of the column starting index and ending index to access the data. The default is (0, maximumColumns - 1). A single index may be used also to return a single column. The first column has an index of 0 (zero).

Type Object (p. 1409)

RowRange A Tuple of the row starting index and ending index to access the data. The default is (0, maximumRows - 1). A single index may be used also to return a single row. The first row has an index of 0 (zero).

Type Object (p. 1409)

Example

This example illustrates data retrieval from a table.

```
# drParent is the DataReference of an object that has been previously created
table1 = drParent.CreateTable(Columns=[("Temperature","Temperature"),("Displacement","Length")])
table1.AddRow(Data=["100 [C]","1.0 [m]"])
table1.AddRow(Data=["200 [C]",".02 [m]"])
table1.AddRow(Data=["300 [C]",".06 [m]"])
table1.GetData(RowRange=1) # get the second row of the table
table1.GetData(RowRange=(1,2)) # get the second and third row of the table
table1.GetData() # complete table in row-major
# Get table as column major
numCol = len(table1.GetData(RowRange=0))
colMajorArr = [0 for i in range(numCol)]
for colIdx in range(numCol):
    colMajorArr[colIdx] = [row[colIdx] for row in table1.GetData()]
colMajorArr
```

GetTabularData

Returns the tabular data associated with the data entity.

Return The returned data in scalar, list, or dictionary format.

Type Object (p. 1409)

Required Arguments

TabularData Data entity that has associated tabular data.

Type DataReference (p. 1371)

Optional Arguments

AsDictionary If set to true, the data will be returned as a dictionary where the keys are variable names and the values are the data for each variable. If set to false, the data will be returned in scalar or list format without the variable names.

Type bool (p. 1360)

Default Value False

ColumnMajor If set to true, the data will be returned in column-major order. If set to false, the data will be returned in row-major order.

Type bool (p. 1360)

Default Value True

EndIndex The end index for requesting a subset of the data (zero-based).

Type int (p. 1394)

Default Value -2147483647

SheetName Specifies the sheet name when the data contains multiple sheets.

Type string (p. 1438)

SheetQualifiers Used to pass in the qualifiers to select between multiple sheets with the same name. This is a dictionary of qualifiers and values.

Type Dictionary (p. 1375)<string (p. 1438), string (p. 1438)>

StartIndex The start index for requesting a subset of the data (zero-based).

Type int (p. 1394)

Default Value 0

Variables Names of the variables for which data is requested (string or list of strings).

Type Object (p. 1409)

Example

In this example, all data is requested for the given tabular data entity.

```
tabData1.GetData()
```

In this example, all data is requested in row-major order.

```
tabData1.GetData(ColumnMajor=False)
```

In this example, all data is requested in dictionary format.

```
tabData1.GetData(AsDictionary=True)
```

In this example, data for variables Density and Temperature is requested in dictionary format.

```
tabData1.GetData(Variables=["Density", "Temperature"], AsDictionary=True)
```

GetTemplate

No details are provided for this entry.

Return No details are provided for this entry.

Type DataReference (p. 1371)

Required Arguments

TemplateName No details are provided for this entry.

Type string (p. 1438)

Optional Arguments

Solver No details are provided for this entry.

Type string (p. 1438)

GetUnitsDisplaySettings

Gets the value of units display settings ("DisplayValuesAsDefined" or "DisplayValuesInProjectUnits").

Return The current value of the Units display setting.

Type string (p. 1438)

GetUnitSystem

Gets a reference to the UnitSystem entity of the given internal name.

Return A reference to the UnitSystem data entity.

Type DataReference (p. 1371)

Required Arguments

UnitSystemName The internal name of the unit system.

Type string (p. 1438)

GetUnitSystemsNames

Gets the internal names of all currently defined unit systems (Predefined and Custom Unit Systems).

Return A list of the internal names of currently defined unit systems.

Type List (p. 1400)<string (p. 1438)>

GetUserFilesDirectory

Gets the current user_files directory.

Return The full path to the user_files directory.

Type string (p. 1438)

GetUserPathRoot

Gets the current user path root.

Return The user path root.

Type string (p. 1438)

ImportFile

Imports a file into the project. The result of the import varies, dependent upon the file type's defined import behavior. Usually a corresponding system is created in the Project Schematic. Sometimes, the file is simply added to the Files pane. To be imported, a file's type must be registered and the ImportFileCommandName property must be non-null. Use FileTypeRepository.RegisterFileTypeTemplate or FileTypeTemplateRecord.ImportFileCommandName to specify the command to be invoked when importing a file of a given type.

Required Arguments

FilePath The full path to the file to import.

Type string (p. 1438)

Optional Arguments

FileType The reference to the type of the file to import. If not specified, the type will be inferred from the file's extension.

Type DataReference (p. 1371)

ImportJournalVariables

Imports any variables defined in the currently-recorded journal file into the scripting environment so they can be accessed in the command window.

ImportProjectInternal

Importing a project into the current project. There are several restrictions / limitations to this

Required Arguments

FilePath Path to the project to import.

Type string (p. 1438)

ImportUnitSystem

Imports a unit system from a Units .xml file.

Return A reference to the created UnitSystem data entity.

Type DataReference (p. 1371)

Required Arguments

FilePath The full path to the file to be imported.

Type string (p. 1438)

Optional Arguments

UnitSystemName The internal name of the unit system to be created.

Type string (p. 1438)

IsProjectUpToDate

No details are provided for this entry.

Return No details are provided for this entry.

Type bool (p. 1360)

Optional Arguments

IncludeParametricComponents No details are provided for this entry.

Type bool (p. 1360)

MergeRsmProject

Merging multiple project tasks solved from RSM design point updates into the current project. This command should *NOT* be used to merge any un-related projects. Solved projects may be either intermediate or final results.

Required Arguments

MergingProjects The dictionary contains full file path of base project of the RSM project as key, and a dictionary of the full file path to the updated RSM project and its design points to be merged into the current project as value. The file paths can be either standard wbpz file or partial wppz file or wbpj file. If the project is a standard wbpz or partial wppz archive, it will be unpacked to a sub-directory (whose name is the same as archive file name) under the same directory as the wbpz directory first before merging.

Type Dictionary (p. 1375)<string (p. 1438), Dictionary (p. 1375)<string (p. 1438), List (p. 1400)<DataReference (p. 1371)>>>

Optional Arguments

DoRevertibleSave Whether do revertible save or not

Type bool (p. 1360)

Default Value True

KeepFilesInRsmProject Determines whether we should keep the project files from the (un-archived) RSM project after it is merged into the current project. Note that keeping the RSM project will significant slow down the project file merging performance and will require more disk space. If this option is false, it is up to the caller to backup the original project first. Also note that if the ProjectFilePath is a *.wbpz file, it will not be affected.

Type bool (p. 1360)

Default Value True

Overwrite If the ProjectFilePath is an archive file (*.wbpz), and if the same unpacked project already exists on the same directory, this flag determines whether we should overwrite it.

Type bool (p. 1360)

SaveFirst Whether to save the project before merging.

Type bool (p. 1360)

Example

The following example illustrates how this command may be used to merge all Design Points from RSM-solved projects into the current project. Existing Rsm Project files are retained. Any existing

archive with the same name in the destination project will be overwritten. The project will be first saved using a revertible save.

```
MergeRsmProject(MergingProjects={(r"Path_to_RSM_Project", {(r"Path_to_current_project", {Updated DesignPoi
                KeepFilesInRsmProject=True,
                Overwrite=True,
                SaveFirst=True,
                DoRevertibleSave=True)
```

MoveFile

Moves a file to a new directory. Either the File argument or the FilePath argument must be specified.

Required Arguments

NewDirectoryPath The full path to the destination directory.

Type string (p. 1438)

Optional Arguments

BackUp Specifies whether to back up the file before the move. This optional argument's default value is true.

Type bool (p. 1360)

Default Value True

File A reference to the file to be moved. A file reference can be obtained from commands and queries like RegisterFileCommand and GetRegisteredFileQuery.

Type DataReference (p. 1371)

FilePath A full path to the file to be moved.

Type string (p. 1438)

Example

This example illustrates a path-based file move procedure. The specified file will be moved to the specified destination directory path.

```
MoveFile(FilePath=r"C:\Users\anyuser\myTextFile.txt",
         NewDirectoryPath=r"C:\Users\anyuser\newfolder")
```

This second example illustrates a reference-based file move procedure. The specified file will be moved to the specified destination directory path.

```
file1 = GetRegisteredFile(FilePath=r"C:\path_to_file\file.txt")
MoveFile(SourceFile=file1,
         NewDirectoryPath=r"C:\Users\anyuser\newfolder")
```

This final example also illustrates a reference-based file move procedure. The specified file will be moved to the specified destination directory path. The file will not be backed up prior to the move.

```
file1 = GetRegisteredFile(FilePath=r"C:\path_to_file\file.txt")
MoveFile(SourceFile=file1,
         NewDirectoryPath=r"C:\Users\anyuser\newfolder",
         Backup=False)
```

Open

Opens a Workbench project from the specified .wbpj file.

Required Arguments

FilePath The full path to the project file to open.

Type string (p. 1438)

Refresh

No details are provided for this entry.

RegisterFile

Registers the specified file with the project. FilePath argument must be specified.

Return Holds the reference to the registered file as the output value.

Type DataReference (p. 1371)

Optional Arguments

FilePath A full path to the file to be registered.

Type string (p. 1438)

FileType The reference to the type of the file to register. If not specified, the type will be inferred from the file's extension.

Type DataReference (p. 1371)

Example

This example illustrates a path-based file registration procedure. The specified file will be registered and the resulting file reference will be returned.

```
file1 = RegisterFile(FilePath=r"C:\Users\anyuser\myTextFile.txt")
```

This second example illustrates a file type based file registration procedure. The specified file will be registered with the corresponding (supplied) file type and the resulting file reference will be returned.

```
fileType1 = GetFileType(Name="MyFileType")
file1 = RegisterFile(FilePath=r"C:\Users\anyuser\myFile.abc",
                     FileType=fileType1)
```

RemovePropertyTable

Remove a table reference from a property to prevent its use in an expression. If the property does not support an expression an error will occur.

Required Arguments

Index The integer which uniquely identifies the Table for use in the expression of the Property.

Type uint (p. 1446)

Parent The DataReference of the object which has the Property that the Table should be removed from.

Type DataReference (p. 1371)

Property The String which provides the path of the property on the Parent.

Type string (p. 1438)

Example

```
tbl1 = pressure1.CreateTable(Columns=[[“X”, “Length”], [“Y”, “Length”], [“Pressure”, “Pressure”]])
bc1.AddPropertyTable(Property=“Magnitude”, Table=tbl1, Independents=[“X”, “Y”], Dependents=[“Pressure”])
# In the table function the x and y refer to field variables
bc1.SetPropertyExpression(Entity=bc1, Name=“Magnitude”, Expression=“table1(x, y)”)
bc1.RemovePropertyTable(Property=“Magnitude”, Index=1);
```

RenameFile

Renames a file. Either the File argument or the FilePath argument must be specified.

Required Arguments

NewName The new file name, excluding the directory path. To change the directory, see MoveFileCommand.

Type string (p. 1438)

Optional Arguments

BackUp Specifies whether to back up the file before renaming it. This optional argument's default value is true.

Type bool (p. 1360)

Default Value True

File A reference to the file to be renamed. A file reference can be obtained from commands and queries like RegisterFileCommand and GetRegisteredFileQuery.

Type DataReference (p. 1371)

FilePath A full path to the file to be renamed.

Type string (p. 1438)

Example

This example illustrates a path-based file rename procedure. The specified file will be renamed with the specified new name.

```
RenameFile(FilePath=r"C:\Users\anyuser\myTextFile.txt",
         NewName="myTextFileRENAMED.txt")
```

This second example illustrates a reference-based file rename procedure. The specified file reference will be renamed with the specified new name.

```
file1 = GetRegisteredFile(FilePath=r"C:\path_to_file\file.txt")
RenameFile(File=file1,
         NewName="fileRENAMED.txt")
```

This final example also illustrates a reference-based file rename procedure. The specified file will be renamed with the specified new name. The file will not be backed up prior to the move.

```
file1 = GetRegisteredFile(FilePath=r"C:\path_to_file\file.txt")
RenameFile(File=file1,
         NewName="fileRENAMED.txt",
         Backup=False)
```

RenameUnitSystem

Renames a unit system.

Required Arguments

UnitSystemName The name of the unit system to be changed.

Type string (p. 1438)

UnitSystemNewName The new name of the unit system.

Type string (p. 1438)

Reset

Resets Workbench to an empty project. Any unsaved changes in the current project are lost.

Resolve

A command used to resolve the data object to make it consistent with the rest of the data model

Required Arguments

Entity The entity to resolve.

Type DataReference (p. 1371)

ResumeBackgroundDesignPointUpdate

No details are provided for this entry.

Optional Arguments

Session No details are provided for this entry.

Type DataReference (p. 1371)

ResumeDesignPointUpdates

No details are provided for this entry.

RunScript

Executes a Workbench script file from the specified location.

Required Arguments

FilePath Full path to the script file to execute.

Type string (p. 1438)

Save

Saves the current project to disk.

Optional Arguments

FilePath The full path of the project file to be saved.

Type string (p. 1438)

Overwrite Whether to overwrite the project save location if it exists.

Type bool (p. 1360)

SetDesignPointUpdateOption

No details are provided for this entry.

Required Arguments

UpdateOption No details are provided for this entry.

Type JobRunMode (p. 1397)

SetProjectUnitSystem

Sets the unit system for the current project.

Return A reference to the UnitSystem entity that is now the project unit system.

Type DataReference (p. 1371)

Required Arguments

UnitSystemName The internal name of the unit system to be used in the project.

Type string (p. 1438)

SetQuantityUnitForUnitSystem

Sets the unit for a Quantity in the specified unit system. Changing Base or Common units will change derived units if appropriate. Note: Quantity unit for a predefined unit system cannot be changed. Note: Only consistent units (SI or US Customary) are allowed.

Return A dictionary of the Quantity names and new units for all resulting quantity changes. This will be more than just the target quantity if a base unit is changed.

Type Dictionary (p. 1375)<string (p. 1438), string (p. 1438)>

Required Arguments

QuantityName The name of the Quantity to be changed.

Type string (p. 1438)

Unit The new Unit for the quantity.

Type string (p. 1438)

UnitSystemName The internal name of the unit system to be changed.

Type string (p. 1438)

Example

The following example illustrates the setting of a quantity unit in a given unit system.

```
changedUnits = SetQuantityUnitForUnitSystem(UnitSystemName="myUnitSystem",
                                            QuantityName="Temperature",
                                            Unit="C")
```

SetScriptVersion

Sets the command API version used when executing a script.

Required Arguments

Version The command API version required to run the script. This string can either a 2 or 3 part version string.

Type string (p. 1438)

SetTableData

SetData will overwrite the data in a table beginning at the specified location.

Required Arguments

Column An Integer index or the String name of the column (case-sensitive) to begin setting data. The first column has an index of 0 (zero).

Type Object (p. 1409)

Data A String or List of Strings which specifies the data to set at the given row and column of the table. The String must be a valid Quantity string in the form "value [units]". If a List is used see the DataOrder argument for behavior.

Type List (p. 1400)<Quantity (p. 1422)>

Row The Integer index of the row of the table to begin setting data. The first row has an index of 0 (zero).

Type int (p. 1394)

Table The DataReference of the table which will have data changed.

Type DataReference (p. 1371)

Optional Arguments

DataOrder An enum of Row or Column which indicates if the Data is entered in a row-major or column-major order. The default is Row.

When the DataOrder is Row the data is set beginning at the specified position and continues in the row. If the operation for the List of data would exceed the number of columns an error will occur.

When the DataOrder is Column the data is set beginning at the specified position and continues in the column. If the operation for the List of data would exceed the number of rows an error will occur.

Type DataOrder (p. 1371)

Default Value Row

Example

This example illustrates setting data in a table.

```
# NameSpace is the Addin namespace in use (e.g. Study, EngData, etc.)  which may have its specific CreateT
# drParent is the DataReference of an object that has been previously created
table1 = NameSpace.CreateTable(Parent=drParent, Columns=[("Temperature","Temperature"),("Displacement","Le
table1.AddRow(Data=["100 [C]","1.0 [m]"])
table1.AddRow(Data=["200 [C]",".02 [m]"])
table1.AddRow(Data=["300 [C]",".06 [m]"])
tableData = ["100 [C]","0.0 [m]"]
table1.SetData(Row=0, Column=0, Data=tableData)
```

SetTabularData

Set tabular data associated with the data entity.

Required Arguments

TabularData Data entity that has associated tabular data.

Type DataReference (p. 1371)

Optional Arguments

Data Sets the data using a dictionary form. The keys are the variable names and the values are the data. The use of this argument is mutually exclusive with "Values" and "Variables".

Type Dictionary (p. 1375)<string (p. 1438), List (p. 1400)<Object (p. 1409)>>

Index Specifies the starting location used to set the data (zero-based). A value of -1 indicates that the data should be appended to the existing data.

Type int (p. 1394)

Default Value 0

SheetName Specifies the sheet name when the data contains multiple sheets.

Type string (p. 1438)

SheetQualifiers Used to pass in the qualifiers to select between multiple sheets with the same name. This is a dictionary of qualifiers and values.

Type Dictionary (p. 1375)<string (p. 1438), string (p. 1438)>

Values List of data values set in conjunction with the "Variables" parameter. This parameter and the "Data" parameter are mutually exclusive.

Type List (p. 1400)<List (p. 1400)<Object (p. 1409)>>

Variables Names of the variables for which data is being set. This parameter and and the "Data" parameter are mutually exclusive.

Type List (p. 1400)<string (p. 1438)>

Example

```
#
# Create a new Engineering Data System and access Structural Steel
#
template1 = GetTemplate(TemplateName="EngData")
system1 = template1.CreateSystem()
engineeringData1 = system1.GetContainer(ComponentName="Engineering Data")
matl1 = engineeringData1.GetMaterial(Name="Structural Steel")
#
# Change the value of a simple single-valued property
#
matlProp1 = matl1.GetProperty(Name="Density")
SetTabularData(matlProp1,
                  Variables="Density",
                  Values="8500 [kg m^-3]")
#
# Set Temperature-dependent data for Elasticity based
# on lists of variables and values.
matlProp2 = matl1.GetProperty(Name="Elasticity")
temperature = ["400 [K]", "600 [K]", "800 [K]"]
E = ["2e5 [MPa]", "1.9e5 [MPa]", "1.6e5 [MPa]"]
SetTabularData(matlProp2,
                  Variables = ["Temperature","Young's Modulus"],
                  Values = [temperature, E])
#
# Change the Temperature for the second table entry.
#
SetTabularData(matlProp2,
                  Index = 1,
                  Variables = "Temperature",
                  Values = "625 [K]")
#
# Set a list for Poisson's Ratio starting at the second table entry.
#
SetTabularData(matlProp2,
                  Index = 1,
                  Variables = "Poisson's Ratio",
                  Values = [0.3, 0.3])
#
# Set Temperature-dependent property data for the Coefficient of Thermal Expansion
# using a dictionary. The dictionary key is the Variable name,
# followed by the list of values for the variable.
#
matlProp3 = matl1.GetProperty(Name="Coefficient of Thermal Expansion")
newData = {"Temperature": ["200 [F]","400 [F]","600 [F]","800 [F]","1000 [F]"],
"Coefficient of Thermal Expansion" : ["6.3e-6 [F^-1]", "7.0e-6 [F^-1]",
                                      "7.46e-6 [F^-1]", "7.8e-6 [F^-1]",
                                      "8.04e-6 [F^-1]"]}
SetTabularData(matlProp3,
                  SheetName="Coefficient of Thermal Expansion",
                  SheetQualifiers={"Definition Method": "Secant", "Behavior": "Isotropic"},
                  Data = newData)
```

SetTabularDataQualifier

Changes the values of a specifiec qualifier in a data table.

Required Arguments

Qualifier The Qualifier to Set.

Type string (p. 1438)

TabularData Data Entity that can be bound to ITabularDataWrite

Type DataReference (p. 1371)

Value The new value.

Type string (p. 1438)

Optional Arguments

SheetName The name of the tabular data sheet that contains the qualifier.

Type string (p. 1438)

SheetQualifiers SheetQualifiers can be used to pass in the qualifiers to select between multiple sheets witht the same name. This is a dictionary of the Qualifier and its Value.

Type Dictionary (p. 1375)<string (p. 1438), string (p. 1438)>

VariableName The name of the Variable that contains the qualifier to be changed.

Type string (p. 1438)

VariableQualifiers VariableQualifiers can used to pass in the qualifiers to select between multiple variables witht the same name. This is a dictionary of the Qualifier and its Value.

Type Dictionary (p. 1375)<string (p. 1438), string (p. 1438)>

Example

The following example changes the 'Derive From' setting within an Isotropic Elasticity material property to be "Bulk Modulus and Poisson's Ratio".

```
matl1 = engineeringData1.GetMaterial(Name="Structural Steel")
matlProp1 = matl1.GetProperty(Name="Elasticity")
SetTabularDataQualifier(matlProp1,
   SheetName="Elasticity",
   Qualifier="Derive from",
   Value="Bulk Modulus and Poisson's Ratio")
```

SetUnitsDisplaySettings

Sets units display settings.

Required Arguments

DisplaySettings The new value to be used for Unit display settings. Allowed values are:

DisplayValuesAsDefined
DisplayValuesInProjectUnits

Type string (p. 1438)

SetUserPathRoot

Sets the current user path root. The root path facilitates portability of session journals by allowing relative rather than absolute paths to be recorded. During command journal replay, paths are reconstructed using the active user path root setting.

Required Arguments

DirectoryPath The new root directory path.

Type string (p. 1438)

Unarchive

Restores a project from an archive to the specified location.

Required Arguments

ArchivePath The project archive file path to open.

Type string (p. 1438)

Overwrite Whether the project should be overwritten if it already exsits.

Type bool (p. 1360)

Optional Arguments

ProjectPath The path to which the archived project will be uncompressed. If not specified, it will be extracted to a directory under "ProjectTemporaryFilesFolder" set by user preference. If the "ProjectTemporaryFilesFolder" is not set, it will be extracted to a directory under system temp directory.

Type string (p. 1438)

Example

This example illustrates an unarchive procedure. The specified archive will be un-archived to the specified project file path. If the project file path already exists, it will be overwritten since Overwrite=True.

```
Unarchive(ArchivePath=r"C:\Users\anyuser\myProjectArchive.wbpz",
          ProjectFilePath=r"C:\Users\anyuser\myProject.wbpj",
          Overwrite=True)
```

UpdateAllDesignPoints

No details are provided for this entry.

Return No details are provided for this entry.

Type DataReference (p. 1371)

Optional Arguments

CannotCompleteBehavior No details are provided for this entry.

Type UpdateErrorBehavior (p. 1446)

Default Value Continue

DesignPoints No details are provided for this entry.

Type List (p. 1400)<DataReference (p. 1371)>

ErrorBehavior No details are provided for this entry.

Type UpdateErrorBehavior (p. 1446)

Default Value Continue

Parameters No details are provided for this entry.

Type List (p. 1400)<DataReference (p. 1371)>

ReupdateInterrupted No details are provided for this entry.

Type bool (p. 1360)

Default Value False

Example

Required example does not exist.

Update

No details are provided for this entry.

Optional Arguments

ContinueCalculation No details are provided for this entry.

Type bool (p. 1360)

Default Value False

UpdateDesignPointsInSteps

No details are provided for this entry.

Required Arguments

UpdateOptions No details are provided for this entry.

Type DpUpdateOptions (p. 1379)

UpdateSystems

No details are provided for this entry.

Required Arguments

Systems No details are provided for this entry.

Type DataReferenceSet (p. 1371)

Sherlock

No details are provided for this entry.

TransferToMechanical

Create a Mechanical System based off the component analysis type

Required Arguments

Container Reference to the data container

Type DataContainerReference (p. 1371)

SherlockCommon

No details are provided for this entry.

ImportJournal

Import a journal file

Required Arguments

Container Reference to the data container

Type DataContainerReference (p. 1371)

JournalFilePath Path to the project file

Type string (p. 1438)

Open3DViewer

Opens the Sherlock 3D Viewer on a particular board and analysis in the project

Required Arguments

Analysis Analysis Name

Type string (p. 1438)

Board Board Name

Type string (p. 1438)

Container Reference to the data container

Type DataContainerReference (p. 1371)

Example

Required example does not exist.

Open2DViewer

Import a journal file

Required Arguments

Board Reference to the data container

Type string (p. 1438)

Container Reference to the data container

Type DataContainerReference (p. 1371)

SaveProject

Update the project path in Sherlock

Required Arguments

Container Reference to a data container

Type DataContainerReference (p. 1371)

Study

This namespace holds top-level commands and queries related to Study

CreatePredefinedGroup

Create a new pre-defined task group based on the given task group name.

Return The newly created group

Type DataReference (p. 1371)

Required Arguments

System The system where the task group component should be created.

Type DataReference (p. 1371)

Type The type of the new group

Type string (p. 1438)

CreateTask

Create a task, given the name for the task type.

Return The component created for the task.

Type DataReference (p. 1371)

Required Arguments

System The system where the task component should be created.

Type DataReference (p. 1371)

Type The task type to create.

Type string (p. 1438)

Optional Arguments

Input Optional, the task component which should be an upstream source for the created component.

Type DataReference (p. 1371)

Output Optional, the task component which should be downstream of the created component.

Type DataReference (p. 1371)

Example

This example details how to create a Modeling task and a Physics Defintion task where the Modeling task is an input to the Physics Definition task.

```
system1 = GetSystem(Name="Study")
geometryModelingComponent1 = Study.CreateTask(
  Type="Geometry Modeling",
  System=system1)
physicsDefinitionComponent1 = Study.CreateTask(
  Type="Physics Definition",
  System=system1,
  Input=geometryModelingComponent1)
```

CreateGroup

Create a new task group based on the given task group name.+

Return The newly created group

Type DataReference (p. 1371)

Required Arguments

Name The name of the new group

Type string (p. 1438)

EvaluateResults

Evaluates the a group of Results to update their outputs based on the current solution

Required Arguments

Entities The result objects to evaluate

Type DataReferenceSet (p. 1371)

GetAllEngineeringDataInProject

Find all engineering data objects in the project.

Return All engineering data objects in the project

Type DataReferenceSet (p. 1371)

GetAllLeafTasksInProject

Returns all leaf tasks in the project.

Return All leaf tasks in the project.

Type DataReferenceSet (p. 1371)

GetAllObjectsOfType

Returns all the objects of a given type for the study

Return A DataReferenceSet with the objects of the given type

Type DataReferenceSet (p. 1371)

Required Arguments

Type The type for which the objects are returned

Type string (p. 1438)

GetAllRootTasksInProject

Returns all root tasks in the project.

Return All root tasks in the project.

Type DataReferenceSet (p. 1371)

GetAllTaskObjectsOfType

For a given task, returns all the objects of a given type

Return A DataReferenceSet with the objects of the given type

Type DataReferenceSet (p. 1371)

Required Arguments

Task The task for which the objects are returned

Type DataReference (p. 1371)

Type The type for which the objects are returned

Type string (p. 1438)

GetAllTasksInProject

Returns all tasks in the project.

Return All tasks in the project.

Type DataReferenceSet (p. 1371)

GetTaskObjectForTask

Return the task object for a given task.

Return The anchor object associated with the task.

Type DataReference (p. 1371)

Required Arguments

Task The task to query.

Type DataReference (p. 1371)

GetCurrentModelForExtension

Command to be invoked from within a callback in order to get the current model in use for the extension The command SetCurrentModelForExtension must be invoked in order to set the model

Return The current model

Type DataReference (p. 1371)

GetInputMesh

If available for a task, return the input mesh; otherwise null.

Return The input mesh.

Type DataReference (p. 1371)

Required Arguments

Task The task to query.

Type DataReference (p. 1371)

GetInputModel

If available for a task, return the input model; otherwise null.

Return The input model.

Type DataReference (p. 1371)

Required Arguments

Task The task to query.

Type DataReference (p. 1371)

GetObjectOfTypeAndName

Returns all the objects of a given type and name for the study. Include the "@" prefix if searching by display text

Return A DataReferenceSet with the objects of the given type

Type DataReference (p. 1371)

Required Arguments

Name The name of the object. Use @ prefix if searching by display text

Type string (p. 1438)

Type The type for which the objects are returned

Type string (p. 1438)

GetOutputMesh

If available for a task, return the output mesh; otherwise null.

Return The output mesh.

Type DataReference (p. 1371)

Required Arguments

Task The task to query.

Type DataReference (p. 1371)

GetOutputModel

If available for a task, return the output model; otherwise null.

Return The output model.

Type DataReference (p. 1371)

Required Arguments

Task The task to query.

Type DataReference (p. 1371)

GetReferences

Returns a list of reference strings of the specified model components.

Return A List of the reference strings of the specified model components.

Type List (p. 1400)<string (p. 1438)>

Required Arguments

ModelComponents The DataReferenceSet of the model components of interest.

Type DataReferenceSet (p. 1371)

GetReferenceString

Returns a reference string for a user object. This is a special case query, for those objects that contain references, not stored in LocationSets. For now, works on Locator objects only, and is to be used to retrieve the surface area of bounded surfaces: Sphere, Torus, and CappingSurface only.

Return Reference string for the specified object.

Type string (p. 1438)

Required Arguments

ObjectReference The DataReference for the specified object that is assumed to be Analytic.

Type DataReference (p. 1371)

GetTasksForTaskGroup

Returns the tasks for a given task group

Return The tasks for the task group

Type DataReferenceSet (p. 1371)

Required Arguments

TaskGroup The task group for which to search.

Type DataReference (p. 1371)

GetTaskUserObjectBelongsTo

Returns a data reference to a Task that the provided UserObject maps to. May return null.

Return The input tasks for the given task.

Type DataReference (p. 1371)

Required Arguments

UserObjectReference The user object to map to a task.

Type DataReference (p. 1371)

InitializeGuidedSimulations

Initializes data when executing Guided Simulations.

InsertPredefinedGroup

Inserts a new task group based on the given task group name.

Return The newly created group

Type DataReference (p. 1371)

Required Arguments

Input The Task or TaskGroup DataReference which should be upstream of the created component.

Type DataReference (p. 1371)

Output The Task or TaskGroup DataReference which should be downstream of the created component.

Type DataReference (p. 1371)

System The system where the task component should be created.

Type DataReference (p. 1371)

Type The name of the new group

Type string (p. 1438)

Example

This example will create a connection between two tasks and then insert a Physics Task Group between the two

```
system1 = GetSystem(Name="Study")
solvePhysicsComponent1 = Study.CreateTask(
  Type="Solve Physics",
  System=system1)

resultsEvaluationComponent1 = Study.CreateTask(
  Type="Results Evaluation",
  System=system1,
  Input=solvePhysicsComponent1)
physicsSolutionGroup1 = Study.InsertPredefinedGroup(
  Type="Physics Solution",
  System=system1,
  Input=solvePhysicsComponent1,
  Output=resultsEvaluationComponent1)
```

InsertTask

Create a task, given the name for the task type.

Return The component created for the task.

Type DataReference (p. 1371)

Required Arguments

Input The Task or TaskGroup DataReference which should be an upstream source for the created component.

Type DataReference (p. 1371)

Output The Task or TaskGroup DataReference which should be downstream of the created component.

Type DataReference (p. 1371)

System The system where the task component should be created.

Type DataReference (p. 1371)

Type The task type to create.

Type string (p. 1438)

Example

This example will create two Tasks and insert a third Tasks between the two

```
system1 = GetSystem(Name="Study")
importComponent1 = Study.CreateTask(
  Type="Import",
  System=system1)
physicsDefinitionComponent1 = Study.CreateTask(
  Type="Physics Definition",
  System=system1,
  Input=importComponent1)
meshingComponent1 = Study.InsertTask(
  Type="Meshing",
  System=system1,
  Input=importComponent1,
  Output=physicsDefinitionComponent1)
```

RemovePropertyTables

Removes all property tables from an expression property.

Required Arguments

Parent The DataReference of the object that has the Propety parameter which tables will be removed from.

Type DataReference (p. 1371)

Property The String which provides the path of the property on the Parent.

Type string (p. 1438)

StudyUI

This namespace holds top-level User Interface based commands and queries related to Study

SetSelectionTo

Action invoked from a script in order to force visibility of model based upon a given user object.

Required Arguments

Entity Entity to set visibility to

Type DataReference (p. 1371)

HideBodies

Action invoked from a script to a hide one or more bodies

Required Arguments

Bodies A list of the ids of bodies to be hidden. For example ["BODY.1", "BODY.2"]

Type List (p. 1400)<string (p. 1438)>

HideFaces

Action invoked from a scrip to a hide one or more faces

Required Arguments

Faces A list of the ids of the faces to be hidden. For example ["FACE.1", "FACE.2"]

Type List (p. 1400)<string (p. 1438)>

SaveImage

Action invoked from a script to save an image to a file.

Required Arguments

FilePath The path to the file where the image is saved

Type string (p. 1438)

Optional Arguments

Size The resolution of the image - Supported values are "640x480", "800x600", "1024x768", "1280x720", "1280x1024", "1400x1120", "1600x1400", "1920x1080", "2048x1536", "3840x2160"

Type String[] (p. 1438)

SetConvergenceDisplayMessage

Action invoked from a script in order to control the display of convergence messages

Required Arguments

IsEnabled Set to False to disable display of convergence messages

Type bool (p. 1360)

SetSelectionFilter

Action invoked from within a script in order to set the selection filter.

Required Arguments

Filter A string representing the type of filter to be set. Valid values are "Body", "Face", "Edge"

Type string (p. 1438)

ShowAll

Action invoked from a scrip to reset visibility for all entities

Turbo

This namespace holds top-level commands and queries related to Turbo Systems

CreateDesignPoints

This command class creates multiple new design points and returns a list of those design points

```
Suppose you want to create 5 design points with a single command.
This can be achieved by the following code:
MyDPlist = Turbo.CreateDesignPoints(
NumberOfPoints=5,
Retained=True)
Here the design points are also flagged as retained.
```

Return List of DataReferences to the new design point entities.

Type List (p. 1400)<DataReference (p. 1371)>

Required Arguments

NumberOfPoints The specified number of design points to create

Type int (p. 1394)

Optional Arguments

Exported Indicates that the newly created design points will be exported.

Type bool (p. 1360)

Default Value False

Retained Indicates that the newly created design points will be retained.

Type bool (p. 1360)

Default Value False

Example

Required example does not exist.

CreateDP

This command class creates a new design point, ensuring the Parameter Manager GUI is updated correctly

Return A DataReference to the new design point entity.

Type DataReference (p. 1371)

Optional Arguments

Exported Indicates that the newly created design point will be exported.

Type bool (p. 1360)

Default Value False

Retained Indicates that the newly created design point will be retained.

Type bool (p. 1360)

Default Value False

DeleteDP

This command class deletes the specified design point, ensuring the Parameter Manager GUI is updated correctly

Required Arguments

DesignPoint The design point data reference.

Type DataReference (p. 1371)

SetParameterInDP

This command class updates the parameter manager with the design point defined by the Performance Map.

```
    The following code uses SetParameterInDP to set the parameter values for mass flow rate (P1) and percentage sp
```

```
designPoint1 = Parameters.GetDesignPoint(Name="1")
paramMassFlowRate = Parameters.GetParameter(Name="P1")
Turbo.SetParameterInDP(
DesignPoint=designPoint1,
Parameter=paramMassFlowRate,
Expression="2 [kg s^-1]")
paramPercentSpeed = Parameters.GetParameter(Name="P2")
Turbo.SetParameterInDP(
DesignPoint=designPoint1,
Parameter=paramPercentSpeed,
Expression="90")
```

Required Arguments

DesignPoint The design point data reference.

Type DataReference (p. 1371)

Expression The string with the expression for the parameter.

Type string (p. 1438)

Parameter The input parameter data reference.

Type DataReference (p. 1371)

Example

Required example does not exist.

Data Types

Data Types

A breakdown of the types represented in this document

ActionMode

Enumeration with the available ROM action modes.

Possible Values

None	No action.
Production	Production mode, when a snapshot file is generated each time a design point is updated.
Consumption	Consumption mode, when a ROM project is used to evaluate the results.

AdaptKrigOutType

Enumeration for the Output Variable Combinations type when refining for the Kriging algorithm.

Possible Values

AK_MAXOUT	Maximum Output
AK_ALLOUT	All Outputs
AK_SUMOUT	Sum of Outputs

Aggressive

Used to specify whether mesh may deviate from geometry as a result of wrapper surface improvement. The default value is MeetTarget.

Possible Values

MeetTarget	Modify the geometry as needed to meet quality target.
MaintainGeometry	Compromise targets to maintain geometry.

AlgorithmType

Available contact detection algorithm types.

Possible Values

Unknown
ProximityBased
TrajectoryBased

AllowedContinuousValues

Enumeration of the possible continuous input parameter types.

Possible Values

Any	Any
ManufacturableValues	Manufacturable Values
SnapToGrid	SnapToGrid

AnalysisType

No details are provided for this entry.

Possible Values

StressLife
StrainLife

AnalysisType

3D/2D import option

Possible Values

AnalysisType_3D	Import all 3D objects
AnalysisType_2D	Import only 2D objects (The model must be in the x-y plane.)

APAMAdvancedSizeFunction

No details are provided for this entry.

Possible Values

ProgramControlled
Curvature
Proximity
ProximityCurvature

Fixed

APAMMidsideNodeOrder

Options for mid side nodes to determine the element order either linear or quadratic.

Possible Values

ProgramControlled
Dropped
Kept

APAMProximitySizeFunctionSources

Used to specify the source type to be used for size function calculation when Advanced Size Function is Proximity or Curvature and proximity. The default value is Faces.

Possible Values

Faces
Edges
FacesAndEdges

ApplicationMethod

Possible options for the applied force

Possible Values

TotalForce
ForcePerUnitArea
ForcePerUnitLength

ApplicationMethod

Possible options for the applying heat generation

Possible Values

TotalHeat
PerUnitVolume
Unknown

AssociationType

Construction strategy.

Possible Values

GLOBAL	One ROM for all the regions of a field.
LOCAL	One ROM per region of a field.

AutoTimeSteppingDefinedBy

Define Auto Time Stepping by Time or Substeps as available in MAPDL solver.

Possible Values

Substeps
Time

AxesRangeModes

Enumeration of the Automatic Range modes for output parameters.

Possible Values

OutputParameterMinMax	The range of the output parameter axis is determined from the minimum and maximum of the parameter (min-max search of DPs min-max)
ChartData	The range of the output parameter axis is determined by the min and max of the chart's data.

AxialDirectionDefinedBy

This enum is used for axial direction to be defined by either geometry selection or reference frame.

Possible Values

GeometrySelection
ReferenceFrame

Axis

The specification of allowed chart axes.

Possible Values

None
X_Axis
Y_Axis
Z_Axis

AxisDefineBy

Options for defining an axis

Possible Values

Direction
Point

AxisType

Options specifying which axis to define

Possible Values

XAxis
YAxis
ZAxis

BeamSectionOffsetType

Available section offset types for beams.

Possible Values

Centroid
ShearCenter
Origin
UserDefined

BeamSectionType

Available section types for beams.

Possible Values

Unknown
Circular
Rectangular
Quadrilateral
TubeShaped
ChannelShaped
ZShaped
LShaped
IShaped
TShaped
HatShaped
HollowRectanglar
ArbitraryShaped

MeshCrossSection

BiasMethod

Used to specify the bias option. Currently, this is always SmoothTransition.

Possible Values

SmoothTransition

BiasType

Used to specify the option for bias type. The default value is Constant.

Possible Values

Right
Left
CenterIn
CenterOut
Constant

BilinearIsotropicHardeningDependents

The Bilinear Isotropic Hardening dependents.

BladeLoftType

Enumeration of the possible blade lofting directions.

Possible Values

Streamwise
Spanwise

BladeType

Blade type for export to BladeGen

Possible Values

IGV
Rotor
OGV

IGV
Rotor
OGV

BladeType

Blade type for export to BladeGen

Possible Values

IGV	IGV
Rotor	Rotor
OGV	OGV

BMunitsType

BladeGen/BladeEditor units type

Possible Values

m	Create blade model in metres
cm	Create blade model in cm
mm	Create blade model in mm
inches	Create blade model in inches
ft	Create blade model in feet

BMunitsType

BladeGen/BladeEditor units type

Possible Values

mm	Create blade model in mm
inches	Create blade model in inches

BodyGroupingType

Type of geometry that is being attached.

Possible Values

MaterialNumber
MaterialNumberAndThickness
NoGrouping

BodyGroupingType

Type of geometry that is being attached.

Possible Values

MaterialNumber
MaterialNumberAndThickness
NoGrouping

bool

This type represents a Boolean value. Valid values are 'True' or 'False'.

BoundaryLayerAlgorithm

Used to specify if pre or post boundary layer algorithm should be used by the mesher. The default value is Pre.

Possible Values

Post
Pre

BoundaryLayerOption

Used to specify the way heights of the boundary layers are determined. The default value is SmoothTransition.

Possible Values

TotalThickness	Creates constant boundary layers using the values of Number of Layers and Growth Rate.
FirstLayerThickness	Creates constant bounadry layers using values of First Layer Height, Maximum Layers and Growth Rate.
SmoothTransition	Mesher ensures smooth rate of volume change using local tetrahedral element size.
FirstAspectRatio	Creates boundary layers using values of First Aspect Ratio, Maximum Layers and Growth Rate.
LastAspectRatio	Creates constant boundary layers using values of First Layer Height, Maximum Layers and Aspect Ratio.

BoxCushionType

Box cushion type options.

Possible Values

Uniform	Uniform cushion around box.
Nonuniform	Non-Uniform cushion around box.

CADImportParameterType

Types of parameters to be imported from CAD True and False ARE NOT enum values to be used, but there to help support the open of old databases.

Possible Values

CADImportParameterType_None	Do NOT import parameters
CADImportParameterType_Independent	Import editable parameters -- default value (like old true value)
CADImportParameterType_All	Import all paramters that match the filter, both independent and dependent (read-only)

CalculationFrequency

CalculationFrequency specifies the frequency to write the specified solution results of OutputType

Possible Values

AllTimePoints
LastTimePoint
EquallySpacedTimePoints
SpecifiedRecurrenceRate
Never

CalculationType

The available analysis types for an analysis.

Possible Values

Unknown	Indicates analysis was never set
None	Indicates analysis was set then cleared
Static	Computation of the system that is time independent.
TimeDependent	Computation of the system that is time dependent.
Modal	Computation of the dynamic response of a system under excitation
FrequencyResponse	Frequency domain analysis of a system
Optimization	Topology optimization analysis
EigenvalueBuckling	Eigenvalue buckling
All	All domains.

CandidatesColoringMethods

Enumeration of the Coloring methods used for Candidates chart.

Possible Values

ColoringPerPointType	The color is used to distinguish the different types of candidate points (Starting Points, Candidate Points, etc).

ColoringPerOutputNature	The color is used to distinguish the nature of the output values (Response Surface or Simulation).

CappingSurfaceConstructionMethod

Options for choosing selection method for capping surface.

Possible Values

CappingSurfaceFacesConstraint	Construct capping surface by capping all laminar loops of selected faces.
CappingSurfaceEdgesConstraint	Construct capping surface by capping all loops of selected edges.
CappingSurfaceLoopsConstraint	Construct capping surface by capping all specified loops.
CappingSurfaceVerticesConstraint	Construct capping surface by capping the loop formed by selected Vertices.

CasePrecision

Precision of FLUENT Session.

Possible Values

Single
Double

CCDTemplateType

Enumeration for the Template Type for CCD algorithm.

Possible Values

CCD_STANDARD_TEMPLATE	Standard
CCD_ENHANCED_TEMPLATE	Enhanced

CentralCompositeDesignType

Enumeration of the available design types for the Central Composite Design algorithm.

Possible Values

CCDTYPE_FACE_CENT	Face-Centered
CCDTYPE_ROT	Rotatable
CCDTYPE_VIF_OPT	VIF-Optimality
CCDTYPE_G_OPT	G-Optimality
CCDTYPE_AUTO	Auto Defined

ChartAxes

The possible chart axes.

Possible Values

XAxis	X axis
YAxis	Y axis
ZAxis	Z axis
XTopAxis	Secondary X axis drawn at the top of the chart.
YRightAxis	Secondary Y axis drawn at the right of the chart.
SweepAxis	An axis used to sweep over an additional parameter.

ChartColoringMethods

Enumeration of the Coloring methods used for Tradeoff and Samples charts.

Possible Values

ColoringPerFront	A different color for each Pareto fronts (several samples have the same color).
ColoringPerSample	A different color for each sample.

ChartStyle

Allowed styles of a multi-axis chart.

Possible Values

PCP	Parallel Coordinate Plot
Spider	Spider Plot

ChartType

Enumeration for the chart type.

Possible Values

ChartUnknown	Unknown
ChartResponse	Response
ChartTradeoff	Tradeoff
ChartSamples	Samples
ChartDistributions	Distributions
ChartCorrelation	Correlation
ChartSpiderResponses	Spider
ChartLocalSensitivity	Local Sensitivity

ChartSensitivities	Sensitivities
ChartStatistics	Statistics
ChartCorrelationScatter	Correlation Scatter
ChartDesignPointsParallel	Parameters Parallel
ChartDesignPointsCurves	Design Points vs. Parameters
ChartDetermination	Determination Matrix
ChartPredictedvsObserved	Predicted vs. Observed
ChartDeterminationHistogram	Determination Histogram
ChartConvergence	Convergence
ChartLocalSensitivityCurves	Local Sensitivity Curves
ChartCustom	Custom Chart
ChartCandidates	Candidates
ChartHistory	History
ChartConvergenceCriteria	History
ChartParameterRelationship	History
ChartPredictionErrorScatterPlot	Prediction Errors

ClearanceType

inpeller clearance type

Possible Values

Ratio	tip clearance specified as a ratio
User	tip clearance specified directly

ClearanceType

Enumeration of the tip clearance specification options.

Possible Values

None
RelativeLayer
AbsoluteLayer

ClearMessagesOption

Options when deleting messages.

Possible Values

KeepMessagesIfSourceMatched	Do not delete message if its source matches the specified one.
ClearMessagesIfSourceMatched	Delete message if its source matches the specified one.
ClearMessagesIfSourceNotMatched	Delete message if its source does not match the specified one.
IgnoreSource	Do not check source attribute.

CollisionAvoidance

Used to specify the strategy to be used for avoiding collisions between inflated surface meshes. The default value is StairStepping.

Possible Values

None	No check for layer collisions.
LayerCompression	Boundary layers are compressed in collision areas.
StairStepping	Prism layers are stair stepped to avoid collision and maintain the gap defined by gap factor.

Color

This type represents a 32-bit Red/Green/Blue/Alpha color.

When working with Workbench Scripting, a color is represented as a four entry string in the form "R G B A", where each entry is in the range 0-255. Alpha (A) is optional and will be set to 255 (opaque) if not supplied. For example:

```
renderStyle1.LineColor = "255 0 0"     # Sets the line to opaque Red.
renderStyle1.LineColor = "0 0 255 128" # Sets the line to translucent Blue.
```

ColorDistributionOption

Supported color distributions.

Possible Values

Linear
Logarithmic

ColorSpaceSetting

Options for color space.

Possible Values

Linear
sRGB

ComparePartsOnUpdateMethod

Compare parts on update options

Possible Values

ComparePartsMethod_None	Do NOT compare parts on update -- default value
ComparePartsMethod_Associatively	Compare parts using associative mechanism, if geometry interface is non-associative expect failures in compare
ComparePartsMethod_NonAssociatively	Compare parts using entity comparisons based on index of second model to original attach

ComparePartsTolerance

Compare parts on update tolerance options

Possible Values

ComparePartsTolerance_Loose	A greater loosening of the default tolerance
ComparePartsTolerance_Normal	Looser tolerance than default to allow some wiggle room for slight deviations
ComparePartsTolerance_Tight	Default, existing behavior

ComponentConfigType

List of available component configurations

Possible Values

ImpellerMainOnly	impeller (main blade only)
ImpellerOneSplitter	impeller (main blade + 1 splitter)
ImpellerTwoSplitters	impeller (main blade + 2 splitters)

ConductanceControlType

Switch to use program controlled conductance value or enter value.

Possible Values

ProgramControlled
Manual

ConnectionEndBehavior

Possible options for connection end behavior

Possible Values

kProgramControlled
kDeformable
kRigid

kCoupled

ConstraintHandlingType

Enumeration of the Constraint Handling types.

Possible Values

AsGoals	Relaxed constraint.
AsHardConstraints	Strict constraint.

ConstraintType

Enumeration of the possible optimization constraint types.

Possible Values

CT_NoPreference	No constraint defined.
CT_NearTarget	Equals target.
CT_LessThanTarget	Less than target.
CT_GreaterThanTarget	Greater than target.
CT_InsideBounds	Inside bounds.

ConstraintType

This enum is used to indicate how the member size is defined

Possible Values

ProgramControlled
Manual

ConstructionAlgorithm

Options for choosing construction algorithm for capping surface.

Possible Values

Delaunay	Construct capping surface by using delaunay triangulation.
FillHole	Construct capping surface by using hole filling algorithms.
ConvexHull	Construct capping surface by using convex hull approach.
Automatic	Construct capping surface by choosing appropriate construction algorithm.

ConstructionType

Construction type

Possible Values

FIXED_NB_MODE	Fixed number of modes
FIXED_ACCURACY	Fixed error

ContactConstraintType

Possible options for contact constraint type.

Possible Values

ProgramControlled
TargetNormalUncoupleUtoROT
TargetNormalCoupleUtoROT
InsidePinballCoupleUtoROT

ContactDetectionPoint

Possible options for contact detection point.

Possible Values

ProgramControlled
DetectOnGaussPoint
DetectNodalNormalFromContact
DetectNodalNormalToTarget
DetectNodalProjectedNormalFromContact

ContactPairOutputYesNo

Activate Contact Pair Output in MAPDL solution.

Possible Values

kNo
kYes

ContactPhysicsToTransfer

Method for selecting the physics type: Program controlled or Manual.

Possible Values

AllRelevant
Specified

ContactType

Type of contact for eg. Bonded, Frictional, Frictionless, etc.

Possible Values

Bonded
NoSeparation
Frictionless
Rough
Frictional

ConvergenceOption

No details are provided for this entry.

Possible Values

NumericalResiduals
AutoHalt
UserHalt

CoordinateInputMethod

Options for defininining vectors

Possible Values

Entry
Location

CoordinateSystemType

Enumeration to specify the coordinate system type for imported data.

Possible Values

Cartesian
Cylindrical

CorrelationAutoStopType

Enumeration of the correlation Auto Stop types.

Possible Values

ExecuteAllSimulations	Execute all Simulations.
EnableAutoStop	Enable Auto Stop: execute Simulations iteratively until the process converges or the max-

	imum number of simulation specified is reached.

CoupledAnalysisType

The valid coupling types

Possible Values

Undefined	This is internal option used to cache the user selection of analysis type. It cannot be specified by the user.
General	Define the coupling by time steps
Transient	Define the coupling by time intervals
Harmonic	Define the coupling by frequency intervals. The harmonic coupled analysis is not currently supported.

CreateIntersectingEdgesForOverlappingBodies

Used to specify whether intersecting edges should be created for overlapping bodies. The default value is No.

Possible Values

Yes
No

CreationMode

Allows clients of a generator to tag objects as created automatically by the program or created manually by the user in a consistent manner.

Possible Values

Automatic
Manual

CurrentSpecificationMethod

No details are provided for this entry.

Possible Values

TotalCurrent
CurrentPerUnitArea
Unknown

DampedSolverType

An enumeration for damped modal solver types as available from MAPDL solver.

Possible Values

ProgramControlled
FullDamped
ReducedDamped

DataContainerReference

A reference to a data container, similar to a DataReference referring to an entity.

DataObject

Implements the virtual data model concepts for native C# code.

DataOrder

An enum of Row or Column which indicates if the Data is entered in a row-major or column-major order. The default is Row.

Possible Values

Column
Column-major
Row
Row-major

DataReference

A Data Reference holds and manages a reference to a data entity in the Workbench data model.

DataReferenceSet

This type contains an ordered set of DataReferences. No modifications can be made to the contents.

DataTransferType

Enumeration of the data transfer options.

Possible Values

BGD
NDF

DataTypeEnumeration

This enumeration represents all of the tensor types supported by the MPC variable.

Possible Values

AType	An unknown type
Scalar	A scalar
VectorXY	A vector in x and y components
VectorYZ	A vector in y and z components
VectorXZ	A vector in x and z components
VectorXYZ	A vector in x, y, and z components
VectorRA	A vector in r and a components
VectorAZ	A vector in a and z components
VectorRZ	A vector in r and z components
VectorRAZ	A vector in r, a, and z components
VectorRI	A vector in r and i components
VectorIA	A vector in i and a components
VectorRIA	A vector in r, i, and a components
Tensor2XYZ	A 2D tensor
Tensor4XYZ	A 4D Tensor
SymTensor2XYZ	A 2D symmetric tensor
SymTensor4XYZ	A 4D symmetric tensor
AntisymTensor2XYZ	An anti-symmetric tensor

DateTime

This type represents a date and time. The default format for printing a DateTime object is "DD/MM/YYYY HH:MM:SS AMPM". Additional properties which can be examined on a DateTime object to extract extra detail include Year, Month, Day, Hour, Minute, Second and Date.

DefineBy

No details are provided for this entry.

Possible Values

ultStrength
SNCurve
Material

DefineBy

This enum is used to set how tolerance is specified. It can be program controlled or user can manual set the property value.

Possible Values

ProgramControlled
UserDefined

DefinitionState

State of the definition

Possible Values

Complete	Fully defined
Incomplete	Data are missing
Empty	Empty definition
ConnectionFailed	Connection with DCS has failed
Failed	Post Operation has failed

DeformationScaling

How much to deform the Results display

Possible Values

None
Actual
FiftyPercent
OneHundredPercent
TwoHundredPercent
FiveHundredPercent
FiveHundrenPercent
Custom

DelimiterType

Enumeration to specify the type of delimiter in imported data.

Possible Values

Comma
Semicolon
Space
Tab
UserDefined

DerivationType

The enum values for the Derivation property to use for the IsotropicElasticity values.

Possible Values

Unknown
YoungsModulusPoissonsRatio
ShearModulusPoissonsRatio
BulkModulusPoissonsRatio

YoungsModulusShearModulus
YoungsModulusBulkModulus
BulkModulusShearModulus

DerivativeApproximationType

Enumeration for the derivative approximation type.

Possible Values

DA_CentralDifference	Central Difference
DA_ForwardDifference	Forward Difference

DesignPointParameter

No details are provided for this entry.

DesignPointRetainedOrExportedUpdate

No details are provided for this entry.

Possible Values

Parameters
FullProject

DesignPointState

Indicates a design point parameter's current state.

Possible Values

NeverSolvedAndFailed	Parameter has never been successfully updated and the last update failed.
NeverSolved	Parameter has never been updated.
Failed	Parameter was previously UpToDate but last update failed.
OutOfDate	Parameter was previously UpToDate or Failed but has become out of date by either design point input value changes or a non-parametric change within in the project.
Interrupted	Parameter is updated but the update was interrupted
UpToDate	Parameter is successfully updated.

DesignPointUpdateOrder

No details are provided for this entry.

Possible Values

UpdateFromCurrentDesignPoint
UpdateDesignPointsInOrder

DesignTypes

Possible design goals for a topology optimization analysis

Possible Values

Strength
NaturalFrequency

DeterminationCoefficientChartModes

Enumeration of the Determination Coefficient chart modes.

Possible Values

Linear	Display linear determination coefficients.
Quadratic	Display quadratic determination coefficients.

Dictionary<Key, Value>

This type represents a data dictionary, where a Key is used to access an associated Value. When used in scripting, a dictionary is created or printed using the form

```
myDict = {key1:value1, key2:value2, ...}
```

Python functionality can be used to examine dictionary keys, test for key existence and perform other useful operations on dictionaries.

The following example shows the use of dictionaries in a Workbench script:

```
>>> template1 = GetTemplate(
        TemplateName="Fluid Flow",
        Solver="CFX")
>>> system1 = template1.CreateSystem()
>>> solution1 = system1.GetContainer(ComponentName="Solution")
>>> cfxSolutionProperties1 = solution1.GetCFXSolutionProperties()
>>> currentProps = cfxSolutionProperties1.GetEntityProperties()
>>> for key in currentProps.Keys:
        print "%s = %s" % (key, currentProps[key])

SolverCommandMode = Foreground
DisplayText = Solution Source
InitializationOption = CurrentSolutionData
LoadMResOptions = LastConfigOnly
ResultsFile = None
>>> myProps={"SolverCommandMode":"Background","InitializationOption":"InitialConditions"}
>>> cfxSolutionProperties1.SetEntityProperties(Properties=myProps)
      # The above is equivalent to:
      #   cfxSolutionProperties1.InitializationOption ="InitialConditions"
      #   cfxSolutionProperties1.SolverCommandMode="Background"
```

DiffuserType

diffuser type

Possible Values

Vaned	vaned diffuser
Vaneless	vaneless diffuser

DimensionsType

Enumeration to specify the dimensionality of imported data.

Possible Values

Dimension2D
Dimension3D

DirectionType

Identifiers for direction type.

Possible Values

NoMagnetization
Cartesian
Cylindrical
Spherical

DirectionType

The direction the streamline is moving, relative to inlets and outlets

Possible Values

Backward
Forward
Both

DirectionType

This enum is used to indicate the direction of the selected axis

Possible Values

AlongAxis
OppositeToAxis
BothDirection

DisplacementType

Possible options for displacement.

Possible Values

Unknown
TranslationAndRotation
TranslationOnly
RotationOnly

DisplayLevelType

Level of Display of Log File

Possible Values

Low	Low
Medium	Medium
High	High

DistributedMassType

No details are provided for this entry.

Possible Values

TotalMass
MassPerUnitArea

DistributionType

Enumeration of the possible parameter distribution types.

Possible Values

Uniform	Uniform
Triangular	Triangular
Gauss	Normal
TruncGauss	Truncated Normal
LogNorm	Lognormal
Exponential	Exponential
Beta	Beta

Weibull	Weibull

DoF

No details are provided for this entry.

Possible Values

None
X
Y
Z
All

DOFBehavior

This enum is used to set the translational and/or rotational degree of freedom as fixed or free.

Possible Values

Fixed
Free

DotStyles

Styles of dot symbols. Default is none

Possible Values

None
Ellipse
Rect
Diamond
Hexagon
Triangle
DTriangle
UTriangle
LTriangle
RTriangle
Cross
XCross
Star
Default

double

This type represents a double-precision floating point number.

DpUpdateOptions

No details are provided for this entry.

DurationType

The methods used to specify duration

Possible Values

NumberOfSteps	The coupling will end at a given number of steps
EndTime	The coupling will end at a given time

EdgeEdgeTreatment

No details are provided for this entry.

Possible Values

Both
ParallelOnly
PerpendicularOnly

EdgeSizingMethod

Used to specify if the edge sizing should be based on the element size or on the number of divisions.

Possible Values

ElementSize
EdgeDivisions

EffType

Impeller efficiency type

Possible Values

Automatic	Efficiencies calculated from correlations
Hydraulic	Hydraulic efficiency calculated. Volumetric, mechanical and overall pump efficiencies user defined.
Volumetric	Volumetric efficiency calculated. Hydraulic, mechanical and overall pump efficiencies user defined.
Mechanical	Mechanical efficiency calculated. Hydraulic, volumetric and overall pump efficiencies user defined.

Pump	Overall pump efficiency calculated. Hydraulic, volumetric and mechanical efficiencies user defined.

ElasticityType

No details are provided for this entry.

Possible Values

IsotropicElasticity
OrthotropicElasticity
NeoHookeanHyperElasticity

ElementMidsideNodes

Used to define whether elements are created with or without midside nodes. The default value is UseEngineeringIntent.

Possible Values

UseEngineeringIntent	Determine whether midside nodes are kept or dropped based on the Engineering Intent.
Dropped	Elements are created without midside nodes.
Kept	Elements are created with midside nodes.

ElementShape

Used to specify the mesh element shape for the bodies associated with this control.

Possible Values

Automatic
Tetrahedrons
Hexahedrons

EndBehavior

Possible options for connection end behavior

Possible Values

ProgramControlled
HeatFluxDistributed
Isothermal
Coupled

EngineeringDataType

Supported types of engineering data.

Possible Values

Unknown
Material
Load
BeamSection
Mixture

EngineeringIntent

Used to define the kind of physics that is being studied in this simulation process (used in part-based Meshing).

Possible Values

Unknown	Intent of this process is not yet known due to insufficient information in connected Physics task(s).
StructuralOrThermalOrElectricConduction	Intent of this process is a structural, electric conduction and/or thermal simulation.
FluidFlow	Intent of this process is a fluid flow, or a fluid-solid heat conduction simulation.

ENMeanStressTheory

No details are provided for this entry.

Possible Values

NoneEN
Morrow
SWT

EntityType

Available expression entity types

Possible Values

Node
Face
Element
Edge

EquationSolverType

Equation Solver Types as available from MAPDL solver.

Possible Values

ProgramControlled
Direct
Iterative
Unsymmetric
Supernode
Subspace

EtaCorrelType

Efficiency correlation type

Possible Values

CaseyRobinson	Casey-Robinson correlation
CaseyMarty	Casey-Marty correlation
Rodgers	Rodgers correlation

EtaCorrelType

Efficiency correlation type

Possible Values

Suhrmann	Suhrmann's correlation
Baines	Baines' correlation

EtaImpType

impeller isentropic efficiency type

Possible Values

LinkToStage	linked to stage efficiency
User	user specified efficiency

EtaType

Stage efficiency type

Possible Values

User	User defined efficiency
Correlation	Efficiency calculated from correlation

EtaType

Stage efficiency type

Possible Values

User	User defined efficiency
Correlation	Efficiency calculated from correlation

EtaUserType

User specified stage efficiency type

Possible Values

Isentropic	user-specified isentropic efficiency
Polytropic	user-specified polytropic efficiency

ExcelConnectionState

Enumeration for the Excel Connection states

Possible Values

NotConnected	The connection with Excel is not established
ConnectionAlive	The connection is alive
ConnectionLost	The connection has been lost

ExecutionControlConflictOptions

Options for handling execution control conflicts on Edit

Possible Values

Default
UseSetupExecutionControl
UseSetupExecutionControlAlways
UseSolutionExecutionControl
UseSolutionExecutionControlAlways

ExecutionControlSource

Enumeration for the execution control conflict resolution.

Possible Values

IssueWarning
UseExecutionControlFromSetup
UseExecutionControlFromSolution

ExitAngleType

Exit angle type

Possible Values

Absolute	Absolute exit angle
Relative	Relative exit angle

Expression<Type>

The Expression type is used to hold an expression that returns a result of a given type or a result in a homogenous vector of the given type.

ExpressionType

Specifies the possible types of parameter expression.

Possible Values

Undefined	An undefined(null) expression.
Constant	An expression without dependency on other parameter.
Derived	An expression with dependency on other parameters.

FaceEdgePreference

Preference for face to edge contact detection.

Possible Values

FaceAndEdgeOfDifferentParts
EdgesOfSolidBodies
EdgesOfSurfaceBodies
All

FaceFacePreference

Preference for face to face contact detection.

Possible Values

FacesOfDifferentParts
All

FatigueUnitType

No details are provided for this entry.

Possible Values

Cycles
Blocks
Seconds
Minutes
Hours
Days
Months

FillStyles

Style of any filled region. The default is None.

Possible Values

None
Solid
Dense
Medium
Sparse
Horizontal
Vertical
Cross
BDiagonal
FDiagonal
CrossDiagonal
Gradient
Default

FittingType

Enumeration for the Response Surface type.

Possible Values

FITTINGTYPE_SRS	Standard Response Surface - Full second order Polynomials
FITTINGTYPE_KRIGING	Kriging
FITTINGTYPE_MARS	Non Parametric Regression
FITTINGTYPE_NN	Neural Network
FITTINGTYPE_SPARSEGRID	Sparse Grid
FITTINGTYPE_GRS	Genetic Aggregation

FixFirstLayer

Used to specify whether the heights or ratios of the first boundary layer will be modified to avoid collision. The default value is No.

Possible Values

Yes
No

FlowSpecification

No details are provided for this entry.

Possible Values

In
Out

FlowType

Enumeration of the flow boundary condition options.

Possible Values

MassFlow
PressureRatio
PressureDifference

FluentBoundary

No details are provided for this entry.

FluidType

Enumeration of the types of available fluids.

Possible Values

IdealGas
RealGas
Liquid

FormatType

Enumeration to specify the format type for imported data.

Possible Values

UserDefined
Delimited
Cdb
Axdt
Tgz
Anf
Brd
Mcm
Sip
Bool
Cond
Edb

FormulationType

Set the contact formulation to be used by the solver for a particular contact pair.

Possible Values

ProgramControlled
MultiPointConstraint
PurePenalty
NormalLagrange
AugmentedLagrange

GARefinementOutType

Enumeration for the Output Variable Combinations type when refining for the GARS algorithm.

Possible Values

GA_MAXOUT	Maximum Output
GA_ALLOUT	All Outputs

GARefinementStatusType

Enumeration for the convergence status for the GARS algorithm.

Possible Values

CS_UNKNOWN	Unknown
CS_NOTCONVERGED	Not Converged
CS_MAXPTS	Maximum Number of Points Reached
CS_STOPPED	Stopped by the User
CS_CONVERGED	Converged

GasModelType

gas model type

Possible Values

Ideal	ideal gas model
Real	real gas model

GasPropType

Gas properties type

Possible Values

Air	Gas props - Air
AFR	Gas props - Air/fuel ratio
Fixed	Gas props - Fixed

GeneralSettings

This is a class meant to provide an abstraction over different kinds of advanced result settings, such as Fatigue Settings.

GeometryAttachType

Type of geometry that is being attached.

Possible Values

ThreeDimensional
TwoDimensional
Unknown

GeometryAttachType

Type of geometry that is being attached.

Possible Values

ThreeDimensional
TwoDimensional
Unknown

GeometryStyleType

Geometry export style type

Possible Values

Interactive	Create interactive geometry
Parametric	Create parametric geometry

GoalType

Enumeration of the possible optimization objective types.

Possible Values

GT_NoPreference	No objective defined.
GT_MaximumPossible	Maximize.
GT_MinimumPossible	Minimize.
GT_SeekTarget	Seek target.

GPUAccelerator

Enumeration for the graphics acceleration library to be used by the Mechanical APDL editor.

Possible Values

None
NVIDIA

GravityDefinition

GravityDefinition defines the three components of gravity and the reference frame it is relative to.

GroundedLocation

Possible locations to be grounded in a spring

Possible Values

None
Location1
Location2

GrowthRateType

Determines height of boundary layers for given initial height and height ratio. The default value is Geometric.

Possible Values

Exponential	Height of boundary layer determined exponentially.

Geometric	Height of boundary layer determined geometrically.
Linear	Height of boundary layer determined linearly.

HubLEBetaType

Hub and Mean LE blade angle option

Possible Values

Cot	Hub/Mean LE blade angle calculated using cotangent (rel to Shroud LE beta)
Cos	Hub/Mean LE blade angle calculated using cosine (rel to Shroud LE beta)
User	User defined hub and mean LE blade angles

ICCombustionSimulationType

Type of IC Engine Combustion simulation type.

Possible Values

ICSector	0 for sector
ICFullEngineFullCycle	1 for full engine full cycle
ICFullEngineClosedValves	2 for full engine IVC to EVO

ICIVCandEVOoption

IVC and EVO options

Possible Values

ICLiftCurv	0 lift curv profile
ICIVCandEVO	1 for IVC EVO option

ICSimulationType

Type of IC Engine Simulation.

Possible Values

ICSimulationColdFlow	0 for Cold Flow Simulation
ICSimulationPortFLow	1 for Port Flow Simulation
ICSimulationCombustion	2 for Combustion Flow Simulation

ImpellerExportType

Impeller export type

Possible Values

Coupled	Coupled to volute
Isolated	Isolated impeller

ImpellerLengthType

impeller length ratio type

Possible Values

Automatic	automatic
User	user specified

ImpellerType

List of available impeller types

Possible Values

Unshrouded	unshrouded impeller
Shrouded	shrouded impeller

ImportAnalysis

No details are provided for this entry.

Possible Values

NaturalFreq
HarmonicVibe
Ictanalysis
MechanicalShock
RandomVibe

ImportanceLevel

Enumeration of the importance levels which can be associated with an optimization objective or constraint.

Possible Values

GI_MediumImportant	Default
GI_LowImportant	Lower
GI_HighImportant	Higher

ImportFacetQuality

Preference for imported facet Quality

Possible Values

PlugInFacetQuality_VeryCoarse	Very Coarse
PlugInFacetQuality_Coarse	Coarse -
PlugInFacetQuality_Normal	Normal -
PlugInFacetQuality_Fine	Fine
PlugInFacetQuality_VeryFine	Very Fine -
PlugInFacetQuality_Source	Source -

ImportParameterType

Defines the compare parts on update option during geometry import.

Possible Values

None
Independent
All

ImportStitchPreference

Stitch Import Preference

Possible Values

kStitchPreference_None	Do NOT stitch surfaces on import
kStitchPreference_ProgramTolerance	Stitch surfaces, but allow the program to determine the tolerance for the stitching
kStitchPreference_UserTolerance	Stitch surfaces, execute the stitching using the user's specified tolerance value

ImportStitchType

Defines the stitching option type during geometry import.

Possible Values

None
ProgramDefined
UserDefined

ImportWeightclass

Preference for imported weightclass, could be used for identifying weightclasses of imported parts

Possible Values

ImportWeightclass_Heavyweight	Heavyweight - all geometry and topology
ImportWeightclass_Middleweight	Middleweight - facet only faces

ImportWeightclass_Lightweight	Lightweight - each body consists of a single facetted face
ImportWeightclass_Featherweight	Featherweight - parts are created along with the tree hierarchy, but no topology

IncidenceType

Incidence type

Possible Values

incidence	specified incidence
choke	specified choke

InitialContactTreatment

An enumeration for initial contact treatment.

Possible Values

ProgramControlled
IgnoreInitialCondition
AutoCorrectInitialCondition

InitialInterfaceTreatmentType

Possible types for initial interface of a contact pair is treated.

Possible Values

Unknown
AdjustToTouch
OffsetWithRamping
OffsetWithoutRamping
ProgramControlled

InitializationMethods

Initilization Methods

Possible Values

ProgramControlled
SolverControlled
ProvideInitialSolution

InitializationOption

Enumeration for the Solution update initialization options.

Possible Values

Automatic
CachedSolutionData
CurrentSolutionData
InitialConditions

InitializationType

The initialization settings

Possible Values

ProgramControlled	Program Controlled
StartTime	Start Time
RestartStepTime	Restart Step and Time

InletAngleType

Inlet angle type

Possible Values

Absolute	Absolute inlet angle
Relative	Relative inlet angle
Calculated	OBSOLETE OPTION. RETAINED IN R16.0 FOR PURELY FOR BACKWARDS COMPATIBILITY Calculated from nozzle area

InletType

No details are provided for this entry.

Possible Values

NormalSpeed
Swirl
Pressure
MassFlow

int

This type represents an Integer number.

long

This type represents a long (64-bit) Integer number.

IntegrationOptions

No details are provided for this entry.

Possible Values

ProgramControlled
PreIntegrated
Mesh

IntegrationType

IntegrationType of elements used by solvers such as Full or Reduced.

Possible Values

Unknown
ProgramControlled
Full
Reduced

InterfaceBehavior

InterfaceBehavior is an abstraction over all kinds of interface behaviors such as JointBehavior, BeamBehavior, LinkBehavior, SpringBehavior and ContactBehavior.

InterfaceBehaviorCreation

It allows you to create one connection behavior and share it or create separate ones for each one of searched interfaces.

Possible Values

Shared
Individual

InterfaceDetectionTypes

Possible interface detection types for auto contact generation.

Possible Values

FaceToFace
FaceToEdge
EdgeToEdge

InterfaceModelSpecification

No details are provided for this entry.

Possible Values

None
Thermal

InterfaceSideSelection

No details are provided for this entry.

Possible Values

Automatic
SpecifyFace

InterfaceToleranceSpecification

It allows to let either program decides the tolerance value for you or you choose to input your own tolerance value

Possible Values

Automatic
Manual

InterfaceType

Type of inerfaces to be generated. Select from Contact, Joint and Mesh Interface.

Possible Values

None
Contact
Joint
MeshInterface

InterpolationType

The enum values for the Interpolation property to use for the AlternatingStress property charting

Possible Values

Unknown
Linear
SemiLog
LogLog

IsotropicElasticityDependents

Used when IsotropicElasticity is constant for all independent variables (e.g. Temperature).

IsovalueMethod

Specifies how to obtain the isovalues for the isosurface

Possible Values

Manual
Automatic

IterationKey

Possible options for defining equilibrium iterations key for non-linear solution in MAPDL solver

Possible Values

Unchanged
ProgramControlled
ForceToOne
ManuallySpecified

JobRunMode

The job running modes

Possible Values

Foreground	Forground mode.
Background	Background mode.
RemoteSolveManager	Submitted to Remote Solve Manager to run the job.

JointFormulation

JointFormulation types used by Mapdl Fixed Joint Connection Behavior

Possible Values

ProgramControlled
RigidLink
RigidBeam
Weld

JointReductionMethod

JointReductionMethod is an enumeration used by Mapdl Fixed Joint Connection Behavior formulations

Possible Values

ProgramControlled
DirectEliminationMethod
LagrangeMultiplierMethod

JointTrimOptimization

JointTrimOptimization is an enumeration used by Joint to trim elements for face-face joints

Possible Values

ProgramControlled
Off

JointType

This is a collection of all available joint types.

Possible Values

Unknown
Fixed
Hinge
Translational
Slot
Cylindrical
Universal
Spherical
Planar
General
Bushing
PointOnCurve

KernelVariationType

Enumeration of the Kernel Variation types.

Possible Values

VARIABLE	Variable
CONSTANT	Constant

LegendColoring

Types of allowed legend coloring.

Possible Values

Banded
Smooth

LevelDisplayType

Enumeration for the level of display type.

Possible Values

LD_Off	Off
LD_Final	Final
LD_FinalWithDetails	Final with details
LD_Iterative	Iterative
LD_IterativeWithDetails	Iterative with details

LinearCorrelationType

Enumeration of the Correlation types.

Possible Values

Spearman	Spearman
Pearson	Pearson

LineModelType

No details are provided for this entry.

Possible Values

Unknown
Beam
Truss
Spring
Cable
ThermalFluid
Pipe

LineStyles

Styles of lines that can be displayed. The default is Solid.

Possible Values

None
Solid
Dense
Medium
Sparse
DashShort
DashMedium
DashLong
DashDot
DashDotDot
DashDashDot
Gradient
Default

List<Type>

This type represents an unordered list of values.

LoadCaseType

Load Case Type for Material Designer

Possible Values

TensileX
TensileY
TensileZ
ShearXY
ShearXZ
ShearYZ

LoadingType

No details are provided for this entry.

Possible Values

FullyReversed
Zero
Ratio

LocalElementMidsideNodes

Used to specify if the elements created for the bodies associated with this control should have midside nodes or not. The default value is UseGlobalSettings.

Possible Values

UseGlobalSettings
Dropped
Kept

LocationSet

An immutable class representing a set of locations

A Location is a key value pair of a model reference string and Point3DUnited (coordinate of the hit point on the model).

The order of the locations is maintained for the lifetime of the object.

In script a LocationSet is represented in one of the following forms:

Model reference strings	["BODY1", "FACE5"]
Objects with LocationSet	[selectionSet1, selectionSet2]
Model reference string with hit point	[('FACE13', ('0.062 [m]', '0.197 [m]', '0.35 [m]')),('FACE23', ('0.07 [m]', '0.569 [m]', '0.502 [m]'))]
LocationSet returned from a function	AllBodies()

MachineSpecification

List of computers to be used for a parallel FLUENT session. The list can be specified directly, or a hosts file containing the list can be specified.

For more details on how to specify the machines to be used for a parallel FLUENT session, please refer to the FLUENT User's Guide.

Possible Values

MachineList
FileName

MachineType

Setup Entity enum definition for machine type

Possible Values

Pump	Pump
AxialCompressor	Axial Compressor
CentrifugalCompressor	Centrifugal Compressor
Fan	Fan
AxialTurbine	Axial Turbine
RadialTurbine	Radial Turbine
HydraulicTurbine	Hydraulic Turbine
Unknown	Undefined machine type

Other	Obsolete option, use 'Unknown' instead.

Material

The material object holds all information that defines the behavior for a specific material.

MaterialAssignment

Assigns a Material to a specified Location.

MaterialNamesList

Database materials list Note that this is currently a fixed list which must correspond to the vistaFluids.xml database

Possible Values

Air	Air
CarbonDioxide	Carbon dioxide
Hydrogen	Hydrogen
Methane	Methane
Nitrogen	Nitrogen
Oxygen	Oxygen
Parahydrogen	Parahydrogen
Propylene	Propylene
R123	R123
R125	R125
R134a	R134a
R141b	R141b
R142b	R142b
R245fa	R245fa
Water	Water

MaterialPropsType

material properties type

Possible Values

Database	select material from database
User	user specified material properties

MeanStressTheory

No details are provided for this entry.

Possible Values

None

Goodman
Soderberg
Gerber

MeshBasedDefeaturing

Used to specify the type of automatic defeaturing for dirty geometries. The default value is AutomaticallyDetermined.

Possible Values

AutomaticallyDetermined	Automatically removes features smaller than or equal to the calculated default tolerance value on dirty geometry.
UserDefined	Automatically removes features smaller than or equal to the user defined defeaturing tolerance value on dirty geometry.
Off	No defeaturing done on dirty geometry.

MeshFileType

MeshFileType is used to identify the format of the mesh data file. This is usually the same as the application that generated the data file.

Possible Values

CFX
ICEM_CFD
FLUENT
POLYFLOW
Unknown

MeshMetric

Used to specify the metric type based on which mesh quality is evaluated. The default value is None.

Possible Values

None	No mesh metric type is selected.
ElementQuality	A composite quality metric is computed ranging between 0 and 1.
AspectRatio	Aspect ratio is calculated based only on corner nodes of elements.
JacobianRatio	Jacobian ratio is computed for all elements except those with no midside nodes or perfectly centered midside nodes.
WarpingFactor	Warping factor is computed for quadrilateral shell elements and quadrilateral faces of bricks, wedges and pyramids.

ParallelDeviation	Parallel deviation is computed ignoring mid-side nodes and based on unit vectors along each element size.
MaximumCornerAngle	Maximum corner angles are computed between adjacent edges and high corner angles can degrade element performance.
Skewness	Skewness is one primary quality measure that determines how close to ideal (that is, equilateral or equiangular) a face or cell is.
OrthogonalQuality	Orthogonal quality is computed using face normal vectors and ranges from 0 to 1.

MeshReading

Indicates the type of restart.

Possible Values

Direct	In this mode, one loads directly the mesh (no conversion).
Convert	In this mode, one converts the *.poly or fluent *.msh into a polyflow *msh file

MeshRestartMode

Indicates the type of mesh import we want : - no initialization from upstream system - select a mesh in a list of mesh files coming from an upstream polyflow system

Possible Values

NoUpstreamMeshFile	in this mode, no mesh file is selected from the upstream polyflow system.
SingleMeshFile	in this mode, a mesh file is selected from a list of meshes coming from the upstream polyflow system.

MessageCategory

The valid message categories.

Possible Values

None
ConditionallyUpToDate

MessageType

The valid message types.

Possible Values

Information	An informational message for the user.
Warning	A warning message for the user.
Error	An error message for the user.
Fatal	A fatal message for the user. On receiving a message of this type, the framework will show the message to the user, and then exit immediately, without giving any chance to save their work or shut down running processes.
Problems	Problem messages (WARNING + ERROR + FATAL).
Debug	A debug message.
Progress	A progress message for the user.
News	A news update for the user.
AttentionRequired	A message that requires user action to address
Standard	All non-debug and non-progress messages.

Method

The method used to create the Single Value Result

Possible Values

FunctionCalculatorMethod
UserDefinedExpressionMethod

Metric

Used to specify the measure to be used for improving the mesh quality. The default value is Skewness.

Possible Values

Skewness
SizeChange
AspectRatio

MetricErrorScatterType

Enumeration of the metric types supported.

Possible Values

ErrNrmInf
RelErrNrm2

MirrorTransformOptionsForMeshing

Class containing all mirror transformation options for Model Assembly workflows in Meshing

MirrorTransformOptionsForSimulation

Class containing all mirror transformation options for Model Assembly workflows in Mechanical

MixedImportPref

Mixed import preference option for mixed dimension parts

Possible Values

MixedImport_None	If mixed dimension part, import None
MixedImport_Solids	If mixed dimension part, import Solids only
MixedImport_Surfaces	If mixed dimension part, import Surfaces only
MixedImport_Lines	If mixed dimension part, import Lines only
MixedImport_SolidsAndSurfaces	If mixed dimension part, import Solids and Surfaces only
MixedImport_SolidsAndLines	If mixed dimension part, import Solids and Surfaces only
MixedImport_SurfacesAndLines	If mixed dimension part, import Surfaces and Lines only
MixedImport_SolidsSurfacesAndLines	If mixed dimension part, import Surfaces and Lines only

MixedResolutionType

Defines the mixed resolution option during geometry import.

Possible Values

None
Solid
Sheet
Wire
Point
SolidSheet
SolidWire
SolidPoint
SheetWire
SheetPoint
WirePoint
SheetWirePoint
SolidWirePoint
SolidSheetPoint
SolidSheetWire
All

MonitorChartType

MonitorChartType enum: Residual or UserDefined

Possible Values

Residual
UserDefined

MPIType

Enumeration for the MPI library to be used by the Mechanical APDL solver.

Possible Values

Undefined
IntelMPI
MSMPI
OpenMPI

MpiTypeValue

No details are provided for this entry.

Possible Values

Ibmmpi
Intel

MResOptions

Enumeration for the load options for Multi-configuration Results.

Possible Values

AllConfigsSingleCase
AllConfigsSeparateCases
LastConfigOnly

MultiAxisType

No details are provided for this entry.

Possible Values

S_X_TYPE
S_Y_TYPE
S_Z_TYPE
S_XY_TYPE
S_YZ_TYPE
S_XZ_TYPE
S_VON_MISES_TYPE
S_SIGNED_VON_MISES_TYPE

S_MAX_SHEAR_TYPE
S_MAX_PRINCIPAL_TYPE
S_ABS_MAX_PRINCIPAL_TYPE
S_CRITICAL_PLANE_TYPE

NegativeBucklingLoadMultiplier

This is used for allowing negative load multiplier for eigenvalue buckling analysis.

Possible Values

ProgramControlled
No
Yes

NeoHookeanHyperElasticityDependents

Neo-Hookean Hyperelasticity dependents.

NewtonRaphsonOption

Possible ways of using Newton-Raphson method.

Possible Values

kAuto
kFull
kModified
kInitial
kUnsymmetrical

NodeAndElementRenumberingMethodType

Enumartion of of the renumbering system

Possible Values

None
Offset
Automatic

NumSampType

Enumeration to specify for the samples type for OSFD algorithm.

Possible Values

SFD_CCD	CCD samples
SFD_LINEAR	Linear model samples

SFD_PUREQUAD	Pure quadratic model samples
SFD_CROSSQUAD	Full quadratic model samples
SFD_USER	User-defined samples

NuUserType

kinematic viscosity calculation type (obsolete)

Possible Values

Sutherland	calculate viscosity from Sutherland's Law (using coeffs for air)
User	user specified kinematic viscosity

Object

This type can represent any generic object. It is used when any type is a valid value.

ObjectRenamingTypeInMechanical

Controls how objects in Mechanical will be named for models assembled from multiple upstream sources

Possible Values

Off
BasedOnCellId
BasedOnSystemName
BasedOnUserInput

ObjectRenamingTypeInMeshing

Controls how objects in Meshing will be named for models assembled from multiple upstream sources

Possible Values

Off
BasedOnCellId
BasedOnSystemName
BasedOnUserInput

OnOffSwitch

This is used for activating or deactivating a setting.

Possible Values

kOn
kOff

OpeningPositionMethod

Enumeration of the inlet/outlet placement options.

Possible Values

Manual
AdjacentBlade

OptimalSpaceFillingType

Enumeration of the available design types for the Optimal Space Filling algorithm.

Possible Values

SFDTYPE_MDIST	Max-Min Distance
SFDTYPE_CL2	Centered L2
SFDTYPE_MAXENT	Maximum entropy

OptimizationMethodSelection

Enumeration of the available optimization method selections.

Possible Values

OMS_Auto	Automated selection - Optimization Method and its settings are controlled automatically by DesignXplorer
OMS_Manual	Manual - Optimization Method and its settings are defined by the User

OptimizationSolverType

Possible selection for optimization solver type

Possible Values

ProgramControlled
SequentialConvexProgramming
OptimalityCriteria

OrientationDefineBy

No details are provided for this entry.

Possible Values

ReferenceFrame
Geometry

OrientationStyle

Allowed orientation of a legend. Default is Vertical

Possible Values

Vertical
Horizontal
Default

OrthotropicElasticityDependents

The Orthotropic Elasticity dependents.

OutletType

No details are provided for this entry.

Possible Values

Pressure
MassFlow
NormalSpeed

Output<Type>

The Output type is used in select instances where a method returns additional information in a method argument as well as the method return. These output arguments are typically optional, and a output variable must be declared before it is used. Once assignment has been made to an output variable, the return value can be evaluated by using the Get() method on the variable.

The following example shows the declaration and use of an output argument.

```
>>> template1 = GetTemplate(TemplateName="EngData")
>>> system1 = template1.CreateSystem()
>>> engineeringData1 = system1.GetContainer(ComponentName="Engineering Data")
>>> matl1 = engineeringData1.GetMaterial(Name="Structural Steel")
>>> matlProp1 = matl1.GetProperty(Name="Density")
>>> matlProp1.SetData(
>>>    Variables="Density",
>>>    Values="-10 [kg m^-3]")
>>> from Ansys.Core.Commands import Output
>>> outMsg = Output[str]()
>>> if not matl1.IsValid(Message=outMsg):
>>>    print "Material is not valid for the following reason:"
>>>    print outMsg.Get()
Material is not valid for the following reason:
The value(s) for Density must be greater than zero.
```

OutputFrequencyType

The entitry stores the options to specify frequency of writing result files

Possible Values

None	No intermediate result files
EveryStep	Every Coupling Step
StepInterval	At defined interval

OutputSource

Source of the output values, indicating the method used to obtain them.

Possible Values

UserEdited	The output values are edited by the user.
Simulation	The output values are obtained by a real simulation.
ResponseSurface	The output values are obtained by evaluating a response surface approximation.

OutputType

OutputType for OutputSpecification such as Deformation, Stress or Strain

Possible Values

None
All
Basic
NodalDofSolution
NodalReactionLoads
NodalVelocity
NodalAcceleration
ElementSolution
ElementNodalLoads
ElementNodalStresses
ElementElasticStrains
ElementThermalStrains
ElementPlasticStrains
ElementCreepStrains
ElementDiffusionStrains
ElementNodalGradients
ElementNodalFluxes
IntegrationPointLocations
StateVariables
ElementMiscellaneousData
ElementEnergies

ParameterizedEntityPropertiesCollection

A ReadOnlyDictionary for parameterized properties. The keys are the data references to the entities that hold the properties, the values are the list (one or more) of the parameterized properties.

ParameterNature

Enumeration of the possible nature of a parameter.

Possible Values

NatureContinuous	Continuous
NatureUsability	Obsolete. Instead of defining a usability parameter, define a continuous parameter with the UseManufacturableValues set to True.
NatureDiscrete	Discrete

ParameterRelationshipType

Enumeration of the possible parameter relationship types.

Possible Values

PRT_LessThanOrEqualTo	Less Than or Equal To
PRT_GreaterThanOrEqualTo	Greater Than or Equal To

ParameterUsage

Specifies the possible ways a parameter can be used or set within the data model.

Possible Values

Input	A parameter whose value will be used by the data model.
ExpressionOutput	An output parameter whose value is based on an expression. The parameter cannot be associated directly with the data model.
DirectOutput	A parameter whose value will be provided directly by the data model. The parameter expression has to be undefined(null).

ParameterValueType

Type of a parameter value.

Possible Values

ActualValue	The actual value of the parameter.

VariationToReference	The variation of the parameter with respect to the current reference point, as a decimal number.

PartialDesignPointUpdate

No details are provided for this entry.

Possible Values

None
Geometry

ParticleTrackColorOption

No details are provided for this entry.

Possible Values

None
Time
TimeStep
Position
TrackId
Velocity
Diameter
Temperature

ParticleTrackDisplayStyle

No details are provided for this entry.

Possible Values

Lines
LinesAndSymbols
Symbols

ParticleTrackShapeDef

No details are provided for this entry.

Possible Values

Wire
Tube

ParticleTrackStyle

No details are provided for this entry.

Possible Values

Continuous
Segments

PeriodicSurfType

Enumeration of the periodic surface options.

Possible Values

OnePiece
ThreePieces

PersistablePosition

No details are provided for this entry.

PersistableStateMessageLevel

No details are provided for this entry.

Possible Values

Error
Warning
Information
Verbose

PhysicsRegion

Specifies the type and location of the fundamental physics in the simulation.

PhysicsType

The available physics types for an analysis.

Possible Values

Unknown	Indicates physics type was never set
None	Indicates physics type was set and then cleared
Structural	A mechanical analysis determines the displacements, stresses, strains, and forces in struc-

	tures or components caused by loads that do not induce significant inertia and damping effects. Steady loading and response conditions are assumed; that is, the loads and the structure's response are assumed to vary slowly with respect to time.
ElectricConduction	An electric analysis supports Steady-State Electric Conduction.
Fluid	A fluid flow analysis that simulates fluid passing through or around an object.
Thermal	You can use a thermal analysis to determine temperatures, thermal gradients, heat flow rates, and heat fluxes in an object that are caused by thermal loads.
Extrusion	Simulation of molten polymers for extrusion (in the laminar regime). The fluid can be modeled by a generalized newtonian law or by a viscoelastic law. Generally, it involves some deformation (needing remeshing) of the flow domain.
Electromagnetics	Denotes static and low frequency electromagnetic physics for analysis of electromechanical, etc. systems.
BlowMolding	Simulation of molten polymers for blow molding / thermoforming (in the laminar regime). The fluid can be modeled by a generalized newtonian law or by a viscoelastic law. It is a transient simulation, involving deformation of the flow domain, and contact with one or more molds.
ElectroStatic	An electric analysis supports Steady-State electric field
All	All physics types.

PIFType

power input factor type

Possible Values

Correlation	correlation
User	user specified

Plane

The analytic definition of a flat plane where the Z axis is the normal direction.

PlaneConstructionMethod

Option for defining the plane construction method

Possible Values

OriginAndOrientation
FromPlane
FromReferenceFrame

PlaneOption

Options for selecting a Reference Frame plane.

Possible Values

XY
YZ
ZX

PlasticityType

Identifiers specifying plasticity types.

Possible Values

None
BilinearIsotropicHardening

Point

Point represented by local coordinates relative to a reference frame.

PointCalculationMethod

Options for calculating a point from geometry selection

Possible Values

Centroid
HitPoint

PorosityMediaMaterial

No details are provided for this entry.

Possible Values

Default
Specified

PorosityType

No details are provided for this entry.

Possible Values

Isotropic
Directional

PositionType

No details are provided for this entry.

Possible Values

Default
Left
Right
Above
Below

PostReportNamesType

CFD Post reports

Possible Values

None
AxialCompressorReport
AxialCompressorRotorReport
CentrifugalCompressorReport
CentrifugalCompressorBladeRowReport
CentrifugalCompressorRotorReport
TurbineReport
TurbineRotorReport
FanNoiseReport
FanReport
GenericReport
HydraulicTurbineReport
HydraulicTurbineRotorReport
PumpReport
PumpImpellerReport
StatorReport
TurbineStatorReport
Custom

PredictionErrorScatterType

Enumeration of the scatter types.

Possible Values

MetricErrorvsSnapshot	Metric Error vs Snapshots
CDFError	Empirical Cumulative Distribution of Error on Snapshots

PredictorKey

Possible options to activate predictor for non-linear solution in MAPDL solver.

Possible Values

Unchanged
ProgramControlled
Off
OnAfterFirstSubstep
OnForAllSubsteps

PreferenceType

Possible options for detecting contacts -- between bodies, or between bodies of different parts or all

Possible Values

BetweenBodies
BetweenBodiesOfDifferentParts
All

PreswirlType

Preswirl type

Possible Values

constant	Constant inlet angle
free	Free vortex
forced	Solid body rotation

PreswirlType

Preswirl type

Possible Values

constant	Constant inlet angle
free	Free vortex
forced	Solid body rotation
linear	Linear variation of Vw

PretensionDefineBy

This enum is used to set how bolt pretension is defined. It can be defined by axial foce, adjustment, lLock, open or increment.

Possible Values

kUnknown
kAxialForce
kAdjustment
kLock
kOpen
kIncrement
kTorque

PretensionModelAs

This enum is used to set how many bolts should be created if location is set to multiple volumes : One for all selected volumes or one per selected volume.

Possible Values

Unknown
OneBoltForAllLocations
OneBoltPerLocation

PrimitiveBoxCreationMethod

Box creation method options.

Possible Values

ByEntitySelection	Box can be created by enclosing single or multiple entities.
ByCoordinate	Box can be created by specifying diagonal coordinates.

PriorityType

Options for override priority.

Possible Values

FaceOverride
EdgeOverride
All

ProcessDistributionOption

No details are provided for this entry.

Possible Values

LocalComputer
MultipleComputers

ProcessorUnit

Various Processore Unit options available for Microsoft Scheduler

Possible Values

Core
Socket
Node

ProgramControlType

Possible values of the underlying property in the current load step

Possible Values

Unchanged
ProgramControlled
On
Off

ProjectHandler

Handle a project file (and optional associated files). This class handles the lifecycle of the file, the instanciation of the ROM API project object and the calls to this object for ROM evaluation.

ProjectionStrategy

Projection strategy for traitement of multi-field and/or multi-component fields

Possible Values

STATE_VECTOR_PROJECTION	Same projection coefficient for all physics
MULTI_FIELD_PROJECTION	Dedicated projection coefficient for each physics
MIXTE_PROJECTION	Dedicated projection coefficient for each physics and same projection for all component of a field of vector

ProximitySizeFunctionSources

Used to specify the source type to be used for size function calculation when Advanced Size Function is Proximity or Curvature and proximity. The default value is Faces.

Possible Values

Faces
Edges
FacesAndEdges

Quantity

This class represents a physical quantity that can be measured. It holds a double value and a string that specifies the value's unit of measurement. The Value and Unit can be accessed individually as properties on this type, and a Quantity can be converted to new units.

Mathematical operations can also be performed on Quantities, and these operations calculate and enforce dimensional consistency between units. Note that most mathematical operations return results in computational units for the current unit system, but this is not guaranteed, and may change in the future for performance or other reasons. If you need to ensure that results are in a particular unit, you should always apply a conversion to the final result of the calculation.

There is an FAQ on units at {Albion}/CoreAddins/Units/FAQ.md

There is a detailed description of the units string syntax at {Albion}/CoreAddins/Units/UnitSyntax.md

RadiationSpecification

No details are provided for this entry.

Possible Values

None
DiscreteOrdinates
MonteCarlo

RadiusDefineBy

No details are provided for this entry.

Possible Values

ProgramControlled
Manual

RampingType

Enum providing ramping options.

Possible Values

None	No ramping.
Linear	Linear profile ramping.

ReadOnlyDictionary<Key, Value>

This type represents a read-only data dictionary, where a Key is used to access an associated Value. When used in scripting, a dictionary is created or printed using the form

```
myDict = {key1:value1, key2:value2, ...}
```

Python functionality can be used to examine dictionary keys, test for key existence and perform other useful operations on dictionaries; however, the contents may not be manipulated unless the dictionary or its contents are cloned into a regular dictionary.

RealGas

Enumeration of the available real gas materials

Possible Values

Air
CarbonDioxide
Hydrogen
Methane
Nitrogen
Oxygen
Parahydrogen
Propylene
R123
R125
R134a
R141b
R142b
R245fa
Water
Custom

ReductionType

ReductionType enum specifies the objective options for the optimization process

Possible Values

Mass
Volume

RefDiameterType

Enumeration of the reference diameter options.

Possible Values

Automatic
User

ReferenceFrame

Frame of reference for modeling and simulation data.

ReferenceFrameDefinitionType

Options for reference frame definition types

Possible Values

OriginAndOrientation
CoincidentWithParent
Transform

ReferenceTopologyType

Defines the topology type for a given reference.

Possible Values

RTT_NoTopology
RTT_Vertex
RTT_Edge
RTT_Face
RTT_Body
RTT_Part
RTT_Assembly
RTT_ConfigurationComponent
RTT_Invalid

ReferenceValueSpecification

Options for specifying convergence reference value: Program controlled or Manual

Possible Values

ProgramControlled
Manual

RegionBehavior

Used to specify the meshing region behavior. Currently, the only possible value is Wrap.

Possible Values

Wrap	Simplifies unclean geometry before sewing.

RegionInterface2

No details are provided for this entry.

RelativeToType

Options for selecting a which reference frame to define relative to

Possible Values

Local
Parent

ResponseChartModes

Enumeration of the available Response chart modes.

Possible Values

Curve2D	2D response chart where an output parameter is plotted versus an input parameter.
Surface3D	3D response chart where an output parameter is plotted versus two input parameters.
Slices2D	2D response chart where an output parameter is plotted versus two input parameters, on the X axis and the other varying over several curves.

ResponseSurfaceRefinementType

Enumeration for the Refinement type.

Possible Values

REFINEMENT_NONE	None
REFINEMENT_AUTO	Automated
REFINEMENT_MANUAL	Manual

ResultReference

No details are provided for this entry.

Possible Values

Relative
Absolute

ResultType

No details are provided for this entry.

Possible Values

Displacement
Velocity
Acceleration

ReverseEdgeLoopOrientation

Options for choosing loop orientation. Required only if construction algorithm is convex hull.

Possible Values

No	Construct capping surface by using default loop orientation.
Yes	Construct capping surface by reversing loop orientation.

RigidTransformOptionsForMeshing

Class containing all rigid transformation options for Model Assembly workflows in Meshing

RigidTransformOptionsForSimulation

Class containing all rigid transformation options for Model Assembly workflows in Mechanical

RomEngine

Engine powering the ROM

Possible Values

DEMO	Demo mode building ROMs suitable for testing the workflow.
SVD_GA	Build ROMs based SVD with genetic aggregation response surface

RotationalDOFBehavior

This enum is used to set the rotational degree of freedom as fixed all, free all, free about x, free about y or free about z.

Possible Values

FixedAll
FreeAll
FreeAboutX
FreeAboutY
FreeAboutZ

RotationType

Enumeration of the machine rotational direction options.

Possible Values

RightHanded
LeftHanded

RoughnessType

surface roughness type

Possible Values

Machined	machined surface finish
Cast	cast surface finish

RshSpecification

Client used to connect to the nodes in a cluster of LINUX machines.

'Other' is used for a custom connect command.

Possible Values

SSH
Other
RSH

RsmQueueDetails

A class to define Rsm Queue details for adding them to UI

RunTimeIndex

Enumeration of the available run time indexes.

Possible Values

RTI_1
RTI_2
RTI_3
RTI_4
RTI_5
RTI_6
RTI_7
RTI_8
RTI_9

SampleGenType

Enumeration for the Sampling type.

Possible Values

SAMPLE_GEN_LHS	LHS
SAMPLE_GEN_WLHS	WLHS

SamplesChartModes

Enumeration of the available Samples chart modes.

Possible Values

Candidates	Draw the samples and highlight the optimization candidates.
ParetoFront	Draw the samples using colors that represent their Pareto front.

SamplingType

Enumeration of the correlation Sampling types.

Possible Values

Auto	Automated - Samples generation is controlled automatically by DesignXplorer.
Custom	Custom - Samples are defined by the User.

Scale

Enum to define the scale of the axis.

Possible Values

Linear
CommonLog
NaturalLog

Scale

Enum to define the scale of the axis.

Possible Values

Linear	Linear scale
CommonLog	Common or log base 10 scale
NaturalLog	Natural log scale

ScaleFactor

No details are provided for this entry.

Possible Values

One
Two
Three
UserDefined

SchedulerSpecification

Various Job Schedulers available on Unix/Linux.

Possible Values

LSF
SGE
PBSPro
SLURM

SearchDirectionMethod

Possible options for detection direction method.

Possible Values

ProgramControlled
Manual

SearchPosition

A filter that can be applied to the SearchString used by GetAllObjects query.

Possible Values

Exact
Contains
StartsWith
EndsWith

SectionOffsetType

Thickness distribution type of surface bodies.

Possible Values

midSurface
OneSide
userDefined

SeedPointDistributionType

Allowable values for seed point sampling.

Possible Values

UniformSample
GridSample
Mesh

SeedPointMeshDistributionType

Allowable values for how mesh is evaluated when sampling seed points.

Possible Values

EveryNthElement
EveryNthVertex

SensitivityChartModes

Sensitivity chart modes.

Possible Values

BarChart	2D Bar chart
PieChart	2D Pie chart

SeparationSurfaceNormal

This enum is used to set how separation surface and tolerance is specified. It can be program controlled or user can manual set the property value.

Possible Values

X
Y
Z

SeparationSurfaceSpecification

This enum is used to set how separation surface and tolerance is specified. It can be program controlled or user can manual set the property value.

Possible Values

ReferenceFrame
GeometrySelection
Unknown

Setup

Entity which manages the setup of the ROM feature. This DataObject must be held by a solver addin container. The solver addin container will then be used as an identifier for the solver addin system.

ShapeChecking

Used to define the shape checking algorithm to be used by the mesher. The default value is UseEngineeringIntent.

Possible Values

UseEngineeringIntent	Determine which algorithm to use based on the Engineering Intent.
StandardMechanical	Specifies the standard mechanical shape checking algorithm for structural simulations.
AggressiveMechanical	Specifies the aggressive mechanical shape checking algorithm for structural simulations.
CFD	Specifies the CFD shape checking algorithm for flow simulations.
None	Disable shape checking.

ShrLEBetaType

Shroud LE blade angle option

Possible Values

Incidence	Shroud LE blade angle calculated from specified incidence
User	User defined shroud LE blade angle

ShroudDiameterType

shroud diameter type

Possible Values

Diameter	user specified shroud inlet diameter
Angle	user specified shroud vane inlet angle
Optimum	optimised shroud inlet diameter (minimum Mrel)

SimulationTopOptAction

No details are provided for this entry.

Possible Values

KExportInitialPmdbAndFinalStl
KExportInitialPmdbAndFinalObj

SimulationType

Enumeration of the simulation types of a parameter.

Possible Values

DesignVariable	Design variable
UncertaintyVariable	Uncertainty variable

float

This type represents a single-precision floating point number.

SixSigmaTableTypes

Enumeration of the Probability Table types.

Possible Values

ProbabilityTable	Quantile-percentile
InverseProbabilityTable	Percentile-quantile

SizingBehavior

Used to specify whether size control settings can be changed by the mesher (Soft) or not (Hard).

Possible Values

Soft
Hard

SlipSpecification

No details are provided for this entry.

Possible Values

NoSlip
FreeSlip

SmallSlidingSetting

Possible options for small sliding to occur in bonded or no separation contacts.

Possible Values

ProgramControlled
On
Off

Smoothing

Used to specify the level of smoothing iterations to be performed. The default value is Medium.

Possible Values

Low
Medium
High

SnapshotCacheHandler

Handle a single snapshot cache file. This class ensure the correct registration and deregistration of the snapshot cache files and expose push/pull methods with Workbench objects.

SolutionProgression

SolutionProgression is a class that specifes convergence, stabilization of solution etc, the typical attributes for monitoring and controlling a solution.

SolutionStep

Solution Step

SolutionUpdateOption

System Coupling Solution Update option

Possible Values

Foreground	Foreground update

SolverRestartMode

Indicates the type of restart.

Possible Values

NoRestartFile	In this mode, no restart file is used to initialize time scheme and fields.
SingleResultFile	In this mode, a result file is used to initialize fields.
CombineResultRestartFiles	In this mode, a result file is used to initialize fields and a restart file is used to initialize time scheme (starting time + derivates).
SingleCsvFile	In this mode, a csv file is used to initialize fields.
CombineCsvRestartFiles	In this mode, a csv file is used to initialize fields and a restart file is used to initialize time scheme (starting time + derivates).
ManyResultFiles	In this mode, a list of polyflow result files will be used to define the flow field on which we evaluate a transient mixing task (computation of a set of trajectories).
ManyCsvFiles	In this mode, a list of polyflow csv files will be selected for a conversion into other kinds of results .. for a future mixing task for example.
InitializationFromCsvFiles	In this mode, csv files are used to initialize fields. The CSV file(s) will be selected in Poly-data.

SourceInformation

Information about the original source of the Engineering Data.

SourceSelectionType

Options for source selection types

Possible Values

BrowseSource
ActiveCADAttach

SpanwiseDistributionType

Spanwise distribution type

Possible Values

General	Blade exported using general spanwise distribution
Radial	Blade exported using radial spanwise distribution

SpringBehaviorType

No details are provided for this entry.

Possible Values

OneDSpring
ThreeDSpring

SpringPreloadType

Possible pre-loading types for a spring connection

Possible Values

None
Load
Length
Torque
Rotation

SpringType

Possible types for a spring connection

Possible Values

Longitudinal
Torsional

StabilizationKey

Possible options for stabilization key

Possible Values

Off
ConstantEnergyDissipation
ReducedEnergyDissipation
ConstantDampingFactor
ReducedDampingFactor

StackingType

stacking type

Possible Values

Radial	Radial stacking
Tan	Beta calculated from tangent
Sin	Beta calculated from sin

StateOfMatter

Identifiers specifying states of matter.

Possible Values

Unknown
None
Solid
Liquid
Gas
Plasma

Status

The current calculation status of the Excel file.

Possible Values

UpToDate	The output parameters are up to date.
OutOfDate	The values of the input parameters are modified and the values of the output parameters are not recalculated yet.
ErrorsWhenCalculating	Errors ocurred during the calculation in Excel.

StepSizeType

The method used to determine the size of integration steps in streamline creation.

Possible Values

Automatic
Fixed

StraightSidedElements

Used to force mesher to create elements with straight edges. The default value is No.

Possible Values

Yes
No

StreamlineColorOption

Allowable values for how a streamline is colored.

Possible Values

None
Time

StreamLineDisplayStyle

Allowable values for how a streamline is displayed.

Possible Values

Lines
LinesAndSymbols
Symbols

StreamLineShapeDef

Allowable values for the shape of the streamline

Possible Values

Wire
Ribbon
Tube

StreamLineStyle

Allowable values for the style of the streamline

Possible Values

Continuous
Segments

StressType

StressType is an enum for the stress types

Possible Values

EquivalentStress
StressIntensity
MaximumShearStress

string

This type represents a String value.

SubStepOption

Possible settings for sub step options

Possible Values

No
Yes
IfNotConverged

Substepping

Define Substepping options such as specified range, fixed number or program controlled as available in MAPDL solver.

Possible Values

Unchanged
ProgramControlled
SpecifiedRange
FixedNumber

SupportFileFormatType

Enumeration for any supporting files

Possible Values

Info

SupportType

Possible types of support

Possible Values

Unknown
Fixed
Frictionless
UserSpecified

SurfaceMonitorChainingOperation

No details are provided for this entry.

Possible Values

And
Or

SurfaceMonitorOperation

No details are provided for this entry.

Possible Values

Monitor
Halt

SurfaceMonitorVariable

No details are provided for this entry.

Possible Values

TotalPressure
StaticPressure
MassFlowRate
NormalSpeed
Temperature
DragX
DragY
DragZ
HeatFlux
MinTotalPressure
MaxTotalPressure
MinStaticPressure
MaxStaticPressure
MinNormalSpeed

MaxNormalSpeed
MinTemperature
MaxTemperature
VolumeFlowRate
HeatFlow
XVelocity
YVelocity
ZVelocity
VelocityMagnitude

SymbolLengthOptions

The allowable values for the control of vector length display.

Possible Values

BasedOnMagnitude
Constant

SymbolShape

The symbol shape used for the associated result

Possible Values

SymbolShape1DArrowHead
SymbolShape1DArrow
SymbolShape2DArrowHead
SymbolShape2DArrow
SymbolShape3DArrowHead
SymbolShape3DArrow
SymbolShape3DCrossHair
SymbolShape3DCube
SymbolShape3DSphere

SystemPropertyDictionary

No details are provided for this entry.

TableInterpolation

An enum of the type of interpolation to use when accessing data from the table at a given index. The valid enum values are; None, Linear, Cubic Spline.

Possible Values

None	No interpolation is done.
Linear	A linear interpolation will be used to compute the return value.

TableInterpolationBeyondBounds

An enum which describes what will occur when the index is outside of the bounds of the table. The valid enum values are; Constant, Fit, Zero, and Error.

Possible Values

Constant	his option will cause the value returned to be the data at the index nearest to the requested index.
Fit	This option will ignore the bounds of the table and use the Interpolation algorithm to fit the data and then compute the value at the requested index.
Zero	This option will cause the value returned to be zero if the index is outside of the bounds of the table.
Error	If the data cannot be determined from the given index (outside of the bounds) then an error will occur.

TessellationRefinement

Used to specify value to be used for tessellation refinement. The default value is AutomaticallyDetermined.

Possible Values

AutomaticallyDetermined	Sets tessellation refinement to 10% of the value of Min Size/Proximity Min Size.
UserDefined	Sets tessellation refinement to user defined value.
Off	Tessellation refinement is not performed.

ThermalWallSpecification

No details are provided for this entry.

Possible Values

Insulated
HeatFlux
HeatFlow
Temperature
Convection
Radiation
Mixed

TimeOption

No details are provided for this entry.

Possible Values

EndTime
UserDefined

TipDiamType

Impeller tip diameter option

Possible Values

Automatic	D2 calculated automatically (from stability factor)
HeadCoeff	D2 calculated from head coefficient
User	D2 user defined

TopologyType

All of the possible values for topology.

Possible Values

Point	A topological point
Curve	A topological curve
Surface	A topological surface
Volume	A topological volume

TrackerMethod

The method used to create the Single Value Result

Possible Values

Undefined
Function
UserDefinedExpression

TradeoffChartModes

Enumeration of the available Tradeoff chart modes.

Possible Values

Curve2D	2D tradeoff chart where a parameter is plotted versus another parameter.

Surface3D	3D tradeoff chart where a parameter is plotted versus two other parameters.

TranscriptType

The allowable types of output in a Transcript monitor.

Possible Values

SolverOutput
PhysicsCouplingDiagnostic
SelectFile

TransferAtType

Enum providing options of when to Transfer Data.

Possible Values

StartOfStep	Transfer data at start of step.
StartOfIteration	Transfer data at start of iteration.

TransferDataFromNewComponentSpec

No details are provided for this entry.

TransferDataToNewComponentSpec

No details are provided for this entry.

TransferType

Transfer type from ACP

Possible Values

Shell
Solid
None

Transform4x4

Represents a combined rotation and translation matrix Rotation matrix is the first 3 rows and columns in row-major order Translation matrix is the 4th column

TransformationType

Enumeration of the available transformation types applicable to a parameter.

Possible Values

TransTypeNone	No transformation
TransTypeBox	Box-Cox
TransTypeYeo	Yeo-Johnson

TransformType

Transform type for assembly

Possible Values

RotationAndTranslation
Mirroring

TransformType

Transform type for assembly

Possible Values

RotationAndTranslation
Mirroring

TransformType

Transform type for assembly

Possible Values

RotationAndTranslation
Mirroring

TranslationalDOFBehavior

This enum is used to set the translational degree of freedom as fixed or free.

Possible Values

Fixed
Free

TriangleSurfaceMesher

Used to specify the surface meshing algorithm to be used by patch conforming mesher. The default value is ProgramControlled.

Possible Values

ProgramControlled	Mesher determines usage of Delaunay or Advancing Front algorithm.
AdvancingFront	Mesher uses Advancing Front algorithm but can switch to Delaunay if problems occur.

TrimType

Trim Type to control trimming such as keeping it on, off or Program Controllled.

Possible Values

ProgramControlled
On
Off

TripletType

An enumeration to describe the type of Triptlet that is used in Coordinate systems, Reference frames, and Vector objects.

Possible Values

Cartesian
Cylindrical
Spherical

TurbulenceSpecification

No details are provided for this entry.

Possible Values

Laminar
SpalartAllmaras
KEpsilonStandard
KEpsilonRealizable
KOmegaStandard
KOmegaSst

TypeOfInitialSampling

Enumeration for the Initial Sampling Type.

Possible Values

E_Screening	Screening
E_OSF	Optimal Space-Filling

E_ConstrainedSampling	Constrained Sampling

uint

This type represents an unsigned integer number. Negative values are invalid.

UniversalVectorDefineBy

Options for defining the vector

Possible Values

Components
MagnitudeAndDirection

UpdatableEntityState

Types of state an updatable entity can have.

Possible Values

Unknown	entity state can not be defined.
OutOfDate	entity is out of date.
UpToDate	entity is up to date.
Error	last update of the entity gave an error.
OutOfDateWithError	entity is out of date and last update of the entity gave an error.
UpToDateWithError	entity is up of date and last update of the entity gave an error.
RefreshRequired	entity needs a refresh.

UpdateContinuation

No details are provided for this entry.

Possible Values

None
TargetOnly
All

UpdateErrorBehavior

No details are provided for this entry.

Possible Values

Stop
SkipDesignPoint

Continue

UpdateStiffnessControlType

Possible options to update the stifness in a contact analysis.

Possible Values

ProgramControlled
EachEquilibriumIteration
EachEquilibriumIterationAggressive
Never
EachEquilibriumIterationNominal
EachEquilibriumIterationExponential

UseAdvancedLocalSizeFunction

Used to specify which advanced size function option should be used by the mesher.

Possible Values

ProgramControlled
ProximityCurvature
Curvature
Proximity
Fixed

UseAdvancedSizeFunction

Used to specify which advanced size function option should be used by the mesher.

Possible Values

Adaptive
ProximityCurvature
Curvature
Proximity
Fixed

UseAllProcessors

Used to specify the usage of all available processors for parallel part-based meshing. The default value is Yes.

Possible Values

Yes	Automatically detects and utilizes all available processors for meshing parts of an assembly in parallel.

No	Manually specify the limit to the number of processors that can be utilized for parallel part-based meshing.

UsePostSmoothing

Used to specify whether post boundary smoothing will be performed. The default value is Yes.

Possible Values

Yes
No

UserObject

No details are provided for this entry.

ValueDefineBy

This propertry decides whether it should use program controlled value or a manually entered factor or value.

Possible Values

Unknown
kProgramControlled
kPhysicalValue
kFactor
kAutoDetectValue

VariableConversionOption

Enumeration to specify the conversion of data

Possible Values

NoConversion
AverageSharedNodes
AverageNodesToElement
AverageNodesToFace
DistributeElementToNodesEqually
DistributeFaceToNodesEqually
AverageCornerToMidsideNodes

VariableExposure

Level of exposure for variables in the CDI

Possible Values

Standard	Default
Expert	Expert

VariableRangeOption

The variable range options

Possible Values

Local
UserSpecified

VariableStyle

Styles that can be used to display the variable. Default is Line

Possible Values

None
Line
Spline
Step
Bar
Default

VectorCalculationMethod

Options for calculating a vector from geometry selection

Possible Values

Automatic
HitPointNormal
WeightedAverageNormal
HitPointTangent
NearestVertexTangent
WeightedAverageTangent
CenterAxis
TwoVertices

VectorDefineBy

This enum is a collection of methods by which a vector can be defined : Directional Components, Magnitude and direction or normal to a surface.

Possible Values

Components
MagnitudeAndDirection
NormalToSurface

VectorType

Motion vector types used in mechanics.

Possible Values

Unknown
Displacement
Velocity
Acceleration
Rotation
AngularVelocity
AngularAcceleration

ViscosityType

viscosity type

Possible Values

Sutherland	Viscosity calculated using Sutherland's Law (2 coefficient method)
Dynamic	User-specified dynamic viscosity
Kinematic	User specified kinematic viscosity

VolumeDataInput

This class defines input parameters used for Rapid Results Exploration(RRE).

VoluteType

Volute style option

Possible Values

Elliptic	Elliptic/circular cross sections
Rectangular	Rectangular cross sections

WallMotionSpecification

No details are provided for this entry.

Possible Values

Stationary
Rotating

WeightingType

Available expression weight types

Possible Values

Simple
Length
Area
Volume
Mass

WrapMethod

Used to specify the wrapping method to be used for the meshing region. The default value is ShrinkWrap.

Possible Values

ShrinkWrap
CutWrap

XAxisQuantity

Enum for quantity used for X-axis.

Possible Values

Iteration
FlowTime
TimeStep

YesOrNo

This is used for activating or deactivating a setting.

Possible Values

kYes
kNo

YN

This is enum for yes/no option

Possible Values

ICYes
ICNo

Zoning

Used to specify the option to be used for zoning. The default value is Body.

Possible Values

Part	Creates a single zone for each part.
Body	Creates a separate zone for each body.
Face	Creates a separate zone for each face.

Index

E

F

G

I

J

K

T

U

V