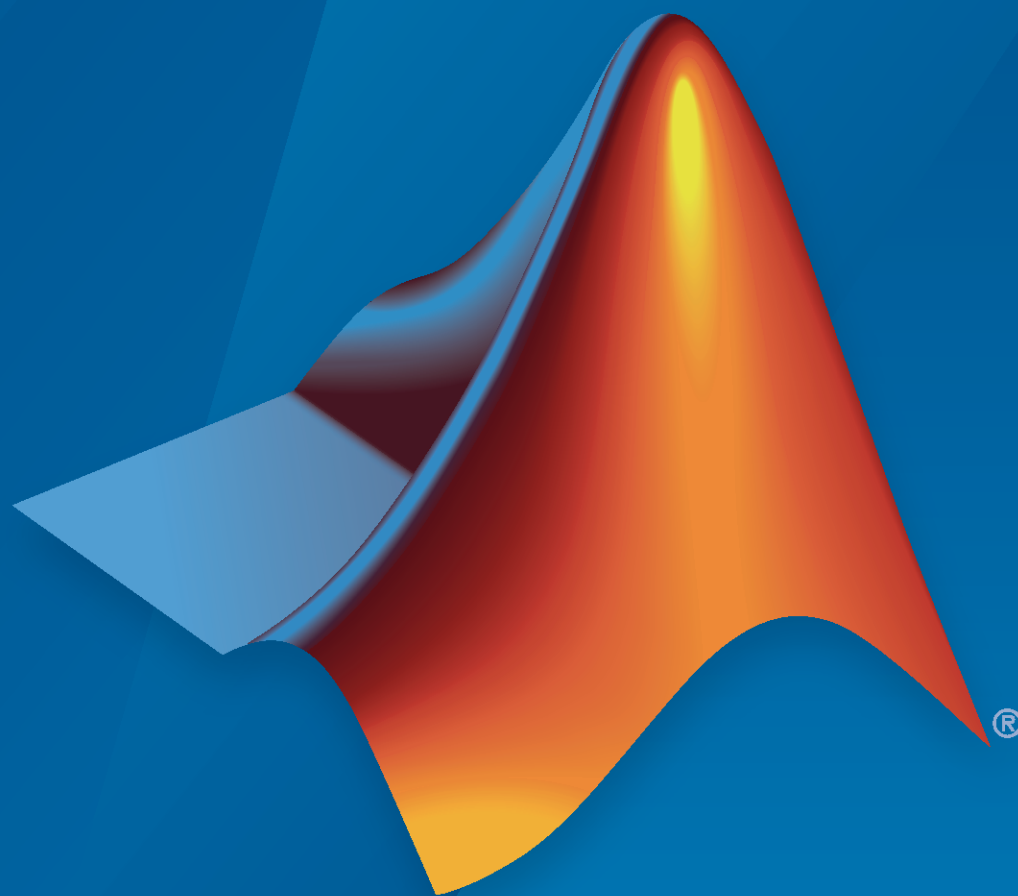# Simscape™ Multibody™ Link

Reference

MATLAB®

MathWorks®

How to Contact MathWorks

Latest news: www.mathworks.com

Sales and services: www.mathworks.com/sales_and_services

User community: www.mathworks.com/matlabcentral

Technical support: www.mathworks.com/support/contact_us

Phone: 508-647-7000

The MathWorks, Inc.
1 Apple Hill Drive
Natick, MA 01760-2098

Trademarks

MATLAB and Simulink are registered trademarks of The MathWorks, Inc. See www.mathworks.com/trademarks for a list of additional trademarks. Other product or brand names may be trademarks or registered trademarks of their respective holders.

Patents

MathWorks products are protected by one or more U.S. patents. Please see www.mathworks.com/patents for more information.

Contents

Register and Use SolidWorks Add-In

3

Register and Use the Inventor Add-In

This chapter describes how to register Simscape Multibody Link software to the Autodesk Inventor CAD platform as an Inventor add-in tool. You must complete the registration before you can export a CAD assembly in Simscape Multibody format.

Enable Simscape Multibody Link Plugin in Inventor Plugin

In this section...
"About the Plugin" on page 1-2
"Enable the Plugin" on page 1-2
"Updating the Plugin" on page 1-2

About the Plugin

The Simscape Multibody Link plugin provides the primary interface for exporting CAD assemblies into Simscape Multibody models. The plugin is compatible with three CAD applications: Autodesk Inventor®, Creo™ Parametric, and SolidWorks®. If you use a different CAD application, you can still export CAD assemblies using the Simscape Multibody Import XML schema.

Enable the Plugin

Once you have downloaded and installed the Simscape Multibody Link plugin, you must enable it on your Autodesk Inventor application. To do this, at the MATLAB® command prompt, enter `smlink_linkinv`. A Simscape Multibody Link menu appears in the Inventor menu when you start or open a CAD assembly.

If your computer has more than one Inventor application, the `smlink_linkinv` command adds the Simscape Multibody Link plugin to all installations simultaneously. However, you must select the Simscape Multibody Link check box in the Add-Ins dialog box individually for each installation you want to export CAD from.

Updating the Plugin

If you are updating your Simscape Multibody Link version, you must disable the current version first. You do this by entering `smlink_unlinkinv` at the MATLAB command prompt. Then, after downloading and installing the new plugin version, follow the procedure for enabling the plugin on your Inventor application.

Constraints and Joints

In this section...

CAD Constraints and Entities

You create a CAD assembly by applying joints and constraints between parts. Each joint or constraint defines a kinematic relationship between constraint entities on the parts it connects. Angle and Insert are examples of constraints. Planes, lines, and points are examples of constraint entities.

Consider the connection between the upper-arm and forearm parts of a robotic-arm assembly. Such a connection allows the two parts to rotate with respect to each other about a single axis and therefore has one rotational degree of freedom. You specify this degree of freedom by applying constraints such as:

- One mate constraint between the cylindrical hinge surfaces of the two parts. This constraint reduces the joint degrees of freedom to two—one translational, along the cylindrical axis, and one rotational, about the same axis.

- One mate constraint between two planes normal to the cylindrical axis. This constraint removes the translational degree of freedom that the first mate constraint provides between the two parts. The combined constraint set allows the parts only to rotate about the common cylindrical axis.

The figure shows the constrained surfaces on the upper-arm and forearm parts.
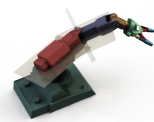
Simscape Multibody software can successfully import an Autodesk Inventor assembly model with any joint (all are supported) and the following constraints:

- Angle offset △
- Mate/flush ⬛
- Insert ⬛

Simscape Multibody software can successfully import CAD assemblies whose constraints join these constraint entities:

- Circle/arc ◯
- Cone ▲

- Cylinder
- Line
- Plane
- Point

Supported constraints are valid only for certain entity pairs. The table shows the entity pairs compatible with the supported constraints. This table is symmetric with respect to the diagonal row.

Simscape Multibody Joint and Constraint Blocks

Joint and Constraint blocks are the Simscape Multibody equivalent of Autodesk Inventor constraints. They apply between two bodies the kinematic constraints that determine how they can move. A Revolute Joint block is an example. This block removes five degrees of freedom between two bodies, allowing them only to rotate about a common axis.

Consider the connection between the upper-arm and forearm parts of the robotic arm assembly. This connection provides a single rotational degree of freedom and therefore acts as a revolute joint. During CAD import, the constraints between the two parts translate into a Revolute Joint block between two rigid body subsystems. The figure shows this joint block in an imported model.

Joint blocks are assortments of joint primitives, basic yet complete joints of various kinds you cannot decompose any further—at least without losing behavior such as the rotational-translational coupling of the lead screw joint. Joint primitives range in number from zero in the Weld Joint block to six in the Bushing Joint block. There are five joint primitives:

- Prismatic — Allows translation along a single standard axis (x, y, or z). Joint blocks can contain up to three prismatic joint primitives, one for each translational DoF. Prismatic primitives are labelled P*, where the asterisk denotes the axis of motion, e.g., Px, Py, or Pz.

- Revolute — Allows rotation about a single standard axis (x, y, or z). Joint blocks can contain up to three revolute joint primitives, one for each rotational DoF. Revolute primitives are labelled R*, where the asterisk denotes the axis of motion, e.g., Rx, Ry, or Rz.

- Spherical — Allows rotation about any 3-D axis, [x, y, z]. Joint blocks contain no more than one spherical primitive, and never in combination with revolute primitives. Spherical primitives are labelled S.

- Lead Screw Primitive — Allows coupled rotation and translation on a standard axis (e.g., z). This primitive converts between rotation at one end and translation at the other. Joint blocks contain no more than one lead screw primitive. Lead screw primitives are labeled LS*, where the asterisk denotes the axis of motion.
- Constant Velocity Joint — Allows rotation at constant velocity between intersecting though arbitrarily aligned shafts. Joint blocks contain no more than one constant velocity primitive. Constant velocity primitives are labelled CV.

The table shows the Joint blocks supported during CAD import, the joint primitives the blocks contain, and the degrees of freedom they provide. T and R denote translational and rotational DOFs. Joint blocks not shown are not supported.

Joint Block	Joint Primitives							Joint DoFs
6-DOF Joint	Px	Py	Pz	Rx	Ry	Rz	S	3 T + 3 R
Cartesian Joint	Px	Py	Pz	Rx	Ry	Rz	S	3 T + 0 R
Cylindrical Joint	Px	Py	Pz	Rx	Ry	Rz	S	1 T + 1 R
Planar Joint	Px	Py	Pz	Rx	Ry	Rz	S	2 T + 1 R
Prismatic Joint	Px	Py	Pz	Rx	Ry	Rz	S	1 T + 0 R
Rectangular Joint	Px	Py	Pz	Rx	Ry	Rz	S	2 T + 0 R
Revolute Joint	Px	Py	Pz	Rx	Ry	Rz	S	0 T + 1 R
Spherical Joint	Px	Py	Pz	Rx	Ry	Rz	S	0 T + 3 R
Telescoping Joint	Px	Py	Pz	Rx	Ry	Rz	S	1 T + 3 R
Universal Joint	Px	Py	Pz	Rx	Ry	Rz	S	0 T + 2 R
Weld Joint	Px	Py	Pz	Rx	Ry	Rz	S	0 T + 0 R

By defining the relative degrees of freedom between two bodies, Joint blocks partially determine how these bodies can move with respect to each other. Constraint blocks enable you to impose additional restrictions on their motion. CAD constraints can translate into these Constraint blocks:

- Angle Constraint
- Distance Constraint

CAD Constraint-Simscape Multibody Joint Mapping

The table shows some of the constraint combinations you can use to obtain a specific joint block during CAD import. Different constraint combinations can map into the same joint. This happens if the constraint combinations provide the same degrees of freedom between the parts they join. For a legend of the icons in the table, see "CAD Constraints and Entities" on page 1-3.

Joint Block	Constraint I	Entities I	Constraint II	Entities II	Offsets	Notes
Cartesian Joint	⬙	▪ ▪	⬙	▪ ▪	$\theta_I = \theta_{II} = 0°$	
Cylindrical Joint	⬚	╱ ╱			$d_I = 0$	
Planar Joint	⬚	▪ ▪			$d_I = 0$	
Prismatic Joint	⬚	╱ ╱	⬚	▪ ▪	$d_I = d_{II} = 0$	1
	⬚	╱ ╱	⬙	▪ ▪	$d_I = 0, \theta_{II} = 90°$	
	⬚	▪ ▪	⬚	▪ ▪	$d_I, d_{II} \geq 0$	2
Rectangular Joint	⬚	▪ ▪	⬙	▪ ▪	$d_I, \theta_{II} \geq 0$	
	⬚	▪ ▪	⬙	╱ ╱	$d_I \geq 0, \theta_{II} = 0°$	
Revolute Joint	⬓	○ ○				3
	⬚	╱ ╱	⬚	▪ ▪	$d_I = 0, d_{II} \geq 0$	
Spherical Joint	⬚	▪ ▪			$d_I = 0$	

Constraint pairs marked with a note number must satisfy additional requirements. This list outlines those requirements:

1 Lines in constraint I must be parallel to planes in constraint II.

2 Planes in constraint I must not be parallel to planes in constraint II.

3 Lines in constraint I must be perpendicular to planes in constraint II.

CAD Constraint-Simscape Multibody Constraint Mapping

The table shows the Constraint blocks that different constraint combinations map into. Different constraints can map into the same Constraint block if they provide the same degrees of freedom. Angle offsets must be either 0 or 90 degrees. Other offsets are not supported. For a legend of the icons in the table, see "CAD Constraints and Entities" on page 1-3.

Constraint Block	CAD Constraint	CAD Entities	Offsets
Angle (Perpendicular)	⬙	▪ ▪, 🛢 🛢, ▲ ▲, or ╱ ╱	$\theta = 90°$
Angle (Parallel)	⬙	▪ ▪, 🛢 🛢, ▲ ▲, or ╱ ╱	$\theta = 0°$
Distance	⬚	• •	$d \geq 0$

Special Constraint Translation Cases

The lack of constraints between parts, combinations of constraints that fully restrict motion between parts, and unsupported constraints are special translation cases. Here is how Simscape Multibody software handles these cases:

- Unsupported constraints between parts translate into rigid connections between rigid bodies. The rigid connections can be in the form of Weld Joint blocks or direct frame connection lines between the rigid bodies. These connections are meant to be temporary. After CAD import, search your model for rigid connections and, if appropriate, replace them with other Joint and Constraint blocks.

- Combinations of constraints that fully restrict motion between parts translate into rigid connections between rigid bodies. The rigid connections can be in the form of Weld Joint blocks or direct frame connection lines between the rigid bodies. These rigid connections accurately model the degrees of freedom between the two bodies and do not need to be replaced.

- The absence of a constraint between a part and the rest of the assembly translates into a 6-DOF Joint block between a rigid body and the World frame. This joint block provides the rigid body the six degrees of freedom that the CAD part has within the CAD assembly.

Constraint-Joint Mapping in Simscape Multibody First Generation

In this section...
"Degrees of Freedom in Simscape Multibody" on page 1-9
"CAD Constraint – Simscape Multibody Joint Mapping" on page 1-9
"Supported Constraint Entity" on page 1-10
"Supported Constraint Entity Combinations" on page 1-10
"Limitations" on page 1-11

Warning This content refers to Simscape Multibody Link First Generation software. First-generation features are slated to be deprecated and should be avoided. To prevent compatibility issues in future releases, the Simscape Multibody Link plug-in no longer supports the translation of CAD models into Simscape Multibody Link First Generation models.

In Autodesk Inventor, unconstrained parts have six mechanical degrees of freedom (DoFs) that describe how the parts move with respect to each other. Of the six degrees of freedom, three are rotational and three are translational. Applying a constraint between two parts eliminates degrees of freedom between the two parts. Constraints can remove between zero and six degrees of freedom.

Degrees of Freedom in Simscape Multibody

Simscape Multibody First Generation assigns zero degrees of freedom to an unconstrained rigid body. Connecting the rigid body to a joint or constraint block increases the mechanical degrees of freedom available to the rigid body.

Rigid Body Condition	First-Generation DoF
Not connected to joints, constraints, or World Frame	0
Connected to Joints or Constraints blocks	Add degrees of freedom according to joint or constraint

CAD Constraint – Simscape Multibody Joint Mapping

During CAD export, Simscape Multibody Link maps Inventor constraints between parts to Simscape Multibody joints between rigid bodies. CAD constraints and Simscape Multibody joints do not follow a one-to-one correspondence — multiple constraints can map into a single joint. All Simscape Multibody joints contain a combination of three joint primitives: Prismatic, Revolute, and Spherical. The Weld Joint block contains zero joint primitives, and therefore zero degrees of freedom. The following table identifies the degrees of freedom of each joint primitive.

Primitive	Abbreviation	Motion Type	Number of DoFs
Prismatic	P	Translational	1
Revolute	R	Rotational	1

Primitive	Abbreviation	Motion Type	Number of DoFs
Spherical	S	Rotational	3

Supported Constraint Entity

Depending on the constraint combination, Simscape Multibody Link utility supports the following Inventor constraint entities:

Entity	Description
Circle/Arc	Circular edge/arc sketch segment*
Ellipse/Arc	Elliptical edge/arc sketch segment*
Cone	Conical face
Cylinder	Cylindrical face
Line	Linear edge/sketch segment/reference axis
Plane	Reference plane or planar face
Point	Vertex/sketch point/reference point

* A complete circle or ellipse is a special case of a circular or elliptical arc.

Supported Constraint Entity Combinations

The following sections list the constraint-entity combinations that Simscape Multibody Link supports for different constraint types.

Note If the Simscape Multibody Link exporter cannot translate a constraint–constraint entity combination into a supported Simscape Multibody joint with DoFs, it converts the combination into a weld (W) primitive.

Coincident Constraint

The following table identifies supported constraint-entity combinations for the Coincident constraint. A ✓ indicates the combination is supported.

		Constraint-Entity 2					
		Point	**Line**	**Plane**	**Cylinder**	**Cone**	**Circle/Arc**
Constraint-Entity 1	**Point**	✓					
	Line		✓	✓			
	Plane		✓	✓			✓
	Cylinder				✓	✓	✓
	Cone				✓	✓	✓
	Circle/Arc			✓	✓	✓	✓

Concentric Constraint

The following table identifies supported constraint-entity combinations for the Concentric constraint. A ✓ indicates the combination is supported.

		Constraint Entity 2					
		Point	**Line**	**Plane**	**Cylinder**	**Cone**	**Circle/Arc**
Constraint Entity 1	**Point**						
	Line					✓	✓
	Plane						
	Cylinder				✓	✓	✓
	Cone		✓		✓	✓	✓
	Circle/Arc		✓		✓	✓	✓

Distance Constraint

The following table identifies supported constraint-entity combinations for the Distance constraint. A ✓ indicates the combination is supported.

		Constraint Entity 2					
		Point	**Line**	**Plane**	**Cylinder**	**Cone**	**Circle/Arc**
Constraint Entity 1	**Point**	✓		✓			
	Line			✓			
	Plane	✓	✓	✓			
	Cylinder						
	Cone						
	Circle/Arc						

Angle Constraint

The following table identifies supported constraint-entity combinations for the Angle constraint. A ✓ indicates the combination is supported.

		Constraint Entity 2					
		Point	**Line**	**Plane**	**Cylinder**	**Cone**	**Circle/Arc**
Constraint Entity 1	**Point**						
	Line		✓				
	Plane			✓			
	Cylinder						
	Cone						
	Circle/Arc						

Limitations

The following limitation applies to CAD export from Inventor.

Weld is Default Joint

If the Simscape Multibody Link utility fails to translate a CAD constraint, a Weld joint replaces the constraint.

Restriction on Point-Point Distance Mate

For Simscape Multibody Link to successfully map the CAD point-point distance constraint onto a Simscape Multibody spherical-spherical massless connector, the constraint must not connect to any other constraint.

Configure Simscape Multibody Link

In this section...
"Simscape Multibody Link Settings" on page 1-13
"Dialog Box" on page 1-13

Warning This content refers to Simscape Multibody Link First Generation software. First-generation features are slated to be deprecated and should be avoided. To prevent compatibility issues in future releases, the Simscape Multibody Link plug-in no longer supports the translation of CAD models into Simscape Multibody Link First Generation models.

Simscape Multibody Link Settings

The Simscape Multibody Link add-in tool provides a Settings option. Use the option to specify:

- Tolerances — linear, angular, and relative

To access the Settings parameters:

1. Open the assembly to export.
2. In the menu bar, click **Add-Ins > Settings**.

The Settings dialog box opens.

Dialog Box

The dialog box contains two panes:

- **Assembly Tolerances** — Specifies linear, angular, and relative tolerances of exported assembly.

Enter the export tolerances for a CAD assembly. During the conversion of CAD constraints to Simscape Multibody joints, Simscape Multibody Link compares the spacing, alignment, and relative numerical errors with the export tolerances.

Field	Default Value	Purpose	Default	Unit
Linear tolerance	1e-005	Smallest significant length difference	1e-5	Unit used in assembly. The default is mm
Angular tolerance	1e-005	Smallest significant angle difference	1e-5	Unit used in assembly. The default is rad
Relative roundoff tolerance	1e-012	Smallest significant relative numerical difference	1e-12	—

Register and Use the Creo Add-In

This chapter describes how to register Simscape Multibody Link software to the Creo (Pro/ENGINEER) CAD platform as a Pro/TOOLKIT application. You must complete the registration before you can export a CAD assembly in Simscape Multibody format.

Enable Simscape Multibody Link Plugin in Creo-Pro/E

In this section...
"About the Plugin" on page 2-2
"Update Registry File" on page 2-2
"Update Configuration File" on page 2-3
"Verify Plugin Is Enabled" on page 2-3
"Updating Plugin Version" on page 2-3

About the Plugin

The Simscape Multibody Link plugin provides the primary interface for exporting CAD assemblies into Simscape Multibody software. The plugin is compatible with three CAD applications: Autodesk Inventor, Creo Parametric, and SolidWorks. If you use a different CAD application, you can still export CAD assemblies using the Simscape Multibody Import XML schema.

To enable the plugin in Creo or Pro/E, you must manually update the contents of protk.dat, a registry file. If you store this file outside of the Creo or Pro/E root and startup folders, you must also update the contents of config.pro, a configuration file. If you have not done so, download and install the Simscape Multibody Link plugin before continuing.

Update Registry File

Locate and open the protk.dat registry file for your Creo or Pro/E application. Search for this file in the application's root and startup folders. Add this code to the file, replacing *matlabroot* with the absolute path to your MATLAB root folder and *os* with your operating system architecture, for example, `win64`.

```
name Simscape Multibody Link
startup dll
exec_file matlabroot/bin/os/cl_proe2sm.dll
text_dir matlabroot/toolbox/physmod/smlink/cad_systems/proe/text
unicode_encoding false
end
```

If you cannot find or edit the file, create a new text file with a name of your choosing and .dat extension. This is your registry file. Save it in an accessible folder.

Example

Suppose your MATLAB root folder is C:/Program Files/MATLAB/R2015b and your operating system architecture is win64. If you are using Creo Parametric software, then your .dat registry file must contain this code:

```
name Simscape Multibody Link
startup dll
exec_file C:/Program Files/MATLAB/R2015b/bin/win64/cl_proe2sm.dll
text_dir C:/Program Files/MATLAB/R2015b/toolbox/physmod/smlink/cad_systems/proe/text
unicode_encoding false
end
```

Update Configuration File

If your .dat registry file is located outside of the root and startup folders, you must also update the config.pro file. Search for this file in your Creo or Pro/E startup folder. Open the file and add this code:

toolkit_registry_file *<full path to your .dat registry file>*

If you cannot find the file, create a new text file and save it as config.pro in your startup folder. You may need administrator privileges.

Example

Suppose you saved your .dat registry file as myprotk.dat in the folder C:/Users/jdoe/Documents/Creo. Here is the code you must add to your config.pro file:

toolkit_registry_file C:/Users/jdoe/Documents/Creo/myprotk.dat

Verify Plugin Is Enabled

Start your Creo Parametric or Pro/ENGINEER application. In Creo Parametric, search for a **Tools** menu with a `Simscape Multibody Link` option. In Pro/ENGINEER Wildfire, search for a **Simscape Multibody Link** menu. If the option or menu is present, the plugin is enabled. You are ready to start exporting CAD assemblies.

Updating Plugin Version

If you are updating your Simscape Multibody Link version, you must update the exec_file and text_dir paths in your .dat registry file. Suppose you upgraded your MATLAB and Simscape Multibody Link software from an older release to R2015b. If the MATLAB root folder is otherwise unchanged, the new exec_file and text_dir lines must be:

```
exec_file C:/Program Files/MATLAB/R2015b/bin/win64/cl_proe2sm.dll
text_dir C:/Program Files/MATLAB/R2015b/toolbox/physmod/smlink/cad_systems/proe/text
```

Constraints and Joints

In this section...
"CAD Constraints and Entities" on page 2-4
"Simscape Multibody Joint and Constraint Blocks" on page 2-5
"CAD Constraint-Simscape Multibody Joint Mapping" on page 2-7
"CAD Constraint-Simscape Multibody Constraint Mapping" on page 2-8
"Special Constraint Translation Cases" on page 2-8

CAD Constraints and Entities

You create a CAD assembly by applying constraints between parts. Each constraint defines a kinematic relationship between constraint entities on the parts it connects. Angle and Centered are examples of constraints. Planes, lines, and points are examples of constraint entities.

Consider the connection between the upper-arm and forearm parts of a robotic-arm assembly. Such a connection allows the two parts to rotate with respect to each other about a single axis and therefore has one rotational degree of freedom. You specify this degree of freedom by applying constraints such as:

- One Centered constraint between the cylindrical hinge surfaces of the two parts. This constraint reduces the joint degrees of freedom to two—one translational, along the cylindrical axis, and one rotational, about the same axis.

- One Coincident constraint between two planes normal to the cylindrical axis. This constraint removes the translational degree of freedom the Centered constraint provides between the two parts. The combined constraint set allows the parts only to rotate about the common cylindrical axis.

The figure shows the constrained surfaces on the upper-arm and forearm parts.
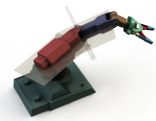
Simscape Multibody software can successfully import CAD assemblies that contain these constraints:

- Angle Offset
- Coincident
- Centered
- Distance
- Parallel
- Normal

Simscape Multibody software can successfully import CAD assemblies whose constraints join these constraint entities:

- Circle/arc ○
- Cone ▲
- Cylinder ▯
- Line ╱
- Plane ▰
- Point ✳

Supported constraints are valid only for certain entity pairs. The table shows the entity pairs compatible with the supported constraints. This table is symmetric with respect to the diagonal row.

Simscape Multibody Joint and Constraint Blocks

Joint and Constraint blocks are the Simscape Multibody equivalent of CAD constraints. They apply between two bodies the kinematic constraints that determine how they can move. A Revolute Joint block is an example. This block removes five degrees of freedom between two bodies, allowing them only to rotate about a common axis.

Consider the connection between the upper-arm and forearm parts of the robotic arm assembly. This connection provides a single rotational degree of freedom and therefore acts as a revolute joint. During CAD import, the constraints between the two parts translate into a Revolute Joint block between two rigid body subsystems. The figure shows this joint block in an imported model.

2-5

Joint blocks are assortments of joint primitives, basic yet complete joints of various kinds you cannot decompose any further—at least without losing behavior such as the rotational-translational coupling of the lead screw joint. Joint primitives range in number from zero in the Weld Joint block to six in the Bushing Joint block. There are five joint primitives:

- Prismatic — Allows translation along a single standard axis (x, y, or z). Joint blocks can contain up to three prismatic joint primitives, one for each translational DoF. Prismatic primitives are labelled P*, where the asterisk denotes the axis of motion, e.g., Px, Py, or Pz.

- Revolute — Allows rotation about a single standard axis (x, y, or z). Joint blocks can contain up to three revolute joint primitives, one for each rotational DoF. Revolute primitives are labelled R*, where the asterisk denotes the axis of motion, e.g., Rx, Ry, or Rz.

- Spherical — Allows rotation about any 3-D axis, [x, y, z]. Joint blocks contain no more than one spherical primitive, and never in combination with revolute primitives. Spherical primitives are labelled S.

- Lead Screw Primitive — Allows coupled rotation and translation on a standard axis (e.g., z). This primitive converts between rotation at one end and translation at the other. Joint blocks contain no more than one lead screw primitive. Lead screw primitives are labeled LS*, where the asterisk denotes the axis of motion.
- Constant Velocity Joint — Allows rotation at constant velocity between intersecting though arbitrarily aligned shafts. Joint blocks contain no more than one constant velocity primitive. Constant velocity primitives are labelled CV.

The table shows the Joint blocks supported during CAD import, the joint primitives the blocks contain, and the degrees of freedom they provide. T and R denote translational and rotational DOFs. Joint blocks not shown are not supported.

Joint Block								Joint DoFs
6-DOF Joint	Px	Py	Pz	Rx	Ry	Rz	S	3 T + 3 R
Cartesian Joint	Px	Py	Pz	Rx	Ry	Rz	S	3 T + 0 R
Cylindrical Joint	Px	Py	Pz	Rx	Ry	Rz	S	1 T + 1 R
Planar Joint	Px	Py	Pz	Rx	Ry	Rz	S	2 T + 1 R
Prismatic Joint	Px	Py	Pz	Rx	Ry	Rz	S	1 T + 0 R
Rectangular Joint	Px	Py	Pz	Rx	Ry	Rz	S	2 T + 0 R
Revolute Joint	Px	Py	Pz	Rx	Ry	Rz	S	0 T + 1 R
Spherical Joint	Px	Py	Pz	Rx	Ry	Rz	S	0 T + 3 R
Telescoping Joint	Px	Py	Pz	Rx	Ry	Rz	S	1 T + 3 R
Universal Joint	Px	Py	Pz	Rx	Ry	Rz	S	0 T + 2 R
Weld Joint	Px	Py	Pz	Rx	Ry	Rz	S	0 T + 0 R

By defining the relative degrees of freedom between two bodies, Joint blocks partially determine how these bodies can move with respect to each other. Constraint blocks enable you to impose additional restrictions on their motion. CAD constraints can translate into these Constraint blocks:

- Angle Constraint
- Distance Constraint

CAD Constraint-Simscape Multibody Joint Mapping

The table shows some of the constraint combinations you can use to obtain a specific joint block during CAD import. Different constraint combinations can map into the same joint. This happens if the constraint combinations provide the same degrees of freedom between the parts they join. For a legend of the icons in the table, see "CAD Constraints and Entities" on page 2-4.

Joint Block	Constraint I	Entities I	Constraint II	Entities II	Notes
Cartesian Joint	ⅠⅭ	■ ■	ⅠⅭ	■ ■	
Cylindrical Joint	ⲧ	╱ ╱			
	⊥	⬤ ⬤			
Planar Joint	ⲧ	■ ■			
Prismatic Joint	⊥	⬤ ⬤	ⲧ or ⅠⅭ	■ ■	1
	ⲧ	╱ ╱	ⲧ or ⅠⅭ	■ ■	2
	ⲧ or ⅠⅭ	■ ■	ⲧ or ⅠⅭ	■ ■	3
Rectangular Joint	ⲧ or ⅠⅭ	■ ■	△, ⅠⅭ, or ⬍	■ ■	
Revolute Joint	ⲧ	╱ ╱	ⲧ or ⅠⅭ	■ ■	4
Spherical Joint	ⲧ	● ●			

Constraint pairs marked with a note number must satisfy additional requirements. This list outlines those requirements:

1 Cylinder axes in constraint I must be parallel to planes in constraint II.

2 Lines in constraint I must be parallel to planes in constraint II.

3 Planes in constraint I must not be parallel to planes in constraint II.

4 Lines in constraint I must be perpendicular to planes in constraint II.

CAD Constraint-Simscape Multibody Constraint Mapping

The table shows the Constraint blocks that different constraint combinations map into. Different constraints can map into the same Constraint block if they provide the same degrees of freedom. For a legend of the icons in the table, see "CAD Constraints and Entities" on page 2-4.

Constraint Block	CAD Constraint	CAD Entities	Offsets
Angle (Perpendicular)	⬍	■ ■, ⬤ ⬤, △ △, or ╱ ╱	
Angle (Parallel)	ⅠⅭ	■ ■	
Distance	ⅠⅭ	● ●	$d \geq 0$

Special Constraint Translation Cases

The lack of constraints between parts, combinations of constraints that fully restrict motion between parts, and unsupported constraints are special translation cases. Here is how Simscape Multibody software handles these cases:

- Unsupported constraints between parts translate into rigid connections between rigid bodies. The rigid connections can be in the form of Weld Joint blocks or direct frame connection lines between the rigid bodies. These connections are meant to be temporary. After CAD import, search your model for rigid connections and, if appropriate, replace them with other Joint and Constraint blocks.

- Combinations of constraints that fully restrict motion between parts translate into rigid connections between rigid bodies. The rigid connections can be in the form of Weld Joint blocks or direct frame connection lines between the rigid bodies. These rigid connections accurately model the degrees of freedom between the two bodies and do not need to be replaced.

- The absence of a constraint between a part and the rest of the assembly translates into a 6-DOF Joint block between a rigid body and the World frame. This joint block provides the rigid body the six degrees of freedom that the CAD part has within the CAD assembly.

Constraint-Joint Mapping in Simscape Multibody First Generation

In this section...

Warning This content refers to Simscape Multibody Link First Generation software. First-generation features are slated to be deprecated and should be avoided. To prevent compatibility issues in future releases, the Simscape Multibody Link plug-in no longer supports the translation of CAD models into Simscape Multibody Link First Generation models.

In Pro/ENGINEER, unconstrained parts have six mechanical degrees of freedom (DoFs) that describe how the parts move with respect to each other. Of the six degrees of freedom, three are rotational and three are translational. Applying a constraint between two parts eliminates degrees of freedom between the two parts. Constraints can remove between zero and six degrees of freedom.

Degrees of Freedom in Simscape Multibody

Simscape Multibody First Generation assigns zero degrees of freedom to an unconstrained rigid body. Connecting the rigid body to a joint or constraint block increases the mechanical degrees of freedom available to the rigid body.

Rigid Body Condition	First-Generation DoF
Not connected to joints, constraints, or World Frame	0
Connected to Joints or Constraints blocks	Add degrees of freedom according to joint or constraint

CAD Constraint – Simscape Multibody Joint Mapping

During CAD export, Simscape Multibody Link maps Pro/ENGINEER constraints between parts to Simscape Multibody joints between rigid bodies. CAD constraints and Simscape Multibody joints do not follow a one-to-one correspondence — multiple constraints can map into a single joint. All Simscape Multibody joints contain a combination of three joint primitives: Prismatic, Revolute, and Spherical. The Weld Joint block contains zero joint primitives, and therefore zero degrees of freedom. The following table identifies the degrees of freedom of each joint primitive.

Primitive	Abbreviation	Motion Type	Number of DoFs
Prismatic	P	Translational	1
Revolute	R	Rotational	1

Primitive	Abbreviation	Motion Type	Number of DoFs
Spherical	S	Rotational	3

Supported Constraint Entity

Depending on the constraint combination, Simscape Multibody Link utility supports the following Creo constraint entities:

Entity	Description
Circle/Arc	Circular edge/arc sketch segment*
Ellipse/Arc	Elliptical edge/arc sketch segment*
Cone	Conical face
Cylinder	Cylindrical face
Line	Linear edge/sketch segment/reference axis
Plane	Reference plane or planar face
Point	Vertex/sketch point/reference point

* A complete circle or ellipse is a special case of a circular or elliptical arc.

Supported Constraint Entity Combinations

The following sections list the constraint-entity combinations that Simscape Multibody Link supports for different constraint types.

Note If the Simscape Multibody Link exporter cannot translate a constraint–constraint entity combination into a supported Simscape Multibody joint with DoFs, it converts the combination into a weld (W) primitive.

Coincident Constraint

The following table identifies supported constraint-entity combinations for constraints:

- Align without offset
- Mate without offset
- Point on Line
- Edge on Surface
- Point on Surface

A ✓ indicates the combination is supported.

		Constraint-Entity 2					
		Point	**Line**	**Plane**	**Cylinder**	**Cone**	**Circle/Arc**
Constraint-Entity 1	**Point**	✓					
	Line		✓	✓			

	Point	Line	Plane	Cylinder	Cone	Circle/Arc
Plane		✓	✓			✓
Cylinder				✓	✓	✓
Cone				✓	✓	✓
Circle/Arc			✓	✓	✓	✓

Insert Constraint

The following table identifies supported constraint-entity combinations for the Insert constraint. A ✓ indicates the combination is supported.

		Constraint Entity 2					
		Point	**Line**	**Plane**	**Cylinder**	**Cone**	**Circle/Arc**
Constraint Entity 1	**Point**						
	Line					✓	✓
	Plane			✓			
	Cylinder		✓		✓	✓	✓
	Cone		✓		✓	✓	✓
	Circle/Arc		✓		✓	✓	✓

Align or Mate Constraint with Translational Offset

The following table identifies supported constraint-entity combinations for the Align or Mate constraints with translational offset. A ✓ indicates the combination is supported.

		Constraint Entity 2					
		Point	**Line**	**Plane**	**Cylinder**	**Cone**	**Circle/Arc**
Constraint Entity 1	**Point**	✓		✓			
	Line			✓			
	Plane	✓	✓	✓			
	Cylinder						
	Cone						
	Circle/Arc						

Align or Mate with Rotational Offset

The following table identifies supported constraint-entity combinations for the Align or Mate constraints with rotational offset. A ✓ indicates the combination is supported.

		Constraint Entity 2					
		Point	**Line**	**Plane**	**Cylinder**	**Cone**	**Circle/Arc**
Constraint Entity 1	**Point**						
	Line		✓				
	Plane			✓			
	Cylinder						

	Cone						
	Circle/Arc						

Limitations

The following limitation applies to CAD export from Pro/ENGINEER.

Weld is Default Joint

If the Simscape Multibody Link utility fails to translate a CAD constraint, a Weld joint replaces the constraint.

Configure Simscape Multibody Link

In this section...
"Simscape Multibody Link Settings" on page 2-14
"Dialog Box" on page 2-14

Warning   This content refers to Simscape Multibody Link First Generation software. First-generation features are slated to be deprecated and should be avoided. To prevent compatibility issues in future releases, the Simscape Multibody Link plug-in no longer supports the translation of CAD models into Simscape Multibody Link First Generation models.

Simscape Multibody Link Settings

The Simscape Multibody Link add-in tool provides a Settings option. Use the option to specify:

- Tolerances — linear, angular, and relative
- Coordinate systems to export

To access the Settings parameters:

1   Open the assembly to export.
2   In the menu bar, click **Tools** > **Simscape Multibody Link**.
3   Click **Settings**.

The Settings dialog box opens.

Dialog Box

The dialog box contains two panes:

- **Assembly Tolerances** — Specifies linear, angular, and relative tolerances of exported assembly.
- **Export Coordinate Systems** — Determines what coordinate systems to export.

Assembly Tolerances

Enter the export tolerances for a CAD assembly. During the conversion of CAD constraints to Simscape Multibody joints, Simscape Multibody Link compares the spacing, alignment, and relative numerical errors with the export tolerances.

Field	Default Value	Purpose	Default	Unit
Linear tolerance	1e-005	Smallest significant length difference	1e-5	Units used in assembly

Field	Default Value	Purpose	Default	Unit
Angular tolerance	1e-005	Smallest significant angle difference	1e-5	Units used in assembly
Relative roundoff tolerance	1e-012	Smallest significant relative numerical difference	1e-12	—

Export Coordinate Systems

Specify which reference coordinate systems to export. The coordinate systems are independent of constraints between parts. Options include:

- **Do not export coordinate systems** — Export no coordinate systems.
- **Export only CSs with this prefix** — Export only coordinate systems with the specified name prefix. If the prefix field is empty, Simscape Multibody Link exports all reference coordinate systems.

Export CAD Assembly from Creo Software

In this section...
"Export CAD Assembly" on page 2-16
"CAD Assembly Export Errors" on page 2-16

Export CAD Assembly

To export a CAD assembly:

1   In the menu bar of the CAD platform, click **Tools**.

2   Click **Simscape Multibody Link > Export**.

3   Click **Simscape Multibody**.

4   Specify the desired XML file name and select a folder to save the file in.

Simscape Multibody Link generates:

• One XML multibody description file.

   The file contains the structure and parameters of the CAD assembly. During CAD import, Simscape Multibody uses the structure and parameters to autogenerate a Simscape Multibody model.

• A set of geometry files in STEP or STL format.

   Each geometry file specifies the 3-D surface geometry of one CAD part. The geometry files are not required to generate the model, but they are required for visualization. If you import a model without the geometry files, during model update and simulation Mechanics Explorer displays a blank screen.

CAD Assembly Export Errors

In the event that a CAD export error occurs:

• A dialog box displays an error message. The message identifies the CAD constraints that Simscape Multibody Link could not translate into joints.

• Simscape Multibody Link generates an error log file. Refer to the log for more information about the CAD export error. The error message identifies the name and location of an error log file.

• Simscape Multibody Link generates the XML file. You can import the file to generate a valid Simscape Multibody model, but the model may not accurately represent the original CAD assembly.

• If Simscape Multibody Link cannot export one or more geometry files, the error message identifies the CAD parts associated with the geometry files.

Register and Use SolidWorks Add-In

This chapter describes how to register Simscape Multibody Link software to the SolidWorks CAD platform as a SolidWorks add-in. You must complete the registration before you can export a CAD assembly in Simscape Multibody format.

Enable Simscape Multibody Link Plugin in SolidWorks

In this section...
"Enable the Plugin" on page 3-2
"Update the Plugin" on page 3-4

The Simscape Multibody Link plugin allows you to export SolidWorks CAD assembly models to Simscape Multibody. To download and install the plugin, see "Install the Simscape Multibody Link Plugin".

Enable the Plugin

To enable the plugin:

1   At the MATLAB command prompt, enter `smlink_linksw`.

2   Start SolidWorks.

3   On the SolidWorks menu bar, click **Tools > Add-Ins**.
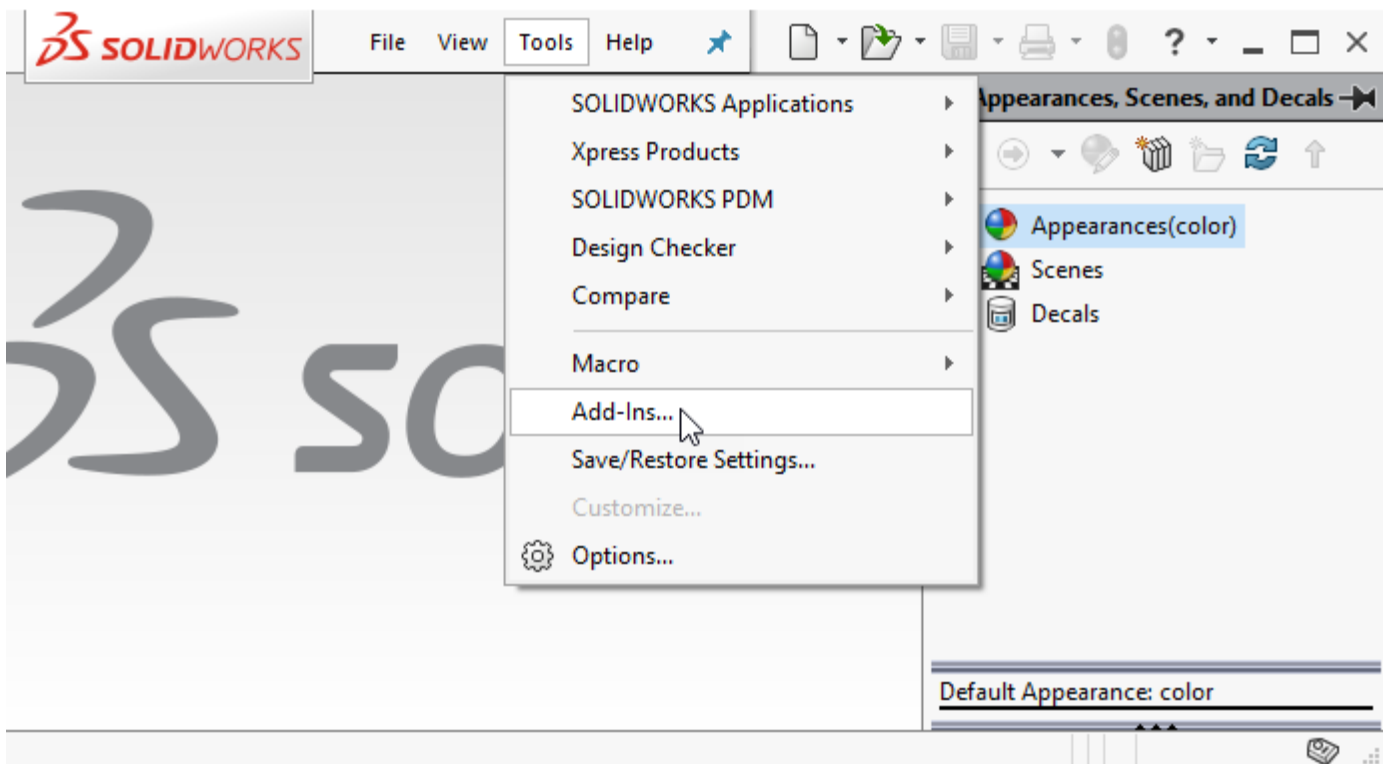
4   In the Add-Ins dialog box, select the **Simscape Multibody Link** check box.
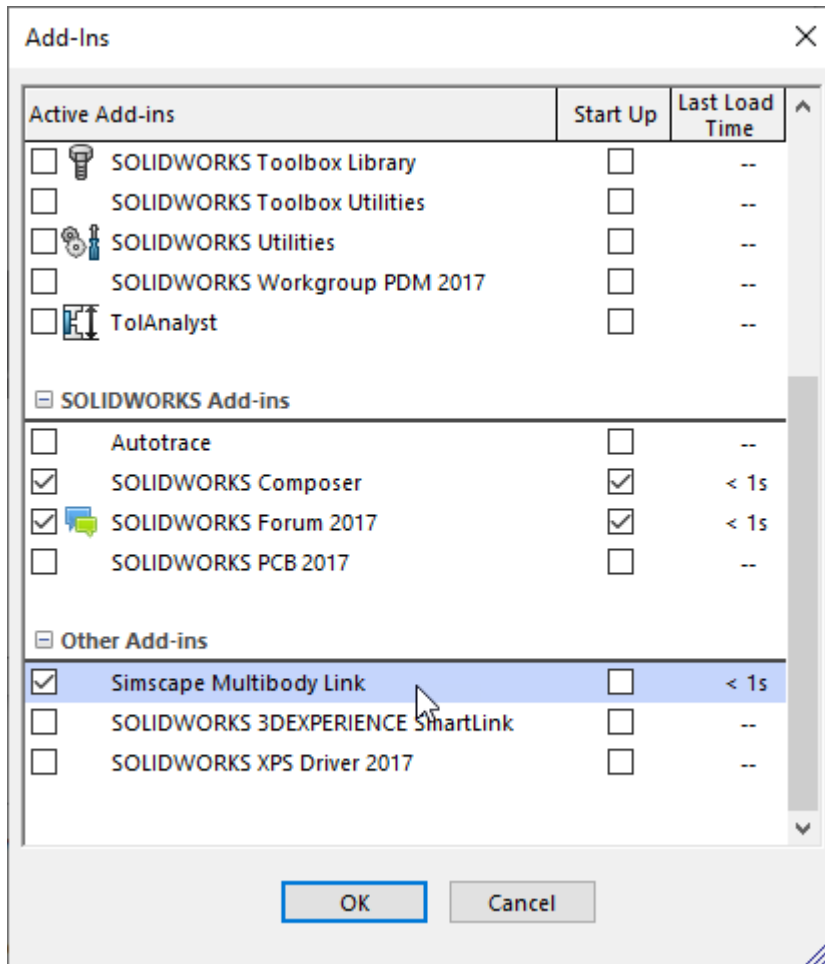
After you enable the plugin, **Simscape Multibody Link** option is available when you start or open a SolidWorks assembly. To export an assembly model, on the menu bar, click **Tools > Simscape Multibody Link > Export > Simscape Multibody**.
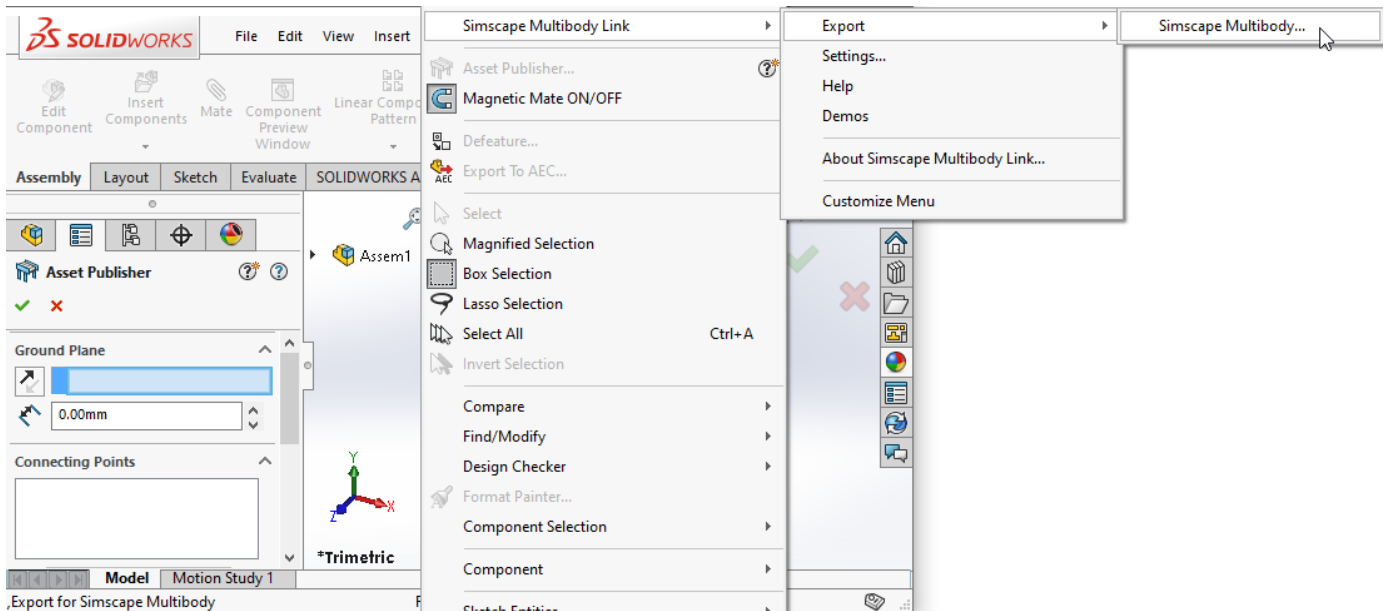
If your computer has more than one copy of SolidWorks, the `smlink_linksw` command adds the Simscape Multibody Link plugin to all installations. However, you must select the **Simscape Multibody Link** check box in the Add-Ins dialog box for each installation from which you want to export CAD assembly models.

Update the Plugin

To update the plugin:

1 Disable the current version by entering `smlink_unlinksw` at the MATLAB command prompt.

2 Download, install, and enable the new plugin. See "Install the Simscape Multibody Link Plugin" for more information.

See Also
`smimport` | `smlink_linksw`

More About
•   "Install the Simscape Multibody Link Plugin"

Mates and Joints

In this section...
"Mates and Entities" on page 3-5
"Joints and Constraints" on page 3-6
"Mate-Joint Mapping" on page 3-8
"Mate-Constraint Mapping" on page 3-9
"Special Mate Translation Cases" on page 3-10

Mates and Entities

In a SolidWorks assembly, you connect parts using mates. Each mate applies a geometric relationship between mate entities on different parts. Mates include parallel, concentric, and coincident types. Mate entities include points, lines, and surfaces.

Consider the connection between the upper-arm and forearm parts of a robotic arm assembly. This connection allows the two parts to rotate with respect to each other about a single axis and therefore has one rotational degree of freedom. You specify this degree of freedom by applying mates such as:

- One concentric mate between the cylindrical hinge surfaces of the two parts. This mate reduces the joint degrees of freedom to two—one translational, along the cylindrical axis, and one rotational, about the same axis.
- One coincident mate between two planes normal to the cylindrical axis. This mate removes the translational degree of freedom between the two parts. These can now only rotate about the common hinge axis.

The figure shows the mated surfaces in a CAD robotic arm assembly.
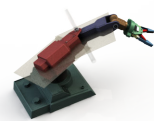
For the purposes of CAD import, Simscape Multibody software supports these mate entities:
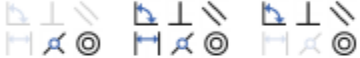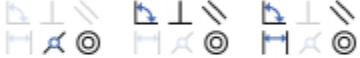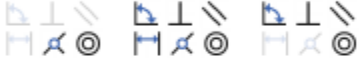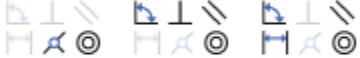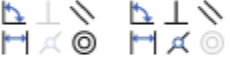
- Circle/arc ⭕
- Cone 🔺
- Cylinder 🟧
- Line ╱
- Plane ◢
- Point ✳

Simscape Multibody also supports these mates:

- Angle 📐

- Coincident ⋊
- Concentric ◎
- Distance ⊢⊣
- Parallel ⫽
- Perpendicular ⊥

Supported mates are valid only for certain entity pairs. The table shows the entity pairs compatible with the supported mates. This table is symmetric with respect to the diagonal row.

Joints and Constraints

Joint and Constraint blocks are the Simscape Multibody equivalent of SolidWorks mates. They apply between two bodies the kinematic constraints that determine how they can move. A Revolute Joint block is an example. This block removes five degrees of freedom between two bodies, allowing them only to rotate about a common axis.

Consider the connection between the upper-arm and forearm parts of the robotic arm assembly. This connection provides a single rotational degree of freedom and therefore is a revolute joint. During CAD import, the mates between the two parts translate into a Revolute Joint block between two rigid body subsystems. The figure shows this joint block in an imported model (polished for clarity).

Joint blocks are assortments of joint primitives, basic yet complete joints of various kinds you cannot decompose any further—at least without losing behavior such as the rotational-translational coupling of the lead screw joint. Joint primitives range in number from zero in the Weld Joint block to six in the Bushing Joint block. There are five joint primitives:

- Prismatic — Allows translation along a single standard axis (x, y, or z). Joint blocks can contain up to three prismatic joint primitives, one for each translational DoF. Prismatic primitives are labelled P*, where the asterisk denotes the axis of motion, e.g., Px, Py, or Pz.

- Revolute — Allows rotation about a single standard axis (x, y, or z). Joint blocks can contain up to three revolute joint primitives, one for each rotational DoF. Revolute primitives are labelled R*, where the asterisk denotes the axis of motion, e.g., Rx, Ry, or Rz.

- Spherical — Allows rotation about any 3-D axis, [x, y, z]. Joint blocks contain no more than one spherical primitive, and never in combination with revolute primitives. Spherical primitives are labelled S.

- Lead Screw Primitive — Allows coupled rotation and translation on a standard axis (e.g., z). This primitive converts between rotation at one end and translation at the other. Joint blocks contain no more than one lead screw primitive. Lead screw primitives are labeled LS*, where the asterisk denotes the axis of motion.
- Constant Velocity Joint — Allows rotation at constant velocity between intersecting though arbitrarily aligned shafts. Joint blocks contain no more than one constant velocity primitive. Constant velocity primitives are labelled CV.

A Joint block can have up to three revolute primitives or, alternatively, one spherical primitive. Each revolute primitive aligns with a different axis in the joint base frame—X, Y, or Z. These are denoted Rx, Ry, and Rz, respectively. The spherical primitive, denoted S, enables rotation about a general axis [X, Y, Z] in the joint base frame.

Similarly, a Joint block can also have up to three prismatic primitives. Each primitive aligns with a different axis in the joint base frame—X, Y, or Z. These are denoted Px, Py, and Pz, respectively. The table shows the Joint blocks that CAD mates can translate into, the joint primitives the blocks contain, and the degrees of freedom they provide. T and R denote translational and rotational DOFs. Joint blocks not shown are not supported.

Joint Block	Joint Primitives							Joint DoFs
6-DOF Joint	Px	Py	Pz	Rx	Ry	Rz	S	3 T + 3 R
Cartesian Joint	Px	Py	Pz	Rx	Ry	Rz	S	3 T + 0 R
Cylindrical Joint	Px	Py	Pz	Rx	Ry	Rz	S	1 T + 1 R
Planar Joint	Px	Py	Pz	Rx	Ry	Rz	S	2 T + 1 R
Prismatic Joint	Px	Py	Pz	Rx	Ry	Rz	S	1 T + 0 R
Rectangular Joint	Px	Py	Pz	Rx	Ry	Rz	S	2 T + 0 R
Revolute Joint	Px	Py	Pz	Rx	Ry	Rz	S	0 T + 1 R
Spherical Joint	Px	Py	Pz	Rx	Ry	Rz	S	0 T + 3 R
Telescoping Joint	Px	Py	Pz	Rx	Ry	Rz	S	1 T + 3 R
Universal Joint	Px	Py	Pz	Rx	Ry	Rz	S	0 T + 2 R
Weld Joint	Px	Py	Pz	Rx	Ry	Rz	S	0 T + 0 R

By defining the relative degrees of freedom between two bodies, Joint blocks partially determine how these bodies can move with respect to each other. Constraint blocks enable you to impose additional restrictions on their motion. CAD mates can translate into these Constraint blocks:

- Angle Constraint
- Distance Constraint

Mate-Joint Mapping

The table shows the Joint blocks that different mate combinations map into. Different mate combinations can map into the same joint. This happens if the mate combinations provide the same

degrees of freedom between the parts they join. For a legend of the icons in the table, see "Mates and Entities" on page 3-5.

Joint Block	Mate I	Entities I	Mate II	Entities II	Notes
Cartesian Joint	⟍	■ ■	⟍	■ ■	
Cylindrical Joint	⋊	╱ ╱			
	◎	▮ ▮			
Planar Joint	⋊	■ ■			
Prismatic Joint	◎	▮ ▮	⋊ or ⊢	■ ■	1
	◎	▮ ▮	⌐ or ⊥	■ ■	
	⋊	╱ ╱	⋊ or ⊢	■ ■	2
	⋊	╱ ╱	⌐ or ⊥	■ ■	
	⋊ or ⊢	■ ■	⋊ or ⊢	■ ■	3
Rectangular Joint	⋊ or ⊢	■ ■	⌐, ⟍, or ⊥	■ ■	
	⋊ or ⊢	■ ■	⟍	╱ ╱	
Revolute Joint	◎	▮ ▮	⋊ or ⊢	■ ■	4
	⋊	╱ ╱	⋊ or ⊢	■ ■	5
	⋊	• •	⋊	■ ■	
Spherical Joint	⋊ or ⊢	• •	⋊ or ⊢	• •	
Universal Joint	⋊ or ⊢	■ ■	⋊ or ⊢	■ ■	
	⋊ or ⊢	■ ■	⋊ or ⊢	■ ■	

Mate pairs marked with a note number must satisfy additional requirements. This list outlines these requirements:

1   Cylinder axes in mate I must be parallel to planes in mate II.

2   Lines in mate I must be parallel to planes in mate II.

3   Planes in mate I must not be parallel to planes in mate II.

4   Cylinder axes in mate I must be perpendicular to planes in mate II.

5   Lines in mate I must be perpendicular to planes in mate II.

Mate-Constraint Mapping

The table shows the Constraint blocks that different mate combinations map into. Different mates map into the same Constraint block if they provide the same degrees of freedom. Angle mates must have values of 0 or 90 degrees. Other mate settings are not supported. For a legend of the icons in the table, see "Mates and Entities" on page 3-5.

Constraint Block	CAD Mate	Entities	Setting
Angle (Perpendicular)	⊾	▮▮, 🛢🛢, ▲▲, or ╱╱	$\theta = 90°$
	⊥	▮▮, 🛢🛢, ▲▲, or ╱╱	
Angle (Parallel)	⊾	▮▮, 🛢🛢, ▲▲, or ╱╱	$\theta = 0°$
	╲╲	▮▮, 🛢🛢, ▲▲, or ╱╱	
Distance	⊢⊣	• •	$d \geq 0$

Special Mate Translation Cases

The lack of mates between parts, combinations of mates that fully constrain two parts, and unsupported mates are special translation cases. Here is how Simscape Multibody software handles these cases:

- Unsupported mates between parts translate into rigid connections between rigid bodies. The rigid connections can be in the form of Weld Joint blocks or direct frame connection lines between the rigid bodies. These connections are meant to be temporary. After CAD import, search your model for rigid connections and, if appropriate, replace them with other Joint and Constraint blocks.

- Mate combinations that fully constrain two parts translate into rigid connections between two rigid bodies. The rigid connections can be in the form of Weld Joint blocks or direct frame connection lines between the rigid bodies. These rigid connections accurately model the degrees of freedom between the two bodies and do not need to be replaced.

- The absence of a mate between a part and the rest of the assembly translates into a 6-DOF Joint block between a rigid body and the World frame. This joint block provides the rigid body the six degrees of freedom that the CAD part has within the CAD assembly.

Mate-Joint Mapping in Simscape Multibody First Generation

In this section...

Warning This content refers to Simscape Multibody Link First Generation software. First-generation features are slated to be deprecated and should be avoided. To prevent compatibility issues in future releases, the Simscape Multibody Link plug-in no longer supports the translation of CAD models into Simscape Multibody Link First Generation models.

In SolidWorks, unmated parts have six mechanical degrees of freedom (DoFs) that describe how the parts can move with respect to each other. Of the six degrees of freedom, three are rotational and three are translational. Applying a mate between two parts eliminates degrees of freedom between the two parts. Mates can remove between zero and six degrees of freedom.

Degrees of Freedom in Simscape Multibody

Simscape Multibody assigns six degrees of freedom to an unconstrained rigid body. The unconstrained rigid body behaves as a free body — it can rotate and translate, about or along three mutually orthogonal axes. The following table lists the degrees of freedom of a rigid body in different configurations.

Rigid Body Condition	Degrees of Freedom
Not connected to joints, constraints, or World Frame	0
Connected to Joints or Constraints blocks	*Add* degrees of freedom as specified by joint or constraint

CAD Mate – Simscape Multibody Joint Mapping

During CAD export, Simscape Multibody Link maps SolidWorks mates between parts to Simscape Multibody joints between rigid bodies. CAD mates and Simscape Multibody joints do not follow a one-to-one correspondence — multiple mates can map into a single joint. All Simscape Multibody joints contain a combination of three joint primitives: Prismatic, Revolute, and Spherical. The Weld Joint block contains zero joint primitives, and therefore zero degrees of freedom. The following table identifies the degrees of freedom of each joint primitive.

Primitive	Abbreviation	Motion Type	Number of DoFs
Prismatic	P	Translational	1
Revolute	R	Rotational	1
Spherical	S	Rotational	3

Supported Constraint Entity

Depending on the constraint combination, Simscape Multibody Link utility supports the following Inventor constraint entities:

Entity	Description
Circle/Arc	Circular edge/arc sketch segment*
Ellipse/Arc	Elliptical edge/arc sketch segment*
Cone	Conical face
Cylinder	Cylindrical face
Line	Linear edge/sketch segment/reference axis
Plane	Reference plane or planar face
Point	Vertex/sketch point/reference point

* A complete circle or ellipse is a special case of a circular or elliptical arc.

Supported Constraint Entity Combinations

The following sections list the constraint-entity combinations that Simscape Multibody Link supports for different constraint types.

Note If the Simscape Multibody Link exporter cannot translate a constraint–constraint entity combination into a supported Simscape Multibody joint with DoFs, it converts the combination into a weld (W) primitive.

Coincident Constraint

The following table identifies supported constraint-entity combinations for the Coincident constraint. A ✓ indicates the combination is supported.

		Mate-Entity 2					
		Point	**Line**	**Plane**	**Cylinder**	**Cone**	**Circle/Arc**
Mate-Entity 1	**Point**	✓					
	Line		✓	✓			
	Plane		✓	✓			✓
	Cylinder				✓	✓	✓
	Cone				✓	✓	✓
	Circle/Arc			✓	✓	✓	✓

Concentric Mate

The following table identifies supported constraint-entity combinations for the Concentric mate. A ✓ indicates the combination is supported.

	Mate-Entity 2

		Point	Line	Plane	Cylinder	Cone	Circle/Arc
Mate-Entity 1	Point						
	Line				✓	✓	✓
	Plane			✓			
	Cylinder		✓		✓	✓	✓
	Cone		✓		✓	✓	✓
	Circle/Arc		✓		✓	✓	✓

Perpendicular Mate

The following table identifies supported constraint-entity combinations for the Perpendicular mate. A ✓ indicates the combination is supported.

		Mate-Entity 2					
		Point	Line	Plane	Cylinder	Cone	Circle/Arc
Mate-Entity 1	Point						
	Line		✓	✓			
	Plane		✓	✓			
	Cylinder						
	Cone						
	Circle/Arc						

Parallel Mate

The following table identifies supported constraint-entity combinations for the Parallel mate. A ✓ indicates the combination is supported.

		Mate-Entity 2					
		Point	Line	Plane	Cylinder	Cone	Circle/Arc
Mate-Entity 1	Point						
	Line		✓	✓			
	Plane		✓	✓			
	Cylinder				✓		
	Cone					✓	
	Circle/Arc						

Distance Mate

The following table identifies supported constraint-entity combinations for the Distance mate. A ✓ indicates the combination is supported.

		Mate-Entity 2					
		Point	Line	Plane	Cylinder	Cone	Circle/Arc
Mate-Entity 1	Point	✓		✓			

	Line			✓			
	Plane	✓	✓	✓			
	Cylinder						
	Cone						
	Circle/Arc						

Angle Mate

The following table identifies supported constraint-entity combinations for the Angle mate. A ✓ indicates the combination is supported.

		Mate-Entity 2					
		Point	**Line**	**Plane**	**Cylinder**	**Cone**	**Circle/Arc**
Mate–Entity 1	**Point**						
	Line		✓				
	Plane			✓			
	Cylinder						
	Cone						
	Circle/Arc						

Limitations

The following limitation applies to CAD export from SolidWorks.

Weld is Default Joint

If the Simscape Multibody Link utility fails to translate a CAD constraint, a Weld joint replaces the constraint.

Restriction on Point-Point Distance Mate

For Simscape Multibody Link to successfully map the CAD point-point distance mate onto a Simscape Multibody spherical-spherical massless connector, the mate must not connect to any other mates.

Configure Simscape Multibody Link

In this section...

Warning This content refers to Simscape Multibody Link First Generation software. First-generation features are slated to be deprecated and should be avoided. To prevent compatibility issues in future releases, the Simscape Multibody Link plug-in no longer supports the translation of CAD models into Simscape Multibody Link First Generation models.

Simscape Multibody Link Settings

The Simscape Multibody Link add-in tool provides a Settings option. Use the option to specify:

- Tolerances — linear, angular, and relative
- Coordinate systems to export

To access the Settings parameters:

1 Open the assembly to export.
2 In the menu bar, select **Simscape Multibody Link** > **Settings**. In certain SolidWorks versions, this option may appear under the **Tools** menu.

The Settings dialog box opens.

Dialog Box

The dialog box contains four panes:

- **Message** — Describes the purpose of the dialog box. The Message box is inactive.
- **Solve Subassemblies Using** — Determines whether to export a subassembly as a rigid or flexible system.
- **Assembly Tolerances** — Specifies linear, angular, and relative tolerances of exported assembly.
- **Export Coordinate Systems** — Determines what coordinate systems to export.

Solve Subassemblies Using

Select how to export CAD subassemblies.

- **Component properties** — Treat rigid subassemblies as rigid, and flexible subassemblies as flexible.
- **Flexible settings** — Treat *all* subassemblies as flexible. This setting applies does not affect the original CAD assembly.

Make Subassemblies Rigid or Flexible in SolidWorks

Subassemblies can be rigid or flexible. Rigid subassemblies behave as a single rigid body. Flexible subassemblies behave as a multibody subsystem. To make a subassembly rigid or flexible:

1. Right-click the subassembly.
2. Click **Component > Properties**.
3. Select between **Flexible** and **Rigid** options.

   Select **Rigid** only if the motion between subassembly parts is not important in Simscape Multibody.

Assembly Tolerances

Enter the export tolerances for a CAD assembly. During the conversion of CAD constraints to Simscape Multibody joints, Simscape Multibody Link compares the spacing, alignment, and relative numerical errors with the export tolerances.

Field	Default Value	Purpose	Default	Unit
Linear tolerance	1e-005	Smallest significant length difference	1e-5	meter (m)
Angular tolerance	1e-005	Smallest significant angle difference	1e-5	radian (rad)
Relative roundoff tolerance	1e-012	Smallest significant relative numerical difference	1e-12	—

Export Coordinate Systems

Specify which reference coordinate systems to export. The reference coordinate systems are independent of mates between parts. Options include:

- **Do not export coordinate systems** — Export no coordinate systems.
- **Export only CSs with this prefix** — Export only coordinate systems with the specified name prefix. If the prefix field is empty, Simscape Multibody Link exports all reference coordinate systems.

Export CAD Assembly from SolidWorks Software

In this section...
"Export an Assembly" on page 3-17
"Files Generated" on page 3-17
"Export Errors" on page 3-17

Export an Assembly

1 In the SolidWorks menu bar, select **Simscape Multibody Link** > **Export** > **Simscape Multibody**. The **Simscape Multibody Link** menu may appear inside the **Tools** menu.

2 Specify a convenient name and folder for the exported assembly.

Files Generated

The Simscape Multibody Link plug-in generates the following files:

- XML multibody description file outlining the CAD parts in the exported assembly, their spatial relationships, and their material properties. During CAD import, Simscape Multibody software uses this data to generate an equivalent Simscape Multibody model.

- Graphics files detailing the 3-D surface geometries of the CAD parts in the exported assembly. These files enable Simscape Multibody software to render the various CAD parts in its visualization utility, Mechanics Explorer. The graphics files can be in STEP or STL format.

Export Errors

Some CAD constraint sets may not translate into equivalent Simscape Multibody Joint or Constraint blocks. In such cases, Simscape Multibody Link issues an error message listing:

- Any CAD Constraints that Simscape Multibody Link could not export.

- Any graphics files and the corresponding part names that Simscape Multibody Link could not export.

The error message includes the name of an error log file. Open this file for more information about the CAD export error.

Functions

smlink_linkinv

Add Simscape Multibody Link to your Windows registry as an Autodesk Inventor plugin

Syntax

```
smlink_linkinv
```

Description

`smlink_linkinv` adds Simscape Multibody Link to your Windows® registry as an Autodesk Inventor plugin. The registry entry makes the plugin discoverable in your Autodesk Inventor installation. Use the plugin to export an assembly model in an XML format compatible with `smimport`—a Simscape Multibody Link function whose purpose it is to convert XML multibody description files into equivalentSimscape Multibody Link models. You may need to select Simscape Multibody Link as an active plugin from within Inventor in order to begin using its features. If you have not yet installed the plugin, do so before running this command. For more information, see Install the Simscape™ Multibody™ Link Plug-In.

See Also
smimport

smlink_linksw

Add Simscape Multibody Link to your Windows registry as a SolidWorks plugin

Syntax

```
smlink_linksw
```

Description

`smlink_linksw` adds Simscape Multibody Link to your Windows registry as a SolidWorks plugin. The registry entry makes the plugin discoverable in your SolidWorks installation. Use the plugin to export an assembly model in an XML format compatible with `smimport`—a Simscape Multibody function whose purpose it is to convert XML multibody description files into equivalent Simscape Multibody models. You may need to select Simscape Multibody Link as an active plugin from within SolidWorks in order to begin using its features. If you have not yet installed the plugin, do so before running this command. For more information, see Install the Simscape™ Multibody™ Link Plug-In.

See Also
`smimport`

smlink_unlinkinv

Remove Simscape Multibody Link from your Windows registry as an Autodesk Inventor plugin

Syntax

```
smlink_unlinkinv
```

Description

`smlink_unlinkinv` removes Simscape Multibody Link from your Windows registry as an Autodesk Inventor plugin. The plugin becomes no longer discoverable in your Autodesk Inventor installation. You can no longer export Autodesk Inventor assembly models for import into the Simscape Multibody environment. Use the `smlink_linkinv` command if you decide to once again register the plugin and continue to use its features.

See Also
`smimport`

smlink_unlinksw

Remove Simscape Multibody Link from your Windows registry as a SolidWorks plugin

Syntax

```
smlink_unlinksw
```

Description

`smlink_unlinksw` removes Simscape Multibody Link from your Windows registry as a SolidWorks plugin. The plugin becomes no longer discoverable in your SolidWorks installation. You can no longer export SolidWorks assembly models for import into the Simscape Multibody environment. Use the `smlink_linksw` command if you decide to once again register the plugin and continue to use its features.

See Also

`smimport`

APIs

pmit_add_cadcs

Add coordinate system to handle object of `PmitCadModelH` class

Description

Warning This content is specific to Simscape Multibody First Generation software. First-generation features are slated to be deprecated and should be avoided.

`PmitError = pmit_add_cadcs(PmitCadModelH pmitCadModelH, PmitCadCSH pmitCadCSH)` returns an error status `PmitError`.

With `pmit_add_cadcs`, you can add a coordinate system to a handle object of `PmitCadModelH` class that represents an API CAD model.

Input Arguments

pmitCadModelH

Handle object of `PmitCadModelH` class representing an API CAD model

Default:

pmitCadCSH

Handle object of `PmitCadCSH` class representing a coordinate system on an API CAD model

Default:

Version History
Introduced in R2010a

See Also
`PmitCadCSH` | `pmit_create_cadcs` | `PmitError`

pmit_add_constrain

Add constraint to handle object of `PmitCadModelH` class

Syntax

`PmitError = pmit_add_constrain(PmitCadModelH pmitCadModelH, PmitConstrainH pmitConstrainH)`

Description

Warning This content is specific to Simscape Multibody First Generation software. First-generation features are slated to be deprecated and should be avoided.

`PmitError = pmit_add_constrain(PmitCadModelH pmitCadModelH, PmitConstrainH pmitConstrainH)` returns an error status `PmitError`.

With `pmit_add_constrain`, you can add a constraint to a handle object of `PmitCadModelH` class that represents an API CAD model.

Input Arguments

`pmitCadModelH`

Handle object of `PmitCadModelH` class representing an API CAD model

Default:

`pmitConstrainH`

Handle object of `PmitConstrainH` class representing an API CAD model constraint

Default:

Version History
Introduced in R2009b

See Also
`PmitCadModelH` | `PmitConstrainH` | `PmitError`

pmit_add_refincadmodel

Add object of `PmitCadModelRefH` class to object of `PmitCadModelH` class

Syntax

`PmitError = pmit_add_refincadmodel(PmitCadModelH pmitCadModelH, PmitCadModelRefH pmitCadModelrefH)`

Description

Warning This content is specific to Simscape Multibody First Generation software. First-generation features are slated to be deprecated and should be avoided.

`PmitError = pmit_add_refincadmodel(PmitCadModelH pmitCadModelH, PmitCadModelRefH pmitCadModelrefH)` returns an error status `PmitError`.

With `pmit_add_refincadmodel`, you can add an object of `PmitCadModelRefH` class to an object of `PmitCadModelH` class, in order to reference a CAD model in an API CAD model hierarchy.

Input Arguments

`pmitCadModelH`

Handle object of `PmitCadModelH` class representing an API CAD model

Default:

`pmitCadModelrefH`

Handle object of `PmitCadModelRefH` class referencing a CAD model in an API CAD model hierarchy

Default:

Version History
Introduced in R2009b

See Also
`PmitCadModelH` | `PmitCadModelRefH` | `PmitError`

pmit_add_refincomp

Add object of `PmitCadModelRefH` class at end of object of `PmitAssemCompH` class

Syntax

`PmitError = pmit_add_refincomp(PmitAssemCompH pmitAssemComp, PmitCadModelRefH pmitCadModelrefH)`

Description

Warning This content is specific to Simscape Multibody First Generation software. First-generation features are slated to be deprecated and should be avoided.

`PmitError = pmit_add_refincomp(PmitAssemCompH pmitAssemComp, PmitCadModelRefH pmitCadModelrefH)` returns an error status `PmitError`.

With `pmit_add_refincomp`, you can add an object of `PmitCadModelRefH` class at the end of an object of `PmitAssemCompH` class, in order to reference an element in an API CAD hierarchy. You construct the full reference with a chain of objects of `PmitCadModelRefH` class. Make the chain as long as needed to reach the desired element in the hierarchy.

Input Arguments

pmitAssemComp

Handle object of `PmitAssemCompH` class representing a component in an API CAD model

Default:

pmitCadModelrefH

Handle object of `PmitCadModelRefH` class referencing a CAD model in an API CAD model hierarchy

Default:

Version History
Introduced in R2009b

See Also
`PmitAssemCompH` | `PmitCadModelRefH` | `PmitError`

PmitAssemCompH

Handle object type to represent component in API CAD model

Description

Warning This content is specific to Simscape Multibody First Generation software. First-generation features are slated to be deprecated and should be avoided.

`PmitAssemCompH` is a C language opaque type.

A variable of this type is a handle object created when you instantiate a Simscape Multibody Link API object representing an API CAD assembly component.

See Also

pmit_add_refincomp | pmit_create_assemcomp | pmit_create_assemcomp_fromstr | pmit_create_constrain | PmitError

PmitCad2SMH

Handle object type to represent API-to-XML translator

Description

Warning This content is specific to Simscape Multibody First Generation software. First-generation features are slated to be deprecated and should be avoided.

`PmitCad2SMH` is a C language opaque type.

A variable of this type is a handle object created when you instantiate a Simscape Multibody Link API object that translates an API CAD model into XML.

See Also

pmit_create_cad2sm | PmitError | pmit_set_tolerances | pmit_write_xml

PmitCadCSH

Handle object type to represent coordinate system

Description

Warning This content is specific to Simscape Multibody First Generation software. First-generation features are slated to be deprecated and should be avoided.

`PmitCadCSH` is a C language opaque type.

A variable of this type is a handle object created when you add a coordinate system to a Simscape Multibody Link API object representing an API CAD model.

See Also
`pmit_add_cadcs` | `pmit_create_cadcs` | `PmitError`

PmitCadModelH

Handle object type to represent API CAD model

Description

Warning This content is specific to Simscape Multibody First Generation software. First-generation features are slated to be deprecated and should be avoided.

`PmitCadModelH` is a C language opaque type.

A variable of this type is a handle object created when you instantiate a Simscape Multibody Link API object representing an API CAD model of an assembly or assembly part.

See Also

pmit_add_constrain | pmit_add_refincadmodel | pmit_cadmodel_setfilename | pmit_cadmodelref_getcadmodel | pmit_create_assemcomp_fromstr | pmit_create_cad2sm | pmit_create_cadmodel | pmit_create_cadmodelref | PmitError

PmitCadModelRefH

Handle object type to reference a CAD model in API CAD model hierarchy

Description

Warning This content is specific to Simscape Multibody First Generation software. First-generation features are slated to be deprecated and should be avoided.

`PmitCadModelRefH` is a C language opaque type.

A variable of this type is a handle object created when you instantiate a Simscape Multibody Link API object referencing an API CAD model component.

See Also

pmit_add_refincadmodel | pmit_add_refincomp | pmit_cadmodelref_getcadmodel | pmit_create_cadmodelref | PmitError | pmit_get_reffixedstatus | pmit_get_refflexiblestatus | pmit_set_reffixedstatus | pmit_set_refflexiblestatus

pmit_cadmodel_setfilename

Specify body geometry file name for handle object of `PmitCadModelH` class

Syntax

`PmitError = pmit_cadmodel_setfilename(PmitCadModelH pmitCadModelH, const char* fileName)`

Description

Warning This content is specific to Simscape Multibody First Generation software. First-generation features are slated to be deprecated and should be avoided.

`PmitError = pmit_cadmodel_setfilename(PmitCadModelH pmitCadModelH, const char* fileName)` returns an error status `PmitError`.

With `pmit_cadmodel_setfilename`, specify the geometry body geometry file name for a handle object of `PmitCadModelH` class representing an API CAD model.

The body geometry file carries no units. This body geometry is interpreted with the units defined for the API session.

Input Arguments

pmitCadModelH

Handle object of `PmitCadModelH` class representing an API CAD model

Default:

fileName

String specifying body geometry file name

Default:

Version History
Introduced in R2009b

See Also
`PmitCadModelH` | `PmitError` | `pmit_set_units`

pmit_cadmodelref_getcadmodel

Get object of `PmitCadModelH` class from children of object of `PmitCadModelRefH` class

Syntax

PmitError = pmit_cadmodelref_getcadmodel(PmitCadModelH* pmitCadModelHOut, PmitCadModelRefH cadModelRefH)

Description

Warning This content is specific to Simscape Multibody First Generation software. First-generation features are slated to be deprecated and should be avoided.

PmitError = pmit_cadmodelref_getcadmodel(PmitCadModelH* pmitCadModelHOut, PmitCadModelRefH cadModelRefH) returns an error status `PmitError`.

With `pmit_cadmodelref_getcadmodel`, you can get an object of `PmitCadModelH` class that represents an API CAD model from whatever is referenced by an object of `PmitCadModelRefH` class.

Input Arguments

cadModelRefH

Handle object of `PmitCadModelRefH` class referencing a CAD model in an API CAD model hierarchy

Default:

Output Arguments

pmitCadModelHOut

Handle object of `PmitCadModelH` class representing an API CAD model

Version History
Introduced in R2009b

See Also
PmitCadModelH | PmitCadModelRefH | PmitError

pmit_connectto_matlab

Connect to MATLAB session

Syntax

PmitError = pmit_connectto_matlab()

Description

Warning This content is specific to Simscape Multibody First Generation software. First-generation features are slated to be deprecated and should be avoided.

PmitError = pmit_connectto_matlab() returns an error status PmitError.

Version History
Introduced in R2009b

See Also
pmit_disconnectfrom_matlab | PmitError | pmit_open_demo | pmit_open_help

PmitConstrainH

Handle object type to represent constraint

Description

> **Warning** This content is specific to Simscape Multibody First Generation software. First-generation features are slated to be deprecated and should be avoided.

`PmitConstrainH` is a C language opaque type.

A variable of this type is a handle object created when you add a constraint to a Simscape Multibody Link API object representing an API CAD model.

See Also

pmit_add_constrain | PmitConstrainType | pmit_create_constrain | PmitError

PmitConstrainType

Enumerated type for specifying constraint type

Description

Warning This content is specific to Simscape Multibody First Generation software. First-generation features are slated to be deprecated and should be avoided.

`PmitConstrainType` is a C language enumerated type.

A variable of this type is defined when you create a constraint in a Simscape Multibody Link API CAD model.

These are the variable's allowed enumerated values.

Value	Constraint Type
`PMIT_CON_UNKNOWN = -1`	Unknown
`PMIT_CON_COINCIDENT = 0`	Coincident points
`PMIT_CON_CONCENTRIC`	Concentric circles or circular arcs
`PMIT_CON_PERPEND`	Perpendicular lines or planes
`PMIT_CON_PARALLEL`	Parallel lines or planes
`PMIT_CON_TANGENT`	Tangent curves or surfaces
`PMIT_CON_DISTANCE`	Fixed distance between points
`PMIT_CON_ANGLE`	Fixed angle between lines
`PMIT_CON_FULL`	Fully fixing one body's position and orientation with respect to another body. Kinematically equivalent to a rigid weld.

See Also

`pmit_add_constrain` | `PmitConstrainH` | `pmit_create_constrain` | `PmitError`

pmit_create_assemcomp

Create object of `PmitAssemCompH` class

Syntax

`PmitError = pmit_create_assemcomp(PmitAssemCompH* const pmitAssemCompHOut)`

Description

Warning This content is specific to Simscape Multibody First Generation software. First-generation features are slated to be deprecated and should be avoided.

`PmitError = pmit_create_assemcomp(PmitAssemCompH* const pmitAssemCompHOut)` returns an error status `PmitError`.

With `pmit_create_assemcomp`, you can create an object of `PmitAssemCompH` class in order to reference child models in the hierarchy of other API CAD models.

Output Arguments

pmitAssemCompHOut

Handle object of `PmitAssemCompH` class representing a component in an API CAD model

Version History
Introduced in R2009b

See Also
`PmitAssemCompH` | `PmitError`

pmit_create_assemcomp_fromstr

Create object of `PmitAssemCompH` class

Syntax

Description

Warning This content is specific to Simscape Multibody First Generation software. First-generation features are slated to be deprecated and should be avoided.

`PmitError = pmit_create_assemcomp_fromstr(PmitAssemCompH* const pmitAssemCompHOut, const char* compName, PmitCadModelH parentModelH)` returns an error status `PmitError`.

With `pmit_create_assemcomp_fromstr`, you can create, from its string representation, an object of `PmitAssemCompH` class that represents an API CAD model component.

Input Arguments

compName

String specifying name of component

Default:

parentModelH

Handle object of class `PmitCadModelH` representing an API CAD model

Default:

Output Arguments

pmitAssemCompHOut

Handle object of `PmitAssemCompH` class representing a component in an API CAD model

Version History
Introduced in R2009b

See Also
PmitAssemCompH | PmitCadModelH | PmitError

pmit_create_cad2sm

Create object of `PmitCad2SMH` class

Syntax

`PmitError = pmit_create_cad2sm(PmitCad2SMH* const pmitCad2SMHOut, PmitCadModelH const pmitCadModelH, const char* createdUsing, const char* createdFrom, const char* createdOn, const char* createdBy, const char* name)`

Description

Warning This content is specific to Simscape Multibody First Generation software. First-generation features are slated to be deprecated and should be avoided.

`PmitError = pmit_create_cad2sm(PmitCad2SMH* const pmitCad2SMHOut, PmitCadModelH const pmitCadModelH, const char* createdUsing, const char* createdFrom, const char* createdOn, const char* createdBy, const char* name)` returns an error status `PmitError`.

With `pmit_create_cad2sm`, you can create an object of `PmitCad2SMH` class to represent an API-to-XML CAD model translator. The header information that you specify in the inputs is written to the final XML file.

Input Arguments

pmitCadModelH

Handle object of `PmitCadModelH` class representing an API CAD model

Default:

createdUsing

String naming the exporter

Default:

createdFrom

String naming the source CAD platform or other external application

Default:

createdOn

String specifying date that the object was created

Default:

createdBy

String specifying name of user creating the object

Default:

name

String naming the assembly model

Default:

Output Arguments

pmitCad2SMHOut

Handle object of `PmitCad2SMH` class representing an API-to-XML translator object

Version History
Introduced in R2009b

See Also
PmitCad2SMH | PmitCadModelH | PmitError | pmit_write_xml

Topics
"A Custom Exporter Module Example"

pmit_create_cadcs

Create object of `PmitCadCSH` class

Syntax

```
PmitError = pmit_create_cadcs(PmitCadCSH* const pmitCadCSHOut, const char*
name, const char* nodeID, double rotation[9], double trans[3])
```

Description

Warning This content is specific to Simscape Multibody First Generation software. First-generation features are slated to be deprecated and should be avoided.

`PmitError = pmit_create_cadcs(PmitCadCSH* const pmitCadCSHOut, const char* name, const char* nodeID, double rotation[9], double trans[3])` returns an error status `PmitError`.

With `pmit_create_cadcs`, you can create an object of `PmitCadCSH` class to represent a coordinate system in an API CAD model.

Input Arguments

name

String naming the coordinate system

Default:

nodeID

String uniquely identifying the coordinate system for associativity purposes

Default:

rotation

Double-type real rotation 9-vector specifying rotational transformation of the origin of this coordinate system with respect to its parent CAD model.

Default:

trans

Double-type real 3-vector specifying translation of the origin of this coordinate system with respect to its parent CAD model.

Default:

Output Arguments

`pmitCadCSHOut`

Handle object of `PmitCadCSH` class representing a coordinate system in an API CAD model

More About

Orthogonal Matrix

A matrix R is orthogonal if it satisfies the matrix multiplication rule $R^T*R = R*R^T = 1$, where 1 is the identity matrix.

Rotational Transformation: Rotation Matrix and Rotation Vector

The rotation vector input is a 9-vector, defined from the 3-by-3 orthogonal rotation matrix R, that represents the rotational orientation of a CAD model component with respect to its parent CAD model.

$$R = \begin{pmatrix} R_{11} & R_{12} & R_{13} \\ R_{21} & R_{22} & R_{23} \\ R_{31} & R_{32} & R_{33} \end{pmatrix}.$$

You define the rotation 9-vector column-wise:

```
rotation = [R(1,1) R(2,1) R(3,1) R(1,2) R(2,2) R(3,2) ...
            ... R(1,3) R(2,3) R(3,3)]
```

Version History
Introduced in R2010a

See Also
`pmit_add_cadcs` | `PmitCadCSH` | `PmitError`

pmit_create_cadmodel

Create object of `PmitCadModelH` class

Syntax

```
PmitError = pmit_create_cadmodel(PmitCadModelH* const pmitCadModelHOut, const
char* name, double mass, const double inertia[6], const double cg[3], double
volume, double sarea, const char* fileName, const PmitVisMatProp* matprops)
```

Description

Warning This content is specific to Simscape Multibody First Generation software. First-generation features are slated to be deprecated and should be avoided.

```
PmitError = pmit_create_cadmodel(PmitCadModelH* const pmitCadModelHOut, const
char* name, double mass, const double inertia[6], const double cg[3], double
volume, double sarea, const char* fileName, const PmitVisMatProp* matprops)
```
returns an error status `PmitError`.

With `pmit_create_cadmodel`, you can create an object of `PmitCadModelH` class to represent an API CAD model.

The body geometry file specified by `fileName` carries no units. This body geometry is interpreted with the units defined for the API session.

Input Arguments

name

String naming the CAD assembly or part model

Default:

mass

Double-type real number specifying the mass of the assembly or part

Default:

inertia

Double-type real 6-vector specifying the rotational inertia of the assembly or part. See "More About" on page 5-23.

Default:

cg

Double-type real 3-vector specifying the position of the center of gravity of the assembly or part

Default:

volume

Double-type real number specifying the volume of the assembly or part

Default:

sarea

Double-type real number specifying the surface area of the assembly or part

Default:

fileName

String specifying body geometry file name

Default:

matprops

Structure of `PmitVisMatProp` class specifying the visualizable properties of the assembly or part

Default:

Output Arguments

pmitCadModelHOut

Handle object of `PmitCadModelH` class representing an API CAD model

More About

Inertia Tensor and Inertia Vector

The inertia vector input is a 6-vector defined from the 3-by-3 symmetric inertia tensor I that depends on the part's mass distribution:

$$I = \begin{pmatrix} I_{11} & I_{12} & I_{13} \\ I_{21} & I_{22} & I_{23} \\ I_{31} & I_{32} & I_{33} \end{pmatrix},$$

where $I_{21} = I_{12}$, $I_{31} = I_{13}$, etc.

You define the inertia 6-vector as:

```
inertia = [I(1,1) I(2,2) I(3,3) I(1,2) I(3,1) I(2,3)]
```

Version History
Introduced in R2009b

See Also
PmitCadModelH | PmitError | pmit_set_units | PmitVisMatProp

pmit_create_cadmodelref

Create object of `PmitCadModelRefH` class

Syntax

```
PmitError = pmit_create_cadmodelref(PmitCadModelRefH* const
pmitCadModelRefHOut, const char* name, const char* nodeID, PmitCadModelH
pmitCadModelH, double rotation[9], double trans[3], double scale, int
isFlexible, int isFixed, const PmitVisMatProp* matprops)
```

Description

Warning This content is specific to Simscape Multibody First Generation software. First-generation features are slated to be deprecated and should be avoided.

```
PmitError = pmit_create_cadmodelref(PmitCadModelRefH* const
pmitCadModelRefHOut, const char* name, const char* nodeID, PmitCadModelH
pmitCadModelH, double rotation[9], double trans[3], double scale, int
isFlexible, int isFixed, const PmitVisMatProp* matprops)
``` returns an error status `PmitError`.

With `pmit_create_cadmodelref`, you can create an object of `PmitCadModelRefH` class to reference a CAD model in an API CAD model hierarchy.

## Input Arguments

**name**

String specifying name of component instance

**Default:**

**nodeID**

String specifying unique identity of model component within parent hierarchy. This identity must be unique within the full model.

**Default:**

**pmitCadModelH**

Handle object of `PmitCadModelH` class representing an API CAD model. This is the same model referenced by the output object `pmitCadModelRefHOut`, an object of `PmitCadModelRefH` class.

**Default:**

**rotation**

Double-type real rotation 9-vector specifying rotational transformation of the origin of this CAD model with respect to its parent CAD model. See "More About" on page 5-26.

**Default:**

**trans**

Double-type real 3-vector specifying translation of the origin of this CAD model with respect to its parent CAD model.

**Default:**

**scale**

Double-type real number specifying overall length scaling of this instance of the model. A value of 1 means no overall scaling.

**Default:**

**isFlexible**

Integer-type flag specifying whether component is rigid or nonrigid. A value of 0 means the component is rigid; a value of 1 means the component is nonrigid.

**Default:**

**isFixed**

Integer-type flag specifying whether component is welded or not to its attachment point in the assembly. A value of 0 means the component is not welded; a value of 1 means the component is welded. See "More About" on page 5-26.

**Default:**

**matprops**

Structure of type `PmitVisMatProp` for defining visualized material properties of the machine

**Default:**

## Output Arguments

**pmitCadModelRefHOut**

Handle object of `PmitCadModelRefH` class referencing a CAD model in an API CAD model hierarchy

## More About

### Orthogonal Matrix

A matrix $R$ is orthogonal if it satisfies the matrix multiplication rule $R^T*R = R*R^T = 1$, where $1$ is the identity matrix.

**Rotational Transformation: Rotation Matrix and Rotation Vector**

The rotation vector input is a 9-vector, defined from the 3-by-3 orthogonal rotation matrix $R$, that represents the rotational orientation of a CAD model component with respect to its parent CAD model.

$$R = \begin{pmatrix} R_{11} & R_{12} & R_{13} \\ R_{21} & R_{22} & R_{23} \\ R_{31} & R_{32} & R_{33} \end{pmatrix}.$$

You define the rotation 9-vector column-wise:

```
rotation = [R(1,1) R(2,1) R(3,1) R(1,2) R(2,2) R(3,2) ...
            ... R(1,3) R(2,3) R(3,3)]
```

Flexible Model

A flexible or nonrigid model is made of components that can move with respect to one another.

An inflexible or rigid model is made of components that cannot move with respect to one another.

Fixed Model

A fixed model cannot move relative to the ground of the assembly model.

A nonfixed model can move relative to the ground of the assembly hierarchy.

Version History
Introduced in R2009b

See Also
PmitCadModelH | PmitCadModelRefH | PmitError | PmitVisMatProp

pmit_create_constrain

Create object of `PmitConstrainH` class

Syntax

Description

> **Warning** This content is specific to Simscape Multibody First Generation software. First-generation features are slated to be deprecated and should be avoided.

`PmitError = pmit_create_constrain(PmitConstrainH* const pmitConstrainhOut, const char* name, PmitConstrainType type, PmitAssemCompH body1Comp, PmitAssemCompH body2Comp, PmitGeomType body1Type, PmitGeomType body2Type, const double body1Loc, const double body1Axis, const double body2Loc, const double body2Axis)` returns an error status `PmitError`.

With `pmit_create_constrain`, you can create an object of `PmitConstrainH` class to represent a constraint in an API CAD model.

Input Arguments

For a complete specification of these inputs, see "More About" on page 5-29.

name

String naming the constraint

Default:

type

Handle object of `PmitConstrainType` class to represent constraint type in an API CAD model

Default:

body1Comp

Handle object of `PmitAssemCompH` class to represent first constrained body in an API CAD model

Default:

body2Comp

Handle object of `PmitAssemCompH` class to represent second constrained body in an API CAD model

Default:

body1Type

Handle object of `PmitGeomType` class to represent the geometry of first constrained body in an API CAD model

Default:

body2Type

Handle object of `PmitGeomType` class to represent the geometry of second constrained body in an API CAD model

Default:

body1Loc

Double-type 3-vector specifying the spatial location of body 1

Default:

body1Axis

Double-type 3-vector specifying the spatial orientation of the axis of body 1

Default:

body2Loc

Double-type 3-vector specifying the spatial location of body 2

Default:

body2Axis

Double-type 3-vector specifying the spatial orientation of the axis of body 2

Default:

Output Arguments

pmitConstrainhOut

Handle object of `PmitConstrainH` class to represent a constraint in an API CAD model

More About

Constraint

A constraint imposes a restriction on how two component bodies can move relative to one another.

You define a constraint by an axis through a point oriented and located, respectively, with respect to body 1.

Body Specification

To impose a constraint, specify the two bodies by their:

- Component handles
- Component body geometry type handles
- Locations in space. The location of body 2 is a translation with respect to the coordinate origin of the CAD model representing body 1.
- Axis directions in space . The axis of body 2 is a direction with respect to the coordinate axes of the CAD model representing body 1.

Version History
Introduced in R2009b

See Also
PmitAssemCompH | PmitConstrainH | PmitConstrainType | PmitError | PmitGeomType

pmit_disconnectfrom_matlab

Disconnect from MATLAB session

Syntax

PmitError = pmit_disconnectfrom_matlab()

Description

Warning This content is specific to Simscape Multibody First Generation software. First-generation features are slated to be deprecated and should be avoided.

PmitError = pmit_disconnectfrom_matlab() returns an error status PmitError.

Version History
Introduced in R2009b

See Also
pmit_connectto_matlab | PmitError

PmitError

Enumerated type for error status

Description

Warning This content is specific to Simscape Multibody First Generation software. First-generation features are slated to be deprecated and should be avoided.

`PmitError` is a C language enumerated type.

A variable of this type is defined whenever you call any Simscape Multibody Link API function.

These are the variable's allowed enumerated values.

Value	Error Type
PMIT_NO_ERROR = 0	No error
PMIT_GENERIC_FAIL	Function call failure not otherwise specified
PMIT_CAD_MODEL_NOTSET	API representation of machine not defined
PMIT_XML_DOM_ERROR	XML error
PMIT_UNHANDLED_CONSTRAIN	Constraint translation error
PMIT_INVALID_CON_COMPS	
PMIT_UNSUPPORTED_INERTIA_UNIT	Mass or inertia unit specified that is not supported by API
PMIT_COULDNOT_CONNECTTO_MATLAB	Failure to connect to MATLAB

PmitGeomType

Enumerated type for specifying geometry of component

Description

Warning This content is specific to Simscape Multibody First Generation software. First-generation features are slated to be deprecated and should be avoided.

`PmitGeomType` is a C language enumerated type.

A variable of this type is defined when you create a component in a Simscape Multibody Link API CAD model.

These are the variable's allowed enumerated values.

Value	Geometry Type
PMIT_GEO_UNKNOWN = -1	Unknown
PMIT_GEO_POINT = 0	Point
PMIT_GEO_LINE	Line
PMIT_GEO_PLANE	Plane
PMIT_GEO_CYL	Cylinder
PMIT_GEO_CONE	Cone
PMIT_GEO_CIRCLE	Circle

See Also
pmit_create_constrain | PmitError

pmit_get_reffixedstatus

Get fixed status of CAD model

Syntax

```
PmitError = pmit_get_reffixedstatus(int* fixedstatusOut, const
PmitCadModelRefH cadModelRefH)
```

Description

Warning This content is specific to Simscape Multibody First Generation software. First-generation features are slated to be deprecated and should be avoided.

`PmitError = pmit_get_reffixedstatus(int* fixedstatusOut, const PmitCadModelRefH cadModelRefH)` returns an error status `PmitError`.

With `pmit_get_reffixedstatus`, you can get the fixed status of a CAD model referenced by an object of `PmitCadModelRefH` class.

Input Arguments

cadModelRefH

Handle object of `PmitCadModelRefH` class referencing a CAD model in an API CAD model hierarchy

Default:

Output Arguments

fixedstatusOut

Integer flag indicating if the model is fixed or not. A value of 0 means the model is not fixed. A value of 1 means the model is fixed. See "More About" on page 5-34.

More About

Fixed Model

A fixed model cannot move relative to the ground of the assembly model.

A nonfixed model can move relative to the ground of the assembly hierarchy.

Version History
Introduced in R2009b

See Also
PmitCadModelRefH | PmitError

pmit_get_refflexiblestatus

Get flexible status of CAD model

Syntax

```
PmitError = pmit_get_refflexiblestatus(int* flexstatusOut, const
PmitCadModelRefH cadModelRefH)
```

Description

Warning This content is specific to Simscape Multibody First Generation software. First-generation features are slated to be deprecated and should be avoided.

`PmitError = pmit_get_refflexiblestatus(int* flexstatusOut, const PmitCadModelRefH cadModelRefH)` returns an error status `PmitError`.

With `pmit_get_refflexiblestatus`, you can get the flexible status of a CAD model referenced by an object of `PmitCadModelRefH` class.

Input Arguments

cadModelRefH

Handle object of `PmitCadModelRefH` class referencing a CAD model in an API CAD model hierarchy

Default:

Output Arguments

flexstatusOut

Integer flag indicating if the model is flexible or not. A value of 0 means the model is inflexible, or rigid. A value of 1 means the model is flexible, or nonrigid. "More About" on page 5-36.

More About

Flexible Model

A flexible or nonrigid model is made of components that can move with respect to one another.

An inflexible or rigid model is made of components that cannot move with respect to one another.

Version History
Introduced in R2009b

See Also
PmitCadModelRefH | PmitError

PmitLengthUnit

Enumerated type for specifying length unit in API session

Description

Warning This content is specific to Simscape Multibody First Generation software. First-generation features are slated to be deprecated and should be avoided.

`PmitLengthUnit` is a C language enumerated type.

You can define a variable of this type globally when you start a Simscape Multibody Link API session.

These are the variable's allowed enumerated values.

Value	Length Unit Type
PMIT_LU_UNKNOWN = -1	Unknown
PMIT_LU_M = 0	Meter
PMIT_LU_CM	Centimeter
PMIT_LU_MM	Millimeter
PMIT_LU_KM	Kilometer
PMIT_LU_IN	Inch
PMIT_LU_FT	Foot
PMIT_LU_MI	Mile
PMIT_LU_YD	Yard

See Also
PmitError | PmitMassUnit | pmit_set_units

PmitMassUnit

Enumerated type for specifying mass unit in API session

Description

Warning This content is specific to Simscape Multibody First Generation software. First-generation features are slated to be deprecated and should be avoided.

`PmitMassUnit` is a C language enumerated type.

You can define a variable of this type globally when you start a Simscape Multibody Link API session.

These are the variable's allowed enumerated values.

Value	Mass Unit Type
PMIT_MU_UNKNOWN = -1	Unknown
PMIT_MU_KG = 0	Kilogram
PMIT_MU_G	Gram
PMIT_MU_MG	Milligram
PMIT_MU_LBM	Pound (mass)
PMIT_MU_OZ	Ounce
PMIT_MU_SLUG	Slug

See Also
PmitError | PmitLengthUnit | pmit_set_units

PmitObjectH

Handle object type to represent any API object

Description

Warning This content is specific to Simscape Multibody First Generation software. First-generation features are slated to be deprecated and should be avoided.

`PmitObjectH` is a C language opaque type.

You can define a variable of this type for any object created by the Simscape Multibody Link API.

See Also
`PmitError` | `pmit_release_object`

pmit_open_demo

Open Simscape Multibody Link examples in MATLAB Help browser

Syntax

PmitError = pmit_open_demo()

Description

Warning This content is specific to Simscape Multibody First Generation software. First-generation features are slated to be deprecated and should be avoided.

PmitError = pmit_open_demo() returns an error status PmitError.

Version History
Introduced in R2009b

See Also
demo | pmit_connectto_matlab | PmitError | pmit_open_help

pmit_open_help

Open product documentation in MATLAB Help browser

Syntax

PmitError = pmit_open_help(const char* helpItem)

Description

Warning This content is specific to Simscape Multibody First Generation software. First-generation features are slated to be deprecated and should be avoided.

PmitError = pmit_open_help(const char* helpItem) returns an error status PmitError.

This function causes MATLAB to issue the command:

doc *StringValue*

StringValue is the value of the string helpItem.

Input Arguments

helpItem

String specifying the product documentation item to display in the MATLAB Help browser

Default:

Version History
Introduced in R2009b

See Also
doc | pmit_connectto_matlab | PmitError | pmit_open_demo

pmit_release_buffer

Release character buffer returned by API function

Syntax

```
PmitError = pmit_release_buffer(char** buffer)
```

Description

Warning This content is specific to Simscape Multibody First Generation software. First-generation features are slated to be deprecated and should be avoided.

`PmitError = pmit_release_buffer(char** buffer)` returns an error status `PmitError`.

Input Arguments

buffer

String specifying the name of the buffer that you want to release

Default:

Version History
Introduced in R2009b

See Also
`PmitError` | `pmit_release_object`

pmit_release_object

Release object used by API session

Syntax

PmitError = pmit_release_object(PmitObjectH objectH)

Description

Warning This content is specific to Simscape Multibody First Generation software. First-generation features are slated to be deprecated and should be avoided.

PmitError = pmit_release_object(PmitObjectH objectH) returns an error status PmitError.

Input Arguments

objectH

Handle object of PmitObjectH class representing the API CAD object that you want to release

Default:

Version History
Introduced in R2009b

See Also
PmitError | PmitObjectH | pmit_release_buffer

pmit_set_reffixedstatus

Set fixed status of CAD model

Syntax

PmitError = pmit_set_reffixedstatus(PmitCadModelRefH cadModelRefH, int status)

Description

Warning This content is specific to Simscape Multibody First Generation software. First-generation features are slated to be deprecated and should be avoided.

PmitError = pmit_set_reffixedstatus(PmitCadModelRefH cadModelRefH, int status) returns an error status PmitError.

With pmit_set_reffixedstatus, you can set the fixed status of a CAD model referenced by an object of PmitCadModelRefH class.

Input Arguments

cadModelRefH

Handle object of PmitCadModelRefH class referencing a CAD model in an API CAD model hierarchy

Default:

status

Integer flag indicating if the model is fixed or not. A value of 0 means the model is not fixed. A value of 1 means the model is fixed. "More About" on page 5-45.

Default:

More About

Fixed Model

A fixed model cannot move relative to the ground of the assembly hierarchy.

A nonfixed model can move relative to the ground of the assembly hierarchy.

Version History
Introduced in R2009b

See Also
PmitCadModelRefH | PmitError

pmit_set_refflexiblestatus

Set flexible status of CAD model

Syntax

```
PmitError = pmit_set_refflexiblestatus(PmitCadModelRefH cadModelRefH, int
status)
```

Description

Warning This content is specific to Simscape Multibody First Generation software. First-generation features are slated to be deprecated and should be avoided.

`PmitError = pmit_set_refflexiblestatus(PmitCadModelRefH cadModelRefH, int status)` returns an error status `PmitError`.

With `pmit_set_refflexiblestatus`, you can set the flexible status of a CAD model referenced by an object of `PmitCadModelRefH` class.

Input Arguments

cadModelRefH

Handle object of `PmitCadModelRefH` class referencing a CAD model within an API CAD model hierarchy

Default:

status

Integer flag indicating if the model is flexible or not. A value of 0 means the model is inflexible, or rigid. A value of 1 means the model is flexible, or nonrigid. "More About" on page 5-47.

Default:

More About

Flexible Model

A flexible or nonrigid model is made of components that can move with respect to one another.

An inflexible or rigid model is made of components that cannot move with respect to one another.

Version History
Introduced in R2009b

See Also
PmitCadModelRefH | PmitError

pmit_set_tolerances

Set linear, angular, and relative tolerances of object of `PmitCad2SMH` class

Syntax

```
PmitError = pmit_set_tolerances(PmitCad2SMH pmitCad2SMH, double linearTol,
double angularTol, double relativeTol)
```

Description

Warning This content is specific to Simscape Multibody First Generation software. First-generation features are slated to be deprecated and should be avoided.

`PmitError = pmit_set_tolerances(PmitCad2SMH pmitCad2SMH, double linearTol, double angularTol, double relativeTol)` returns an error status `PmitError`.

With `pmit_set_tolerances`, you can set the linear, angular, and relative tolerances of an object of `PmitCad2SMH` class representing an API-to-XML translator.

Input Arguments

pmitCad2SMH

Handle object of `PmitCad2SMH` class representing an API-to-XML translator

linearTol

Error tolerance when comparing linear alignments and spacings, measured in length unit specified by `PmitLengthUnit`

Default:

angularTol

Error tolerance when comparing angular alignments and spacings, measured in radians

Default:

relativeTol

Smallest significant relative numerical difference

Default:

Version History
Introduced in R2009b

See Also
PmitCad2SMH | PmitError | PmitLengthUnit | pmit_set_units

pmit_set_units

Set units for API session

Syntax

PmitError = pmit_set_units(PmitMassUnit massUnit, PmitLengthUnit lenUnit)

Description

Warning This content is specific to Simscape Multibody First Generation software. First-generation features are slated to be deprecated and should be avoided.

PmitError = pmit_set_units(PmitMassUnit massUnit, PmitLengthUnit lenUnit) returns an error status PmitError.

With pmit_set_units, you can set the units for an API session.

Input Arguments

massUnit

Input of enumerated type PmitMassUnit specifying the mass unit system

Default:

lenUnit

Input of enumerated type PmitLengthUnit specifying the length unit system

Default:

Version History
Introduced in R2009b

See Also
PmitError | PmitLengthUnit | PmitMassUnit | pmit_set_tolerances

PmitVisMatProp

Structure type for defining visualized material properties of API CAD object

Description

Warning This content is specific to Simscape Multibody First Generation software. First-generation features are slated to be deprecated and should be avoided.

`PmitVisMatProp` is a C language structure type.

You can define a variable of this type for any object in a Simscape Multibody Link API CAD model. This variable specifies the visualized material properties of the object, usually a part component of a CAD assembly.

You refer to the fields of the structure as `PmitVisMatProp.`*`field`*. These are the structure fields and their possible values, which all range from 0 to 1.

Field	Values
rgb	3-vector [r g b] specifying red, green, and blue color intensities r, g, and b
ambient	Intensity of the ambient component of light falling on the component
diffuse	Intensity of the diffuse component of light falling on the component
specular	Intensity of the specular component of light falling on the component
shininess	Shininess coefficient of the component's material
transparency	Transparency factor of the component's material. 0 means the material is not transparent. 1 means it is fully transparent.
emission	Intensity of emission from the component's material

See Also
pmit_create_cadmodel | pmit_create_cadmodelref | PmitError

External Websites
OpenGL resources on visualized lighting and material properties

pmit_write_xml

Write output of object of `PmitCad2SMH` class

Syntax

`PmitError = mit_write_xml(char** const pconstrainErrorOut, PmitCad2SMH pmitCad2SMH, const char* filename)`

Description

Warning This content is specific to Simscape Multibody First Generation software. First-generation features are slated to be deprecated and should be avoided.

`PmitError = mit_write_xml(char** const pconstrainErrorOut, PmitCad2SMH pmitCad2SMH, const char* filename)` returns an error status `PmitError`.

With `pmit_write_xml`, you can write the output of an object of `PmitCad2SMH` class to an XML multibody description file.

Input Arguments

pmitCad2SMH

Handle object of `PmitCad2SMH` class representing an API-to-XML translator object

Default:

filename

String specifying the name of the XML file to which the API representation is written

Default:

Output Arguments

pconstrainErrorOut

String indicating constraint errors, if any, encountered while writing the XML file

Version History
Introduced in R2009b

See Also
`PmitCad2SMH` | `pmit_create_cad2sm` | `PmitError`

Topics
"A Custom Exporter Module Example"