Dymola

Dynamic Modeling Laboratory

Dymola Release Notes

Dymola 2023x Refresh 1

Document version: 1

Dassault Systèmes AB
Ideon Gateway
Scheelevägen 27 – Floor 9
SE-223 63 Lund
Sweden

Support:	https://www.3ds.com/support
URL:	https://www.dymola.com/
Phone:	+46 46 270 67 00

Contents

1 Important notes on Dymola

Installation on Windows

To translate models on Windows, you must also install a supported compiler. The compiler is not distributed with Dymola. Note that administrator privileges are required for installation. Three types of compilers are supported on Windows in Dymola 2023x Refresh 1:

Microsoft Visual Studio C++

This is the recommended compiler for professional users. Both free and full compiler versions are supported. Refer to section "Compilers" on page 58 for more information. **Notes**:

- From Dymola 2020x, Visual Studio C++ compilers older than version 2012 are no longer supported.
- From Dymola 2022x, Visual Studio 2013 is not supported anymore, due to the logistics of supporting multiple old versions for all solvers. (Visual Studio 2012 is however still supported, due to the logistics of changing the oldest supported version.)

Intel

Important. The support for Intel compilers is discontinued from the previous Dymola 2022 release.

MinGW GCC

Dymola 2023x Refresh 1 has limited support for the MinGW GCC compiler, 32-bit and 64-bit. For more information about MinGW GCC, see section "Compilers" on page 58, the section about MinGW GCC compiler.

WSL GCC (Linux cross-compiler)

Dymola 2023x Refresh 1 has limited support for the WSL (Windows Subsystem for Linux) GCC compiler, 64-bit. For more information about WLS GCC, see section "Compilers" on page 58, the section about WSL GCC compiler.

Installation on Linux

To translate models, Linux relies on a GCC compiler, which is usually part of the Linux distribution. Refer to section "Supported Linux versions and compilers" on page 61 for more information.

2 About this booklet

This booklet covers Dymola 2023x Refresh 1. The disposition is similar to the one in Dymola User Manuals; the same main headings are being used (except for, e.g., Libraries and Documentation).

3 Dymola 2023x Refresh 1

3.1 Introduction

3.1.1 Additions and improvements in Dymola

A number of improvements and additions have been implemented in Dymola 2023x Refresh 1. In particular, Dymola 2023x Refresh 1 provides:

- Support for Modelica Language Specification 3.6 (page 56)
- Basic GUI for selective model extension (page 9)
- File browser displaying also non-model files (page 9)
- Improved unit checking (page 12)
- Discontinued support for Windows 8.1 (page 29)
- Support for FMI 3.0:
    - Array handling (page 52)
    - Terminals and icons (*beta feature*) (page 52)
    - Support of hybrid co-simulation (*beta feature*) (page 52)
- Support for FMI 1.0 will be discontinued in a future Dymola release (page 53)
- eFMI improvements (page 56)
- SSP support for SRMD (page 51)
- Improvements of the sweeping functionality:
    - New GUI for Monte Carlo sweeping (page 33)
    - Advanced options dialog (page 37)
    - Sweeping two or more parameters with trajectory result (page 41)
    - Changes and improvements in the Design.Experimentation package (page 41)
- Versioning: Extended Git support (page 42)
- Improvements for plotting:
    - Further improvements for predefined plots (page 17)
    - Export curve data to CSV or SDF format (page 20)
- “Features Under Development” – preview of features to be formally released in a later version (page 29)

3.1.2 New and updated libraries

New libraries

There are no new libraries in this Dymola version.

Updated libraries

The following libraries have been updated:

- Aviation Systems Library, version 1.4.1
- Battery Library, version 2.5.0
- Brushless DC Drives Library, version 1.3.0
- ClaRa DCS Library, version 1.7.1
- ClaRa Grid Library, version 1.7.1
- ClaRa Plus Library, version 1.7.1
- Claytex Library, version 2023.1
- Claytex Fluid Library, version 2023.1
- Cooling Library, version 1.5.0
- Dassault Systemes Library, version 1.10.0
- Design, version 1.2.0
- Dymola Commands Library, version 1.15
- Dymola Embedded Library, version 1.0.1
- Dymola Models Library, version 1.6.0
- eFMI Library, version 3.0.1
- Electric Power Systems Library, version 1.6.2
- Electrified Powertrains Library (ETPL), version 1.7.0
- Flexible Bodies Library, version 2.4.0
- Fluid Dynamics Library, version 2.15.0
- Fluid Power Library, version 2023.1
- FTire Interface Library, version 1.1.5
- Human Comfort Library, version 2.15.0
- HVAC (Heating, Ventilation, and Air Conditioning) Library, version 2.15.0
- Hydrogen Library, version 1.3.8
- Modelica_DeviceDrivers Library, version 2.1.1
- Multiflash Media Library, version 1.1.2
- Pneumatic Systems Library, version 1.6.0
- Testing Library, version 1.6.1
- Thermal Systems Library, version 1.11.0
- Thermal Systems Mobile AC Library, version 1.11.0
- VeSyMA (Vehicle Systems Modeling and Analysis) Library, version 2023.1
- VeSyMA - Engines Library, version 2023.1
- VeSyMA - Powertrain Library, version 2023.1
- VeSyMA - Suspensions Library, version 2023.1
- VeSyMA2ETPL Library, version 2023.1
- Visa2Base, version 1.14

- Visa2Paper, version 1.14
- Visa2Steam, version 1.14

For more information about the updated libraries, please see the Release Notes section in the documentation for each library, respectively.

3.2 Developing a model

3.2.1 Basic GUI for selective model extension

Selective model extension is included in the Modelica Language Specification 3.6 (see also https://github.com/modelica/ModelicaSpecification/tree/master/RationaleMCP/0032) and is now more fully supported in the GUI.

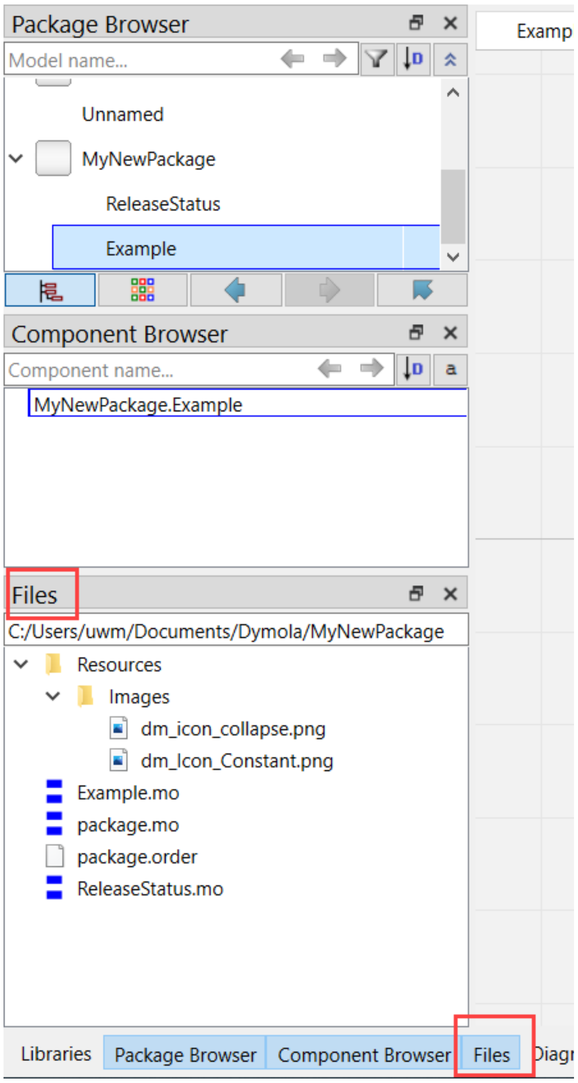
Attempting to delete an inherited component or connection in the diagram layer gives you the possibility to instead *deselect* the inherited element. The prompt can be disabled by setting the flag `Advanced.Editor.DeselectWarning=false;`.

Notes:

- When you want to deselect components, the best is to graphically select the components in the diagram and then press the **Delete** key.
- To regret the deselection of a component, you can right-click the deselected component in the component browser and click the activated **Deselect** menu entry.
- There is currently no graphical support to regret the deselection of a connection. Please do so in the text layer.

3.2.2 File browser displaying also non-model files

You can now activate a file browser that display all files in the root folder for an open model. You activate it by clicking **Files** in the lower line of the Dymola window. An example:

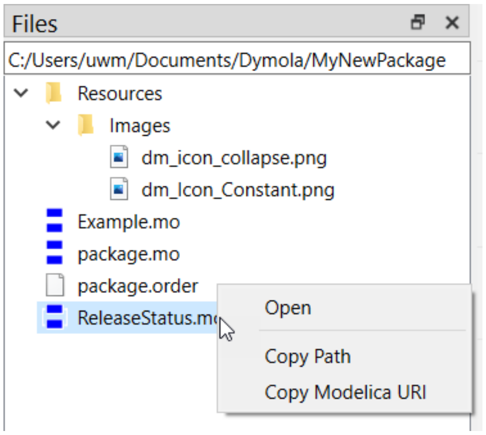


When you open a model, the file browser switches to its root folder.

The path to the root folder is displayed in a top line of the file browser. If the path is long, you can click on the line and use the keys **Home** and **End**. (You can also widen the browser.)

You can drag images from the file browser to the diagram or icon layer.

There is a context menu for the items in the File browser:



Open opens the file in any of:

- For `mo`, `moe`, or `mos` files they are opened in Dymola.
- For other file types, the file opens in the default application.

(You can also double-click the file to get this action.)

Copy path copies the file path to the clipboard. An example could be `E:/MyExperiments/MyNewPackage/ReleaseStatus.mo`.

Copy Modelica URI copies the Modelica URI to the clipboard. An example could be `modelica://MyNewPackage/ReleaseStatus.mo`.

3.2.3 Matching and variable selection

Top-level input and output variables not included anymore in the default selection

As soon as variable selections are used, there is a default selection **Parameters and States** present. This selection previously contained:

- Parameters
- State variables
- Top-level input and output variables (if any)

In Dymola 2023x Refresh 1, the last item, “Top-level input and output variables (if any)” are not included anymore in **Parameters and States**.

This means that top-level input and output variables are handled in the variable selections like auxiliary variables.

3.2.4 Improved unit checking

General

Some general improvements:

- Functions such as sin, cos, asin etc. are now unit-checked, that is, checked that they conform to the units in Modelica.Math. The feature can be disabled by setting the new flag `Advanced.Modelica.CheckUnitsMathFunctions = false`. (The flag is by default `true`.)
- The special case of mixing unit "m" and "mm" now get a special message.
- Unit checking for expressions involving numbers can be improved by setting the new flag `Advanced.Modelica.CheckUnitsProposedSimple = true`. (The flag is by default `false`). This checking may report additional errors. A simple explanation is that with this flag activated, unit checking ignores numbers – thus x+2 and 2*x will both have the same unit as x, whereas the default unit checking treats number as wildcards, so that 2*x is seen to have an unknown unit. Note that this flag enables a recently proposed revision to the Modelica language.

Unit propagation

To handle failures when propagating units, the new integer flag `Advanced.Translation.Log.UnitPropagationFailure` has the following alternatives:

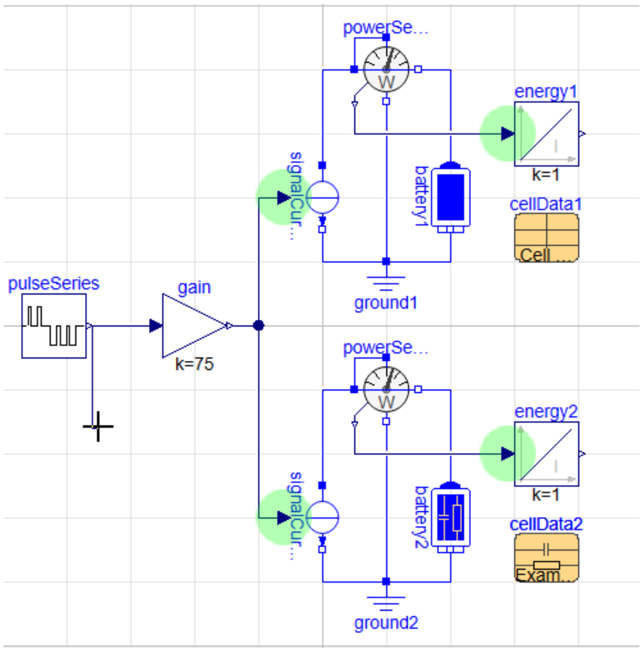
- 0 – (default) Don´t report propagation failure
- 10 – Report that propagation failed (if it does)
- 20 – Report where it failed

3.2.5 Minor improvements

Option to highlight good choices when starting a connection

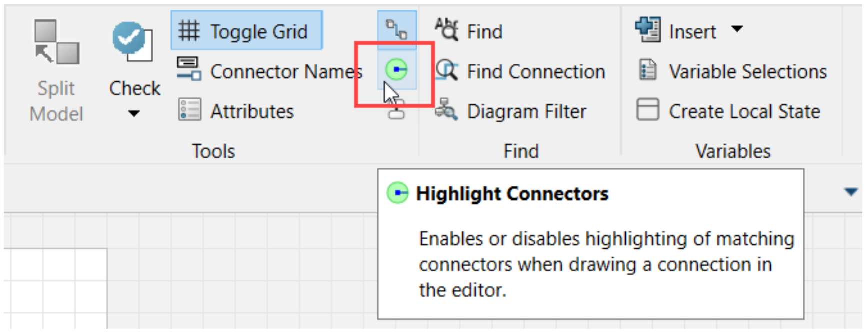
In Dymola 2023x Refresh 1, when you start drawing a connection, the suitable connectors to connect to are highlighted.

An example (Modelica.Electrical.Batteries.Examples.BatteryDischargeCharge, duplicated to be able to edit):



In this case, the connection was drawn from the output connector of the pulseSeries instance. The suitable output connectors are highlighted in green.

The option is by default activated. It can be deactivated by clicking the **Graphics > Highlight Connectors** button:



Highlight Connectors

Enables or disables highlighting of matching connectors when drawing a connection in the editor.

The feature corresponds to the flag

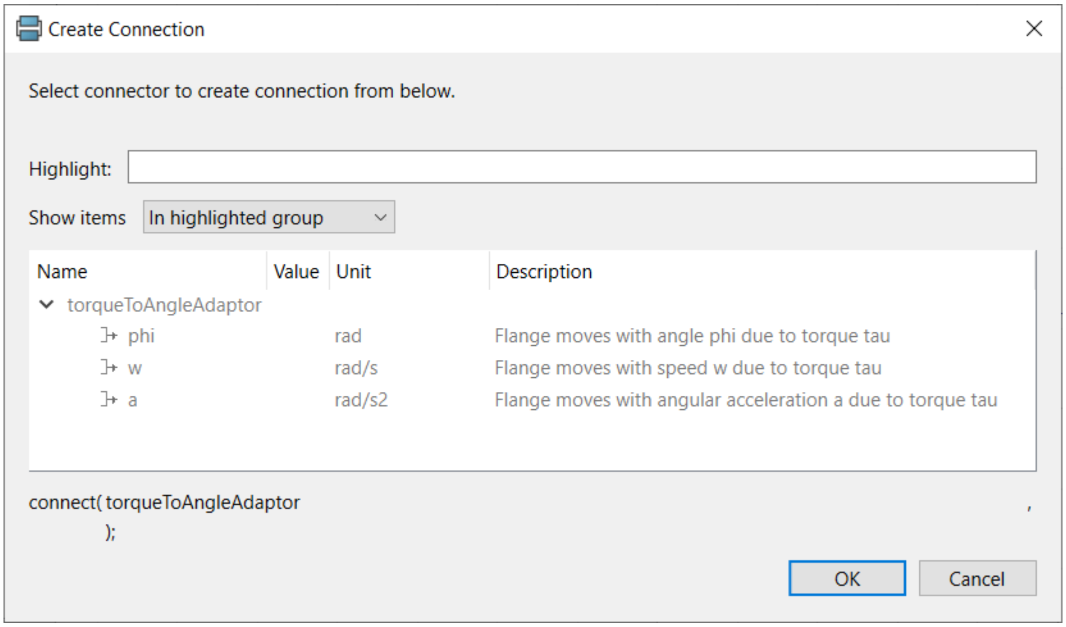
```
Advanced.Editor.Highlight.MatchingConnectors
```

(The flag is by default true.) The value of the flag is saved between sessions.

(This feature was a beta feature in the previous Dymola version.)

Improved handling of the context command Create Connector when creating a connection

For components with many connectors close together, you now get a dialog to help you select the wanted connector for creating the connection and corresponding connector when dragging a connection line and using the context menu entry **Create Connector**. An example of such a component is `Modelica.Mechanics.Rotational.Components.TorqueToAngleAdaptor`. Dragging a connection line from such a component and right-clicking and selecting **Create Connector** may display:



Select connector to create connection from below.

Name	Value	Unit	Description
torqueToAngleAdaptor			
phi		rad	Flange moves with angle phi due to torque tau
w		rad/s	Flange moves with speed w due to torque tau
a		rad/s2	Flange moves with angular acceleration a due to torque tau

```
connect(torqueToAngleAdaptor                                     ,
        );
```

Notes:

- This feature is also used for the context command **Create Node**.
- What result you get depends on the size of the component and where you click when you start dragging the connection. If you get a hit on a connector, you get the corresponding connection and node directly.

Improved handling of locked classes when applying renaming

If you are using the possibility to lock classes for editing, note that now there is a new flag:

```
Advanced.Editor.BreakLockWithConversion
```

The flag can be used to handle the issue that a locked class by default cannot be modified, and thus the normal renaming mechanism that work on all classes (conversion scripts and renaming with the default smart renaming) will silently skip renaming these locked classes, which will lead to an inconsistent result.

The flag can have any of the following values:

- 0: Locked classes are ignored when elements are renamed; this was the previous behavior and will cause such classes to be broken.
- 1: A specific warning is given before overriding the locks.
- 2: Locks are silently overridden when renaming classes. This is the current default value.

The flag value is currently not stored between sessions.

Option to use an earlier version of a library without updating it

In some cases, you may wish not to use the latest version of a library, but an earlier version. If the differences between the library versions are not affecting the package or library where you use the library (except the uses annotations), you can now set the new flag

```
Advanced.Editor.DelayConversionIfNotNeeded = true
```

If this flag is set, and you use such an earlier version of a library, only the uses annotations will be changed; the library will not be updated when you load it. However, this used older version will made read-only, to prevent changing it, since there is a newer version already available.

If you use an earlier version of a library where changes in the new version *do* affect the package or library where you use it, you will a get a warning when loading it, and the loaded library will be updated like in previous versions.

The flag is by default false.

Displaying a list of libraries that must be loaded for a certain model

You can display a list of all libraries that must be loaded for a certain model, by the new built-in function `getDependentLibraries`. For more info, see section “Displaying library dependencies for a certain model” on page 22.

Support for national characters – improved UTF-8 handling

The following changes have been implemented for UTF-8 handling:

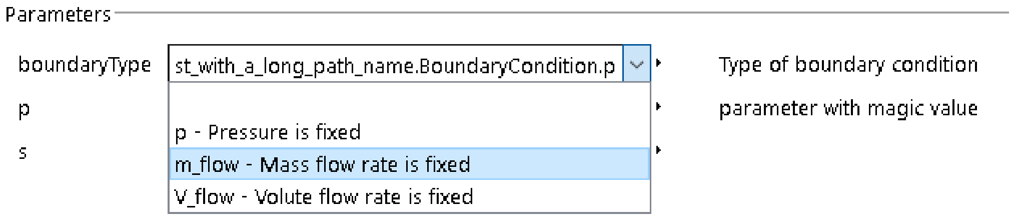
- The flag `Advanced.File.PreferWritingLatin1` has been removed.
- The default is now to write UTF-8 without byte-order-mark, the previous behavior with byte-order-mark can be activated by setting the new flag `Advanced.File.SupportUnicodeFiles=2;`. If you want compatibility with older Dymola versions please contact support to activate the new feature in old Dymola versions.
- The flag `Advanced.File.AlwaysWriteUTF8` only has an effect if `Advanced.File.SupportUnicodeFiles=2;` (since an UTF-8 file without byte-order-mark only containing ASCII characters also is an ASCII file).
- The above flags will be removed in a future Dymola version.

Improved display of enumeration values in the parameter dialog

The parameter dialog for enumerations is updated in two ways:

- The enumerator is always shown right aligned, also for the default value. This means that the enumerator is visible even if the full path is long.
- The drop-down menu shows both the enumerator name and the description (previously only the description was shown).

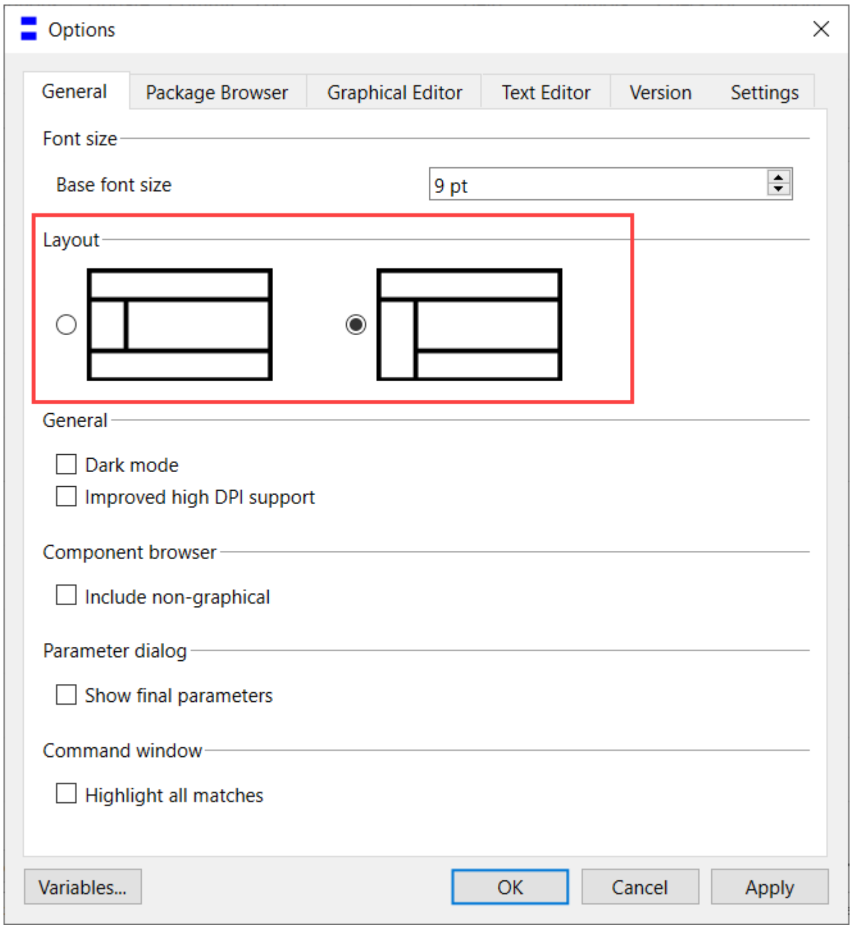
An example of parameter dialog with enumerations:


Parameters
boundaryType
st_with_a_long_path_name.BoundaryCondition.p
Type of boundary condition
p
parameter with magic value
s
p - Pressure is fixed
m_flow - Mass flow rate is fixed
V_flow - Volute flow rate is fixed

Option to control the width of the bottom pane in the Dymola window

You can now select if the bottom pane (usually the Logs/Commands pane) should span the entire Dymola window, or just the part to the right.

You select this by using the command **Tools > Options**, the **General** tab:



Options

General | Package Browser | Graphical Editor | Text Editor | Version | Settings

Font size

Base font size: 9 pt

Layout

○ [✓ ●]

General

☐ Dark mode
☐ Improved high DPI support

Component browser

☐ Include non-graphical

Parameter dialog

☐ Show final parameters

Command window

☐ Highlight all matches

Variables... | OK | Cancel | Apply

Note that the default is now the selected alternative in the figure above; previously the whole window width was used.

You must restart Dymola for any change of this option.

3.3 Simulating a model

3.3.1 Plot tab

Further improvements for predefined plots

Already Dymola 2021x supported much of the now accepted proposal **Annotation for Predefined Plots**, which is now included in the Modelica Language Specification version 3.6, but some features were not implemented. Now the following features are also implemented:

- Empty legends
- Variable expansion, such as `%(variable:x)` and `%x`, in the caption
- Modelica links, such as `%(modelica:///Modelica.Blocks)` and `%[link](modelica:///Modelica.Blocks)`
- Arbitrary links, such as `%(htttp://www.modelica.org)` and `%[link](http://www.modelica.org)`

For the first item, empty legends, see next section.

The other items are covered by the following implemented substitutions:

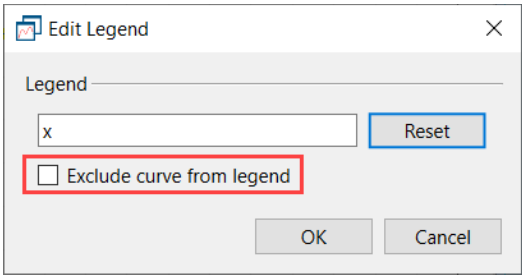
Pattern	**Shown as**	**Comment**
`%[text](…)`	`text`	User-defined substitution text
`%(modelica:///path)`	`path`	
`%(http://x)`	`http://x`	
`%(https://x)`	`https://x`	
`%(variable:x)`	`x`	
`%(plot:x)`	`<plot>`	References to other plots not implemented
`%{variable}`	`?`	Value substitution not implemented

Note: Links are not clickable in the dialog displaying the caption.

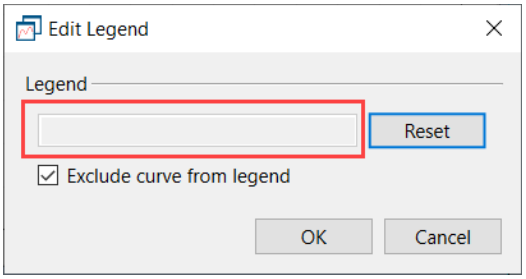
Not displaying the legend for a curve variable (GUI)

There is sometimes a need to plot curves but not show the variable in the legend. Previously, it was only possible to remove the entire legend. It should be noted that this is a feature of plots as defined in the Modelica specification.

To not display the legend for a curve variable, there is now a new option **Exclude curve from legend** in the context menu entry **Edit Legend** of the curve in a plot:

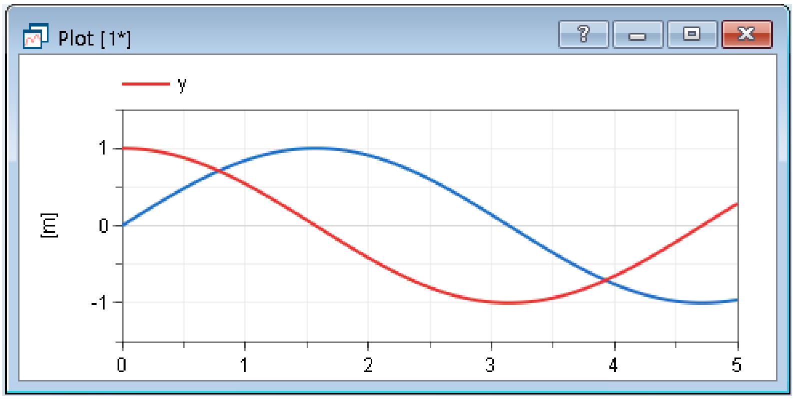


Activating this option gives:

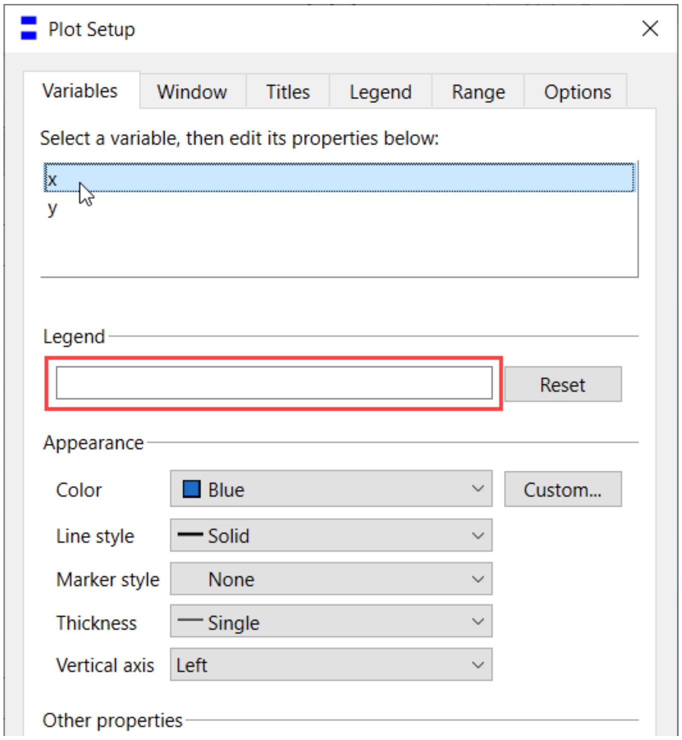


☑ Exclude curve from legend

In the plot, the result is (an example, in the below figure, the blue curve is the curve for the variable x):



In the plot setup (reached by right-clicking the plot window and selecting **Setup…**), it looks the following:



Select a variable, then edit its properties below:

x
y

Legend

Appearance

Color: Blue

Line style: Solid

Marker style: None

Thickness: Single

Vertical axis: Left

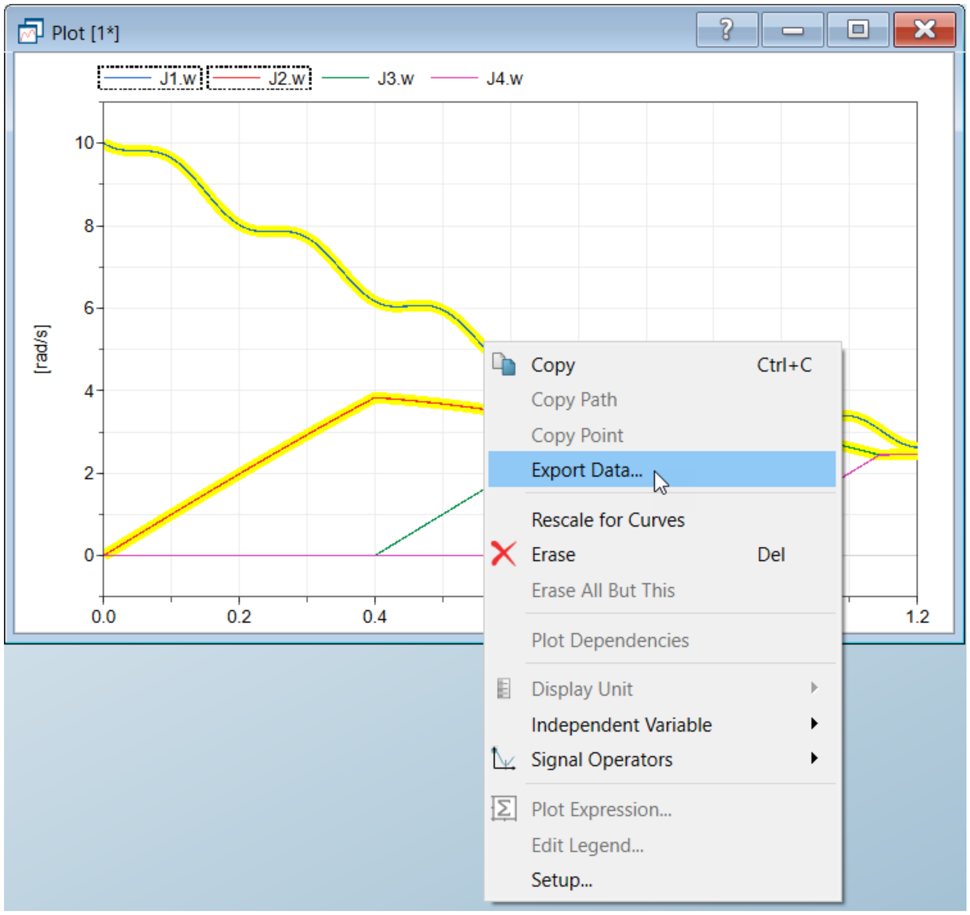
Other properties

Note that the option cannot be handled from this dialog.

For scripting, see section “Not displaying the legend for a curve variable (scripting)” on page 24.

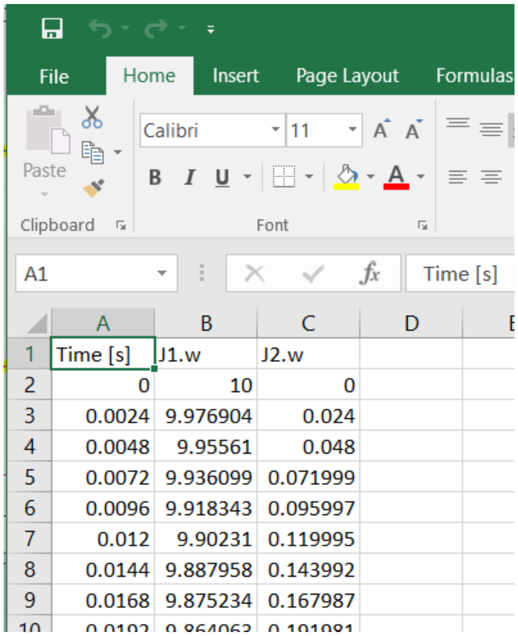
Exporting curve values to CSV or SDF format

You can select one or several curves in a plot and export the curve values to CSV or SDF by right-clicking a (selected) curve and selecting **Export Data…**:



The command opens a window where you can select the name of the file, the location, and if it should be saved in CSV of SDF format.

As an example, exporting the above selection to a CSV file and opening that file gives:



A1 | Time [s]

	A	B	C	D
1	Time [s]	J1.w	J2.w	
2	0	10	0	
3	0.0024	9.976904	0.024	
4	0.0048	9.95561	0.048	
5	0.0072	9.936099	0.071999	
6	0.0096	9.918343	0.095997	
7	0.012	9.90231	0.119995	
8	0.0144	9.887958	0.143992	
9	0.0168	9.875234	0.167987	
10	0.0192	9.864063	0.191981	

Note that the export command is also available from the context menu of the curve legends.

3.3.2 Scripting

Displaying library dependencies for a certain model

You can display a list of all libraries that must be loaded for a certain model, by the new built-in function `getDependentLibraries`:

```
getDependentLibraries(modelName, dependentModels=false)
```

The string input argument `modelName` is the full path to the model. If the Boolean input argument `dependentModels` is set to true, a list of the models needed to be loaded is displayed instead of the libraries (note that such a list can be long).

An example of a call and the result:

```
getDependentLibraries("MyLibrary.MyPackage.MyModel") =
{"MyLibrary", "Modelica", "ModelicaServices", "Optimization"}
```

The built-in function simulateMultiExtendedModel improved

New input argument

The built-in function `simulateMultiExtendedModel` now has a new Boolean input argument `autoLoad`. The default value is `true`, meaning that the result files are loaded in the plot window (and variables are replotted).

This argument is used together with the present argument `resultFileNames`. The argument `resultFileNames` is used to set specific names on the created simulation result files. If such names are set, and `autoload` is `true`, these result files are loaded and used for plotting. A typical example is parameter sweeping (the built-in function `simulateMultiExtendedModel` is used by the sweeping functionality in the Design library).

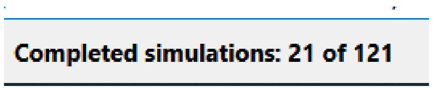
However, if `resultFileNames` has the default value `fill("", 0)`, meaning that the array is empty, nothing is loaded even if `autoload` is `true`. If `autoload` is `false`, nothing is loaded even if simulation results names are specified in `resultFileNames`.

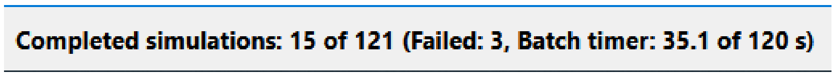
(Note that the argument `autoLoad` is already available in the built-in function `simulateExtendedModel`.)

(A note about naming of simulation result files: If the argument `resultFileNames` contains names, they are used to name the result files. If the argument `resultFileNames` doesn´t contain any names, the argument `resultFile` is used instead to give name to the result files (by default `dsres1`, `dsres2` etc.))

Status messages

Status messages are displayed dynamically in the status bar in the bottom of the Dymola window when the built-in function is used (for example when sweeping). Some examples of messages:


Completed simulations: 21 of 121


Completed simulations: 15 of 121 (Failed: 3, Batch timer: 35.1 of 120 s)

The built-in function simulateMultiResultsModel improved

New input arguments

The built-in function `simulateMultiResultsModel` now has two new input arguments:

- A Boolean input argument `autoLoad`.
- A String vector input argument `resultFileNames`.

For the use of these arguments, see the above section about `simulateMultiExtendedModel`, the use is the same in this built-in function.

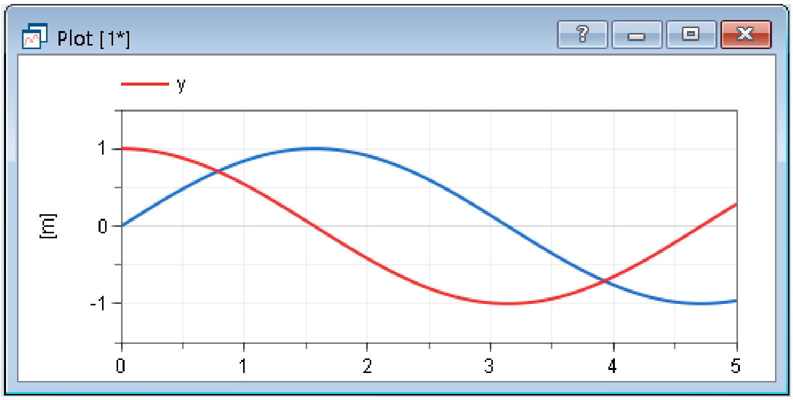
In addition, it is now possible to call the built-in function `simulateMultiResultsModel` with `outputInterval` specified, rather than `numberOfIntervals`. This affects the computation of the size of the output `resultValues`.

Status messages

Status messages are displayed dynamically in the status bar in the bottom of the Dymola window when the built-in function is used (for example when sweeping). For some examples of messages, see the above function.

Not displaying the legend for a curve variable (scripting)

For more about this feature, and the corresponding GUI, see section "Not displaying the legend for a curve variable (GUI)" on page 18. In that section, the following curve was used:



For scripting, removing a variable from the legend is done by setting the legend to exactly one blank character. For example, the above figure (where the legend for the variable x is not displayed) can be the result of

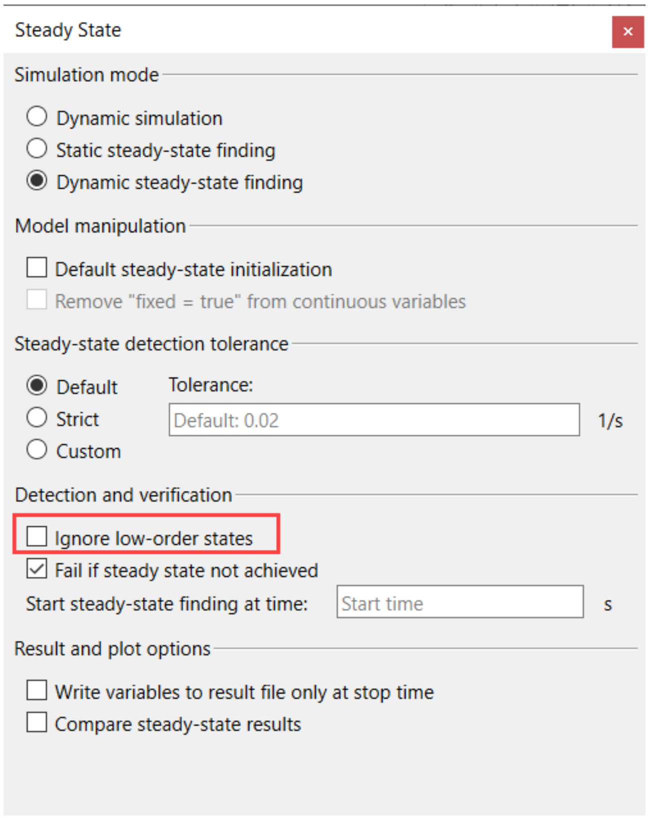
```
plot({"x", "y"}, legends={" ", ""});
```

3.3.3 Minor improvements

Steady state solver interface – Ignoring low-order states for static and dynamic steady-state finding

In some cases, you want to look at the higher-order derivatives when you want to search for steady state. An example is to look at a rotating body. If it rotates at constant speed, the acceleration is zero, that is, you can consider this as a steady state.

To evaluate steady state by looking at the higher-order states, you can now activate the setting **Ignore low-order states** in the steady state solver interface (reached by the command **Simulation > Steady State**). The setting is by default deactivated.


Steady State

Simulation mode

- [ ] Dynamic simulation
- [ ] Static steady-state finding
- [✓] Dynamic steady-state finding

Model manipulation

- [ ] Default steady-state initialization
- [ ] Remove "fixed = true" from continuous variables

Steady-state detection tolerance

- [✓] Default
- [ ] Strict
- [ ] Custom

Tolerance: Default: 0.02 1/s

Detection and verification

- [ ] Ignore low-order states
- [✓] Fail if steady state not achieved

Start steady-state finding at time: Start time s

Result and plot options

- [ ] Write variables to result file only at stop time
- [ ] Compare steady-state results

The setting corresponds to the flag

```
Advanced.Simulation.SteadyStateIgnoreLowOrderStates
```

(The flag is by default false.) The setting can be changed without retranslation. The setting also works for FMU export. The setting is not kept between sessions.

Improved stop cooperation between Dymola and the executable dymosim

When the user stops a simulation by the command **Simulation > Stop** (“the **Stop** button”) Dymola and the executable dymosim performs a stop cooperation to get a controlled termination of the simulator.

This process has now been improved:

- By default, Dymola now waits up to 5 seconds for dymosim to close.
- It is possible to terminate immediately dymosim by clicking the **Stop** button again.
- The above actions are noted in the translation log.
- The simulators now check for a stop request in more places in the code, and more often, at most every 4 seconds, matching the waiting time of Dymola.

Note. For inline integration and code compilation there is no stop cooperation, for these stop happens immediately when pressing the **Stop** button.

When exporting plots, the format is now the same as when exporting ordinary result files

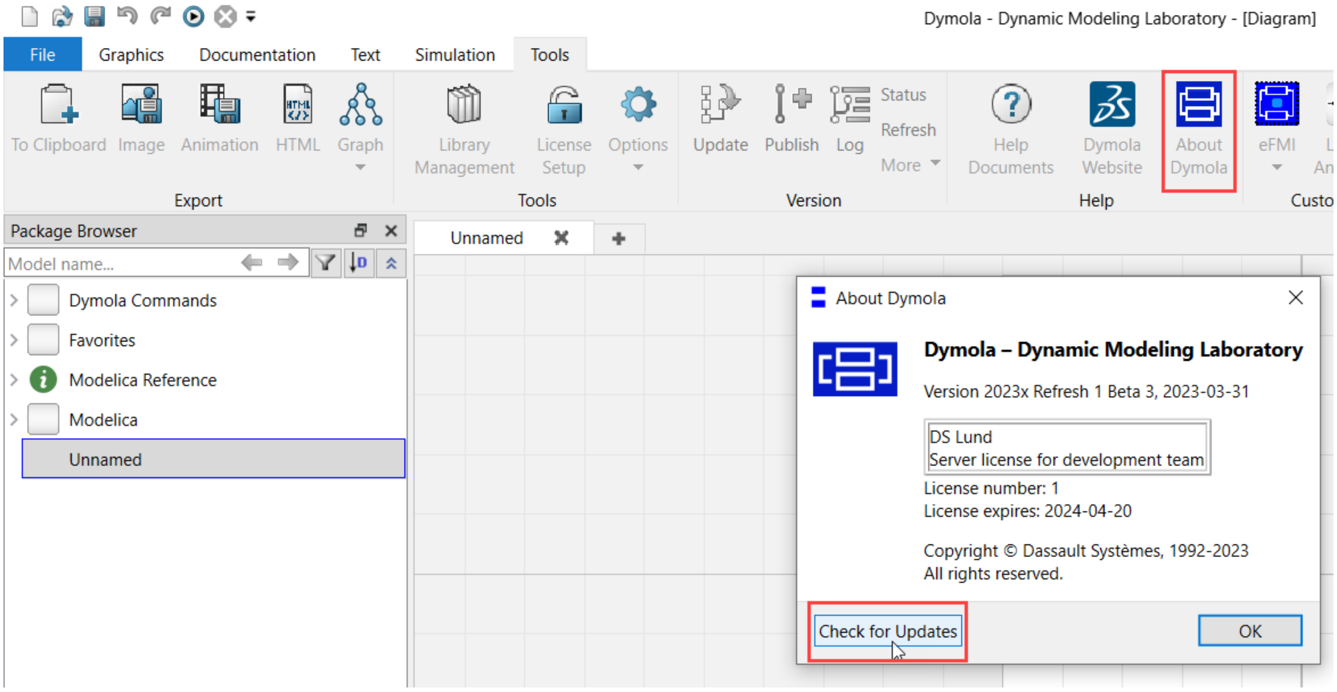
When you export plots by right-clicking the result file in the variable browser and selecting the context command **Export Result > Only Plot Window…** you now export in the same Trajectory 1.1 format as you do when you use the command **Export Result > All…**. This format allows the exported result to include descriptions and units for variables, and allows a unified view of result files.

3.4 Installation

For the current list of hardware and software requirements, please see chapter "Appendix – Installation: Hardware and Software Requirements" starting on page 58.

3.4.1 Checking for updates

A new button available in the information window displayed by the command **Tools > About Dymola** allows you to check if you have the currently released version of Dymola.



Dymola – Dynamic Modeling Laboratory

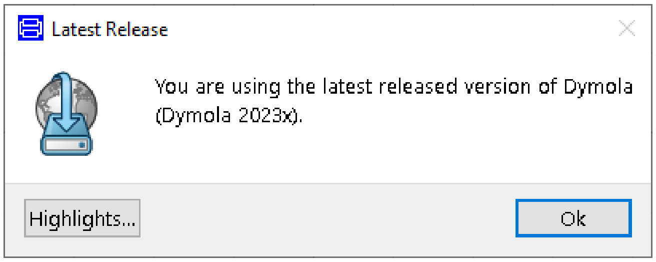
Version 2023x Refresh 1 Beta 3, 2023-03-31

DS Lund
Server license for development team

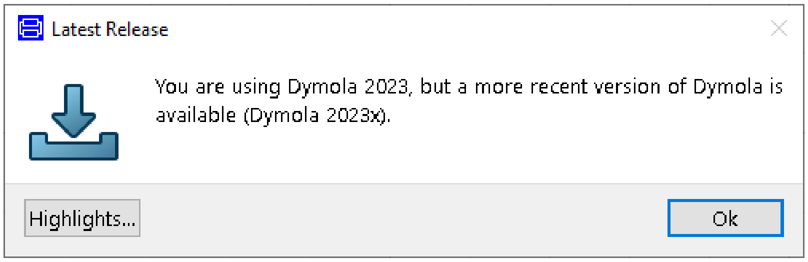
License number: 1
License expires: 2024-04-20

Copyright © Dassault Systèmes, 1992-2023
All rights reserved.

The result of the command could be any of (note that the images are mockups, this command is not available in Dymola 2023x):


Latest Release

You are using the latest released version of Dymola (Dymola 2023x).


Latest Release

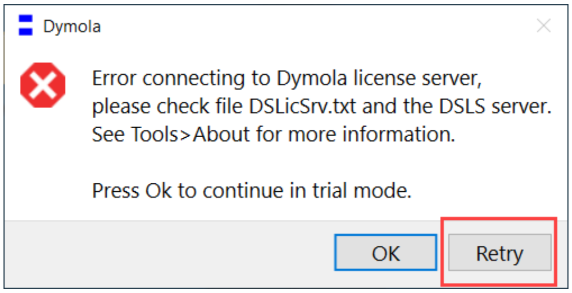
You are using Dymola 2023, but a more recent version of Dymola is available (Dymola 2023x).

Note: In previous Dymola versions, you could get information about the latest Dymola release by using the command **Tools > About Dymola**, and then clicking the button **Latest Release** in the displayed dialog. This button is now available in the dialogs above, but removed to **Highlights…**.

3.4.2 Minor improvements

Possibility to retry if license server does not respond when starting Dymola

The message given if the license server is not found when starting Dymola now also includes a **Retry** button. This gives you the possibility (as an example) to open a VPN tunnel before trying to reach the license server again.

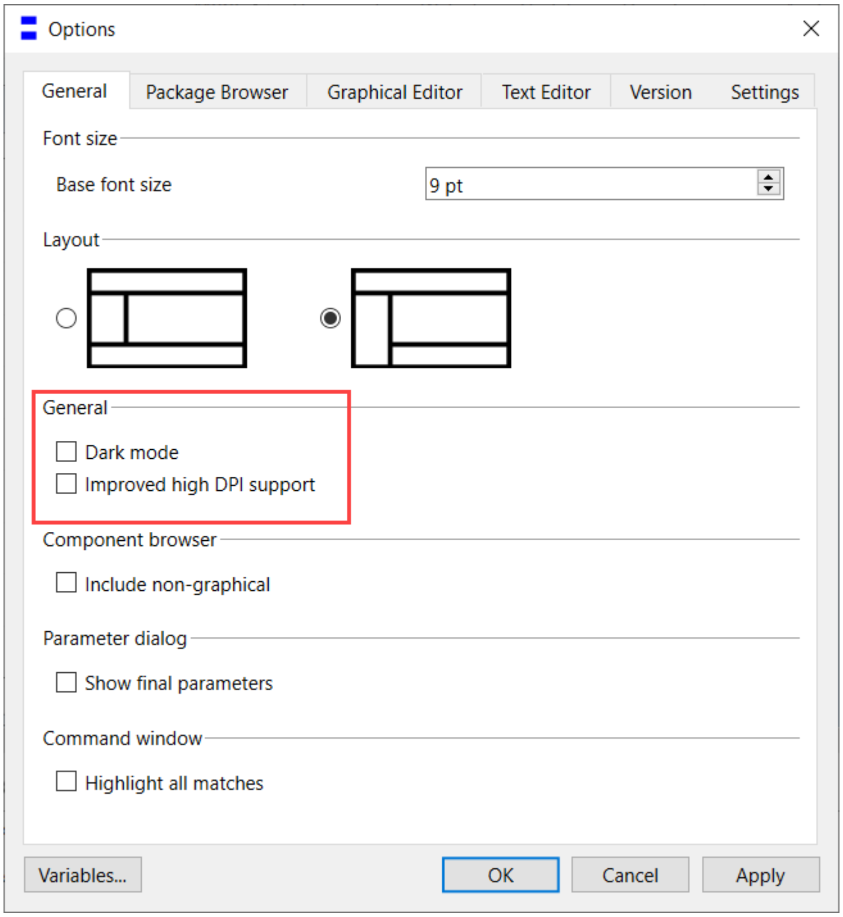

Dymola

Error connecting to Dymola license server, please check file DSLicSrv.txt and the DSLS server. See Tools>About for more information.

Press Ok to continue in trial mode.

Dark mode and improved high DPI support can be activated from GUI

In previous Dymola versions, you could activate Dymola in dark mode, and high DPI support, respectively, by using command line arguments when starting Dymola. Now you can also use new settings in the Options dialog, reached by **Tools > Options**, the **General** tab:



Font size

Base font size: 9 pt

Layout

( ) [✓ ●]

General

☐ Dark mode

☐ Improved high DPI support

Component browser

☐ Include non-graphical

Parameter dialog

☐ Show final parameters

Command window

☐ Highlight all matches

By default, none of these options is activated. Note that you must restart Dymola if changing any of these settings.

The settings are saved between sessions.

3.4.3 Installation on Windows

Discontinued support for Windows 8.1

From this 2023x Refresh 1 version of Dymola, the support of Windows 8.1 is discontinued.

Bundled Visual Studio toolsets not supported

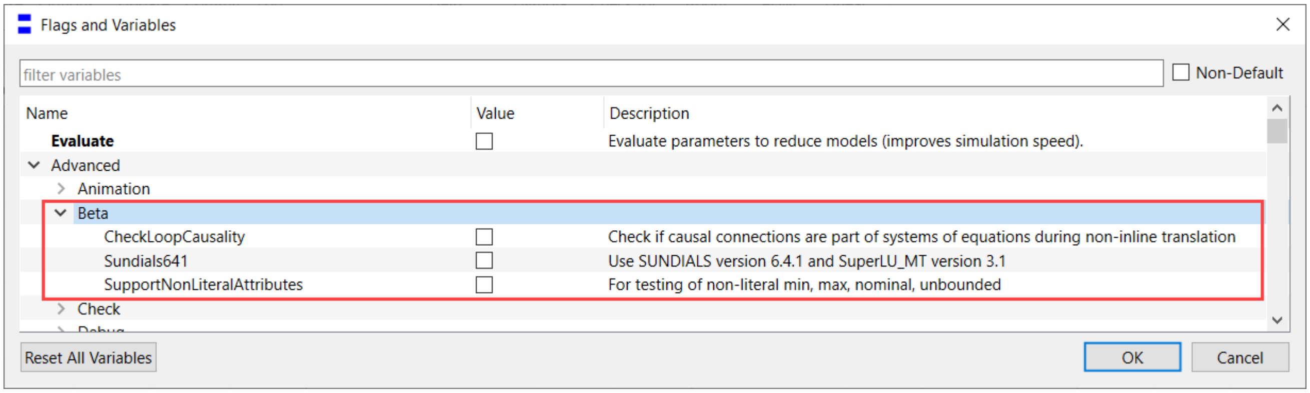
Bundled Visual Studio toolsets are not supported. The workaround is to install the older build tools separately instead of bundled in the Visual Studio 2022 installation.

3.5 Features Under Development

In this section you will find features that are "under development", that is, they are not finalized, nor fully supported and documented, but will be when they are formally released in a later Dymola version. You may see this as a "technology preview".

Note that they are only documented here in the Release Notes until they are finally released, then they are also documented in the manuals.

These features are grouped by `Advanced.Beta` flags in **Tools > Options > Variables…**: An example from Dymola 2023x Refresh 1:



Name	Value	Description
Evaluate	☐	Evaluate parameters to reduce models (improves simulation speed).
Advanced		
Animation		
Beta		
CheckLoopCausality	☐	Check if causal connections are part of systems of equations during non-inline translation
Sundials641	☐	Use SUNDIALS version 6.4.1 and SuperLU_MT version 3.1
SupportNonLiteralAttributes	☐	For testing of non-literal min, max, nominal, unbounded
Check		

The features are by default not activated, to activate any of them, activate the corresponding flag.

When the features have been released, the name of the flag is changed. As an example, a previous beta flag `Advanced.Beta.HighlightMatchingConnectors` is now named `Advanced.Editor.Highlight.MatchingConnectors`, (note the extra period added in this flag, that is not the usual case) and the default value is now true. See "Option to highlight good choices when starting a connection" on page 12.

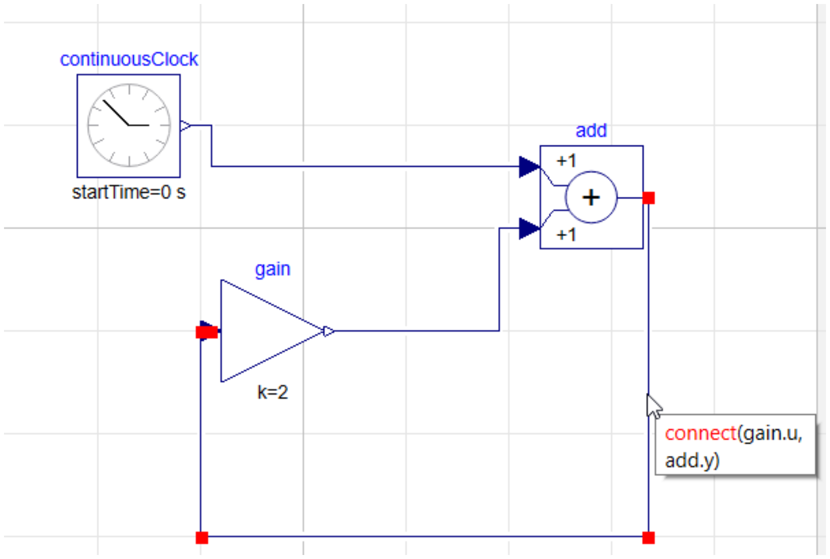
In Dymola 2023x Refresh 1, the following "under development", features are available:

Option to get diagnostics when reversing causality

To use this option, activate the flag `Advanced.Beta.CheckLoopCausality`. We recommend also setting `Evaluate = false` when using this option, otherwise some case might be missed.

An example model might be:

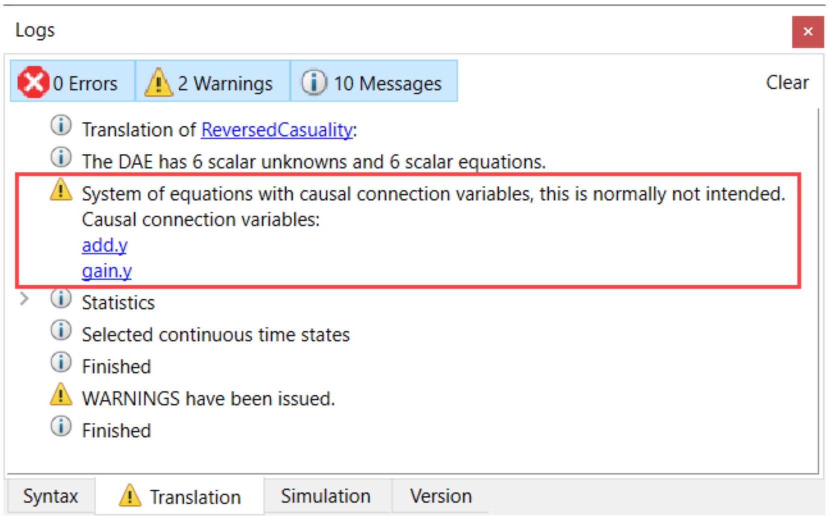
```
model ReversedCasuality
  Modelica.Blocks.Math.Gain gain(k=2)
    annotation (Placement(transformation(extent={{-16,-30},{4,-10}})));
  Modelica.Blocks.Math.Add add
    annotation (Placement(transformation(extent={{46,-4},{66,16}})));
  Modelica.Blocks.Sources.ContinuousClock continuousClock
    annotation (Placement(transformation(extent={{-44,10},{-24,30}})));
equation
  connect(gain.y, add.u2) annotation (Line(points={{5,-20},{38,-20},{38,0},{
          44,0}}, color={0,0,127}));
  connect(add.u1, continuousClock.y) annotation (Line(points={{44,12},{-18,12},
          {-18,20},{-23,20}}, color={0,0,127}));
  connect(gain.u, add.y) annotation (Line(points={{-18,-20},{-20,-20},{-20,
          -60},{67,-60},{67,6}}, color={0,0,127}));
  annotation (Icon(coordinateSystem(preserveAspectRatio=false)), Diagram(
        coordinateSystem(preserveAspectRatio=false)),
    experiment(StopTime=2, __Dymola_Algorithm="Dassl"));
end ReversedCasuality;
```



connect(gain.u, add.y)

Here, the connection between the add block and the gain block constitutes a reversed causality issue.

Activating the flag `Advanced.Beta.CheckLoopCausality`, and setting `Evaluate=false`, and then simulating, you will get, in the translation log:


Logs

0 Errors | 2 Warnings | 10 Messages | Clear

```
Translation of ReversedCasuality:
The DAE has 6 scalar unknowns and 6 scalar equations.
System of equations with causal connection variables, this is normally not intended.
Causal connection variables:
add.y
gain.y
Statistics
Selected continuous time states
Finished
WARNINGS have been issued.
Finished
```

Syntax | Translation | Simulation | Version

Support for the IDA solver from SUNDIALS and upgrade to SUNDIALS 6.4.1 and SuperLU_MT 3.1

Upgrade to SUNDIALS 6.4.1 and SuperLU_MT 3.1

By setting the flag `Advanced.Beta.Sundials641 = true`, the SUNDIALS 6.4.1 and SuperLU_MT 3.1 version will be used.

This applies to the following solvers:

- CVode (SUNDIALS 6.4.1)
- CVode sparse (SUNDIALS 6.4.1 and SuperLU_MT 3.1)
- Radau, Esdirk*, Sdirk34hw sparse (SuperLU_MT 3.1)

The option can be used in the following cases:

- Simulation directly in Dymola using any of the above solver options
- FMU export with "Co-simulation using Dymola solvers" using any of the above solver options

Limitations:

- Requires Visual Studio 2015 or newer

Note that the following are currently not supported:

- Dassl, Lsodar sparse (still only uses old version of SuperLU_MT, even with the flag set)
- FMU export with "Co-simulation using Cvode"
- FMU source code export

Support for the IDA solver from SUNDIALS

The SUNDIALS IDA solver is now available as a part of the SUNDIALS 6.4.1 beta option.

IDA is a modern variable-stepsize, variable-order solver of hybrid DAEs. In the core, it implements a BDF integrator and is therefore most similar to Dassl and CVode.

Just as Dassl, IDA allows integration of DAEs. This is a benefit over SUNDIALS CVode, which is a pure hybrid ODE solver.

Among others, the Dymola IDA implementation supports:

- Sparse linear solvers for large-scale systems, by setting `Advanced.SparseActivate = true`.
- Dymola DAE mode for efficient integration of dynamic models with large algebraic loops, by setting `Advanced.Define.DAEsolver = true`. Just as the other Dymola DAE solvers, IDA DAE mode uses the efficient and accurate event handling as described in Henningsson, Olsson, and Vanfretti: *DAE Solvers for Large-Scale Hybrid Models* (http://dx.doi.org/10.3384/ecp19157491).

To enable the IDA beta option in Dymola 2023x Refresh 1 you need to do the following:

- Set the flag `Advanced.Beta.Sundials641 = true`
- Set the flag `Advanced.Beta.Sundials641IDA = true`
- In Simulation Setup, reached by the command **Simulation > Setup**, the **General** tab, select the algorithm **Cvode – variable order**.

The IDA solver can be used in the following cases:

- Simulation directly in Dymola
- FMU export with "Co-simulation using Dymola solvers"

Limitations:

- Requires the compiler Microsoft Visual Studio 2015 or newer.

Note that the following are currently not supported:

- FMU export with "Co-simulation using Cvode"
- FMU source code export

Down-sampling plotted curves to improve drawing speed

Drawing plots in Dymola can be considerably slowed down if curves have a large number of data points, tens or even hundreds of thousands. From a plotting perspective this is any case meaningless because we will only have a few hundred pixels to paint (horizontally or vertically), which means that every pixel is painted several times.

Where this slowdown becomes critical varies, but for e.g. a Remote desktop connection it can be as low as 10.000. When this happens, Dymola becomes very hard to use because everything is lagging.

The down-sampling has been done with some care, as just picking points at regular intervals might hide important behavior. Instead, points that either overlap completely, or intermediate points of a straight-line segment are deleted. This is not completely undetectable due to certain drawing artifacts when the lines are rendered (pixel rounding, anti-aliasing).

The flag `Advanced.Beta.Plot.DownsampleLimit` has been introduced to set the limit where curves are down-sampled to improve drawing speed. The default is 0, which means down-sampling is off (backward compatible). Note that if you zoom in on a curve, the number of points drawn will eventually fall below this limit and any down-sampling artifacts will thus disappear.

FMI 3: Support for hybrid co-simulation

For this feature, see section "Support of hybrid co-simulation for FMI 3" on page 52.

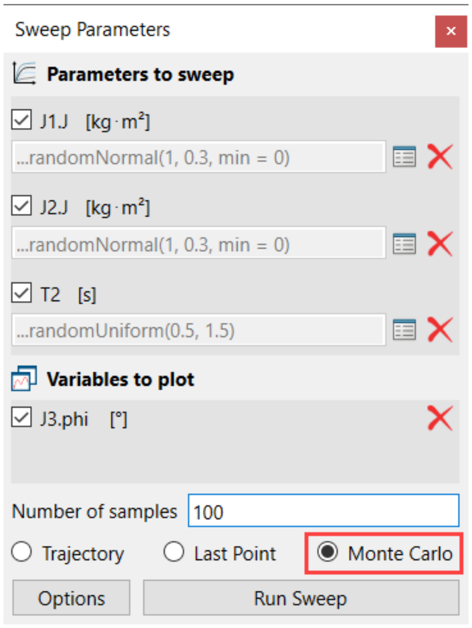
FMI 3: Support for terminals and icons

For this feature, see section "Terminals and Icons" on page 52.

3.6 Model Experimentation

3.6.1 New GUI for Monte Carlo sweeping

There is now a new top-level option in the **Sweep Parameters** dialog: **Monte Carlo**:



Sweep Parameters

Parameters to sweep

☑ J1.J [kg·m^2]
...randomNormal(1, 0.3, min = 0)

☑ J2.J [kg·m^2]
...randomNormal(1, 0.3, min = 0)

☑ T2 [s]
...randomUniform(0.5, 1.5)

Variables to plot

☑ J3.phi [°]

Number of samples: 100

Trajectory   Last Point   [✓ Monte Carlo]

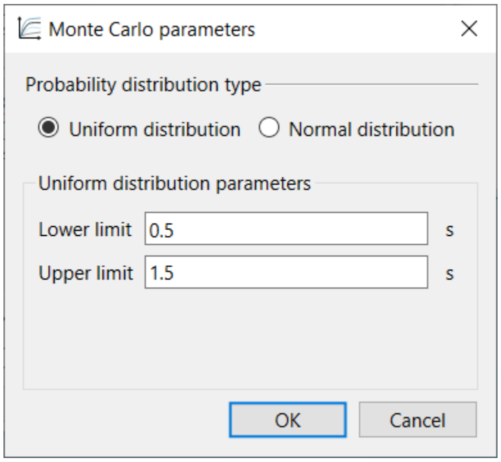
When selecting this option and running the sweep, a Monte Carlo analysis is performed and the result at the stop time is plotted for the selected plot variables. In contrast to the other sweep options, the values of the parameters are not defined by the user before the sweep, but rather they are sampled from user-defined random distributions. Since each parameter value is drawn independently of each other, the set of samples do not generate a regular grid.

Using Monte Carlo is recommended when investigating the effect of many parameters at the same time. A regular grid of values, in many parameters, suffers from the curse of dimensionality, meaning that the number of required simulations grows exponentially. In contrast, Monte Carlo samples the space spanned by the parameters. With well-chosen distributions for the parameters and a large enough number of samples, Monte Carlo analysis can be expected to give a good picture of the effects of each parameter, and the combinations of them.

In the Sweep Parameters widget, two random distributions are available: uniform distribution and normal distribution. (For additional options, please refer to the MonteCarloAnalysis functions in the Design library.)

The distribution options are found in the "Monte Carlo parameters" dialog available for each sweep parameter.

For uniform distribution, you can specify the lower and upper limit of the distribution.


Monte Carlo parameters

Probability distribution type

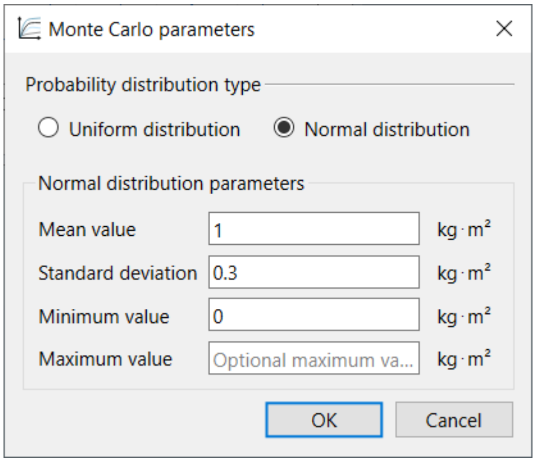
[✓ Uniform distribution] Normal distribution

Uniform distribution parameters

Lower limit 0.5 s

Upper limit 1.5 s

For normal distributions, the options include the mean and the standard deviation. You may optionally supply minimum and maximum values. Any sample outside of this bound will be discarded and a new value will be drawn.


Monte Carlo parameters

Probability distribution type

○ Uniform distribution [✓ ● Normal distribution]

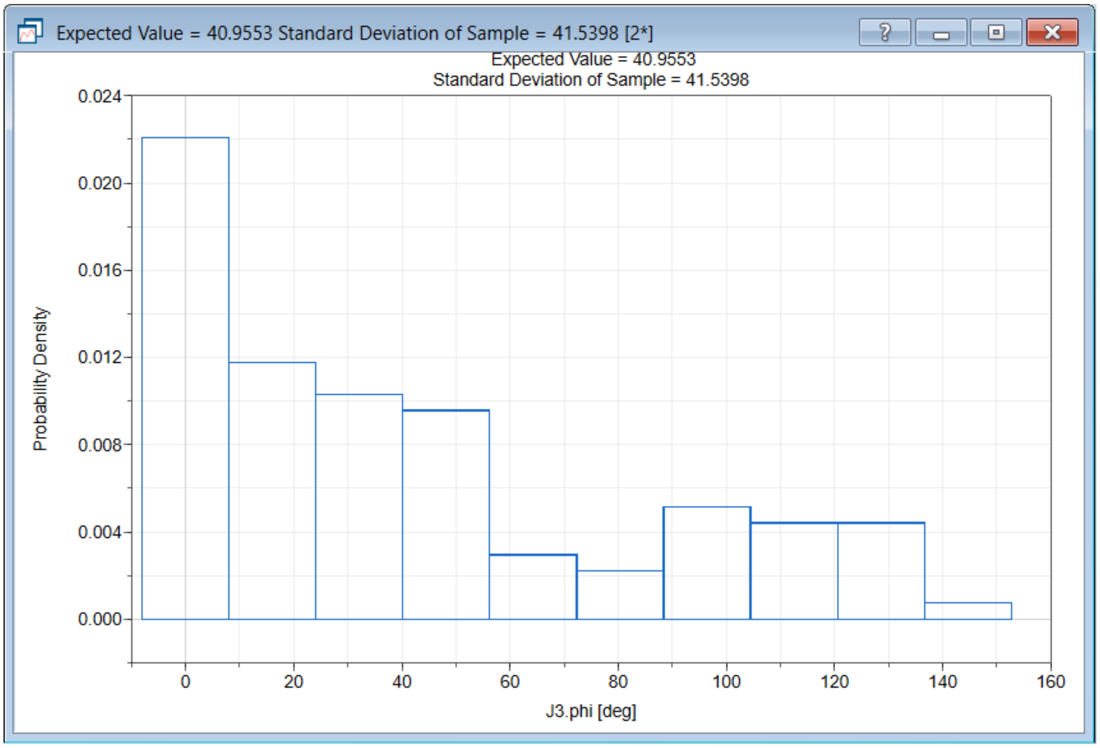
Normal distribution parameters

Mean value	1	kg·m²
Standard deviation	0.3	kg·m²
Minimum value	0	kg·m²
Maximum value	Optional maximum va...	kg·m²

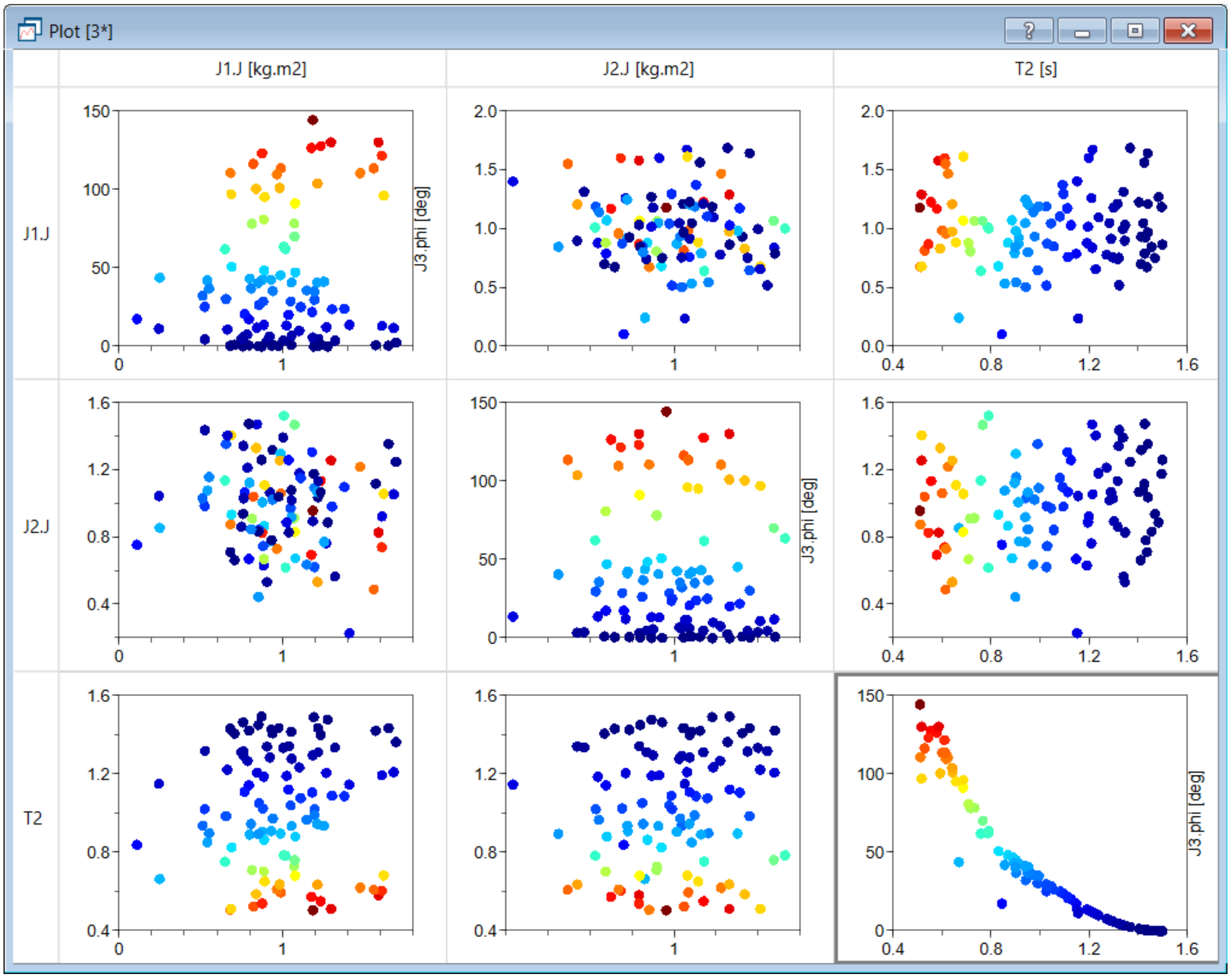
Note that you must edit the parameters by the dialogs above; you cannot edit directly the corresponding field in the **Sweep Parameters** dialog. (This is indicated by the three periods in the field.)

Before running the Monte Carlo analyses, the number of samples must be specified (see the first figure).

The results of an analysis are presented in distribution plots and scatter plots, using the values of the plot variables at the stop time. For each plot variable, its distribution is estimated by a histogram. (The height of each bar depends on the number of results that have a value in the range of the bar. This range of a bar is given by its extent on the x-axis.) Note that the distribution of a plot variable is not only affected by the model, but also by the distributions chosen for the parameters.



For each plot variable, a scatter plot is also generated using the values at the stop time. These show the effect of each parameter on the plot variable (diagonal subplots). In addition, it shows the combined effect of each pair of parameters (off-diagonal subplots). For more on scatter plots see the manual "*Dymola User Manual 2A: Model Development Tools*", chapter "Model Experimentation", section "Sweeping parameters using new GUI", subsection "Sweeping more than two parameters with last point result", but note the difference that, for Monte Carlo, the samples are not randomly perturbed before plotting. This is not required here as the grid is not regular and therefore the samples are not expected to end up on top of each other.

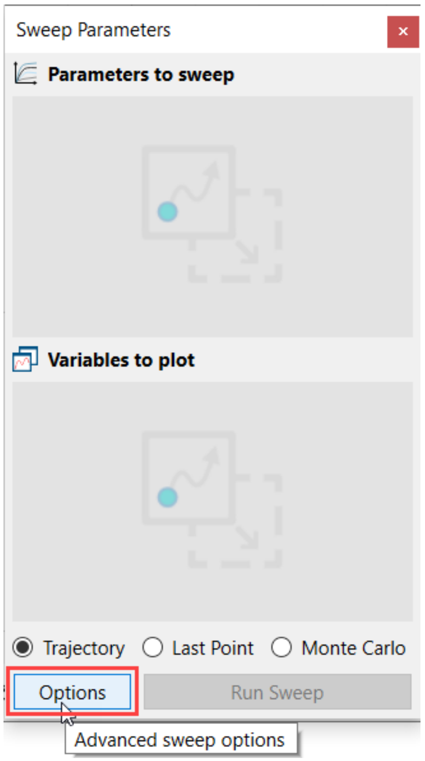


All the advanced sweep options (see next section) are also available for Monte Carlo analysis. Especially, note the option **Fixed seed for Monte Carlo and scatter plots**. When selected a fixed seed will be used for drawing samples, this can e.g. be used for regression testing.

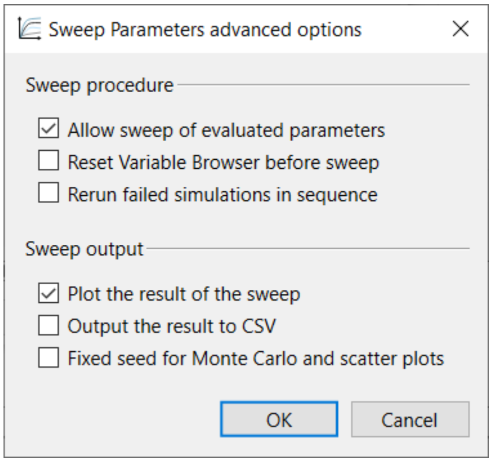
Note that previous setup is remembered, that is, when you switch back and forth to Monte Carlo. As an example, if you have not used **Monte Carlo** before, the previous setup for **Trajectory** or **Last Point** is used to define a randomUniform distribution with the min and max of the values from the previous sweep setup. If you already have done some distribution setup for a variable, that setup is used instead. This goes the other way around as well; whatever you have specified is stored internally and used when switching back and forth to **Monte Carlo**.

3.6.2 Sweep Parameters Advanced Options Dialog

A new button **Options** is available in the sweeping GUI:



Clicking this button, you get (with default settings):



Sweep Parameters advanced options

Sweep procedure

- [✓] Allow sweep of evaluated parameters
- [ ] Reset Variable Browser before sweep
- [ ] Rerun failed simulations in sequence

Sweep output

- [✓] Plot the result of the sweep
- [ ] Output the result to CSV
- [ ] Fixed seed for Monte Carlo and scatter plots

The options are:

Allow sweep of evaluated parameters If this option is activated, also evaluated parameters can be swept. The option was already available by setting the flag

`Advanced.Translation.SmartSimulateExtended = true`. The GUI alternative makes it easier to activate the feature. The option is by default not activated.

Reset Variable Browser before sweep If this option is activated, non-swept parameters that have been modified in the variable browser are reset to default values at sweeping. This option was already available as an argument in the underlying functions; the GUI alternative makes it easier to set this argument. The option is by default not activated.

Rerun failed simulations in sequence If this option is activated, failed simulations are rerun to generate trajectory and log. The option is by default not activated.

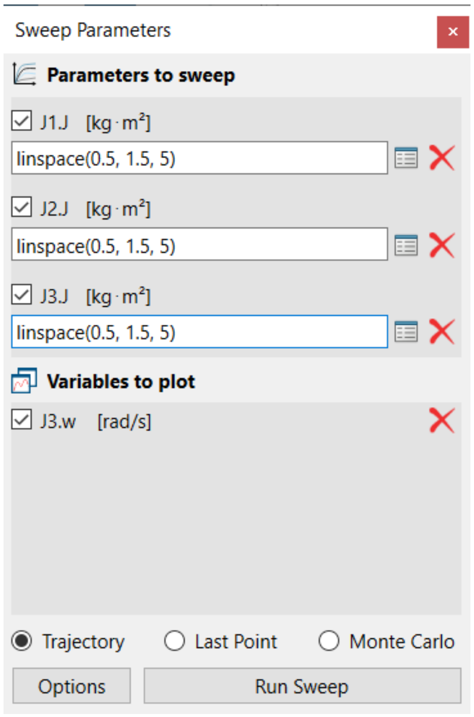
Plot the result of the sweep By default, the result of the sweeping is plotted. If this option is deactivated, no plot is generated; the results are instead written to the command log. Note that displayUnits are not used for the command log output.

Output the result to CSV If this option is activated; the results are also output as CSV files. This option was already available as an argument in some of the underlying functions, but now this argument is available for all sweep functions. By default, this option is not activated.

Fixed seed for Monte Carlo and scatter plots By default, when sweeping with three or more parameters, the resulting scatter plots use random perturbations to avoid dots ending up on top of each other. If this fixed seed option is activated, the seed is set to 1.0 in all cases. This allows for regression checking. By default, this option is not activated.

3.6.3 Sweeping two or more parameters with trajectory result

In previous Dymola versions, you could sweep two or more parameters with last point result (scatter plots). Now you can also sweep any number of parameters with trajectory result. An example:


Sweep Parameters

Parameters to sweep

- [✓] J1.J [kg·m²] — linspace(0.5, 1.5, 5)
- [✓] J2.J [kg·m²] — linspace(0.5, 1.5, 5)
- [✓] J3.J [kg·m²] — linspace(0.5, 1.5, 5)

Variables to plot

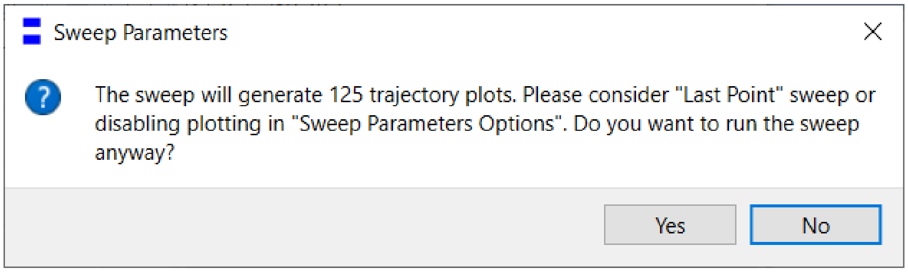
- [✓] J3.w [rad/s]

[✓ Trajectory] Last Point Monte Carlo

If two or more parameters are selected, the new function `Design.Experimentation.sweepManyParametersTrajectory` in the Design package is called.

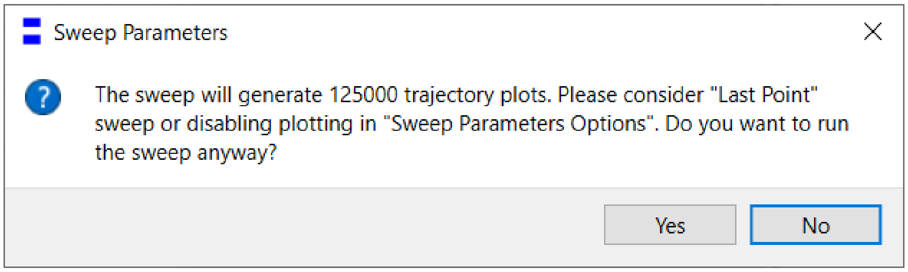
Output to Commands and to CSV is also supported.

Please note that when more than one parameter is selected, the number of result trajectories quickly becomes large. In addition, if each simulation has a large number of output intervals, then the total number of result values may grow unfeasibly large. Warning messages are given when a sweep is expected to give a result that is difficult to interpret or to take a long time to process. As an example, when trying to run the above selection the following warning is given:


Sweep Parameters

The sweep will generate 125 trajectory plots. Please consider "Last Point" sweep or disabling plotting in "Sweep Parameters Options". Do you want to run the sweep anyway?

If you change the number of output points in the interval from 5 to 50 for each swept parameter, you instead get the warning:


Sweep Parameters

The sweep will generate 125000 trajectory plots. Please consider "Last Point" sweep or disabling plotting in "Sweep Parameters Options". Do you want to run the sweep anyway?

3.6.4 Minor improvement: Better presentation of result of sweeping two parameters

When you sweep two parameters, the result is now presented by a separate 3D plot window for each plot variable.

3.6.5 Changes and improvements in the Design.Experimentation package

Unified output order of the sweep functions

The output order of the sweep functions have been unified. For functions with a modified output interface, old versions are kept in `Design.Experimentation.Deprecated`.

- The functions `Design.Experimentation.sweepTwoParametersMulti` and `Design.Experimentation.Analysis.sweepTwoParametersMulti` have been renamed to `Design.Experimentation.sweepTwoParameters` and `Design.Experimentation.Analysis.sweepTwoParameters`, respectively.
- The old `Design.Experimentation.sweepTwoParameters` and `Design.Experimentation.Analysis.sweepTwoParameters` have been removed. The new `sweepTwoParameters` should be used instead. Note that output to CSV is now available for all sweep functions; see below.

New options for the sweep functions

The sweep functions in `Design.Experimentation` have been updated with new options. The new features apply to all the sweep functions in `Design.Experimentation` and `Design.Experimentation.Analysis`:

- Display units are exposed as input options and can also be used for the Analysis functions.
- Option to output sweep results to CSV is now available for all sweep functions.
- Option to rerun failed simulations to generate trajectory and log.
- Option to reset Variable Browser before sweeping.
- Improved plotting of `sweepTwoParameters` results, see “Minor improvement” above.

New function for sweeping two or more parameters with trajectory result

To sweep two or more parameters with trajectory result, you can use the function `Design.Experimentation.sweepManyParametersTrajectory`.

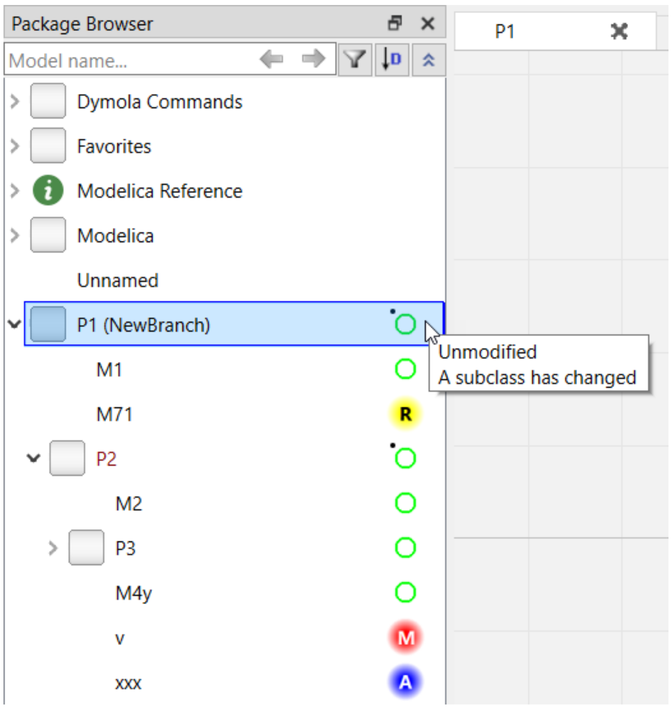
3.7 Model Management

3.7.1 Extended Git support

Git status visible in the package/project browser

To better support version control systems, the current Git status is displayed in the package/project browser.

An example:



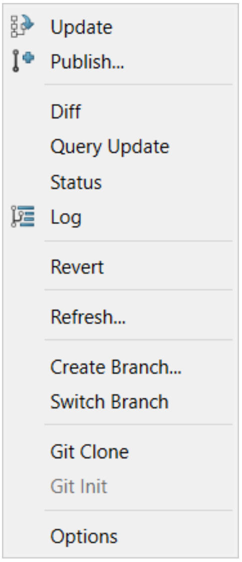
Unmodified
A subclass has changed

The presentation of the status:

Status	Icon	Comments
Unmodified, and all subclasses unmodified	○	
Unmodified, but at least one subclass modified, renamed, or added	•○	
Modified	M	
Added	A	
Renamed	R	
Untracked	▼	
(Deleted)	<no icon>	A deleted file is not displayed in the package browser.

Git versioning commands in Dymola

The Git versioning commands in Dymola can be found in, for example, as the context menu **Version** for a class in the package browser:



The below table lists the Dymola Git versioning commands and what “pure” Git commands that are executed.

Dymola commands	Git commands executed
Update	`git pull -v`

Publish	`git add`
	`git commit`
	
	If a remote branch is missing, the following is executed:
	`git push --set-upstream origin MyBranch`
	Otherwise, the following is executed:
	`git push`
	
	If push fails, everything is set back by executing:
	`git reset --mixed HEAD^1`

Diff	`git diff`

Query Update	`git fetch`
	`git diff --name-status MyBranch..origin/MyBranch`

Status	`git status`
Log	`git log --graph --name-status --date-order`

Revert	If the command is run for the top-level package, the whole package is reverted:
	`git reset --hard`
	Otherwise the current file is restored:
	`git restore`

Refresh	As previously, reloads models. No Git command is executed.

Create Branch	If the user wants to change to the new branch:
	`git switch -c MyBranch`
	Otherwise the branch is just created with:
	`git branch MyBranch`
	
	If a remote repository is present, also the following is executed:
	`git push --set-upstream origin MyBranch`

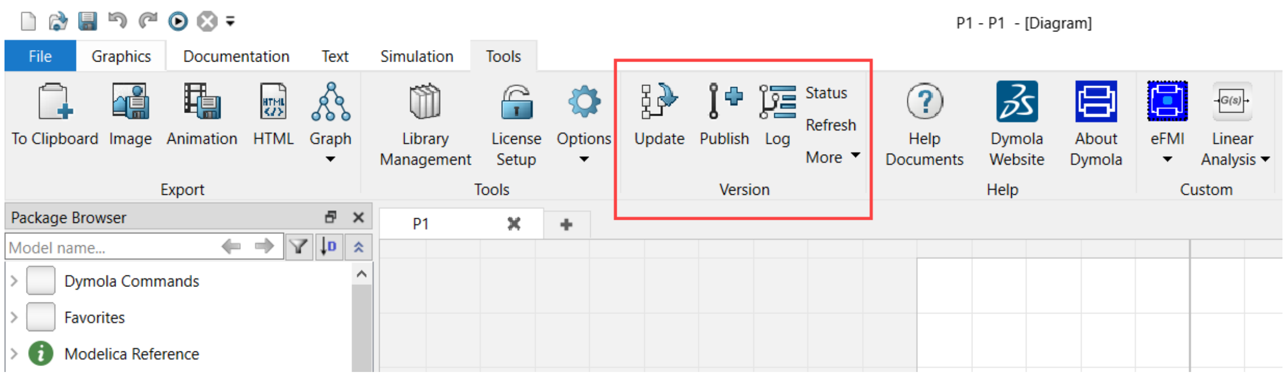
Switch Branch	To change to a local branch:
	`git switch`
	
	To change to a remote branch:
	`git checkout -b MyBranch origin/MyBranch`

Git Clone	`git clone`
Git Init	`git init`

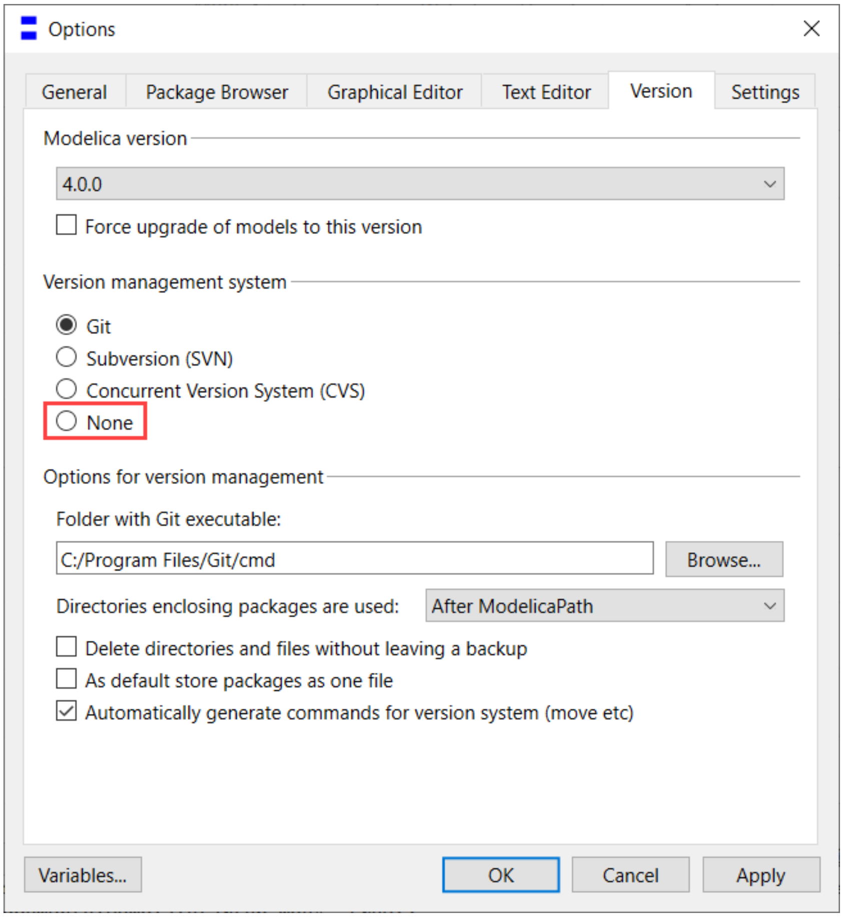
3.7.2 Minor improvements

Expanded Version section in the Tools ribbon

The Version section in the **Tools** ribbon is expanded, that is, the commands from the previous **Tools > Version** command list are available in the section, some as separate buttons, and some in the **More** list:



The content of the **More** command list depends on what version management system is selected in **More > Options**:



Modelica version

4.0.0

☐ Force upgrade of models to this version

Version management system

● Git
○ Subversion (SVN)
○ Concurrent Version System (CVS)
○ None

Options for version management

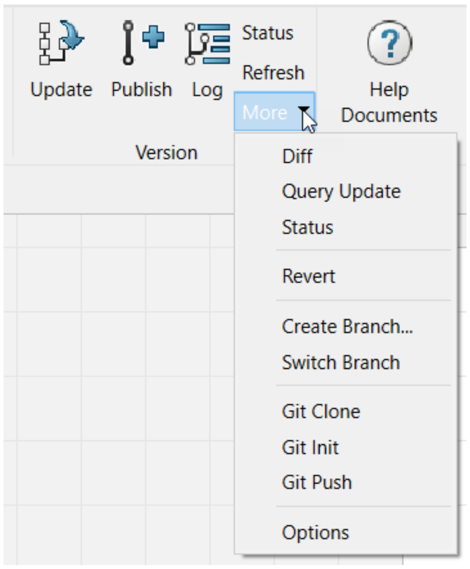
Folder with Git executable: C:/Program Files/Git/cmd

Directories enclosing packages are used: After ModelicaPath

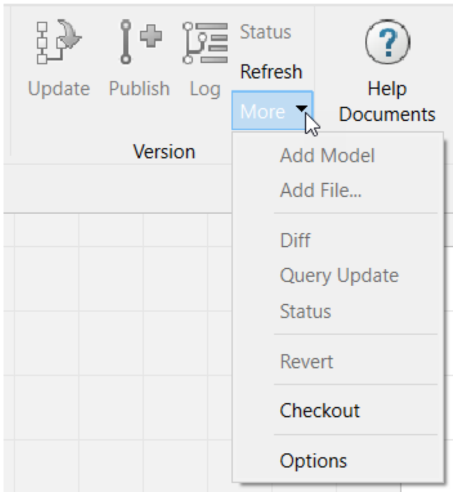
☐ Delete directories and files without leaving a backup
☐ As default store packages as one file
☑ Automatically generate commands for version system (move etc)

(Note that this is a shortcut to the command **Tools > Options**, the **Version** tab. Note also the new option **None**, framed above.)

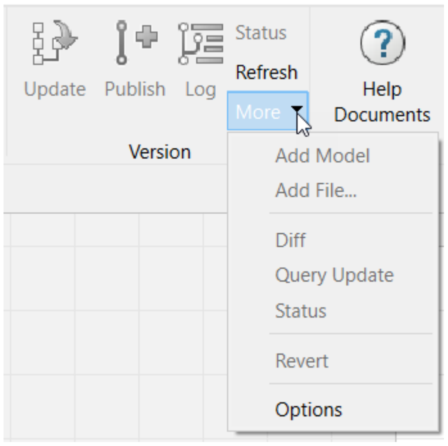
If **Git** is selected:



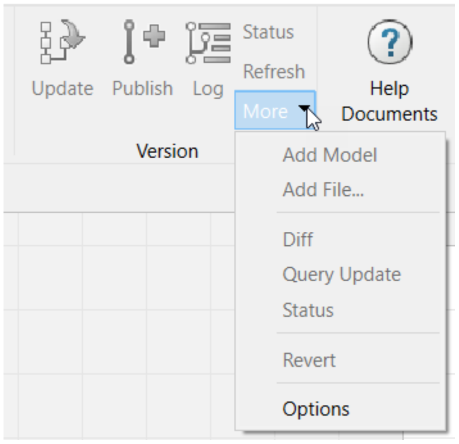
If **Subversion (SVN)** is selected:



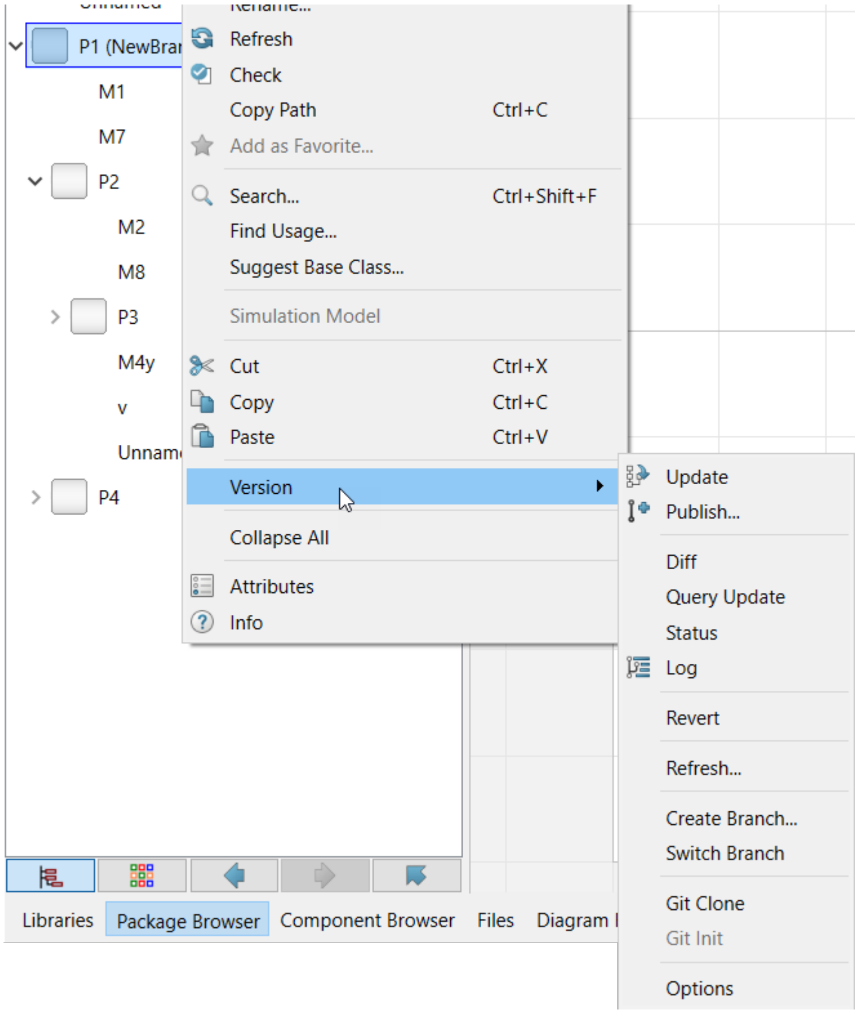
If **Concurrent Version System (CVS)** is selected:



If **None** is selected, only the **Options** command is selectable:



The Versioning commands are also available from the context menu of the class in the package browser/project browser, below an example using Git:



3.8 Other Simulation Environments

3.8.1 Dymola – Matlab interface

Compatibility

The Dymola – Simulink interface now supports Matlab releases from R2018a (ver. 9.4) up to R2022b (ver. 9.13). On Windows, only Visual Studio C++ compilers are supported to generate the DymolaBlock S-function. On Linux, the gcc compiler is supported. The LCC compiler is not supported, neither on Windows nor on Linux.

3.8.2 Real-time simulation

Compatibility – dSPACE

Dymola 2023x Refresh 1 officially supports the DS1006, MicroLabBox, and SCALEXIO systems for HIL applications. For these systems, Dymola 2023x Refresh 1 generated code has been verified for compatibility with the following combinations of dSPACE and Matlab releases:

- dSPACE Release 2018-A with Matlab R2018a
- dSPACE Release 2018-B with Matlab R2018b
- dSPACE Release 2019-A with Matlab R2019a
- dSPACE Release 2019-B with Matlab R2019b
- dSPACE Release 2020-A with Matlab R2020a
- dSPACE Release 2020-B with Matlab R2020b
- dSPACE Release 2021-A with Matlab R2021a
- dSPACE Release 2021-B with Matlab R2021b
- dSPACE Release 2022-A with Matlab R2021b and R2022a
- dSPACE Release 2022-B with Matlab R2021b, R2022a, and R2022b

The selection of supported dSPACE releases focuses on releases that introduce support for a new Matlab release and dSPACE releases that introduce a new version of a cross-compiler tool. In addition, Dymola always support the three latest dSPACE releases with the three latest Matlab releases. Although not officially supported, it is likely that other combinations should work as well.

New utility functions – dym_rti_build2 and dym_rtimp_build2

Dymola 2021 introduced a new function, `dym_rti_build2`, which replaces `dym_rti_build` for building dSPACE applications from models containing DymolaBlocks. The new function uses the new dSPACE RTI function `rti_build2` instead of the old function `rti_build`.

A corresponding new multi-processor build function, `dym_rtimp_build2`, is also introduced.

These functions are supported with dSPACE Release 2019-B and later.

Note on dym_rti_build and dSPACE Release 2017-A and later

The function `rti_usrtrcmerge` is no longer available in dSPACE Release 2017-A and later. Therefore, it is required to run the standard `rti_build` function (with the ‘CM’ command) after `dym_rti_build` to get your _usr.trc content added to the main .trc file. For example:

```
>> dym_rti_build('myModel', 'CM')
>> rti_build('myModel', 'Command', 'CM')
```

Note that this note applies the new functions `dym_rti_build2` and `rti_build2` as well.

Compatibility – Simulink Real-Time

Compatibility with Simulink Real-Time has been verified for all Matlab releases that are supported by the Dymola – Simulink interface, which means R2018a (Simulink Real-Time ver. 6.8) to R2022b (Simulink Real-Time ver. 8.1). Only Microsoft Visual C compilers have been tested.

3.8.3 Java, Python, and JavaScript Interface for Dymola

New or improved built-in functions available

A number of new and improved built-in functions are available in the interfaces.

For more information, see the corresponding sections in "Scripting" starting on page 22.

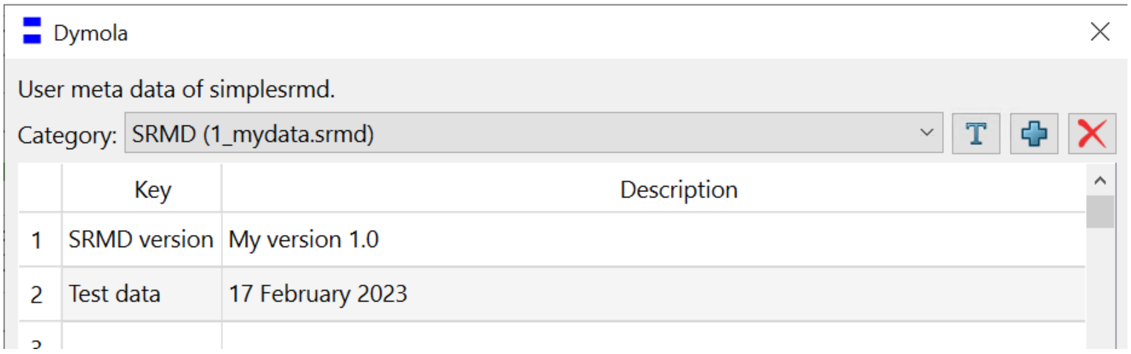
3.8.4 SSP Support

SSP support for SRMD

Simulation Resource Meta Data (SRMD) was developed in the SET Level project to provide a common framework for specifying metadata, which can be used as the foundation for a credible simulation process. More recently, SRMD has been endorsed by System Structure and Parameterization (SSP).

Dymola supports reading and writing SSP files with embedded SRMD metadata. The data on file is converted to metadata annotations in the Modelica model, and can be edited as needed (by the Dymola command **Documentation > Meta Data**). It should be noted that SRMD is not the only metadata definition supported by Dymola; however, SRMD metadata is identified as such in the metadata category (or heading in the documentation viewer).

To create new SRMD definitions that can saved in the Modelica model or exported to an SSP file, the metadata category should be identified as SRMD. This is done by making the SRMD filename part of the category.



User meta data of simplesrmd.

Category: SRMD (1_mydata.srmd)

	Key	Description
1	SRMD version	My version 1.0
2	Test data	17 February 2023

The part enclosed in parenthesis (1_mydata.srmd) is treated as the SRMD filename, which is stored in the SSP file. If multiple sections of metadata are defined, starting each filename with an ordinal number ensures repeatable order of the metadata.

3.8.5 FMI Support in Dymola

Unless otherwise stated, features are available for FMI version 1.0, 2.0, and 3.0.

Support for FMI 3.0

General

Dymola 2023x Refresh 1 includes limited support for FMI version 3.0, with support for the new features described in the below sections. Note that all existing import and export features in Dymola are implemented in FMI 3.0.

Note!

The declaration order of alias variables in FMI 3.0 will not match the order of the original variables, due to a restriction in the specification. Thus, multibody animations of imported FMUs will not work.

For general information about FMI 3.0, see:

- The FMI 3.0 specification
- A paper describing the new features of FMI 3.0

These links are also available using **Tools > Help Documents**.

FMI 3.0 array handling

While earlier FMI versions transformed a Modelica array into a series of scalar values, FMI 3.0 allows keeping the array as one variable.

This feature is supported in Dymola 2023x Refresh 1 by the flag

```
Advanced.FMI3.MinimumArraySize
```

The flag controls at what size Dymola start to use arrays. The default value is -1, which means that arrays are exported as scalar variables, as in FMI 2. This enables FMUs to be used by tools that do not support the new FMI 3.0 arrays.

The FMI 3.0 standard also allows for dynamically resized arrays. This is currently not supported by Dymola.

Terminals and Icons

Note! *This feature is in the beta-stage and performance and reliability will later be improved. As beta feature, it is not documented in the manual.*

FMI 3.0 introduces a new file to support terminals and icons. Dymola 2023x Refresh 1 can use this to store input and output variables. You can export terminals and icons in FMUs by setting the flag `Advanced.Beta.FMI3.ExportTerminalsAndIcons = true`. (The flag is by default `false`.)

Support of hybrid co-simulation for FMI 3

Note! *This feature is in the beta-stage and performance and reliability will later be improved. As beta feature, it is not documented in the manual.*

An *exported* FMI 3 FMU supports early for events.

An *imported* FMI 3 FMU supports hybrid co-simulation if the following is satisfied:

- The flag `Advanced.Beta.FMI3.HybridCoSim` is set to `true`. (The default value of the flag is `false`.)
- The imported FMU indicates that it supports:
    - Returning early for events
    - Getting and setting the states
    - Variable size communication step size.

The imported FMU has some additional hooks, since it is necessary to propagate the potential early return to other parts of the model (including other co-simulation FMUs).

Support for FMI 1.0 will be discontinued in a future release

The support for FMI 1.0 will be discontinued in a future release of Dymola.

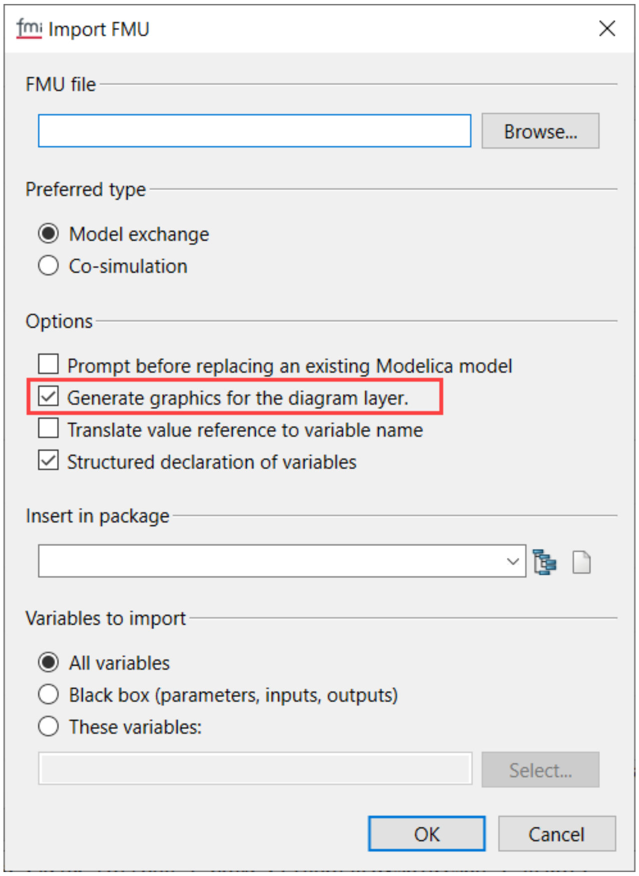
FMI import

Option to show FMU model image also in FMU diagram layer

When you export an FMU, you can select to include a model image. That image is displayed in the icon layer of the imported FMU.

By activating the new option **Generate graphics for the diagram layer**, the model image is also shown in the diagram layer of the imported FMU.

The setting is available both in the dialog you get when importing an FMU (by, for example, the command **File > Open > Import FMU…**), and in the simulation setup, the **FMU Import** tab:



Import FMU

FMU file

Browse...

Preferred type

● Model exchange

○ Co-simulation

Options

☐ Prompt before replacing an existing Modelica model

☑ Generate graphics for the diagram layer.

☐ Translate value reference to variable name

☑ Structured declaration of variables

Insert in package

Variables to import

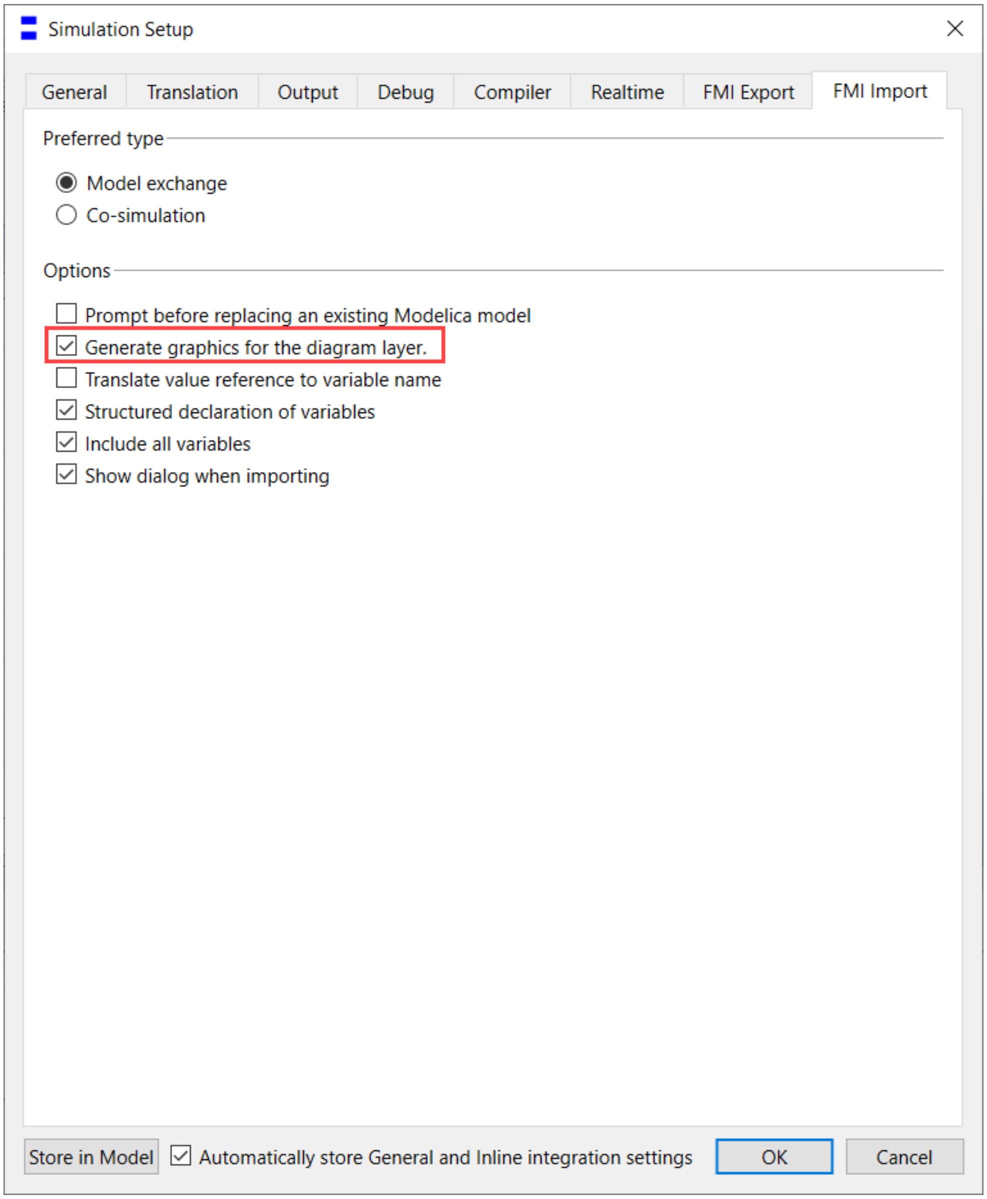
● All variables

○ Black box (parameters, inputs, outputs)

○ These variables:

Select...

OK Cancel



Preferred type

- [✓ Model exchange]
- Co-simulation

Options

- ☐ Prompt before replacing an existing Modelica model
- ☑ Generate graphics for the diagram layer.
- ☐ Translate value reference to variable name
- ☑ Structured declaration of variables
- ☑ Include all variables
- ☑ Show dialog when importing

☑ Automatically store General and Inline integration settings

(The simulation setup is reached by the command **Simulation > Setup**.)

The option is by default not activated. The option is saved between sessions.

The option default value corresponds to the flag `Advanced.FMI.GenerateDiagramGraphics = false`.

Note that if the FMU you import does not contain any model image, the standard FMI icon is used instead. However, the use of this icon is not affected by this new option, this icon is only displayed in the icon layer of the FMU.

3.9 eFMI Support in Dymola

Note that Dymola 2023x Refresh 1 is the last version for which a separate download for CATIA ESP is provided on request (the “eFMI kit supplementary”). CATIA ESP has been released and with the next Dymola version a separate CATIA ESP license on the 3DEXPERIENCE platform will be required; Dymola will then deprecate the “eFMI kit supplementary” and instead provide a seamless interface to CATIA ESP on 3DEXPERIENCE.

The following improvements have been implemented for Dymola 2023x Refresh 1:

- The `DymolaEmbedded` and accompanying libraries are updated to version 1.0.1. Please check the included library documentation and release notes (`DymolaEmbedded.UsersGuide.ReleaseNotes`).
- Generated GALEC programs now preserve the originating model's interface in case of multidimensional inputs, outputs and tunable parameters.
- Added FMI 3 export support for eFMUs with multi-dimensional in-, outputs and tunable parameters; the respective dimensionalities will be preserved by the exported FMU. eFMUs without multi-dimensional in-, outputs or tunable parameters are exported as FMI 2 FMUs as used to.
- Initialization of generated GALEC code is demand-driven; only variables needed to compute sampling steps are initialized (this is particularly important for parameters).
- Generated GALEC code is optimized with respect to pushing parameters towards constantness if they are not tunable nor depend on tunable parameters. The optimization satisfies the GALEC onion-layer of variability and its restrictive “who can be assigned a value when” guarantees, including separate initialization methods for each variability.
- All Modelica functions for eFMI GALEC built-in functions in `.DymolaEmbedded.BuiltinFunctions` are now actually working implementations. The target platform/architecture specific functions are backed by respective CATIA ESP implementations.

3.10 Advanced Modelica Support

3.10.1 Support for Modelica Language Specification 3.6

Dymola 2023x Refresh 1 is compliant with the new Modelica Language Specification 3.6. The most important new features are:

- Selective model extension (see page 9).

- Removing an existing modification of a parameter by using the keyword **break**. (In Dymola you can do this by applying the command **Set to No Value** in the context menu of the parameter dialog entry.)
- Clearer handling of expandable connectors, including hierarchical connections (partially supported in Dymola previously).
- Support of `annotation(mustBeConnected="message")` and `annotation(mayOnlyConnectOnce="message")` instead of rule requiring conditional connectors to be connected.

Examples of other supported features are:

- Can also use `choicesAllMatching` for records
- Removed 0.0 start-value in Real
- Have directorySelector – in addition to load/save

For more information, see https://modelica.org/modelicalanguage.html. The specification is also included in the Dymola distribution, click here.

3.10.2 Minor improvements

Enhanced support for the Modelica attribute unbounded

The Modelica attribute `unbounded` is now supported also for iteration variables of nonlinear systems of equations. It works the same as in the integrators. That is, scaling and error checking only considers the absolute tolerance (based on the nominal value of the variable). This means that the relative tolerance (based on the current value of the variable) is ignored.

3.11 Modelica Standard Library and Modelica Language Specification

The current version of the Modelica Standard Library is version 4.0.0. The current version of the Modelica Language Specification is 3.6.

3.12 Documentation

General

In the software, distribution of Dymola 2023x Refresh 1 Dymola User Manuals of version "March 2023" will be present; these manuals include all relevant features/improvements of Dymola 2023x Refresh 1 presented in the Release Notes.

3.13 Appendix – Installation: Hardware and Software Requirements

Below the current hardware and software requirements for Dymola 2023x Refresh 1 are listed.

3.13.1 Hardware requirements/recommendations

Hardware requirements

- At least 2 GB RAM
- At least 400 MB disc space

Hardware recommendations

At present, it is recommended to have a system with an Intel Core 2 Duo processor or better, with at least 2 MB of L2 cache. Memory speed and cache size are key parameters to achieve maximum simulation performance.

A dual processor will be enough if not using multi-core support; the simulation itself, by default, uses only one execution thread so there is no need for a "quad" processor. If using multi-core support, you might want to use more processors/cores.

Memory size may be significant for translating big models and plotting large result files, but the simulation itself does not require so much memory. Recommended memory size is 6 GB of RAM.

3.13.2 Software requirements

Microsoft Windows

Dymola versions on Windows and Windows operating systems versions

Dymola 2023x Refresh 1 is supported, as 64-bit application, on Windows 10. Since Dymola does not use any features supported only by specific editions of Windows ("Home", "Professional", "Enterprise" etc.), all such editions are supported if the main version is supported.

Compilers

Please note that for the Windows platform, a Microsoft C/C++ compiler, or a GCC compiler, must be installed separately. The following compilers are supported for Dymola 2023x Refresh 1 on Windows:

Microsoft C/C++ compilers, free editions:

Note. When installing any Visual Studio, make sure that the option "C++/CLI support…" is also selected to be installed. (Also note that Bundled Visual Studio toolsets are not supported. The workaround is to install the older build tools separately instead of bundled in e.g. a Visual Studio 2022 installation.)

- Visual Studio 2012 Express Edition (11.0)
- Visual Studio 2015 Express Edition for Windows Desktop (14.0)
- Visual Studio 2017 Desktop Express (15) **Note!** This compiler only supports compiling to Windows 32-bit executables.
- Visual Studio 2017 Community 2017 (15)
- Visual Studio 2017 Build Tools **Notes:**
    - The recommended selection to run Dymola is the workload "Visual C++ build tools" + the option "C++/CLI Support…"
    - Installing this selection, no IDE (Integrated Development Environment) is installed, only command line features
    - This installation is not visible as a specific selection when later selecting the compiler in Dymola, the alternative to select is the same as for any Visual Studio 2017 alternative: **Visual Studio 2017/Visual C++ 2017 Express Edition (15)**.
    - For more information about installing and testing this compiler with Dymola, see www.Dymola.com/compiler.
- Visual Studio 2019 Community (16)
- Visual Studio 2019 Build Tools **Notes:**
    - The recommended selection to run Dymola is the workload "C++ build tools" + the option "C++/CLI Support…"
    - Installing this selection, no IDE (Integrated Development Environment) is installed, only command line features
    - This installation is not visible as a specific selection when later selecting the compiler in Dymola, the alternative to select is the same as for any Visual Studio 2019 alternative: **Visual Studio 2019/Visual C++ 2019 (16)**.
    - For more information about installing and testing this compiler with Dymola, see www.Dymola.com/compiler.
- Visual Studio 2022 Community (17)
- Visual Studio 2022 Build Tools **Notes:**
    - The recommend selection to run Dymola is the workload "Desktop development with C++" + the option "C++/CLI Support…"
    - Installing this selection, no IDE (Integrated Development Environment) is installed, only command line features
    - This installation is not visible as a specific selection when later selecting the compiler in Dymola, the alternative to select is the same as for any Visual Studio 2019 alternative: **Visual Studio 2022/Visual C++ 2022 (17)**.
    - For more information about installing and testing this compiler with Dymola, see www.Dymola.com/compiler.

Microsoft C/C++ compilers, professional editions:

Note. When installing any Visual Studio, make sure that the option "C++/CLI support…" is also selected to be installed. (Also note that Bundled Visual Studio toolsets are not supported. The workaround is to install the older build tools separately instead of bundled in e.g. a Visual Studio 2022 installation.)

- Visual Studio 2012 (11.0)
- Visual Studio 2015 (14.0)
- Visual Studio Professional 2017 (15)
- Visual Studio Enterprise 2017 (15)
- Visual Studio Professional 2019 (16)
- Visual Studio Enterprise 2019 (16)
- Visual Studio Enterprise 2022 (17)
- Visual Studio Professional 2022 (17)

Intel compilers

Note!

Important. The support for Intel compilers are discontinued from the previous Dymola 2022 version.

MinGW GCC compiler

Dymola 2023x Refresh 1 has limited support for the MinGW GCC compiler. The following versions have been tested and are supported:

- For 32-bit GCC: version 6.3 and 8.2
- For 64-bit GCC: version 7.3 and 8.1

Hence, at least the versions in that range should work fine.

To download any of these free compilers, please visit http://www.Dymola.com/compiler where the latest links to downloading the compilers are available. Needed add-ons during installation etc. are also specified here. Note that you need administrator rights to install the compiler.

Also, note that to be able to use other solvers than Lsodar, Dassl, and Euler, you must also add support for C++ when installing the GCC compiler. Usually, you can select this as an add-on when installing GCC.

Current limitations with 32-bit and 64-bit GCC:

- Embedded server (DDE) is not supported.
- Support for external library resources is implemented, but requires that the resources support GCC, which is not always the case.
- FMUs must be exported with the code export option[1] enabled.

[1] Having the code export options means having any of the license features **Dymola Binary Model Export** or the **Dymola Source Code Generation**.

- For 32-bit simulation, parallelization (multi-core) is currently not supported for any of the following algorithms: RadauIIa, Esdirk23a, Esdirk34a, Esdirk45a, and Sdirk34hw.
- Compilation may run out of memory also for models that compile with Visual Studio. The situation is better for 64-bit GCC than for 32-bit GCC.

In general, 64-bit compilation is recommended for MinGW GCC. In addition to the limitations above, it tends to be more numerically robust.

WSL GCC compiler (Linux cross-compiler)

Dymola on window supports cross-compilation for Linux via the use of Windows Subsystem for Linux (WSL) GCC compiler. The default WSL setup is 64-bit only and Dymola adopts this limitation. Notes:

- WSL is usually not enabled on Windows, so you need to enable WSL on your computer and install needed software components.
- You must download and install a suitable Linux distribution, including a C compiler. We recommend Ubuntu 20 since it is the most tested version for Dymola. In particular, the integration algorithms RadauIIa, Esdirk23a, Esdirk34a, Esdirk45a, and Sdirk34hw have been confirmed to work with Ubuntu 20, but not with Ubuntu 18.
    - **Note** however that if you want to exchange FMUs with the Systems Simulation Design app (sometimes referred as "SID"), you should use Ubuntu 18.04 instead.
- The WSL Linux environment can compile the generated model C code from Dymola in order to produce a Linux executable dymosim or a Linux FMU. (To generate Linux FMUs, you must use a specific flag as well.)

Dymola license server

For a Dymola license server on Windows, all files needed to set up and run a Dymola license server on Windows using FLEXnet, except the license file, are available in the Dymola distribution. (This includes also the license daemon, where Dymola presently supports FLEXnet Publisher version 11.16.2.1. This version is part of the Dymola distribution.)

As an alternative to FLEXnet, Dassault Systèmes License Server (DSLS) can be used. Dymola 2023x Refresh 1 supports DSLS R2023x. Earlier DSLS versions cannot be used.

Note that running the Dymola license server on virtual machines is not a supported configuration, although it might work on some platforms.

Linux

Supported Linux versions and compilers

Dymola 2023x Refresh 1 runs on Red Hat Enterprise Linux 8.4, 64-bit, with gcc version 8.5.0, and compatible systems. (For more information about supported platforms, do the following:

- Go to https://doc.qt.io/
- Select the relevant version of Qt, for Dymola 2023x Refresh 1 it is Qt 6.3.
- Select Supported platforms)

Any later version of gcc is typically compatible. In addition to gcc, the model C code generated by Dymola can also be compiled by clang.

You can use a dialog to select compiler, set linker flags, and test the compiler by the **Verify Compiler** button, like in Windows. This is done by the command **Simulation > Setup**, in the **Compiler** tab.

You can however still change the compiler by changing the variable `CC` in `/opt/dymola-<version>-x86-64/bin/dsbuild.sh`. As an example, for a Dymola 2023x Refresh 1 application:

```
/opt/dymola-2023xRefresh1-x86_64/bin/dsbuild.sh
```

Dymola 2023x Refresh 1 is supported as a 64-bit application on Linux.

Notes

- 32-bit compilation for simulation might require explicit installation of 32-bit libc. E.g. on Ubuntu: `sudo apt-get install g++-multilib libc6-dev-i386`
- Dymola is built with Qt 6.3 and thereby inherits the system requirements from Qt. This means:
    - Since Qt 6.3 no longer supports embedding of the XCB libraries, these must now be present on the platform running Dymola. See the table in https://doc.qt.io/qt-6/linux-requirements.html for the list of versions of the ones starting with "libxcb". Note that the development packages ("-dev") mentioned outside the table are not needed.
    - The library `libxcb-xinput.so.0` might require explicit installation.
- For FMU export/import to work, zip/unzip must be installed.

Note on libraries

- The library UserInteraction is not supported on Linux.

Dymola license server

For a Dymola license server on Linux, all files needed to set up and run a Dymola license server on Linux, except the license file, are available in the Dymola distribution. (This also includes the license daemon, where Dymola presently supports FLEXnet Publisher 11.16.2.1.)

As an alternative to FLEXnet, Dassault Systèmes License Server (DSLS) can be used. Dymola 2023x Refresh 1 supports DSLS R2023x. Earlier DSLS versions cannot be used.

Note that running the Dymola license server on virtual machines is not a supported configuration, although it might work on some platforms.

"

'