Dymola

Dynamic Modeling Laboratory

Dymola Release Notes

Dymola 2023x

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:

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 43 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 has limited support for the MinGW GCC compiler, 32-bit and 64-bit. For more information about MinGW GCC, see section “Compilers” on page 43, the section about MinGW GCC compiler.

WSL GCC (Linux cross-compiler)

Dymola 2023x 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 43, 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 46 for more information.

2 About this booklet

This booklet covers Dymola 2023x. 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

3.1 Introduction

3.1.1 Additions and improvements in Dymola

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

- Support for FMI 3.0 (page 39)
- Support for Microsoft Visual Studio 2022 compiler (page 29)
- Separating editable and write-only content (“project” and “library”) (page 9)
- Improved license handling (page 24):
    - Dassault Systèmes License Server (DSLS) format is now default when ordering a new license
    - Nodelocked licenses in the Dassault Systèmes License Server (DSLS) format supported
    - Improved handling of license selection
- Improved handling of resources when copying or renaming classes (page 11)
- Improved error messages (page 11)
- Modifying the selection of iteration variables for non-linear systems of equations (page 16)
- Improvements for plotting
    - Relative size and position of plot windows and table windows (page 17)
    - Improved color-coding of simulation result file tables (page 18)
- ModelManagement included in the Python interface (page 38)
- “Features Under Development” – preview of features to be formally released in a later version (page 34)

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 1.4.0
- Battery Library, version 2.4.1

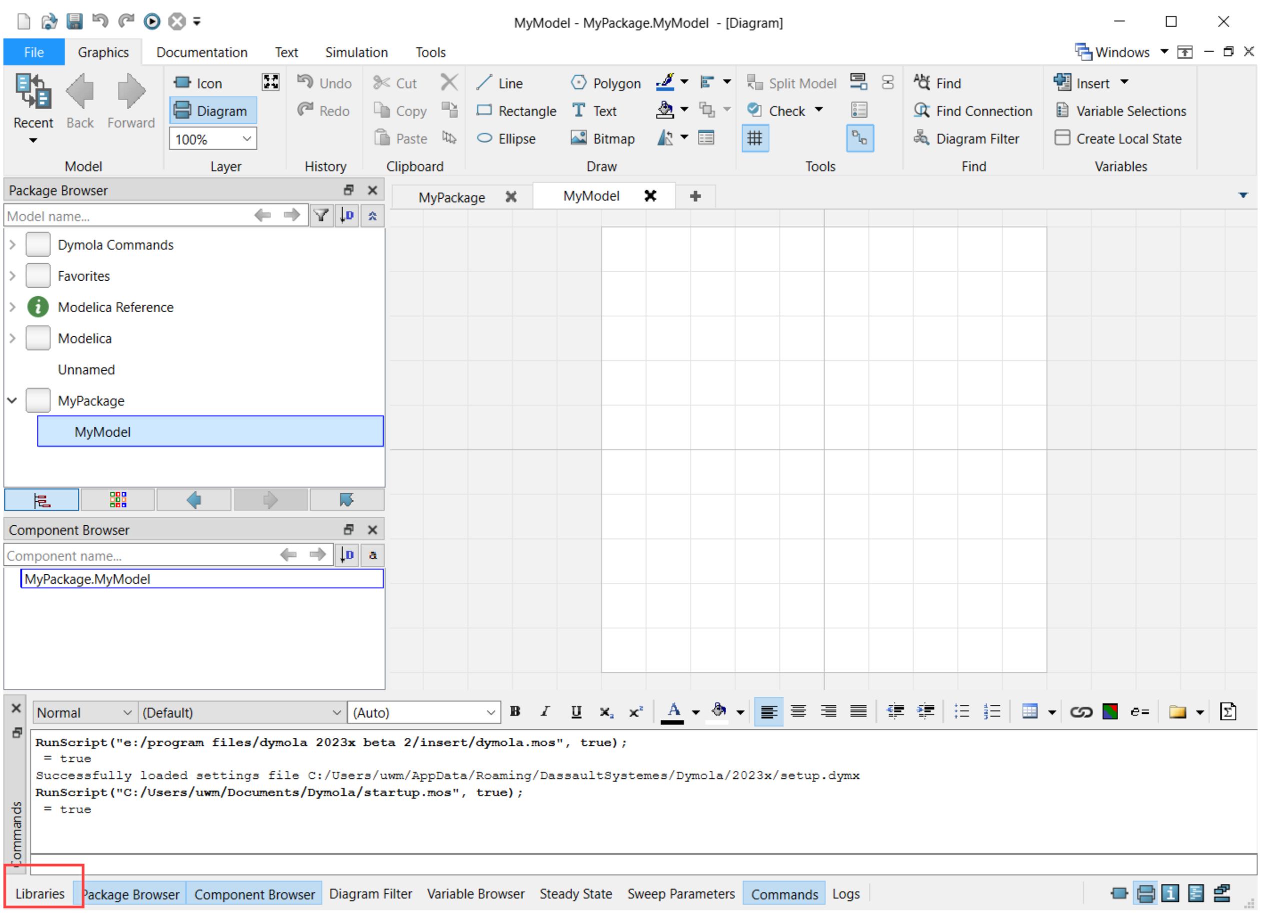
- Brushless DC Drives Library, version 1.2.1
- ClaRa DCS Library, version 1.7.0
- ClaRa Grid Library, version 1.7.0
- ClaRa Plus Library, version 1.7.0
- Claytex Library, version 2022.2
- Claytex Fluid Library, version 2022.2
- Cooling Library, version 1.4.5
- Dassault Systemes Library, version 1.9.0
- Design, version 1.1.4
- Dymola Commands Library, version 1.14
- Dymola Models Library, version 1.5.1
- Electric Power Systems Library, version 1.6.1
- Electrified Powertrains Library (ETPL), version 1.6.0
- Fluid Dynamics Library, version 2.14.0
- Fluid Power Library, version 2022.2
- FTire Interface Library, version 1.1.4
- Human Comfort Library, version 2.14.0
- HVAC (Heating, Ventilation, and Air Conditioning) Library, version 2.14.0
- Hydrogen Library, version 1.3.7
- Modelica_DeviceDrivers Library, version 2.1.0
- Multiflash Media Library, version 1.1.1
- Optimization Library, version 2.2.6
- Pneumatic Systems Library, version 1.5.1
- Smart Electric Drives Library, version 1.5.1
- Testing Library, version 1.6.0
- Thermal Systems Library, version 1.10.0
- Thermal Systems Mobile AC Library, version 1.10.0
- VeSyMA (Vehicle Systems Modeling and Analysis) Library, version 2022.2
- VeSyMA - Engines Library, version 2022.2
- VeSyMA - Powertrain Library, version 2022.2
- VeSyMA - Suspensions Library, version 2022.2
- VeSyMA2ETPL Library, version 2022.2
- Visa2Base, version 1.13
- Visa2Paper, version 1.13
- Visa2Steam, version 1.13

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

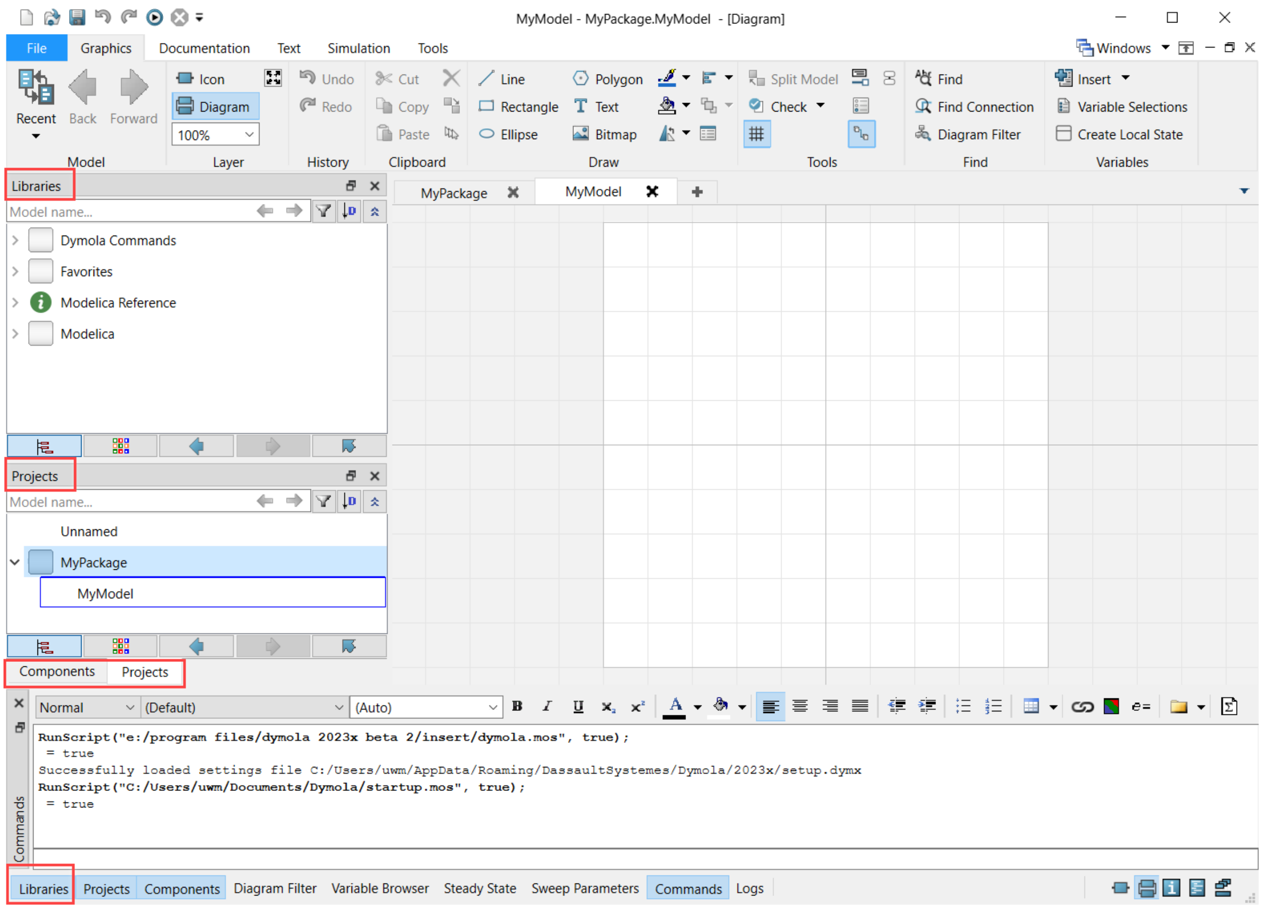
3.2.1 Separating editable and read-only content ("projects" and "libraries")

In Dymola 2023x it is possible to separate editable and read-only content. This is done by activating the new button **Libraries** in the lower left of the Dymola window:



```
RunScript("e:/program files/dymola 2023x beta 2/insert/dymola.mos", true);
 = true
Successfully loaded settings file C:/Users/uwm/AppData/Roaming/DassaultSystemes/Dymola/2023x/setup.dymx
RunScript("C:/Users/uwm/Documents/Dymola/startup.mos", true);
 = true
```

Activating this button changes the above image to:



```
RunScript("e:/program files/dymola 2023x beta 2/insert/dymola.mos", true);
 = true
Successfully loaded settings file C:/Users/uwm/AppData/Roaming/DassaultSystemes/Dymola/2023x/setup.dymx
RunScript("C:/Users/uwm/Documents/Dymola/startup.mos", true);
 = true
```

The Libraries browser contains only read-only packages; the Projects browser contains only editable items. These browsers can be seen a split of the Package browser into one read-only part (“Libraries”) and one editable part (“Projects”).

When in this mode, the Component browser is named Components, and it becomes a tab in the docking space shared with Projects (see the figure above).

The command **Windows > New Dymola Window** adapts to the selected mode; the new Dymola window has the mode selected when giving the command.

The selection of mode is remembered between sessions.

3.2.2 Improved handling of resources

Copying resources

When copying a class by, for example, the command **File > New > Duplicate Class**, the corresponding external resources are by default *not* copied. This corresponds to the new flag `Advanced.Editor.CopyResources = false`.

By setting this flag to `true`, the following external resources are now copied:

- Resources of directories pointed to by the annotations `LibraryDirectory` and `IncludeDirectory`
- Images
- Resources pointed to by the arguments of the built-in functions `Modelica.Utilities.Files.loadResource` or `ModelicaServices.ExternalReferences.loadResource`

Notes:

- For FMU export, handling of external resources is already implemented by the setting **Copy resources to FMU** (and the corresponding flag).
- The ModelManagement library is updated to respect the value of the new flag. (To be precise, the function `ModelManagement.Structure.AST.Misc.CopyClass` is updated.)

Moving resources

When moving a class by, for example, right-clicking the class in the package browser and selecting **Rename…**, the corresponding external resources are by default moved. This corresponds to the new flag `Advanced.Editor.MoveResources = true`. Moving the external resources can be prevented by setting this flag to `false`.

The external resources effected are the same as for copying resources above.

3.2.3 Improved error messages

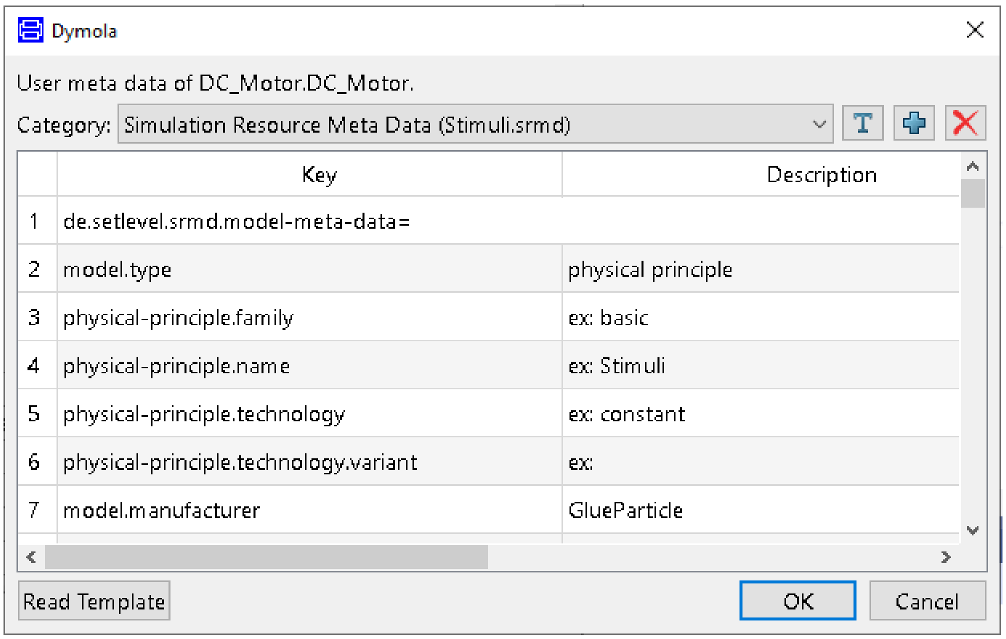
The error messages have been improved:

- More links in error messages
- List of visible connectors rather than the corresponding variables
- Highlighting of faulty components and connections in the diagram layer
- Diagnostics when reversing causality – this is an "under development feature", see "Features Under Development" starting on page 34.

3.2.4 Minor improvements

Support of multiple blocks of model metadata

The editing dialog for metadata has been updated to support multiple selections of metadata. An example:



User meta data of DC_Motor.DC_Motor.

Category: Simulation Resource Meta Data (Stimuli.srmd)

	Key	Description
1	de.setlevel.srmd.model-meta-data=	
2	model.type	physical principle
3	physical-principle.family	ex: basic
4	physical-principle.name	ex: Stimuli
5	physical-principle.technology	ex: constant
6	physical-principle.technology.variant	ex:
7	model.manufacturer	GlueParticle

The new features are:

A combobox to select which section of metadata to display.

An Edit name button. Click the button to enable editing of the section name in the combobox.

A button to add a new section of metadata.

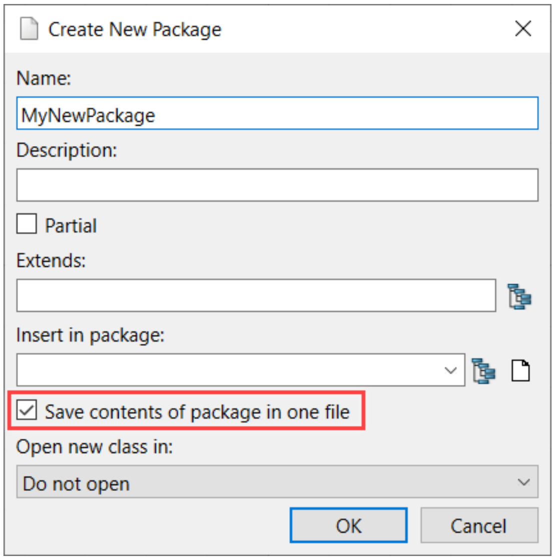
A button to remove the current section of metadata.

Option to highlight good choices when starting a connection

You can highlight good choices when starting a connection – this is an “under development feature”, see “Features Under Development” starting on page 34.

Selection of storing the contents of a new package in one file or as separate files kept between sessions

By default, when creating a new package for the first time, using the command **File > New > Package**, the contents of the package is saved as one file:



Your selection will be the new default, that is, your selection is saved between sessions. This also includes if you change this setting by using the command **File > Save > Save As…** for a package.

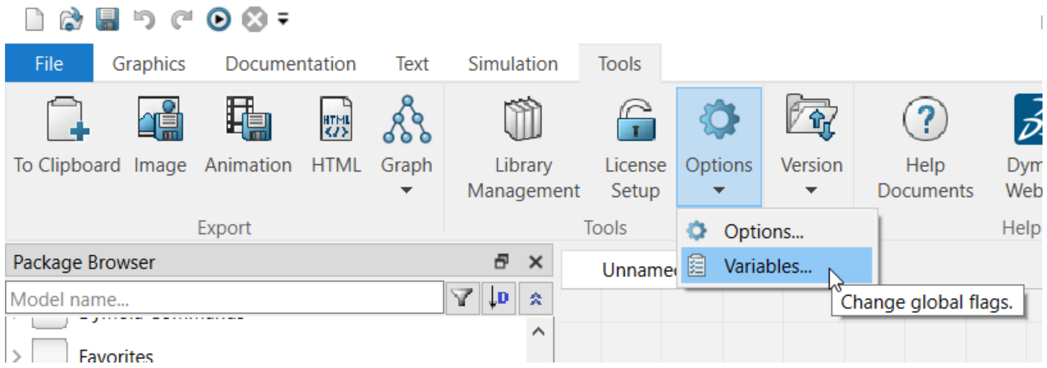
The setting corresponds to the flag `Advanced.File.DefaultStoreAsOneFile`. The flag is by default `false`.

Note that the option of changing how to store a *new* package differs from changing the storage of a *present* package. You can change how to store a present package by using **Attributes > Advanced settings for Hierarchical storage**. This does not change the default setting of how to store a new package.

Easier access to global flags settings

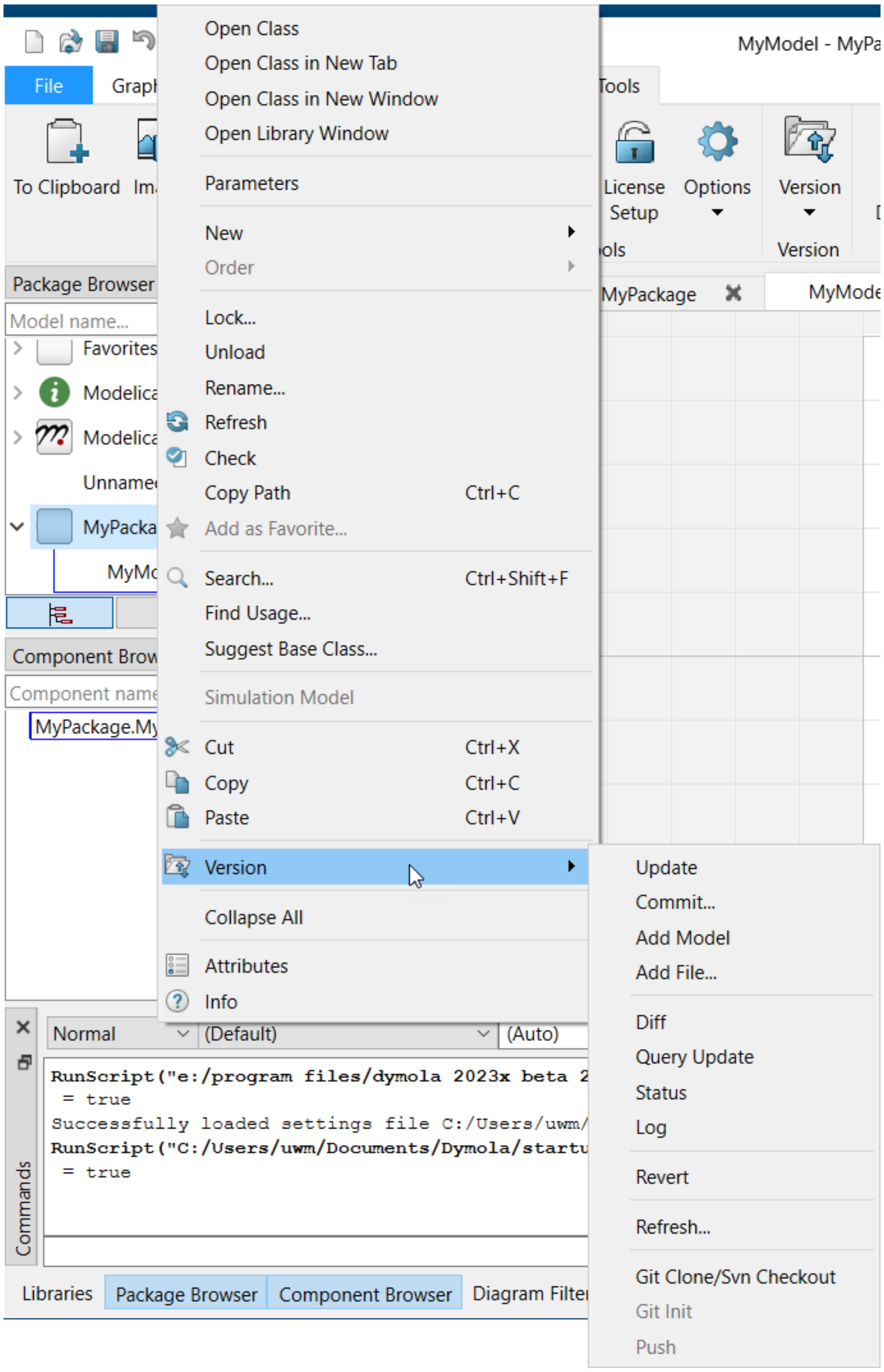
Previously the GUI of global flags settings was only accessible by the command **Tools > Options**, then clicking the **Variables** button.

In Dymola 2023x, the **Options** button has a submenu where you can directly access the GUI of global flags by selecting **Variables…**:



Easier access to versioning commands

Previously the versioning commands were only accessible by the command **Tools > Version**. Now these commands are also available in the context menu for items in the package browser:



```
RunScript("e:/program files/dymola 2023x beta 2
 = true
Successfully loaded settings file C:/Users/uwm/
RunScript("C:/Users/uwm/Documents/Dymola/startu
 = true
```

References in the documentation updated when converting or renaming classes

When renaming a class using a conversion script, or by a rename command, the corresponding references in the documentation are updated. Note that only the references are updated, not any texts.

Minor change of default text font and size

In Dymola 2023x, the default text font is the system default text font. To get the same visual appearance as in earlier Dymola versions, the default font size is now 9 pt.

If you have previous versions of Dymola installed, the default size in Dymola 2023x will still be 9 pt., but if you have changed the default font size in the previous Dymola version, that font size will be used also in Dymola 2023x.

3.3 Simulating a model

3.3.1 Modifying the selection of iteration variables for non-linear systems of equations

When activating the setting **List non-linear iteration variables** in the **Translation** tab of the simulation setup (reached by the command **Simulation > Setup**), Dymola will generate the file `dsmodelIterationSelect.mof`. This file contains the annotation `__Dymola_SimulationIterationVariables`, which contains the iteration variables used by all nonlinear simulation systems.

(These iteration variables are also listed in the translation log, under the header *Statistics > Variables appearing in the nonlinear systems of equations > System simulation.nonlinear > Iteration variables*. Note that the order here is alphabetical, in the file the order is an internal order.)

The annotation `__Dymola_SimulationIterationVariables` can be used at top level in a model to make Dymola prefer the specified variables as iteration variables when tearing. The model `dsmodelIterationSelect` (defined in `dsmodelIterationSelect.mof`) extends the original model with the annotation `__Dymola_SimulationIterationVariables` . Therefore, it can be used as a template.

It is recommended to copy the content of `dsmodelIterationSelect.mof` into a new model and renaming it fitting the original model. If another selection of iteration variables is wanted then the list of variables can be modified.

For example, consider the model Modelica.Electrical.Analog.Examples.SimpleTriacCircuit. When translating this model Dymola creates the nonlinear system *simulation.nonlinear[1]*. The iteration variables are listed in `dsmodelIterationSelect.mof`:

```
    annotation(__Dymola_SimulationIterationVariables={"L.p.v",
    "simpleTriac.thyristor1.vGK", "simpleTriac.idealDiode1.s",
    "simpleTriac.idealDiode.s"});
```

If we wish to use the voltage between gate and cathode in the component thyristor as an iteration variable, we may modify the annotation as in the following model:

```
model SimpleTriacCircuit_IterationVariableSelect
  extends Modelica.Electrical.Analog.Examples.SimpleTriacCircuit;
  annotation(__Dymola_SimulationIterationVariables={"L.p.v",
    "simpleTriac.thyristor.vGK", "simpleTriac.idealDiode1.s",
    "simpleTriac.idealDiode.s"});
end SimpleTriacCircuit_IterationVariableSelect;
```

We should also provide a start guess for the iteration variable simpleTriac.thyristor.vGK. However, in this case the default start guess of 0.0 is the correct solution to the simulation system at start time.

Note that the preferred way to influence Dymola’s choice of iteration variables is to set start guesses for those that you want. Then Dymola will prioritize them. The annotation can be used in cases where there are several good choices to bias Dymola’s selection.

Further, note that Dymola chooses iteration variables that minimizes the size of the torn system. Too heavy usage of the annotation `__Dymola_SimulationIterationVariables` may therefore lead to unnecessarily large systems. Consider, for example, the system of equations:

$$\begin{cases} 0 = y + f(x,t) \\ 0 = g(x,y,t) \end{cases}$$

where x and y are unknowns, t is time, and f and g are nonlinear functions (with no easily determined inverses). Then Dymola will select x as iteration variable as y can easily be computed from x using the first equation. Using the annotation to specify y as an iteration variable will result in a “torn” system with both x and y as iteration variables as x cannot (easily) be computed from y.

3.3.2 Plot tab

Relative size and position of plot windows and table windows

Dymola 2023x supports relative size and position of plot and table windows. This is done by changing the input parameter `position={x0, y0, width, height}` from Integer to Real in the built-in functions `createPlot` and `createTable`.

For any of x0, y0, width and height values, the following now apply:

- **0 < value <= 1**: The value is calculated as a fraction of the available size (main window size minus size of dock areas)
- **value > 1**: The value is an absolute size in pixels (as in previous versions)

Note: When you specify the size in pixels, you specify the inner size of the plot/table window, excluding the frame and title bar. For relative size, you specify the outer size, that is, the whole plot/table window.

Improved color-coding of simulation result file tables

The color-coding of simulation result file tables has been improved in two ways:

- The foreground color adapts to the background color, leading to white text when darker background colors. An example from using the **Heat** color map:

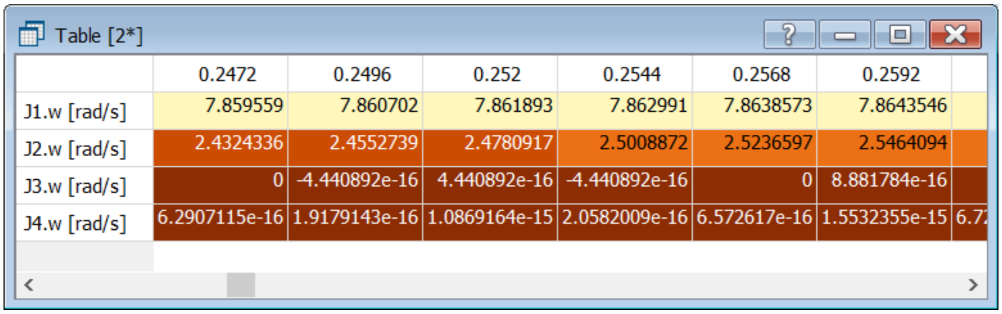


Table [2*]

	0.2472	0.2496	0.252	0.2544	0.2568	0.2592
J1.w [rad/s]	7.859559	7.860702	7.861893	7.862991	7.8638573	7.8643546
J2.w [rad/s]	2.4324336	2.4552739	2.4780917	2.5008872	2.5236597	2.5464094
J3.w [rad/s]	0	-4.440892e-16	4.440892e-16	-4.440892e-16	0	8.881784e-16
J4.w [rad/s]	6.2907115e-16	1.9179143e-16	1.0869164e-15	2.0582009e-16	6.572617e-16	1.5532355e-15

- A **Monochrome** color map has been added. An example:

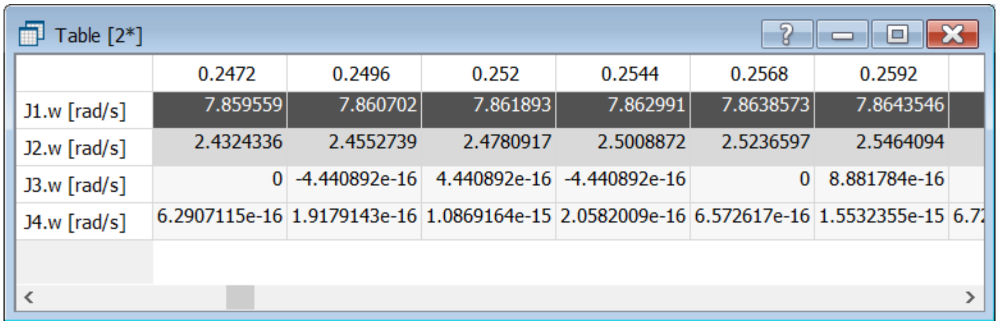


Table [2*]

	0.2472	0.2496	0.252	0.2544	0.2568	0.2592
J1.w [rad/s]	7.859559	7.860702	7.861893	7.862991	7.8638573	7.8643546
J2.w [rad/s]	2.4324336	2.4552739	2.4780917	2.5008872	2.5236597	2.5464094
J3.w [rad/s]	0	-4.440892e-16	4.440892e-16	-4.440892e-16	0	8.881784e-16
J4.w [rad/s]	6.2907115e-16	1.9179143e-16	1.0869164e-15	2.0582009e-16	6.572617e-16	1.5532355e-15

3.3.3 Animation window

Relative size and position of animation windows

Dymola 2023x supports relative size and position of animation windows, in the same way as for plot and table windows. To use this for animation windows, you have to use the built-in function `animationPosition` to create the animation window. This function has an input parameter `positon` that is now handled in the same way as for plot and table windows. See that section above for more information.

3.3.4 Scripting

Option to allow changes of evaluated parameters after translation, including sweeps, with automatic re-translation

By default, evaluated parameters cannot be changed after translation. This can prevent the built-in functions `simulateExtendedModel`, `simulateMultiExtendedModel`, and `simulateMultiResultsModel` to be successful, if evaluated parameters are specified as initial values. The same is the case for sweeping, since it use these underlying functions.

It is possible to handle this by setting these values using modifiers, but it makes the use of these functions more complicated.

By setting the new flag

```
Advanced.Translation.SmartSimulateExtended = true
```

you can change evaluated parameters after translation, including sweeps, but there will be an automatic re-translation of the model to handle these changes. (The flag is by default `false`.)

Below an example of usage, first a failed simulation command due to specifying an evaluated parameter as initial value, then setting the flag, and finally rerunning the simulation command with success. The commands are in bold text. Note the messages given by Dymola.

```
simulateExtendedModel("Modelica.Fluid.Examples.BranchingDynamicPipes", initialNames=
{"pipe2.length"}, initialValues={100});
Warning: Setting pipe2.length has no effect in model.
After translation you can only set literal start-values and non-evaluated parameters.
Setting Advanced.Translation.SmartSimulateExtended=true allows changes of evaluated
parameters including sweeps, but will re-translate the model.
 = false, {}

Advanced.Translation.SmartSimulateExtended=true;

simulateExtendedModel("Modelica.Fluid.Examples.BranchingDynamicPipes", initialNames=
{"pipe2.length"}, initialValues={100});
Automatic re-translation of model triggered by changing structural parameter
pipe2.length and having Advanced.Translation.SmartSimulateExtended=true;
 = true, {}
```

Better handling of gradients in the graphics

Already in Dymola 2023, the built-in function `updateModelicaAnnotations` contained a new Boolean argument `correctGradient`. The argument is by default `true`.

The background is that some classes have icon annotations containing a `fillPattern` that should give a gradient, but they have `lineColor` explicitly equal to `fillColor` that results in a solid fill instead. For such cases, using black as `lineColor` gives a good visual gradient, and this is obtained by default by the new argument `correctGradient` when applying `updateModelicaAnnotations`.

Handling relative size and position for plot, table, and animation windows

The following built-in functions have been improved to handle relative size and position:

- createPlot
- createTable
- animationPosition

For more information about the first two ones, see section “Relative size and position of plot windows and table windows” on page 17.

For the last one, see section “Relative size and position of animation windows” on page 18. Note that this built-in function is now also available in the Dymola Commands library.

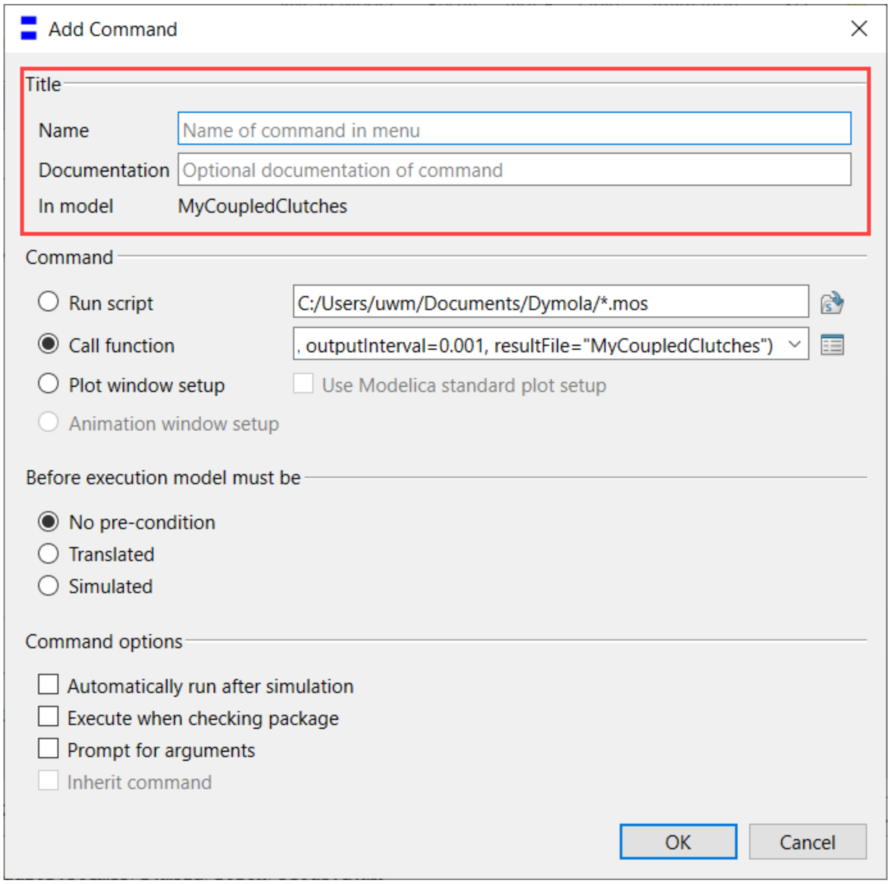
3.3.5 Minor improvements

Improved handling of storing commands in model

When you store a command in the model by the command **Simulation > Commands > Add Command**, the command is now stored in the model that is selected to be the active simulation model. If such a model is not selected, the command is stored in the currently shown model (this was always the case in previous Dymola versions).

(You can select a model as current simulation model by using the context command **Simulation Model** for the model in the package browser or the model tab. If you don´t select any model as active simulation model, the currently displayed model is the active simulation model.)

The menu that appears when you give the command **Simulation > Commands > Add Command** may now looks like:


Add Command

Title

Name: Name of command in menu

Documentation: Optional documentation of command

In model: MyCoupledClutches

Command

( ) Run script: C:/Users/uwm/Documents/Dymola/*.mos

(●) Call function: , outputInterval=0.001, resultFile="MyCoupledClutches")

( ) Plot window setup — ☐ Use Modelica standard plot setup

( ) Animation window setup

Before execution model must be

(●) No pre-condition

( ) Translated

( ) Simulated

Command options

☐ Automatically run after simulation

☐ Execute when checking package

☐ Prompt for arguments

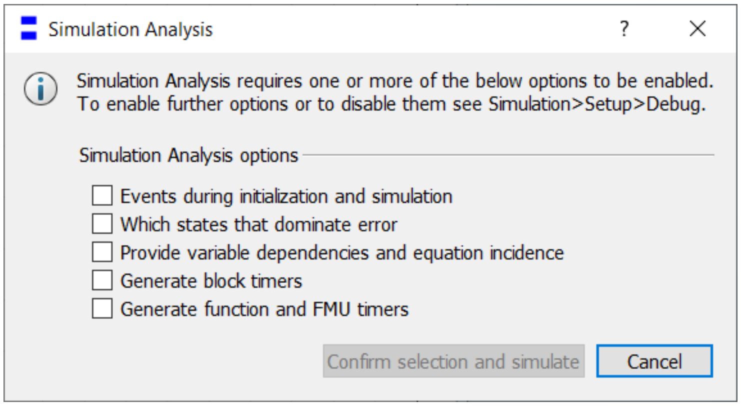
☐ Inherit command

Name is the name of the command; **Documentation** can contain a description of the command, and **In model** indicates in which model the command will be stored.

The command **Simulation > Commands > Organize Commands**, now also work with the current simulation model, like the **Add Command**.

Simplified activation of needed flags for simulation analysis

To use Simulation Analysis, you must set some flags. If you try to apply the command **Simulation > Simulation Analysis** without any of these flags set, you will get the dialog:

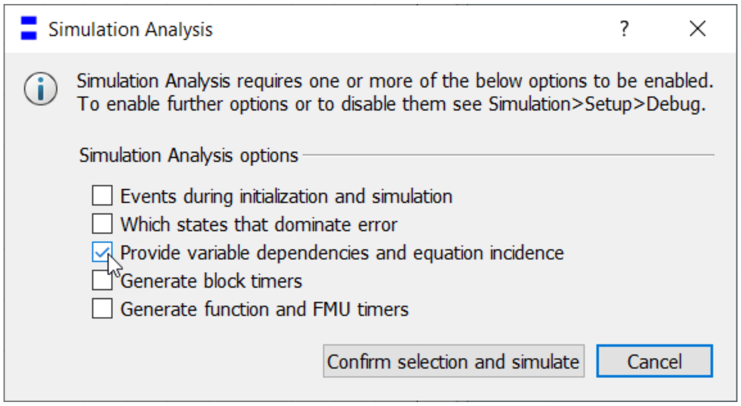

Simulation Analysis

Simulation Analysis requires one or more of the below options to be enabled.
To enable further options or to disable them see Simulation>Setup>Debug.

Simulation Analysis options

- [ ] Events during initialization and simulation
- [ ] Which states that dominate error
- [ ] Provide variable dependencies and equation incidence
- [ ] Generate block timers
- [ ] Generate function and FMU timers

If you activate one or more flags from this dialog, you get:


Simulation Analysis

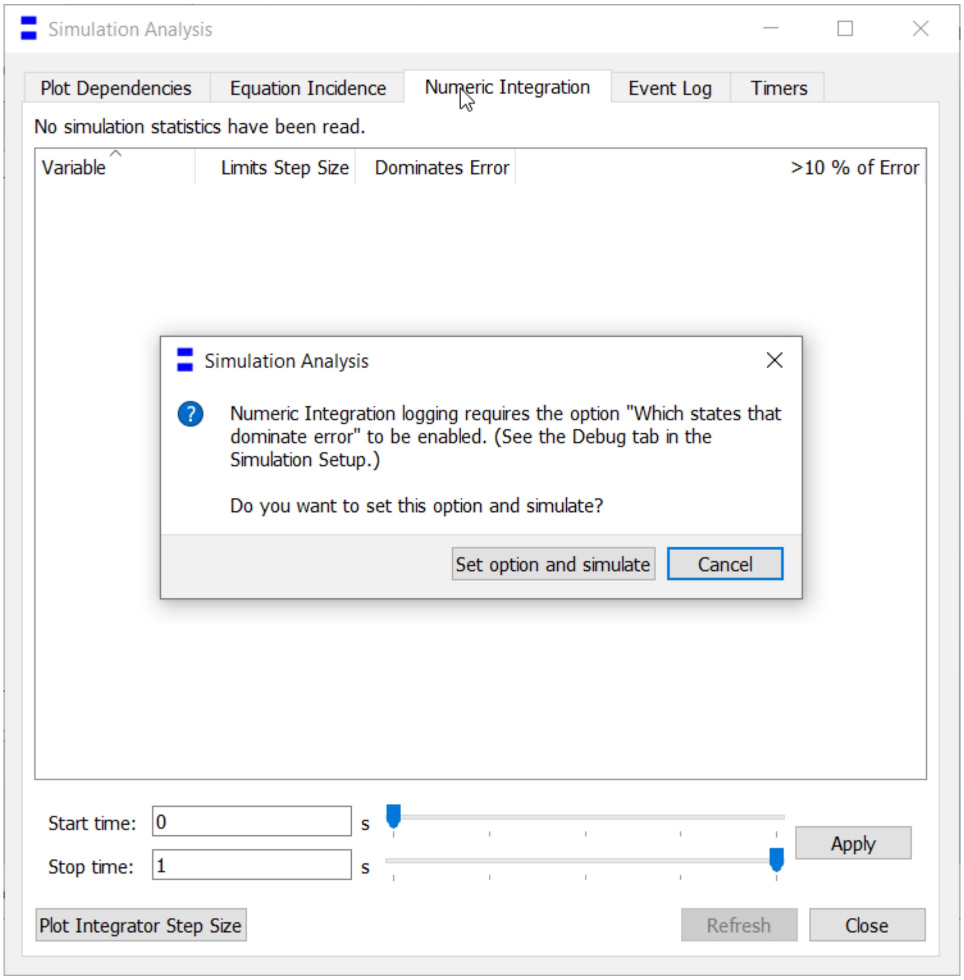
Simulation Analysis requires one or more of the below options to be enabled.
To enable further options or to disable them see Simulation>Setup>Debug.

Simulation Analysis options

- [ ] Events during initialization and simulation
- [ ] Which states that dominate error
- [x] Provide variable dependencies and equation incidence
- [ ] Generate block timers
- [ ] Generate function and FMU timers

If you select **Confirm selection and simulate**, you set these flags in the simulation setup and then simulate.

If you work with the Simulation Analysis interface, and activate a tab that needs a flag you have not set, you will get a dialog like:



No simulation statistics have been read.

Start time: 0 s

Stop time: 1 s

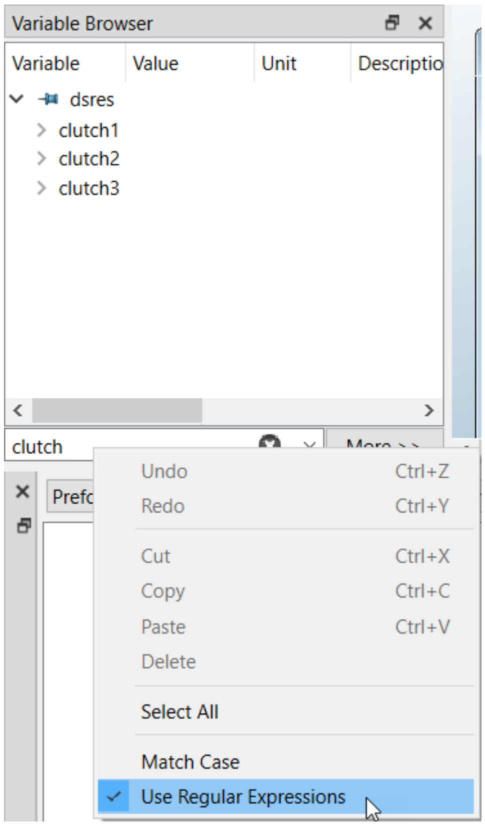
Numeric Integration logging requires the option "Which states that dominate error" to be enabled. (See the Debug tab in the Simulation Setup.)

Do you want to set this option and simulate?

If you select **Set option and simulate**, you set this (or these) flags in the simulation setup and then you simulate.

Option to use plain text search when filtering variables in the variable browser

Previously the use of regular expressions was always activated when filtering in the variable browser. Now the use is optional, controlled by a new option **Use Regular Expressions** in the context menu of the filter field:



The option is by default activated, that is, regular expressions is used like in previous Dymola versions. If you deactivate this option, you use plain text search instead. The change is remembered during the session.

3.4 Installation

For the current list of hardware and software requirements, please see chapter "Appendix – Installation: Hardware and Software Requirements" starting on page 43.

3.4.1 Improved license handling

Dassault Systèmes License Server (DSLS) is now default license format when ordering a new license

From Dymola 2023x, if you order a new license, you by default will get a license in the Dassault Systèmes License Server (DSLS) format. You don´t have to order new license keys when you update to a new DSLS version, the default is to keep the existing license keys when updating DSLS.

If you instead want a license in the FLEXnet license format, you must specify that when ordering the license.

Notes:

- Only FLEXnet licenses can be used as runtime license for the executable `dymosim.exe` or for a generated FMU.
- DSLS license server has to be updated more often that FLEXnet license servers if you update to a new Dymola version. For DSLS the following is the case:
    - For each “autumn release” of Dymola, a new DSLS version is released, and you have to update to that one. (As an example, if you upgrade to Dymola 2023x, you must upgrade the DSLS to 2023x as well.)
    - If you update to the “spring version” you don´t have to update the DSLS license, the previous DSLS version is still valid.
    - Note that older versions of Dymola can use newer versions of DSLS, for example, Dymola 2022x can use DSLS 2023x.
- FLEXnet licenses use other ports that DSLS licenses.

Nodelocked licenses in the Dassault Systèmes License Server (DSLS) format supported

In Dymola 2023x, you can use nodelocked licenses in the Dassault Systèmes License Server (DSLS) format.

For selection of this format, see next section.

Improved handling of license selection

Selection of license when starting Dymola with a command

Previously you had to start Dymola with a specific command to use a DSLS server license, even if only a DSLS license was available, In Dymola 2023x, Dymola by default starts looking for a FLEXnet license when starting up, but if a FLEXnet license is not found, it looks for a DSLS license instead. To start Dymola in this default mode you can use the command `dymola.exe` or start Dymola from the Start menu.

To force Dymola to use a DSLS license even if a FLEXnet license is also available, you must start Dymola with the command `dymola.exe /DSLS` (as you had to do in any case to use a DSLS license in previous Dymola versions). Using this command, the existence of a FLEXnet license is not checked.

The use of a FLEXnet license is default, but if you want to force it anyway, you can use the command `dymola.exe /FLEXnet` to start Dymola. Using this command, the existence of a DSLS license is not checked.

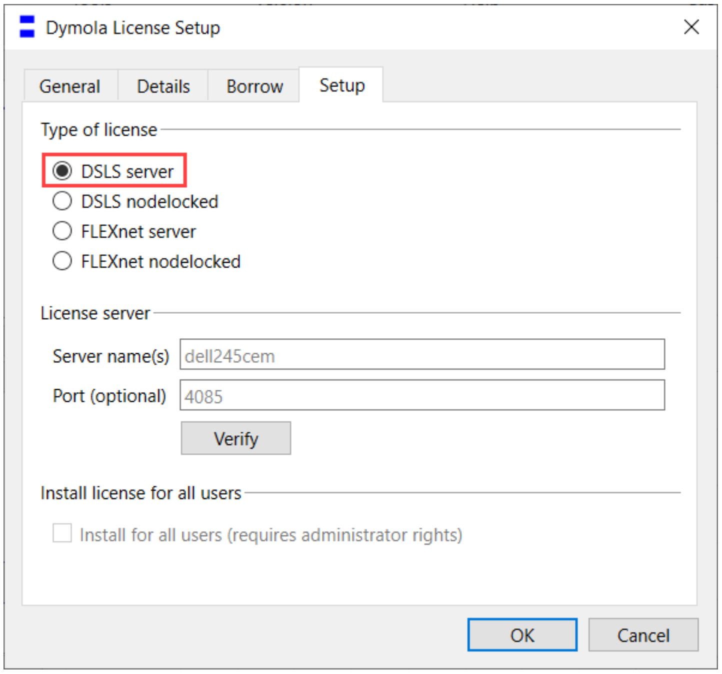
The options above are not case sensitive.

If a valid license is not found, Dymola will start in trial mode.

Selection and setup of license using Tools > License Setup, the Setup tab

When you use the command **Tools > License Setup**, the **Setup** tab has been improved; for each license alternative only relevant items are displayed.

If you select DSLS license format, you have now two alternatives. For sharable licenses, you can select:



Dymola License Setup

General | Details | Borrow | Setup

Type of license

(●) DSLS server
( ) DSLS nodelocked
( ) FLEXnet server
( ) FLEXnet nodelocked

License server

Server name(s): dell245cem

Port (optional): 4085

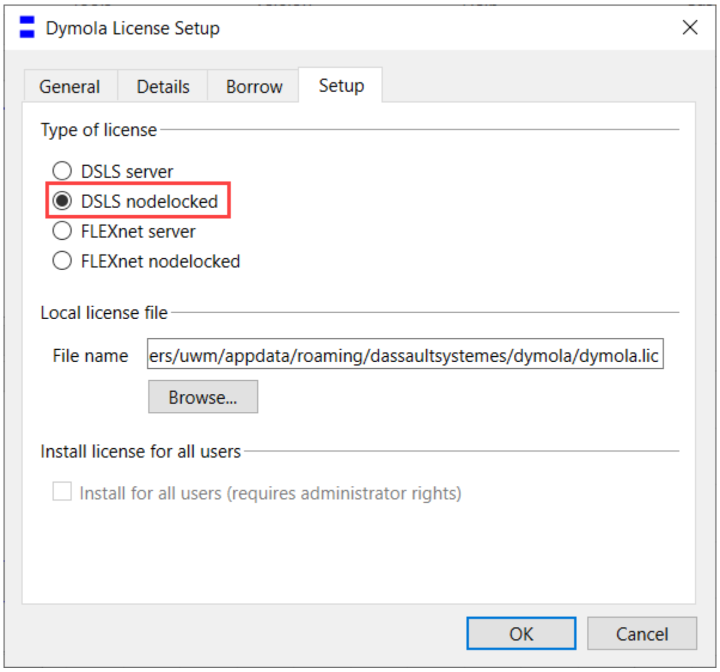
Verify

Install license for all users

[ ] Install for all users (requires administrator rights)

OK | Cancel

For a nodelocked DSLS license, you can select:


Dymola License Setup

General | Details | Borrow | Setup

Type of license

- ○ DSLS server
- ● DSLS nodelocked
- ○ FLEXnet server
- ○ FLEXnet nodelocked

Local license file

File name: ers/uwm/appdata/roaming/dassaultsystemes/dymola/dymola.lic

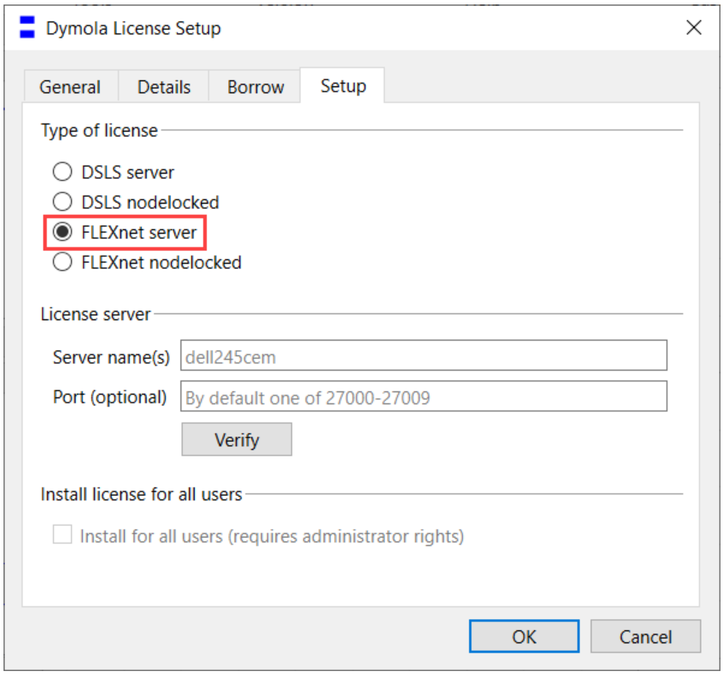
Browse...

Install license for all users

☐ Install for all users (requires administrator rights)

OK | Cancel

For the FLEXnet license format, you have two alternatives. For sharable licenses, select:


Dymola License Setup

General | Details | Borrow | Setup

Type of license

- ○ DSLS server
- ○ DSLS nodelocked
- ● FLEXnet server
- ○ FLEXnet nodelocked

License server

Server name(s): dell245cem

Port (optional): By default one of 27000-27009

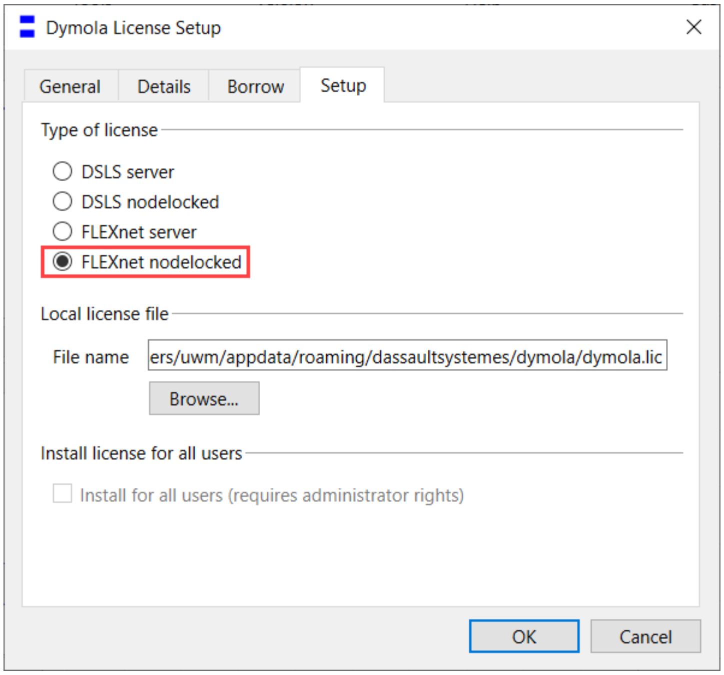
Verify

Install license for all users

☐ Install for all users (requires administrator rights)

OK | Cancel

For a nodelocked FLEXnet license, you can select:


Dymola License Setup

Type of license

○ DSLS server
○ DSLS nodelocked
○ FLEXnet server
● FLEXnet nodelocked

Local license file

File name: ers/uwm/appdata/roaming/dassaultsystemes/dymola/dymola.lic

Install license for all users

☐ Install for all users (requires administrator rights)

The default is whatever license format currently used.

Note the list of differences between the licenses listed in the section about DSLS being the default format when ordering a license, see page 24.

Extended checking for Termination of Support (TOS) in license

Checking for Termination of Support (TOS) has been extended for licenses administrated by FLEXnet. TOS will be applied to each licensed feature individually (including Dymola itself). If you get a message indicating that you have no current license, please renew support or use an older version of the library (issued before the TOS date).

Notes:

- The **Tools > About** dialog says when a TOS license actually expires.
- The TOS check feature was implemented for licenses administered by DSLS (Dassault Systèmes License Server) already in Dymola 2022x.
- For more information about this feature, please see the document Dymola 2023x: Updates for FLEXnet licenses. Note that this document is valid for DSLS as well.

3.4.2 Installation on Windows

Microsoft Visual Studio compilers

Support for Visual Studio 2022 compiler

Dymola 2023x supports the Visual Studio 2022 compiler, the following editions:

Visual Studio Community 2022 (17)

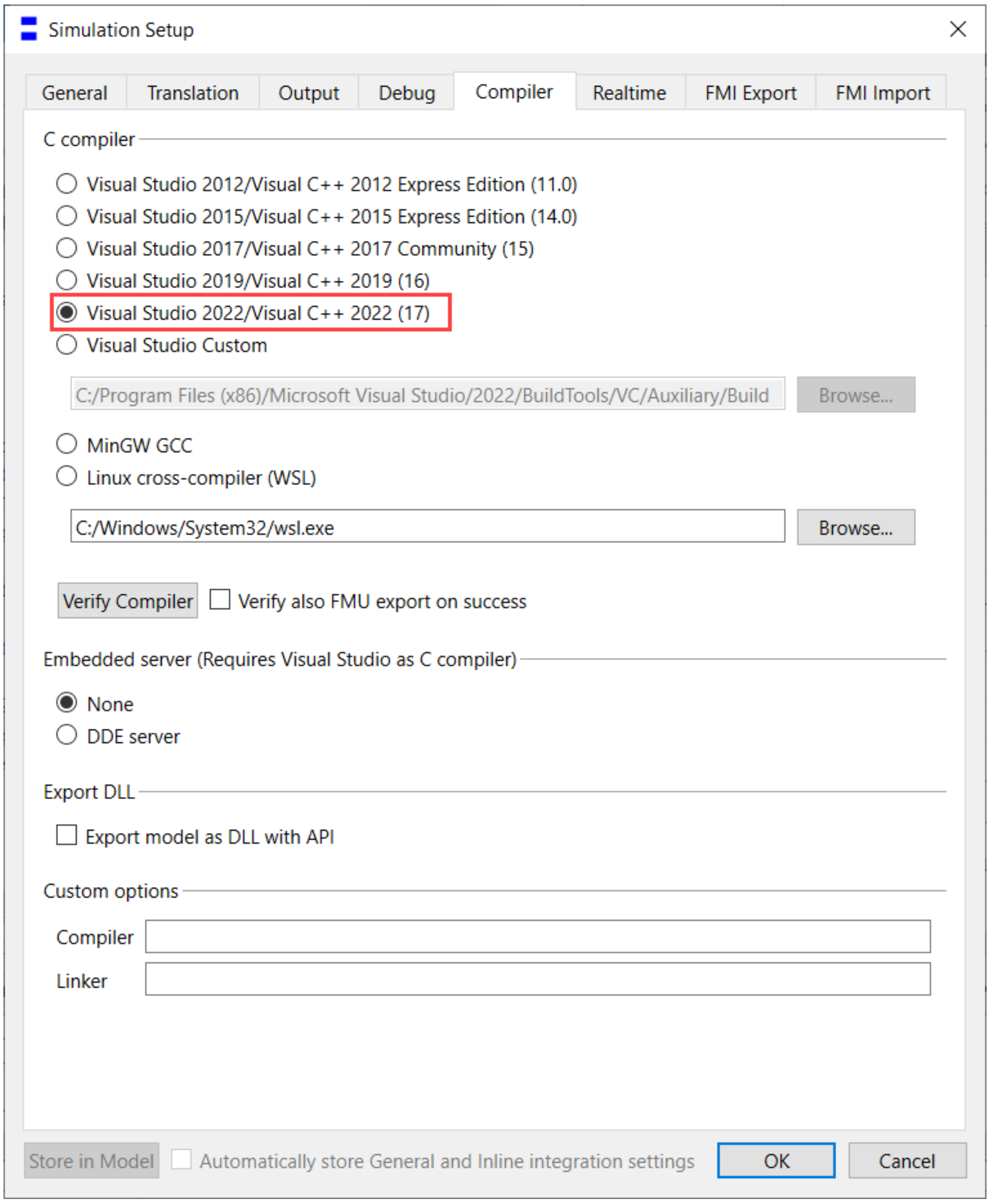
Visual Studio Enterprise 2022 (17)

Visual Studio Professional 2022 (17)

Visual Studio Build Tools 2022 **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)**.

The compiler is selected in the simulation setup, reached by the command **Simulation > Setup**, in the **Compiler** tab:


Simulation Setup

C compiler

- ○ Visual Studio 2012/Visual C++ 2012 Express Edition (11.0)
- ○ Visual Studio 2015/Visual C++ 2015 Express Edition (14.0)
- ○ Visual Studio 2017/Visual C++ 2017 Community (15)
- ○ Visual Studio 2019/Visual C++ 2019 (16)
- ● Visual Studio 2022/Visual C++ 2022 (17)
- ○ Visual Studio Custom

C:/Program Files (x86)/Microsoft Visual Studio/2022/BuildTools/VC/Auxiliary/Build

- ○ MinGW GCC
- ○ Linux cross-compiler (WSL)

C:/Windows/System32/wsl.exe

☐ Verify also FMU export on success

Embedded server (Requires Visual Studio as C compiler)

- ● None
- ○ DDE server

Export DLL

☐ Export model as DLL with API

Custom options

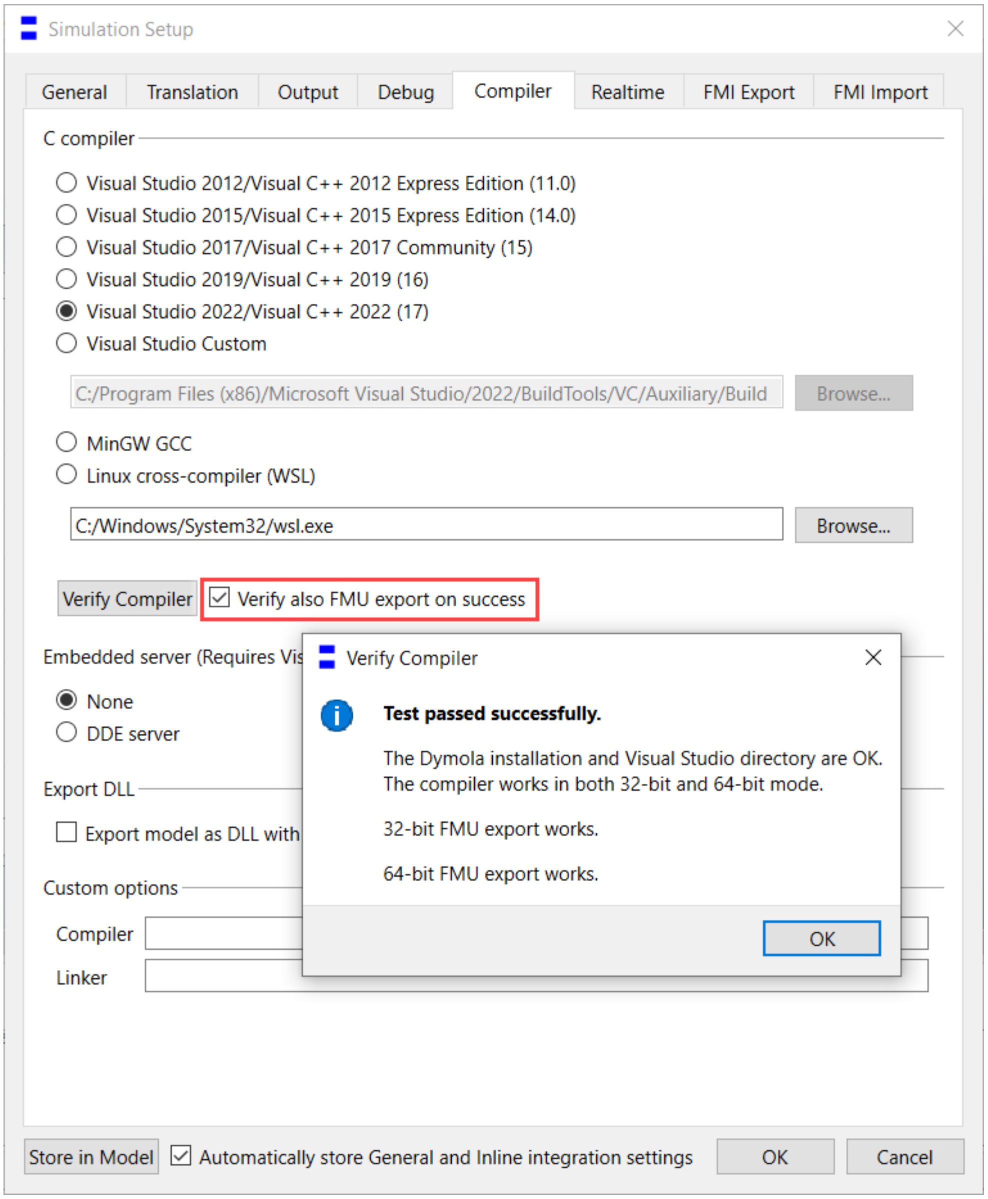
Compiler

Linker

☐ Automatically store General and Inline integration settings

Option to also test FMU export when verifying compiler

Already in Dymola 2023, you could activate an FMU export test when verifying the compiler:


Simulation Setup

General | Translation | Output | Debug | Compiler | Realtime | FMI Export | FMI Import

C compiler

- ◯ Visual Studio 2012/Visual C++ 2012 Express Edition (11.0)
- ◯ Visual Studio 2015/Visual C++ 2015 Express Edition (14.0)
- ◯ Visual Studio 2017/Visual C++ 2017 Community (15)
- ◯ Visual Studio 2019/Visual C++ 2019 (16)
- ● Visual Studio 2022/Visual C++ 2022 (17)
- ◯ Visual Studio Custom

C:/Program Files (x86)/Microsoft Visual Studio/2022/BuildTools/VC/Auxiliary/Build   Browse...

- ◯ MinGW GCC
- ◯ Linux cross-compiler (WSL)

C:/Windows/System32/wsl.exe   Browse...

Verify Compiler   ☑ Verify also FMU export on success

Embedded server (Requires Vis

- ● None
- ◯ DDE server

Export DLL

☐ Export model as DLL with

Custom options

Compiler

Linker

Store in Model   ☑ Automatically store General and Inline integration settings   OK   Cancel

Verify Compiler

Test passed successfully.

The Dymola installation and Visual Studio directory are OK.
The compiler works in both 32-bit and 64-bit mode.

32-bit FMU export works.

64-bit FMU export works.

OK

If you activate this option and then click **Verify Compiler**, Dymola will perform a test of exporting an FMU if the compiler test itself is successful. Both 32-bit and 64-bit FMU export is tested. The message from a successful test is shown in the figure above.

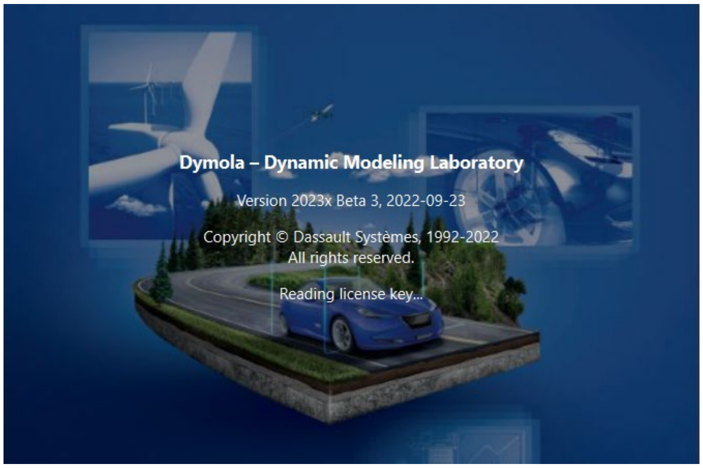
Note that this is a temporary activation of an additional test; the activation is not remembered when closing the dialog.

Updated Qt version

Dymola 2023x is built with Qt 6.3.

Splash screen when starting Dymola

To provide feedback that Dymola is starting up, a splash screen is now shown at startup:


Dymola – Dynamic Modeling Laboratory
Version 2023x Beta 3, 2022-09-23
Copyright © Dassault Systèmes, 1992-2022
All rights reserved.
Reading license key...

If you don´t want any splash screen at startup, you can start dymola using the option `–nosplash` or `/nosplash` (that is, `Dymola.exe –nosplash` or `Dymola.exe /nosplash`).

3.4.3 Installation on Linux

Updated Linux version

Dymola 2023x is supported 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 it is 6.3. For earlier versions of Dymola, to find the relevant Qt version, see the corresponding Dymola Release Notes.
- Select Supported platforms)

Any later version of gcc is typically compatible.

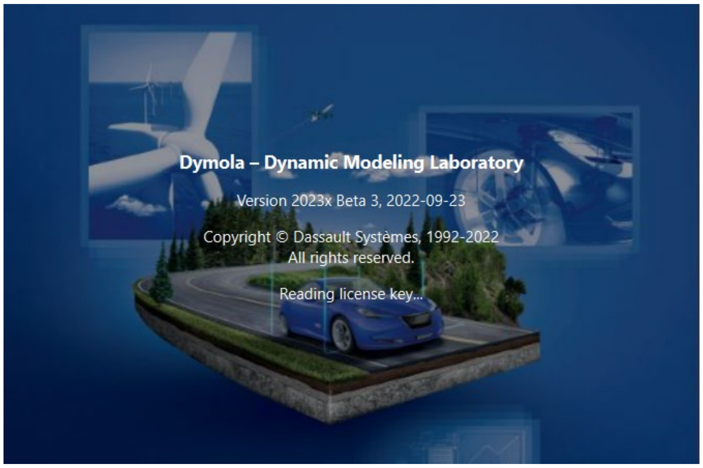
In addition to gcc, the model C code can also be compiled by clang.

Updated Qt version

Dymola 2023x is built with Qt 6.3.

Splash screen when starting Dymola

To provide feedback that Dymola is starting up, a splash screen is now shown at startup:

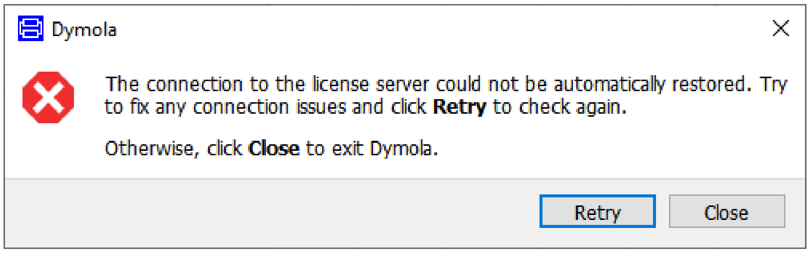

Dymola – Dynamic Modeling Laboratory
Version 2023x Beta 3, 2022-09-23
Copyright © Dassault Systèmes, 1992-2022
All rights reserved.
Reading license key...

If you don´t want any splash screen at startup, you can start dymola using the option `–nosplash` or `/nosplash` (that is, `Dymola.exe –nosplash` or `Dymola.exe /nosplash`).

3.4.4 Dymola license server on Windows and Linux

Checking for lost server connection for FLEXnet

If you use the FLEXnet license server, Dymola checks the contact to the license server every 5 minutes. If the connection is lost, warning messages are displayed every 5 minutes. If the connection has been down for 20 minutes, Dymola displays the message:



The connection to the license server could not be automatically restored. Try to fix any connection issues and click **Retry** to check again.

Otherwise, click **Close** to exit Dymola.

In this case, you are locked out from normal use of Dymola and you have two options:

- Fix the connection problem and click on **Retry**. Dymola will check again if the connection to the server works, and if so, you are returned to the Dymola session. If the check fails, Dymola loops back to the above dialog.
- Give up and accept that the connection to the license sever cannot be restored. You then click on **Close**, meaning that Dymola will go through the usual process for the command **File > Exit**, giving you a chance to save modified models before closing Dymola.

For details, please see the document Dymola 2023x: Updates for FLEXnet licenses.

Checking for lost server connection for DSLS

For the DSLS (Dassault Systèmes License Server) for Dymola, the above feature was implemented already in Dymola 2022x. However, the final message above is now also implemented for DSLS. (The final message gives the option to retry or close.)

Updated DSLS version

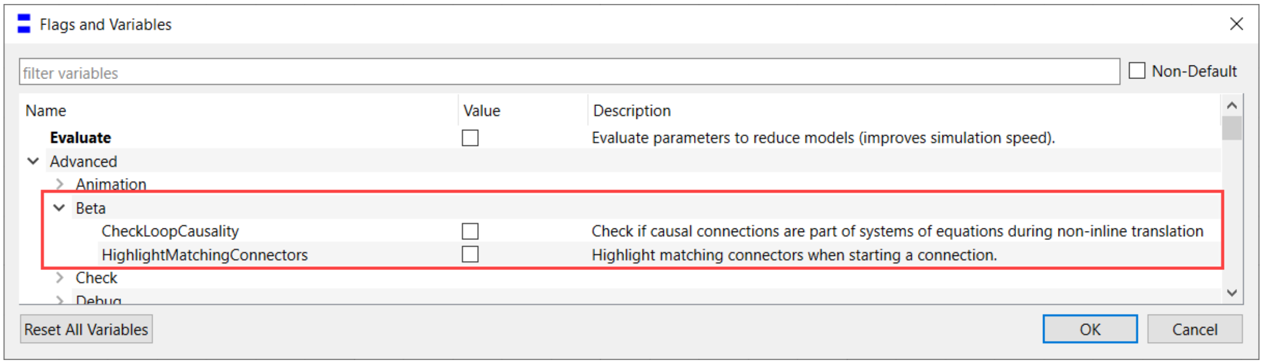
Dymola 2023x supports DSLS R2023x. Earlier DSLS versions cannot be used.

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.UnderDevelopment` flags in **Tools > Options > Variables…**:



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
HighlightMatchingConnectors	☐	Highlight matching connectors when starting a connection.
Check		
Debug		

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

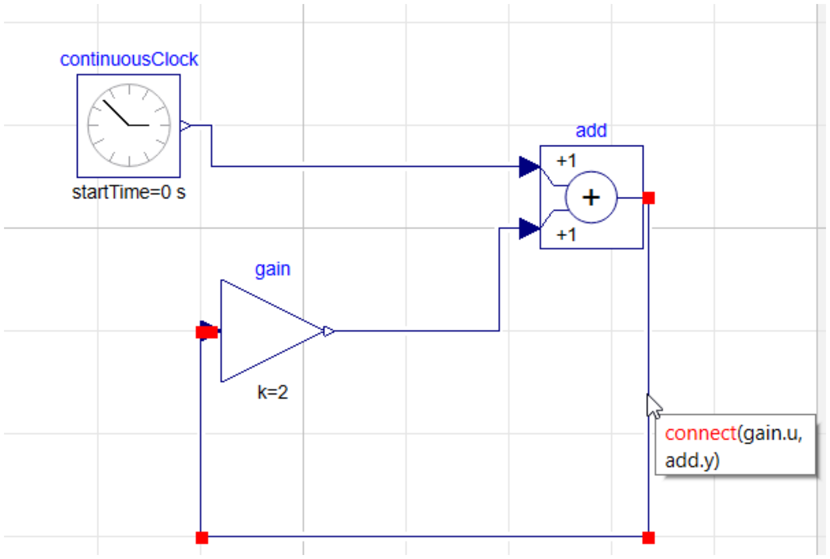
In Dymola 2023 x, the following "under development", features are available:

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.

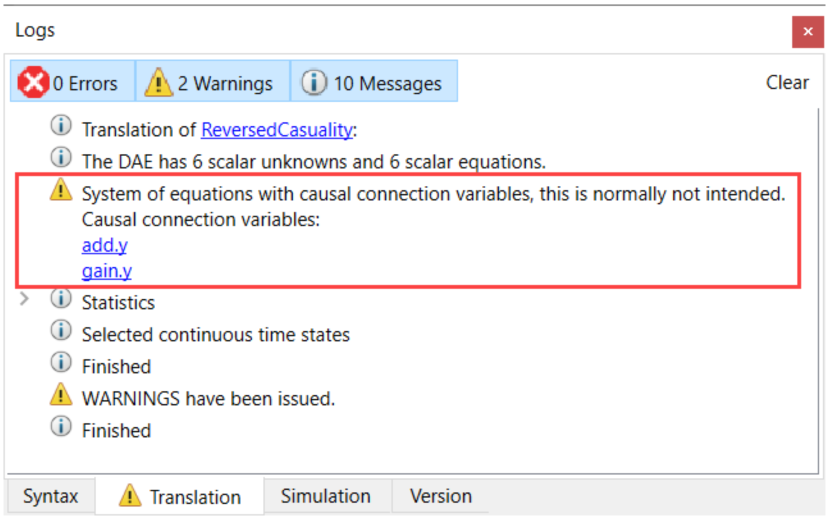
An example model might be:

```
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;
```



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:



```
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
```

Option to highlight good choices when starting a connection

To use this option, activate the flag `Advanced.Beta.HighlightMatchingConnectors`.

When this option is activated, and you start drawing a connection, the suitable connectors are highlighted.

3.6 Model Experimentation

3.6.1 Option to allow changing of evaluated parameters when sweeping

For more about this option, see section “Option to allow changes of evaluated parameters after translation, including sweeps, with automatic re-translation” on page 19.

3.6.2 Minor improvements

Regression testing supported when sweeping with three or more parameters

The function `sweepManyParametersScatter` in the package `Design.Experimentation` now has a new argument `seed`. The scatter plots use random perturbations to avoid dots ending up on top of each other. The new argument makes it possible to set the seed for this randomization. This allows for regression testing.

3.7 Other Simulation Environments

3.7.1 Dymola – Matlab interface

Compatibility

The Dymola – Simulink interface now supports Matlab releases from R2017a (ver. 9.2) up to R2022a (ver. 9.12). 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.

3.7.2 Real-time simulation

Compatibility – dSPACE

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

- dSPACE Release 2017-A with Matlab R2017a
- dSPACE Release 2017-B with Matlab R2017b
- 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 R2021a and R2021b
- dSPACE Release 2022-A with Matlab R2021a, R2021b, and R2022a

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 R2017a (Simulink Real-Time ver. 6.6) to R2022a (Simulink Real-Time ver. 8.0). Only Microsoft Visual C compilers have been tested.

3.7.3 Dymosim DLL

Postponed discontinuation of dymosim DLL support

Dymosim DLL is still supported in Dymola 2023x. However, it will be removed in a future release of Dymola. It will be replaced by the use of FMI.

3.7.4 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 19.

ModelManagement now included in the Python interface

ModelManagement is now part of the Python interface. It is included in `dymola.egg` but in a separate module, `model_management.py`.

The functions are included as a straight list. The naming convention is that dots are replaced with underscore. For example, the function `ModelManagement.Structure.AST.Classes.GetAnnotation` is available as `ModelManagement_Structure_AST_Classes_GetAnnotation`.

An example of usage:

```
from dymola.dymola_interface import DymolaInterface
from dymola.model_management import *

dymola = DymolaInterface()
model_management = ModelManagement(dymola)

s = model_management.ModelManagement_Structure_AST_Classes_GetAnnotation(
"Modelica.Blocks.Examples.PID_Controller", "experiment.StopTime")
print(s) # 4

s = model_management.ModelManagement_Structure_AST_Misc_ClassShownInBrowser()
print(s) # Unnamed

b = model_management.ModelManagement_Structure_AST_Misc_ClassExists(
"Modelica.Mechanics.Rotational.Examples.CoupledClutches")
print(b) # True

list = model_management.ModelManagement_Structure_AST_Misc_ClassesInPackage(
"Modelica.Mechanics.Rotational.Components")
print(len(list)) # 24
print(list)
```

There are three examples included in Dymola: `ModelManagementExample.py`, `ModelManagementExample2.py`, and `ModelManagementExample3.py`. They are available in the same folder as the other Python examples.

`checkLibrary` and `compareModels` have no default values in Python. This is correct, since they mix default and non-default parameters, which is not allowed in Python.

3.7.5 SSP support

SSP import

Option to disable file name prompting when saving package created from imported SSP file

A new Boolean argument `silentSave` is added in the built-in function `importSSP`. The default value is `false`. If you set this argument to `true`, the package created by the imported SSP file is silently saved with the default file name, overwriting any existing file with this name.

3.7.6 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

Dymola 2023x includes limited support for FMI version 3.0. The following is implemented:

- FMU export: All existing export features in Dymola are implemented in FMI 3.0, but no new FMI 3.0 export features.

- FMU import: All existing import features in Dymola are implemented in FMI 3.0, but no new FMI 3.0 import features.

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 models will not work.

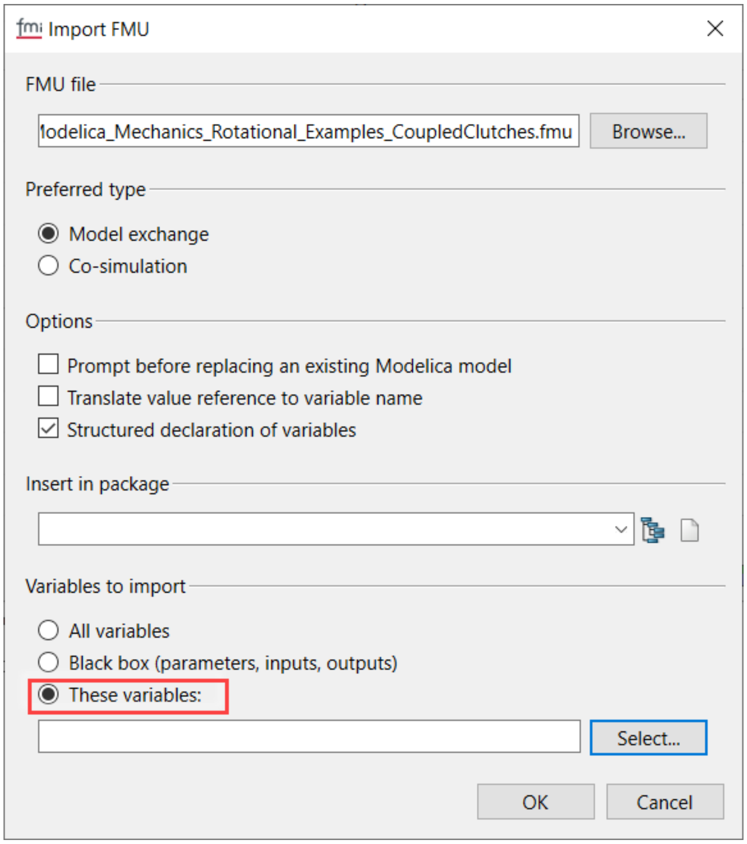
For details about FMI 3.0, see:

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

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

Minor improvement: Consistent default value of the GUI for selecting what variables to import when importing an FMU

When importing an FMU, and selecting to specify yourself what variables to import (as in the image below), the resulting dialog now always by default displays the selection corresponding to the black box alternative; that is, parameters, inputs and outputs are selected to be exposed by default.



Import FMU

FMU file: lodelica_Mechanics_Rotational_Examples_CoupledClutches.fmu

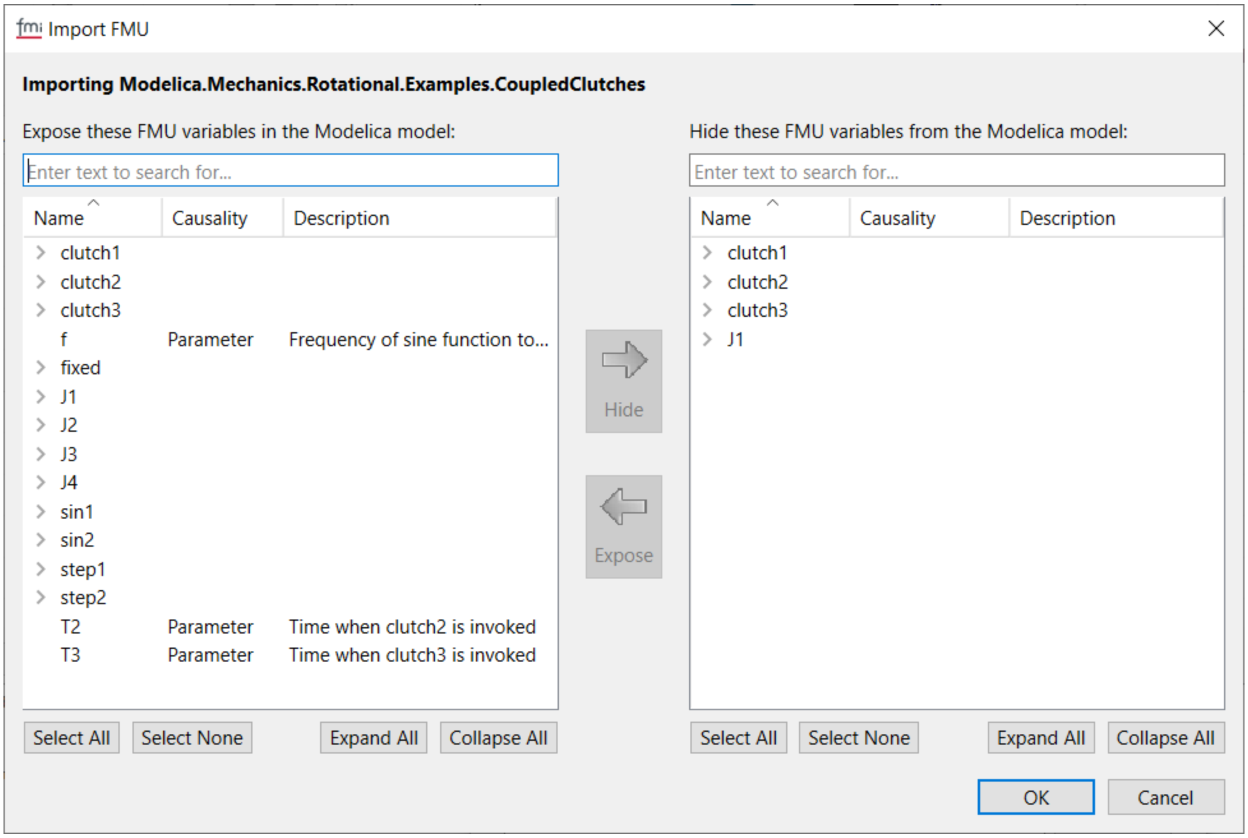
Preferred type: [✓ Model exchange] Co-simulation

Options:
- [ ] Prompt before replacing an existing Modelica model
- [ ] Translate value reference to variable name
- [x] Structured declaration of variables

Insert in package:

Variables to import: All variables; Black box (parameters, inputs, outputs); [✓ These variables:]

An example below, from importing the demo Coupled Clutches as an FMU:


Import FMU

Importing Modelica.Mechanics.Rotational.Examples.CoupledClutches

Expose these FMU variables in the Modelica model:

Name	Causality	Description
clutch1		
clutch2		
clutch3		
f	Parameter	Frequency of sine function to...
fixed		
J1		
J2		
J3		
J4		
sin1		
sin2		
step1		
step2		
T2	Parameter	Time when clutch2 is invoked
T3	Parameter	Time when clutch3 is invoked

Hide these FMU variables from the Modelica model:

Name	Causality	Description
clutch1		
clutch2		
clutch3		
J1		

Previously, in some cases, all variables were selected to be exposed. Note that you can easily obtain this from the above, by clicking **Select All** under the pane to the right, and then clicking **Expose**.

3.8 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 3.5.

Note that the Modelica Standard Library version 4.0.0 is compliant with the Modelica Language Specification 3.4.

3.9 Documentation

General

In the software, distribution of Dymola 2023x Dymola User Manuals of version “September 2022” will be present; these manuals include all relevant features/improvements of Dymola 2023x presented in the Release Notes.

Update of “Installing and Testing Microsoft Visual Studio Build Tools Compiler for Dymola”

The document “Installing and Testing Microsoft Visual Studio Build Tools Compiler for Dymola” has been enhanced to cover the installation of Visual Studio 2022 and the testing on Dymola 2023x. The document is available in the page http://www.Dymola.com/compiler - see the link “Installing and testing Microsoft Visual Studio Build Tools” on that page.

3.10 Appendix – Installation: Hardware and Software Requirements

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

3.10.1 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.

3.10.2 Software requirements

Microsoft Windows

Dymola versions on Windows and Windows operating systems versions

Dymola 2023x is supported, as 64-bit application, on Windows 8.1, and 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 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.

- 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

- 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 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.
- For 32-bit simulation, parallelization (multi-core) is currently not supported for any of the following algorithms: RadauIIa, Esdirk23a, Esdirk34a, Esdirk45a, and Sdirk34hw.

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

- 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.
- 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 supports DSLS R2023x. Earlier DSLS versions cannot be used.

Linux

Supported Linux versions and compilers

Dymola 2023x 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 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/insert/dsbuild.sh`. As an example, for a 64-bit Dymola 2023x application:

```
/opt/dymola-2023x-x86_64/insert/dsbuild.sh
```

Dymola 2023x 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 supports DSLS R2023x. Earlier DSLS versions cannot be used.

"