Dymola

Dynamic Modeling Laboratory

Dymola Release Notes

Dymola 2022

The information in this document is subject to change without notice.

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 2022:

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 46 for more information. **Note** that from Dymola 2020x, Visual Studio C++ compilers older than version 2012 are no longer supported.

Intel

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

GCC

Dymola 2022 has limited support for the MinGW GCC compiler, 32-bit and 64-bit. For more information about GCC, see section "Compilers" on page 46; the section about GCC compilers.

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 48 for more information.

2 About this booklet

This booklet covers Dymola 2022. 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 2022

3.1 Introduction

3.1.1 Additions and improvements in Dymola

A number of improvements and additions have been implemented in Dymola 2022 In particular, Dymola 2022 provides:

- Support for Modelica Language Specification 3.5 (page 44)
- Cross-compilation for Linux on Windows (page 27)
- Improvements of plotting
    - Plotting bar charts and area charts (page 14)
    - Support for plotting of external files in HTML, PNG, or SVG format (page 22)
    - Copying a 3D plot data to clipboard (page 17)
- Better support for including translation settings and output settings in scripting (page 18)
- Setting maximum run time for running simulations (page 11)
- Option to activate a transient mode to reach steady-state (page 12)
- Support for Intel compilers is discontinued from this release (page 31)
- Last Dymola version supporting dymosim DLL (page 38)
- FMI support improvements
    - Filtering signals when exporting an FMU (page 38)
    - Warning about missing code export license (page 40)
    - Option to activate the code export license for the FMI export dialog (page 41)
- New white paper “Exploring Model Structure with Equation Incidence” (page 45)

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 1.1.0
- Battery Library, version 2.2.1

- Brushless DC Drives Library, version 1.1.3
- ClaRa DCS Library, version 1.4.1
- ClaRa Grid Library, version 1.4.1
- ClaRa Plus Library, version 1.4.1
- Claytex Library, version 2021.1
- Claytex Fluid Library, version 2021.1
- Cooling Library, version 1.4.2
- Dassault Systemes Library, version 1.6.0
- Design, version 1.1.1
- Dymola Commands Library, version 1.11
- Dymola Models Library, version 1.3.0
- Electric Power Systems Library, version 1.4.1
- Electrified Powertrains Library (ETPL), version 1.3.4
- Fluid Dynamics Library, version 2.11.0
- Fluid Power Library, version 2021.1
- FTire Interface Library, version 1.1.1
- Human Comfort Library, version 2.11.0
- HVAC (Heating, Ventilation, and Air Conditioning) Library, version 2.11.0
- Hydrogen Library, version 1.3.4
- Model Management, version 1.3
- Pneumatic Systems Library, version 1.4.2
- Testing Library, version 1.4.0
- Thermal Systems Library, version 1.7.0
- Thermal Systems Mobile AC Library, version 1.7.0
- VeSyMA (Vehicle Systems Modeling and Analysis) Library, version 2021.1
- VeSyMA - Engines Library, version 2021.1
- VeSyMA - Powertrain Library, version 2021.1
- VeSyMA - Suspensions Library, version 2021.1
- VeSyMA2ETPL Library, version 2021.1
- Visa2Base, version 1.10
- Visa2Paper, version 1.10
- Visa2Steam, version 1.10
- Wind Power, version 1.1.3

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 Minor improvements

Option to evaluate default parameters values when exporting an icon or the diagram layer

When you export an icon using the built-in function `exportIcon`, or the diagram layer using the built-in function `exportDiagram`, you can now decide if parameter default values should be evaluated before the export, by the new Boolean input parameter `evaluate`. The parameter is `false` by default.

Specific action to move the Modelica text formatting margin

To prevent moving by accident the text-formatting margin when working with Modelica text, you now have to press **Shift** to be able to move the margin by dragging the margin line by the mouse.

Option to disable the indication of replaceable classes

The indication of replaceable classes (which now also includes classes in replaceable packages) in the diagram can now be controlled by the flag `Advanced.Editor.Highlight.ReplaceableClass`. The default value of the flag is `false`, meaning that replaceable classes in the diagram are not highlighted. (Note that since previously there is another flag `Advanced.Editor.Highlight.Replaceable` that controls the indication of replaceable or redeclared *components* in the diagram. That flag is by default `true`.)

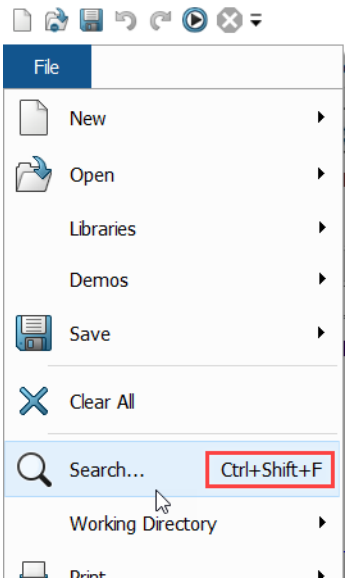
Improved symbolic size check of parameter-conditional sizes

The symbolic size check has been changed to not report errors in parameter-conditional sizes (e.g, `if n>0 then {...} else {...}` where the two arrays may be of different size).

As before, if you want a more detailed log of size issues you can activate more logging by setting the flag `Advanced.LogSymbolicSizeCheck=true`.

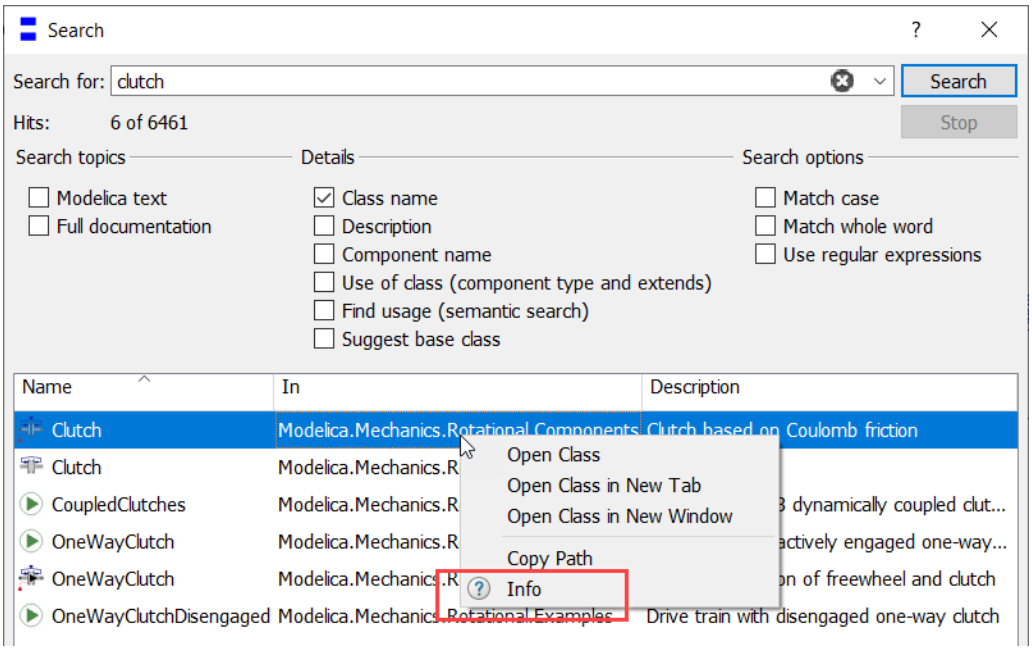
Keyboard shortcut for the command File > Search

You can now use the keyboard shortcut **Ctrl+Shift+F** to activate the command **File > Search...**.



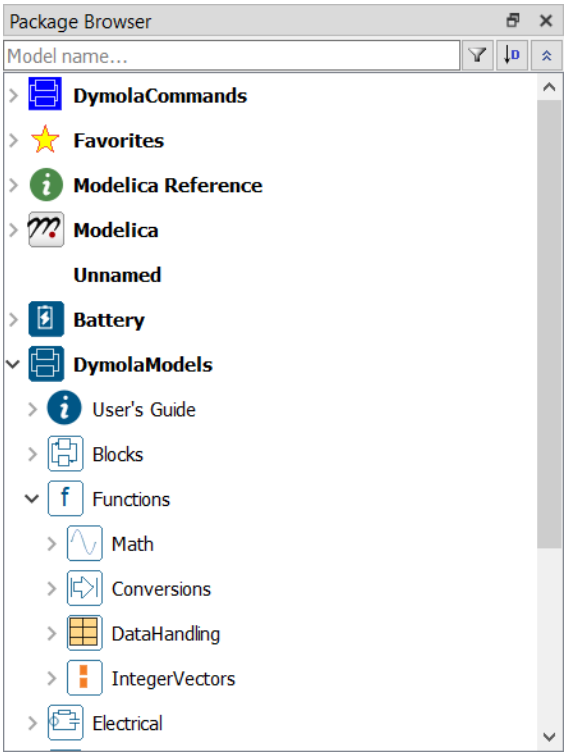
Info command added in context menu of the File > Search... dialog

The **Info** command is now available in the context menu of any **File > Search...** dialog item:



Option to highlight all top-level classes

To facilitate easier navigation in the package browser, in particular when you have opened several sub-packages, you can select to have all top-level classes highlighted:



You activate this option by setting the flag

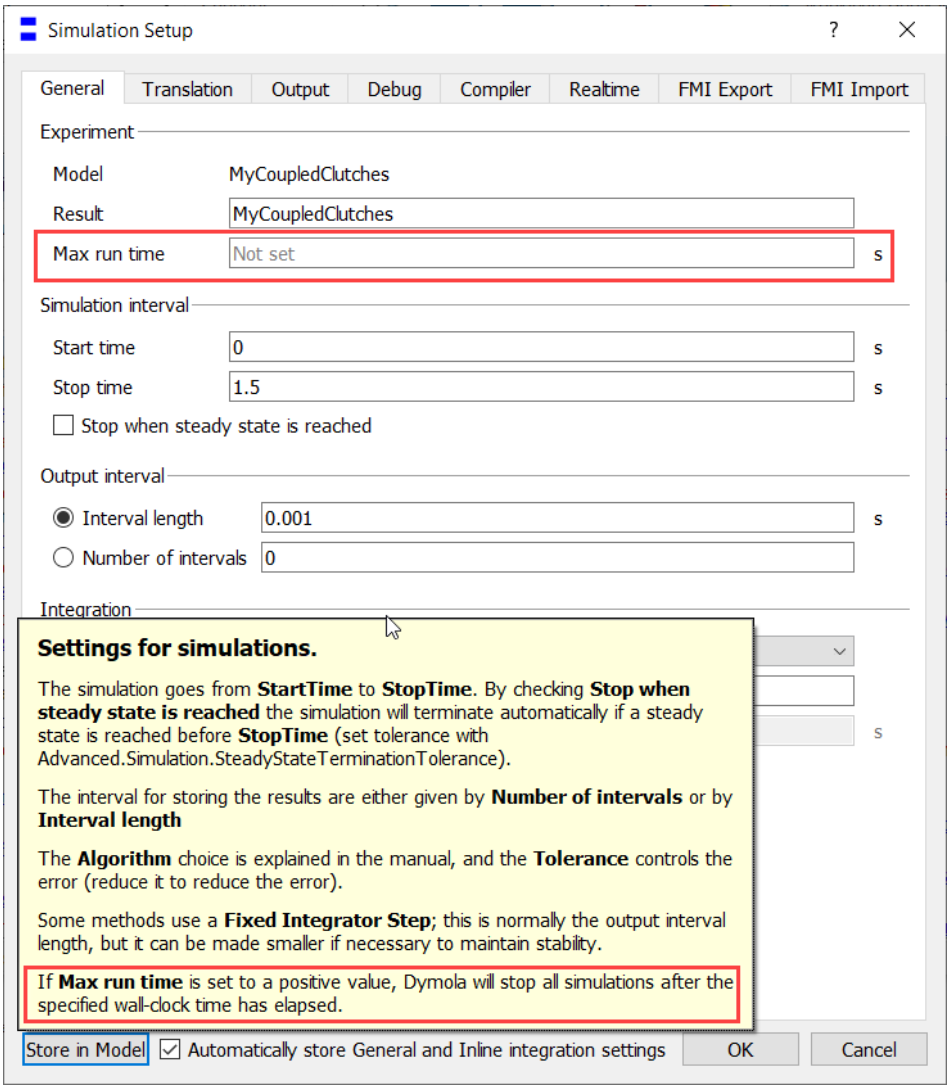
```
Advanced.UI.HighlightPackageInBrowser = true
```

The flag is by default `false`.

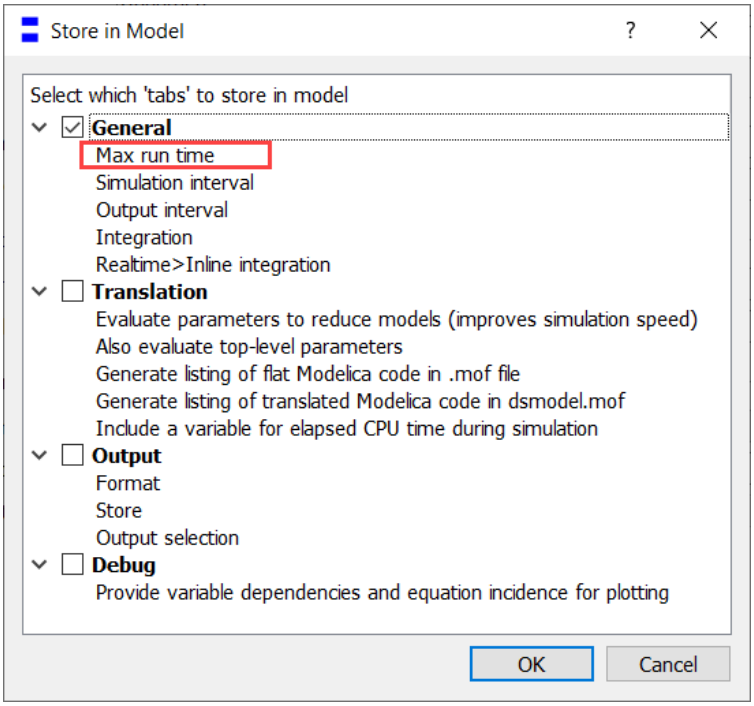
3.3 Simulating a model

3.3.1 Option to set the maximum run time for running simulations

You can set Dymola to stop any running simulation after a given time. You can do this in the simulation setup (reached by the command **Simulation > Setup**), the **General** tab, by the setting **Max run time** (the figure below shows the tab, but also the tooltip from clicking the question mark in the header of the window, and then clicking in the window):



The new setting is also saved by the command button **Store in Model**:



The default setting **Not set** corresponds to the flag

```
Advanced.Simulation.MaxRunTime = 0.0
```

meaning that there is no limitation on the run time.

Notes:

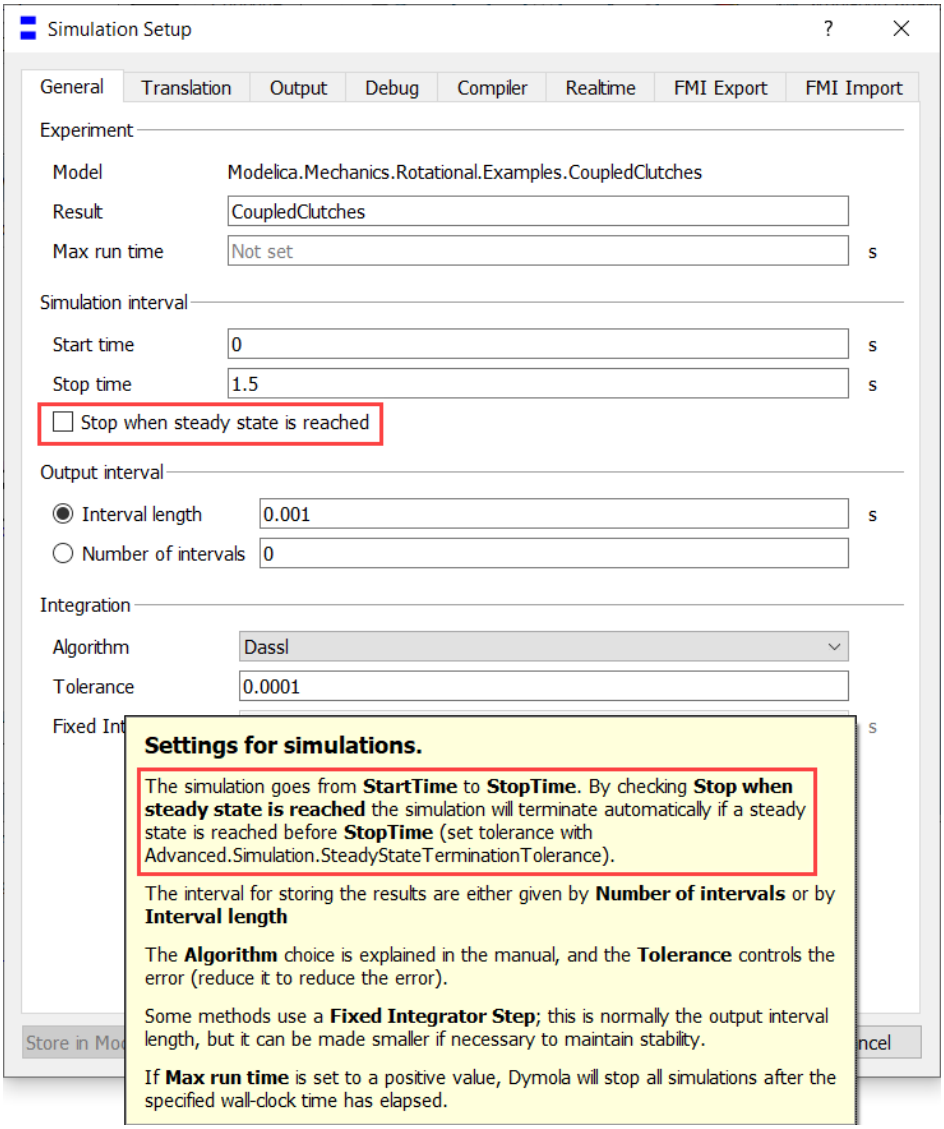
- The wall-clock time is measured, not the CPU time.
- The run time unit is independent of the simulation time unit.
- The option works for both single simulations and batch runs.
- Because the max time is checked by Dymola, the feature works even if the simulation is stuck in code that is not produced by Dymola, such as code of an imported FMU or external C code.

3.3.2 Option to activate a transient mode to reach steady-state

You can activate a transient mode with an undefined stop time that just runs until the transients damp out, that is, when you have reached steady state. However, if the ordinary stop time is reached, the simulation is stopped even if steady state is not reached.

You can activate the feature in the simulation setup (reached by the command **Simulation > Setup**), the **General** tab, by activating the setting **Stop when steady state is reached** (the

figure below shows the tab, but also the tooltip from clicking the question mark in the header of the window, and then clicking in the window):



Note that the **Stop time** stops the simulation when that time is reached, even if steady state is not yet reached at that time.

The tolerance is by default 0.02, with the time scale taken into account; see below for details.

As an alternative, you can set the tolerance to an absolute value by using the flag `Advanced.Simulation.SteadyStateTerminationTolerance`. The default value of this flag is `0.0`, meaning that the flag is not used.

Formally, this gives the test if steady state is reached for each state x_i as:

```
|der(x_i)| <= tolerance * (|x_i| + |nominal_i|) / 2.0
where
  tolerance = Advanced.Simulation.SteadyStateTerminationTolerance
if this flag is set, or
  tolerance = 0.02 / min(stopTime - startTime, 500*interval_length)
```

For unbounded variables, the term |x_i| is removed from the test.

The option **Stop when steady state is reached** corresponds to the flag `Advanced.Simulation.SteadyStateTermination`.

Note: The feature is implemented by checking if the state derivatives are close to zero, as above. This means that the following cannot be detected:

- Periodic steady states
- Steady states where a subset of the states are still varying

3.3.3 Plot tab

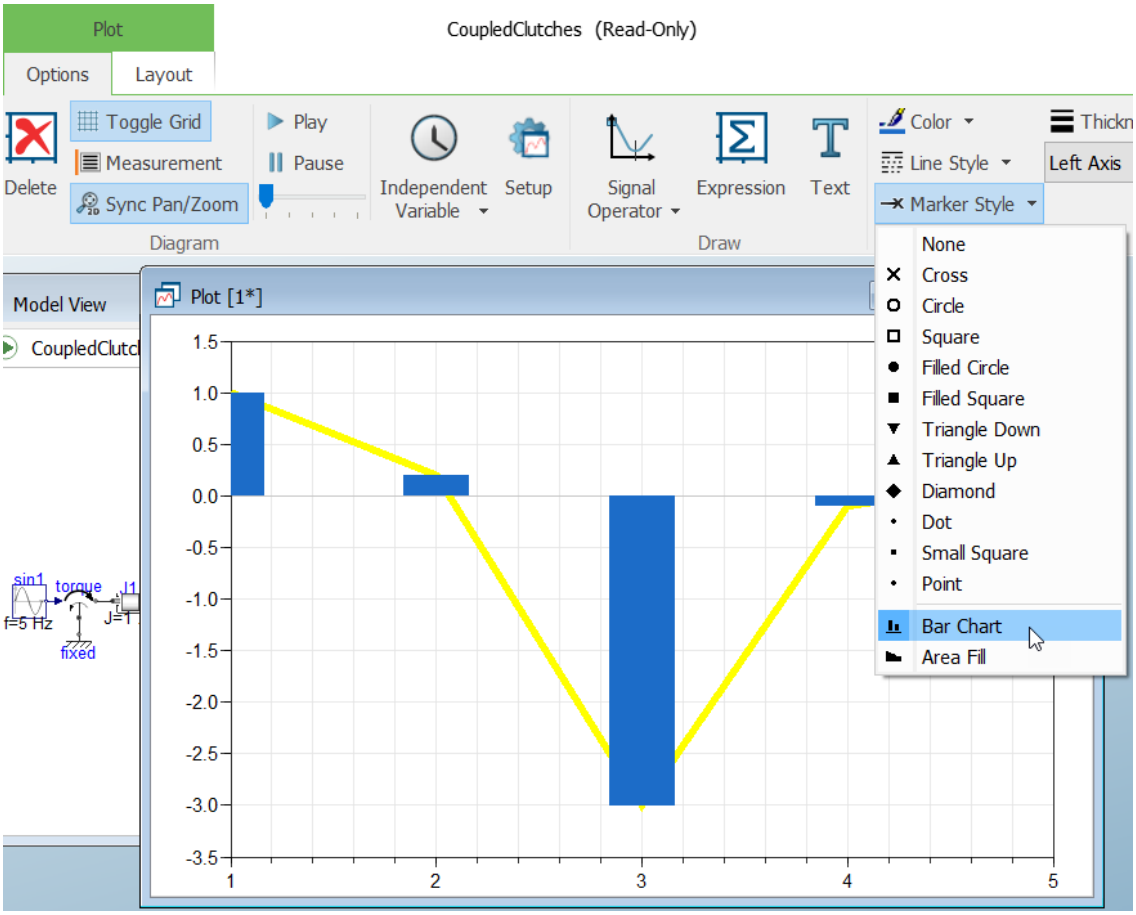
Plotting bar charts and area charts

You can now plot bar charts and area charts by using two new marker styles, **Bar Chart** and **Area Fill**.

For an example of bar charts, enter, in the command input line of the **Simulation** tab, the plot command

```
plotArray({1,2,3,4,5}, {1,0.2,-3,-0.1,0.2})
```

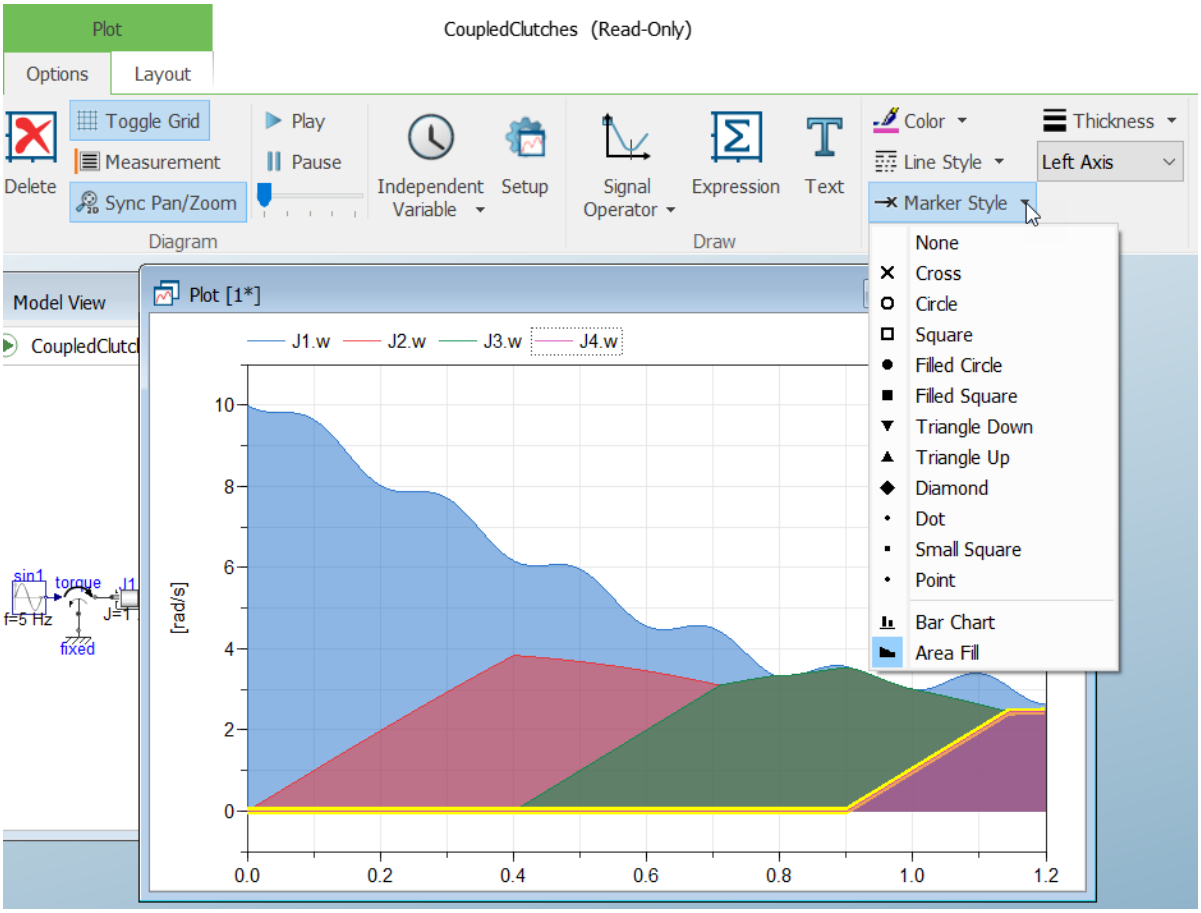
and then select the curve and apply **Marker Style > Bar Chart**:



Notes:

- The yellow representation of the original curve disappears by clicking outside it.
- To select the underlying curve, you must click any corresponding point, that is, at the center of any bar top.

For an example of area charts, the below is the demo Coupled Clutches with the curves selected and the marker style **Area Fill** applied on each of the curves:



Notes:

- The opacity can be controlled by the flag `Advanced.Plot.AreaFillOpacity`. The default value of the flag is `0.5`.
- Thickness and Line Style can be applied to the curve as well.

Plotting bar charts and area charts are supported by scripting as well, see section “Scripting support for plotting bar charts and area charts” on page 22.

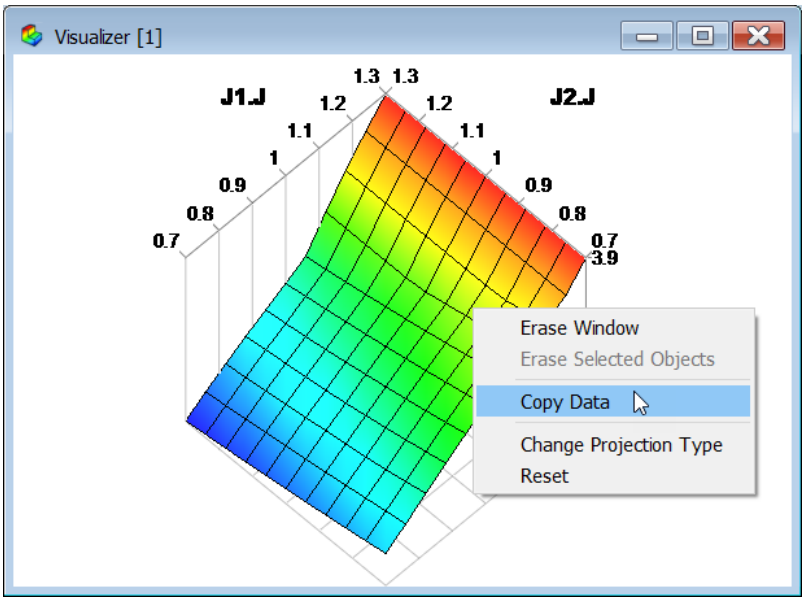
Plotting of external files

Dymola 2022 supports plotting of external files in the HTML, SVG, or PNG format. In addition, HTML plotting scripts generated in Python by for example matlabplot or plotly are supported. The files can be plotted by the new built-in function `plotExternal`. For details about the built-in function `plotExternal`, and examples, see “Scripting support for plotting of external files” on page 22.

3.3.4 Visualizer window

Copying a 3D plot to clipboard

You can now copy the 3D plots of a visualizer window to the clipboard, for further use in for example MS Excel, by right-clicking in the window and selecting **Copy Data**:



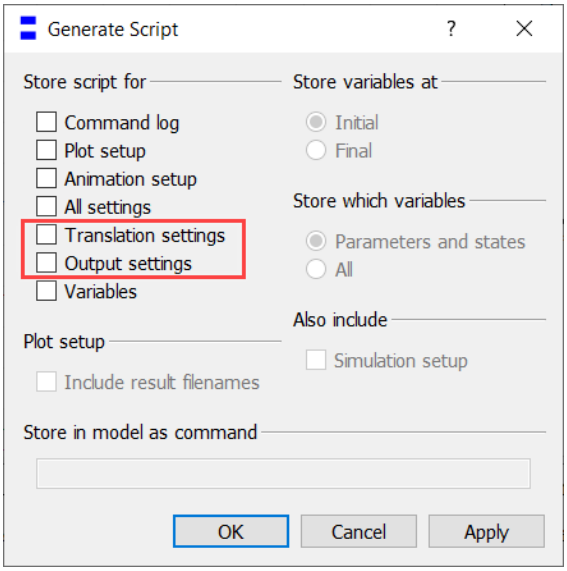
The data is copied in matrix form, x, y, and z. If you have more than one 3D plot in the window, the matrices are named x1,y1,z1, x2,y2,z2 etc.

3.3.5 Scripting

Better support for including translation settings and output settings in scripting

New options in the command Simulation > Generate Script

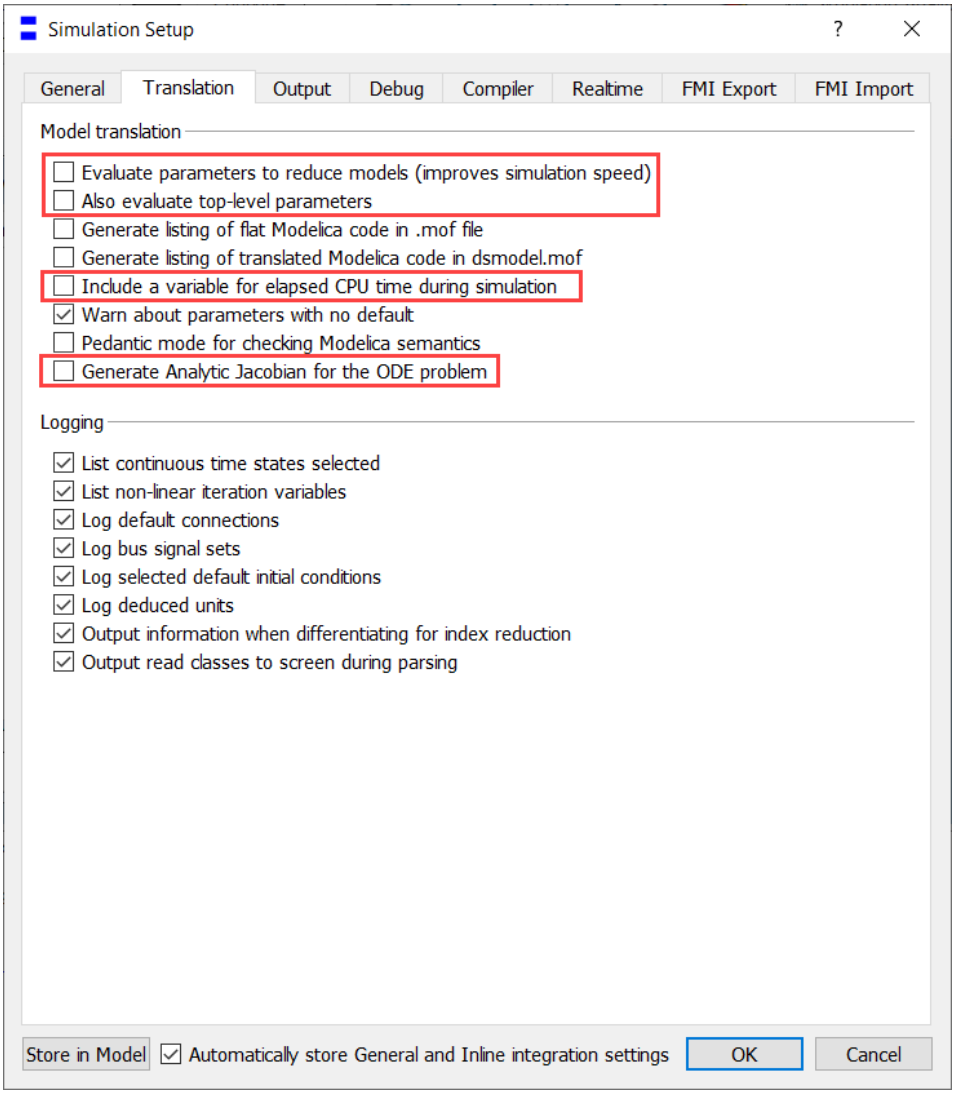
The command **Simulation > Generate Script** now has two new options:



Translation settings.

Activating **Translation settings** will include a number of flags corresponding to settings in the **Model translation** group in the **Translation** tab of the simulation setup, reached by the command **Simulation > Setup**, the **Translation** tab.

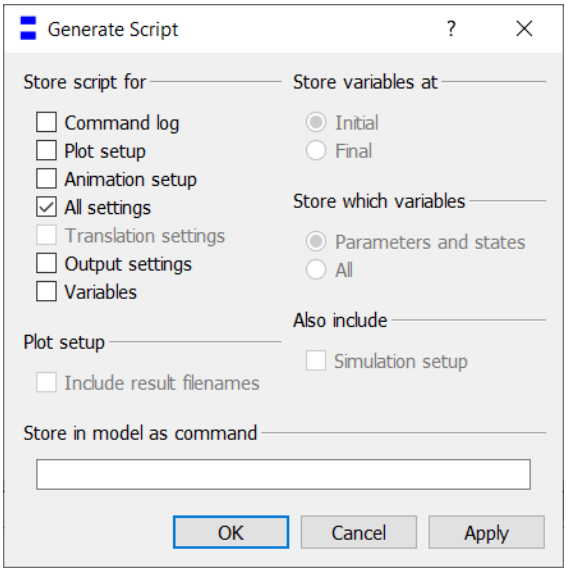
To be specific, the flags corresponding to the framed settings below are included:



Note that there are some differences when comparing this new feature with storing the translation settings in the model using the button **Store in Model** (in the lower left corner in the figure above):

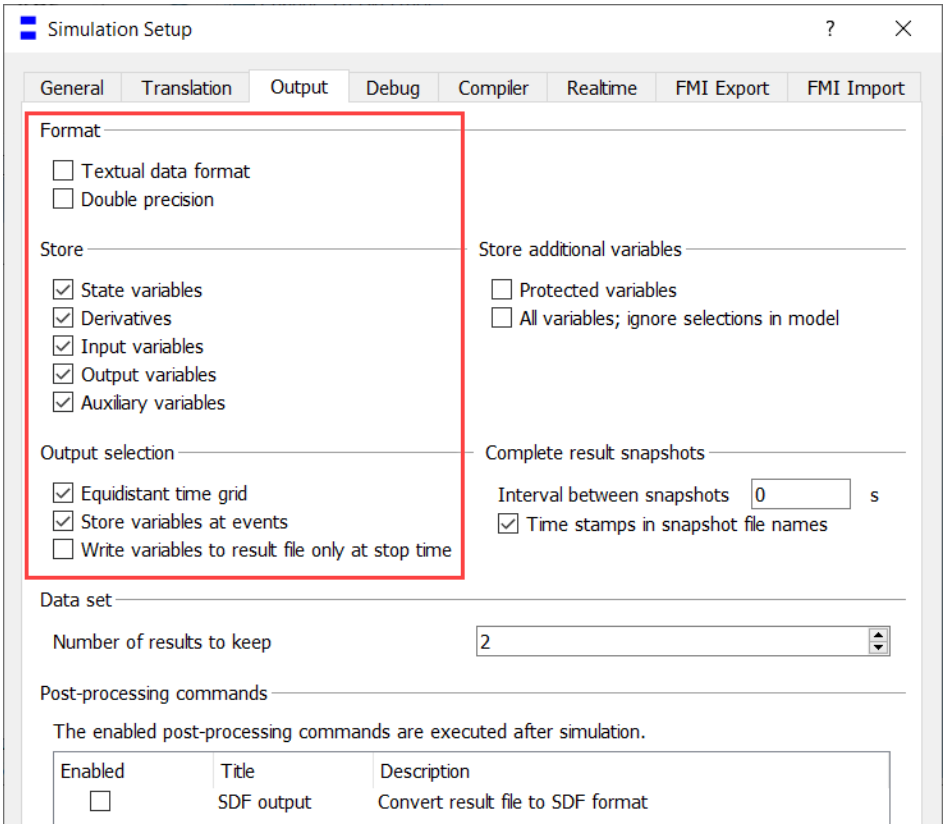
- Using **Store in Model**, the settings **Generate listing of flat Modelica code in .mof file** and **Generate listing of translated Modelica code in dsmodel.mof** are also included. The reason that they are not included in the new feature is that they do not influence the generated C-code.
- Using **Store in Model**, the setting **Generate Analytic Jabobian for the ODE problem** is not included. Using the new feature, it is included.

Note also that **Translation settings** is also included in the option **All Settings**, as before; if you activate **All Settings**, the **Translation settings** is grayed out:



Output settings.

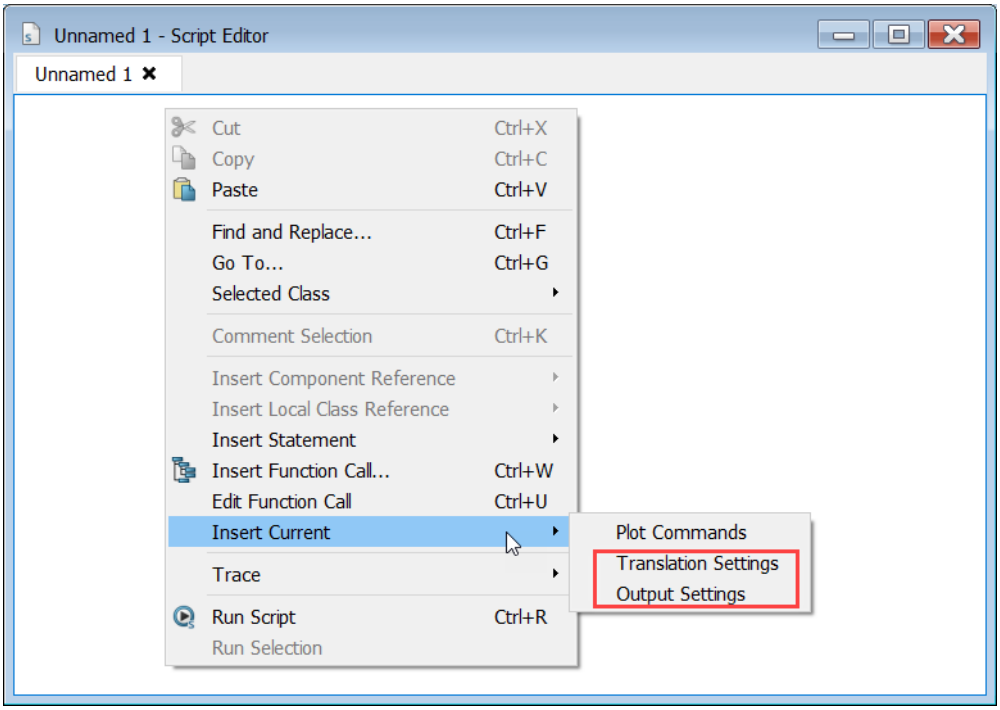
Activating **Output settings** will include the built-in function `experimentSetupOutput` in the script; this built-in function includes the settings in the groups **Format**, **Store**, and **Output selection** in the **Output** tab of the simulation setup, reached by the command **Simulation > Setup**, the **Output** tab – the settings framed:



Note that this setting is not included in the setting **All Settings**, this is a new option.

New options when generating a script by the script editor

The script editor has been enhanced with two new context commands:



These commands fully corresponds to the new options in the generate script command described above.

Scripting support for plotting bar charts and area charts

Plotting of bar charts and area charts, described in section “Plotting bar charts and area charts” on page 14, is also supported by scripting; the marker styles for bar chart and area fill are present as `MarkerStyle.BarChart` and `MarkerStyle.AreaFill`, in the same way as other marker styles.

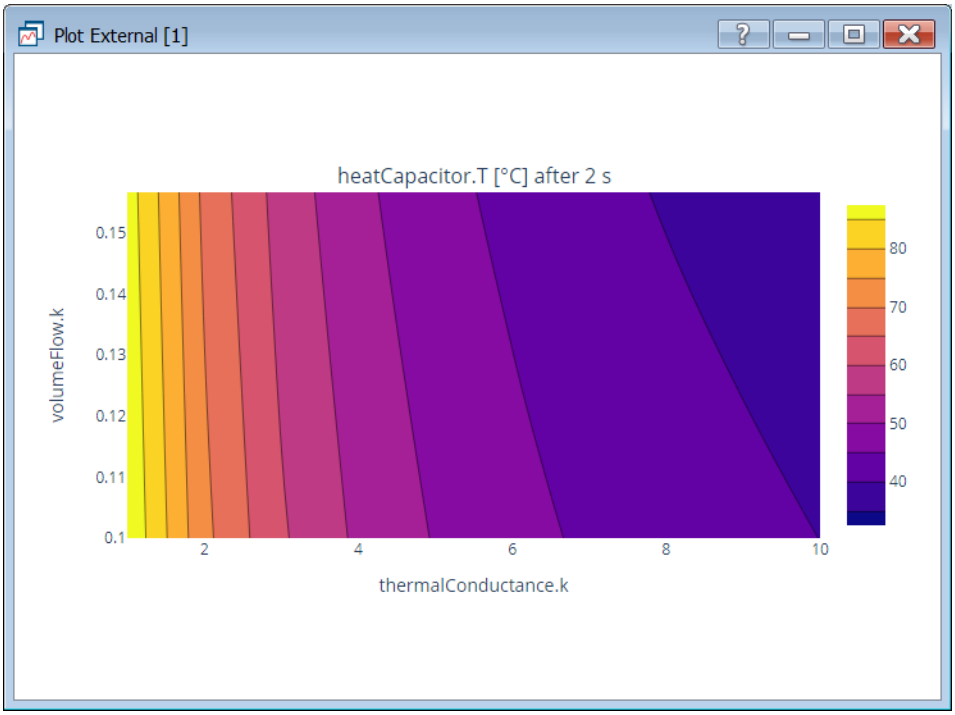
Scripting support for plotting of external files

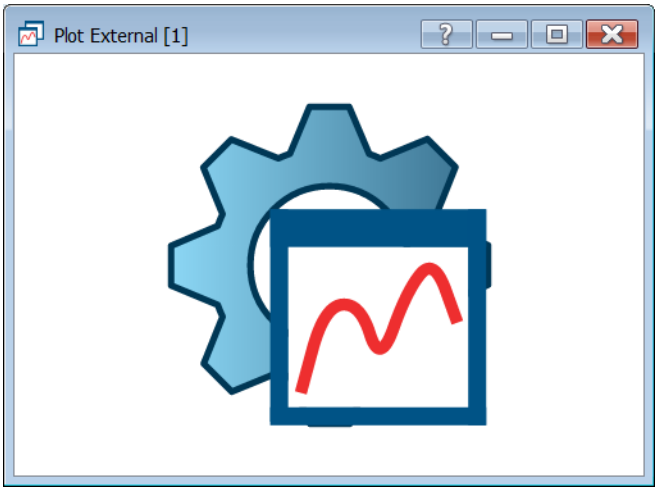
Dymola 2022 supports plotting of external files in the HTML, SVG, or PNG format. In addition, HTML plotting scripts generated in Python by for example matlabplot or plotly are supported.

The files can be plotted by the new built-in function `plotExternal`:

```
function plotExternal "Show external document or image"
   input String filename "Path to HTML, SVG or PNG file";
   output Integer window_id;
end plotExternal;
```

Examples of plotted files, from a plot file generated by plotly, and an SVG image:





The result of the built-in function supports:

- The built-in functions `plotTitle` and `plotDocumentation`
- Exporting an image by the command **Tools > Image** or the built-in function `ExportAsImage`
- Copying an image to clipboard by, for example, the command **Tools > To Clipboard**

- Printing the result

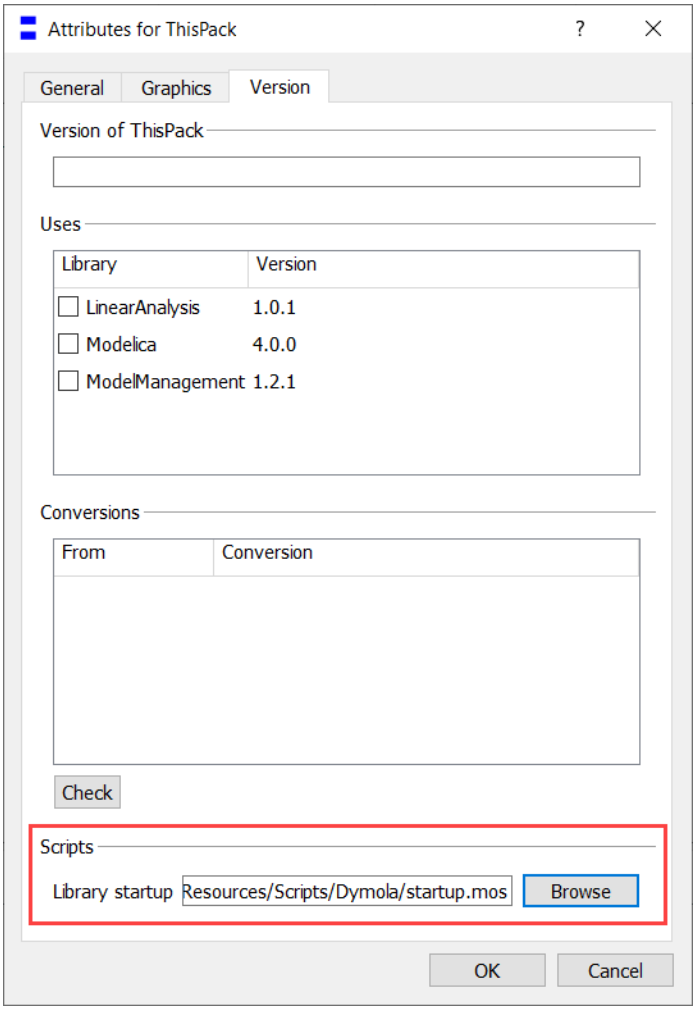
Library startup script

You can now specify what startup file you want to use for a top-level package, by using the annotation `__Dymola_startup`. An example:

```
package ThisPack
  annotation(__Dymola_startup =
  "modelica://ThisPack/Resources/Scripts/Dymola/startup.mos");
end ThisPack;
```

If `__Dymola_startup` is specified, that value is used, otherwise the default value, that value is specified in the example above.

To browse for the startup file to use, you can use the command **Attributes > Version**:



Note that the above figure is the default for any new package you create, you only have to change the path here, and then the corresponding annotation is created automatically.

New optional argument in the built-in function removeResults

The built-in function `removeResults` could in previous versions of Dymola only be used to close all simulation result files, removing them from the variable browser.

In Dymola 2022 the new argument `input String results[:]` can be used to specify which results to close. If keeping the default value of the new argument, the built-in function will work as in previous versions, closing all result files.

Improved built-in function translateModelFMU

The built-in function `translateModelFMU` now contains a new argument for handling filtering of signals when exporting an FMU:

```
input String includeVariables[:] = fill("", 0) "Variables in
model to be included in the xml, if empty all that passes
selected filters will be used";
```

The default value is an empty array. If this is not changed the variables included are the same as in previous versions, where this option to select variables was not present.

If however variables are added in this array these variables will be included in the generated FMU.

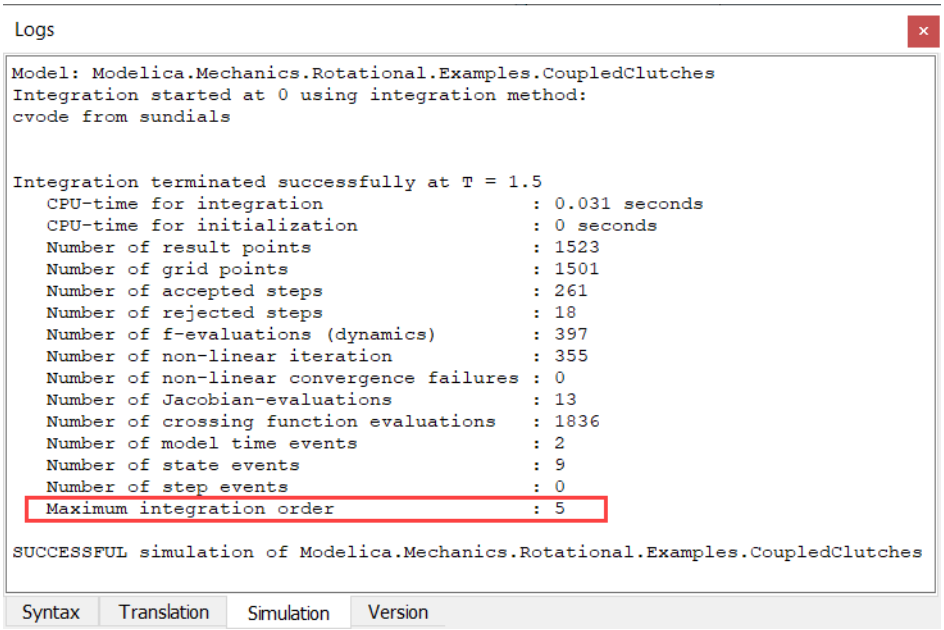
If variables are added that are not in the model, you will get warnings when exporting the FMU.

These improvements of the built-in function `translateModelFMU` corresponds to the new GUI when exporting an FMU, see section “Filtering individual signals when exporting an FMU” starting on page 38.

3.3.6 Improved simulation logging

Maximum integration order in Cvode simulation log

In Dymola 2022, the maximum integration order is displayed in the simulation log for the Cvode solver. An example:



Notes:

- The above is also the case for Cvode in Co-simulation FMUs.
- For other solvers, the maximum integration order is already displayed in previous versions.

3.3.7 Minor improvements

Improved steady-state initialization

The option to generate code for steady-state initialization, activated by setting the flag

```
Advanced.Translation.DefaultSteadyStateInitialization = true;
```

has now a slight improvement to make it easier to apply it to models that were not intended for steady-state initialization.

If you set

```
Advanced.Translation.DefaultSteadyStateInitializationLevel = 1;
```

All start values in the model are ignored. In general, `Advanced.Translation.DefaultSteadyStateInitializationLevel` means that all start values at that level or higher are ignored; where start-values directly in the actual model have level 1, and start-values directly in component models have level 2. (The default value of the flag is 0, meaning that the flag is not used.)

Minor changes in the plot context menus

Some minor changes are made in the plot context menus:

- The context menu for the plot window, without anything selected:
    - The entry **Setup…** has been added.
    - The entries **Layout** and **Plot Expression…** has been removed; they can be accessed from the tab.
- The context menu for a curve/legend:
    - The entry **Setup…** has been added.

3.4 Installation

For the current list of hardware and software requirements, please see chapter "Appendix – Installation: Hardware and Software Requirements" starting on page 45.

3.4.1 Installation on Windows

For the full list of supported compilers, see "Compilers" on page 46.

Cross-compilation for Linux on Windows

Introduction

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

Install WSL

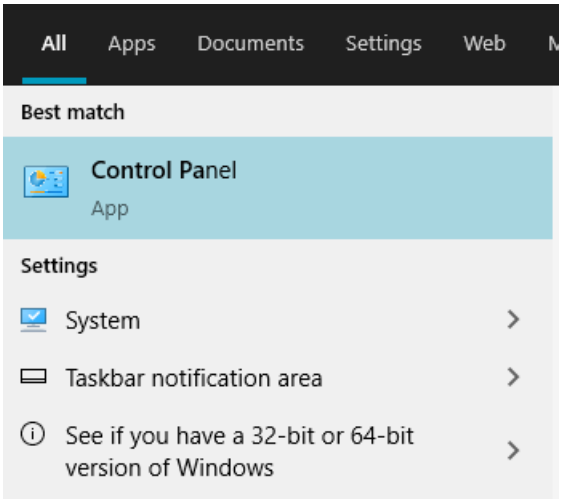
Installing WSL will give you a command-line Linux environment on your Windows computer. This environment can then compile code in a native Linux environment, such as Ubuntu 18.04 LTS as used as an example below.

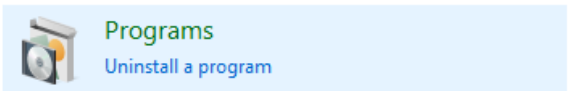
The WSL Linux environment can compile the generated model C code from Dymola in order to produce a Linux executable dymosim or a Linux FMU.

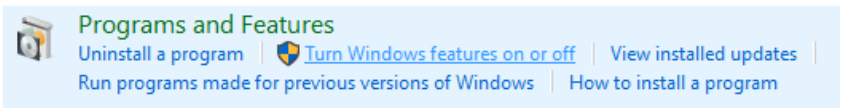
Prerequisites

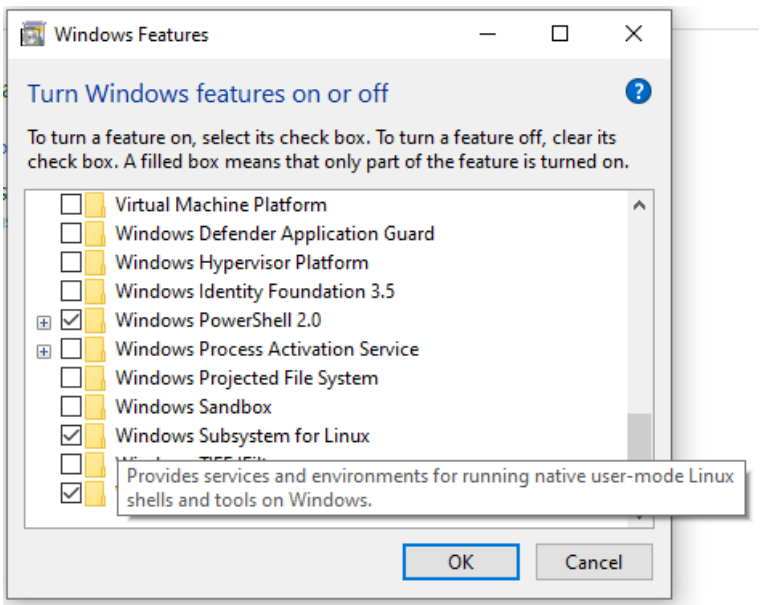
WSL is usually not enabled on Windows, so you need to enable WSL on your computer and install needed software components.

- Turn on feature Windows Subsystem for Linux.


All
Apps
Documents
Settings
Web
Best match
Control Panel
App
Settings
System
Taskbar notification area
See if you have a 32-bit or 64-bit version of Windows


Programs
Uninstall a program

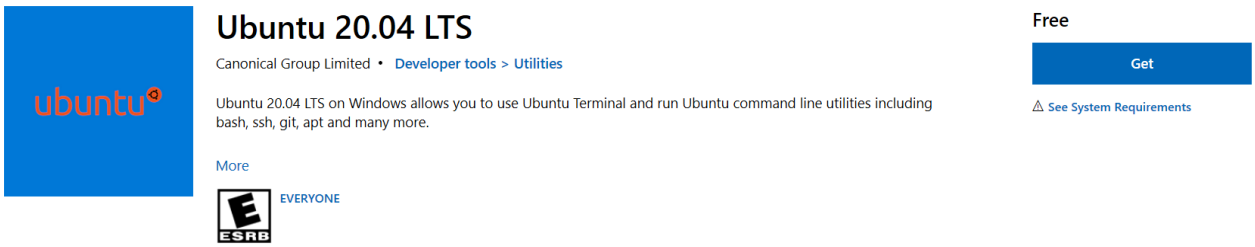

Programs and Features
Uninstall a program
Turn Windows features on or off
View installed updates
Run programs made for previous versions of Windows
How to install a program


Windows Features
Turn Windows features on or off
To turn a feature on, select its check box. To turn a feature off, clear its check box. A filled box means that only part of the feature is turned on.
Virtual Machine Platform
Windows Defender Application Guard
Windows Hypervisor Platform
Windows Identity Foundation 3.5
Windows PowerShell 2.0
Windows Process Activation Service
Windows Projected File System
Windows Sandbox
Windows Subsystem for Linux
Provides services and environments for running native user-mode Linux shells and tools on Windows.
OK
Cancel

Install Linux

Download a suitable Linux distribution, in our case Ubuntu 20.04 LTS, from Microsoft Store.

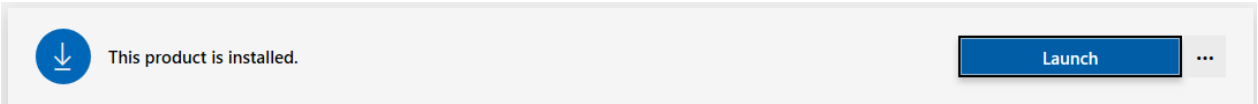
https://www.microsoft.com/en-us/p/ubuntu-2004-lts/9n6svws3rx71?activetab=pivot:overviewtab



Linux setup

Install Linux and do the initial setup, including installation of a C compiler.

- Install Linux by clicking the **Launch** button.



- Create user account when prompted.
- Check that you have the “dos2unix” feature installed in WSL, by giving the command:

```
which dos2unix
```

- If you don´t have it installed, install it by the command:

```
sudo apt install dos2unix
```

- Update your system and install the C compiler.

```
sudo apt-get update
sudo apt install gcc
sudo apt install g++
sudo apt install zip
```

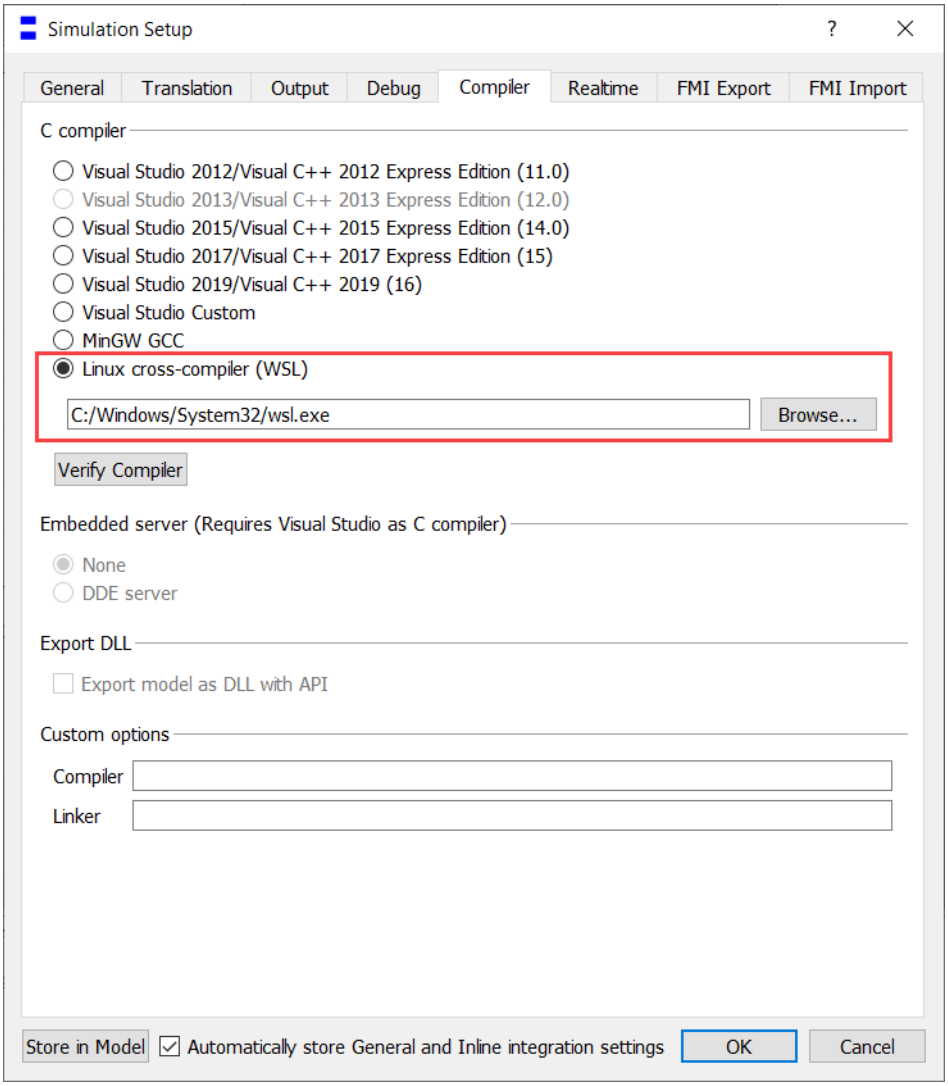
- Ensure that WSL can change file permissions. That can be done in WSL by ensuring that the file `/etc/wsl.conf` has the following two lines (and creating that file if it does not exist):

```
[automount]
options = "metadata"
```

Important! Reboot the computer after any changes of this file.

For more information about the general WLS installation, see also https://docs.microsoft.com/en-us/windows/wsl/install-win10.

To select the WSL installation, select **Linux cross-compiler (WSL)** in the simulation setup, reached by the command **Simulation > Setup**, the **Compiler** tab:



Now, to check you selection, click **Verify Compiler**. If you merely wish to use WSL as the main compiler, your setup is done here. For cross-compilation for FMU exports, see below.

The settings are saved between sessions.

FMU export for multiple platforms

Dymola partially supports cross-compilation when exporting FMUs on Windows. It is partial in the sense that your main compiler environment is limited to Visual Studio or MinGW and the cross-compiler environment is limited to WSL. So first, you need to select either Visual

Studio or MinGW in the Simulation setup. Then, to activate cross-compilation, set the flag `Advanced.FMI.CrossExport=true`. Now when exporting an FMU, you will get 64-bit Linux binaries in addition to Windows binaries.

Important! In this case, you should *not* select **Linux cross-compiler (WSL)**; the flag sets the suitable cross-compilation.

Note that the value of the flag is *not* saved between sessions.

Updated Qt version

Dymola 2022 is built with Qt 5.15.

Discontinued support for Intel compilers

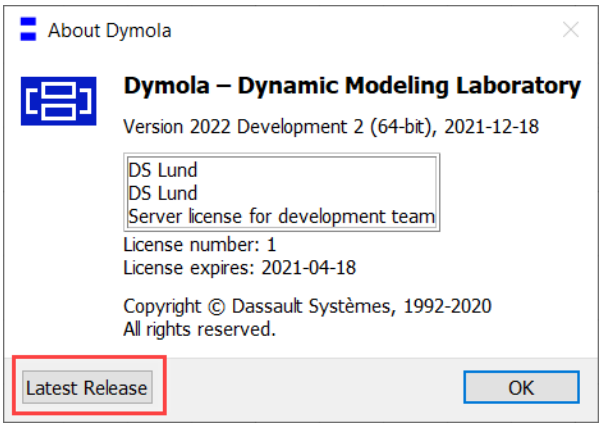
From this Dymola 2022 release, the support of Intel compilers is discontinued.

GCC compilers

From Dymola 2022, the support is discontinued for MinGW GCC compilers with versions lower than 5. For supported and tested versions, see “Compilers” on page 46.

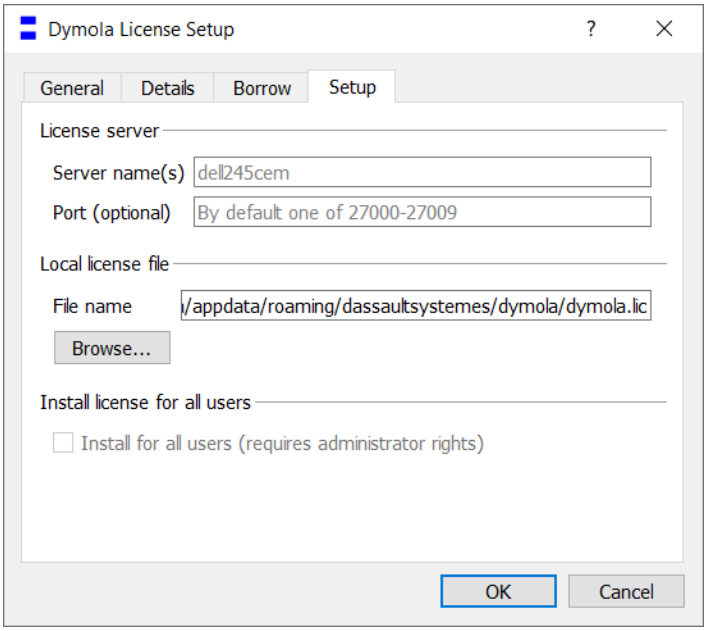
Getting information about the latest Dymola release

In Dymola 2022 you have a new command in the window opened by the command **Tools > About Dymola**, you can in that window click **Latest Release** to get information about the latest release available.



Current license server shown in license setup

To help debugging, the current license server is shown in gray text in the license setup reached by the command **Tools > License Setup**, the **Setup** tab:

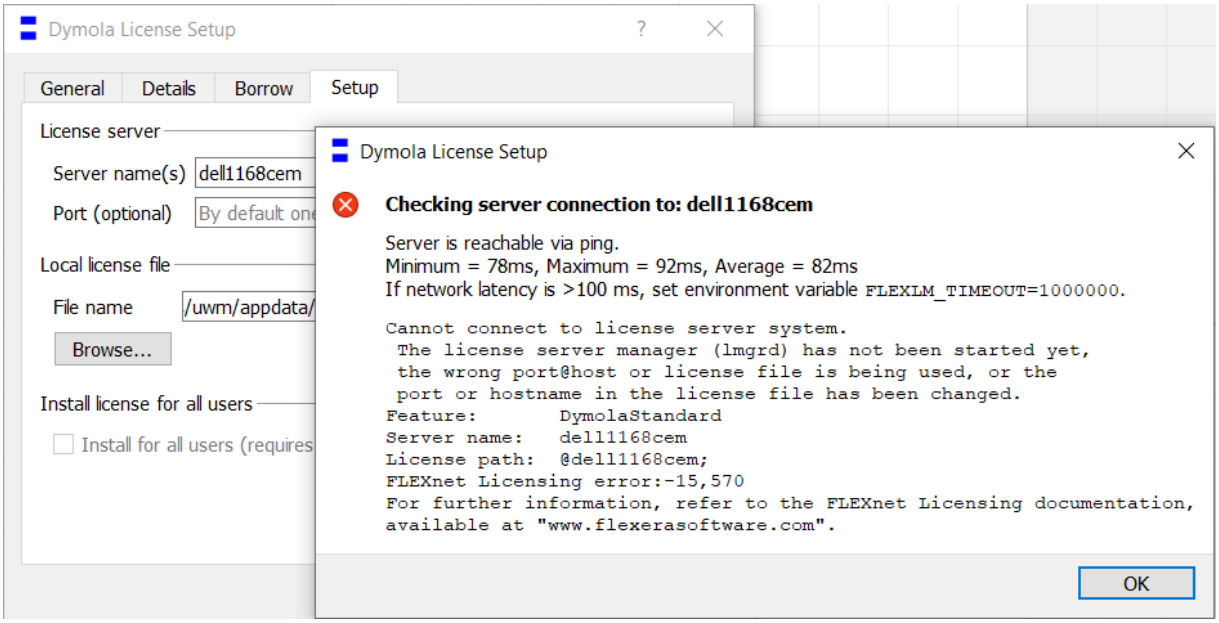


(You can still type in a new server name, of course.)

If the name cannot be detected, or if you have a nodelocked license, then nothing is shown.

Improved diagnostics when connecting to license server

When setting up a license server (by the command **Tools > License Setup**, the **Setup** tab) Dymola tries to contact the license server. If that fails, Dymola will attempt a "ping" to check if the network is alive, and if timing problems occur. The result is reported in the corresponding error message. An example:



Note the tip about the environment variable to increase the time of the network timeout, if needed.

3.4.2 Installation on Linux

Updated Qt version

Dymola 2022 is built with Qt 5.15.

Getting information about the latest Dymola release

See the corresponding section in “Installation on Windows”.

Current license server shown in license setup

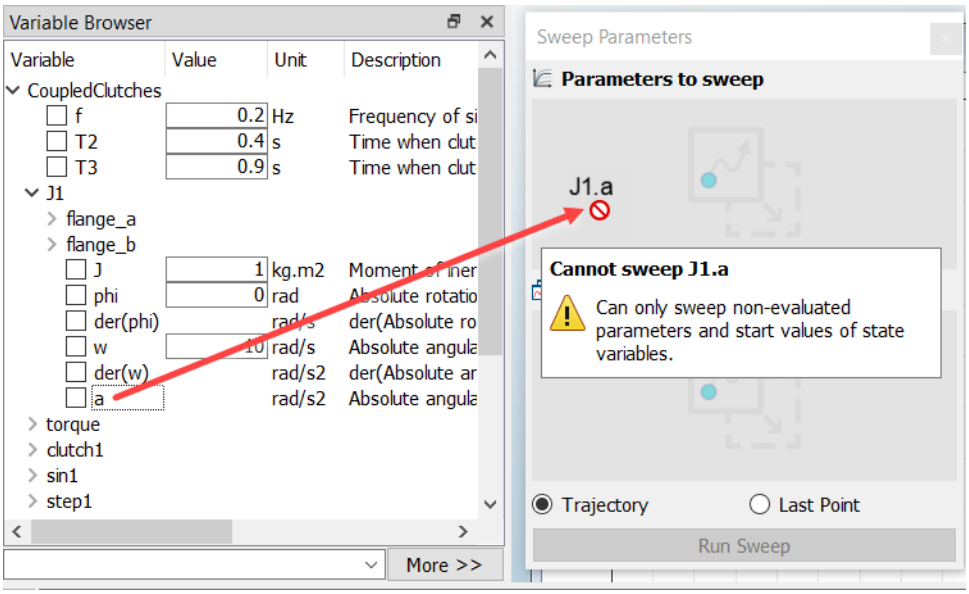
See the corresponding section in “Installation on Windows”.

3.5 Model Experimentation

3.5.1 Improvements for sweeping parameters

Improved indication for trying to sweep

In the previous version of Dymola, if you could not sweep a variable, it could not be added to the Parameters to sweep pane. In Dymola 2022, there is also a tooltip to indicate why:



3.6 Model Management

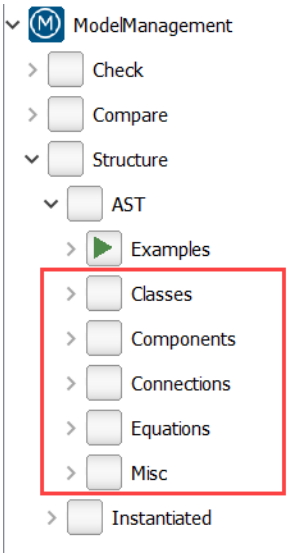
3.6.1 Improved model editing API

Introduction

The functions to create and edit Modelica models using function calls, located in the package `ModelManagement.Structure.AST` have been restructured and extended.

Improved AST package structure

The AST functions in the Model Management package have been grouped into five sub-packages for a better structure:



The five subpackages describe which part of a model that is edited, for example classes, components, connections, or other equations.

Automatic conversion

Automatic conversion is provided to upgrade existing code that uses the Model Management package to the new structure.

New AST functions

The following new functions have been added.

- New examples (in the **Examples** subpackage):

Name	Description
createNewModel	Creates a simple model using AST commands
makeReplaceable	Makes an existing component replaceable

- For editing classes (in the **Classes** subpackage):

Name	Description
CreateClass	Creates a new class with the given name
CreateExtends	Creates an extends in the designated class

- For editing components in a model (in the **Components** subpackage):

Name	Description
CreateComponent	Creates a new component in a class
SetComponentPlacement	Sets the graphical placement of a component in a model
SetComponentReplaceable	Set component replaceable with optional constraining type
SetComponentType	Sets the type of a component
SetAnnotation	Sets the annotation of a component

- For editing connections (in the **Connections** subpackage)

Name	Description
CreateConnection	Creates (or replaces) a connection without any annotation
DeleteConnnection	Deletes a connection between two connectors in the specified model
ConnectionsText	Returns the connections of the class as an array of text strings

- For editing equations (in the **Equations** subpackage)

Name	Description
CreateEquation	Creates one or more equations

3.7 Other Simulation Environments

3.7.1 Dymola – Matlab interface

Compatibility

The Dymola – Simulink interface now supports Matlab releases from R2016a (ver. 9.0) up to R2020b (ver. 9.9). 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 2022 officially supports the DS1005, DS1006, MicroLabBox, and SCALEXIO systems for HIL applications. For these systems, Dymola 2022 generated code has been verified for compatibility with the following combinations of dSPACE and Matlab releases:

- dSPACE Release 2016-A with Matlab R2016a
- dSPACE Release 2016-B with Matlab R2016b
- 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 R2019b and R2020a
- dSPACE Release 2020-B with Matlab R2019b, R2020a, and R2020b

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`, that 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. As a consequence, 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 R2016a (Simulink Real-Time ver. 6.4) to R2020b (Simulink Real-Time ver. 7.0). Only Microsoft Visual C compilers have been tested.

3.7.3 OPC communication

Discontinued OPC communication support

From the previous Dymola version, Dymola 2021x, OPC communication is not supported. For alternatives, please contact support: https://www.3ds.com/support.

3.7.4 Dymosim DLL

Last Dymola version supporting dymosim DLL

Dymola 2022 is the last version supporting dymosim DLL. From the next Dymola version, Dymola 2022x, dymosim DLL is not supported.

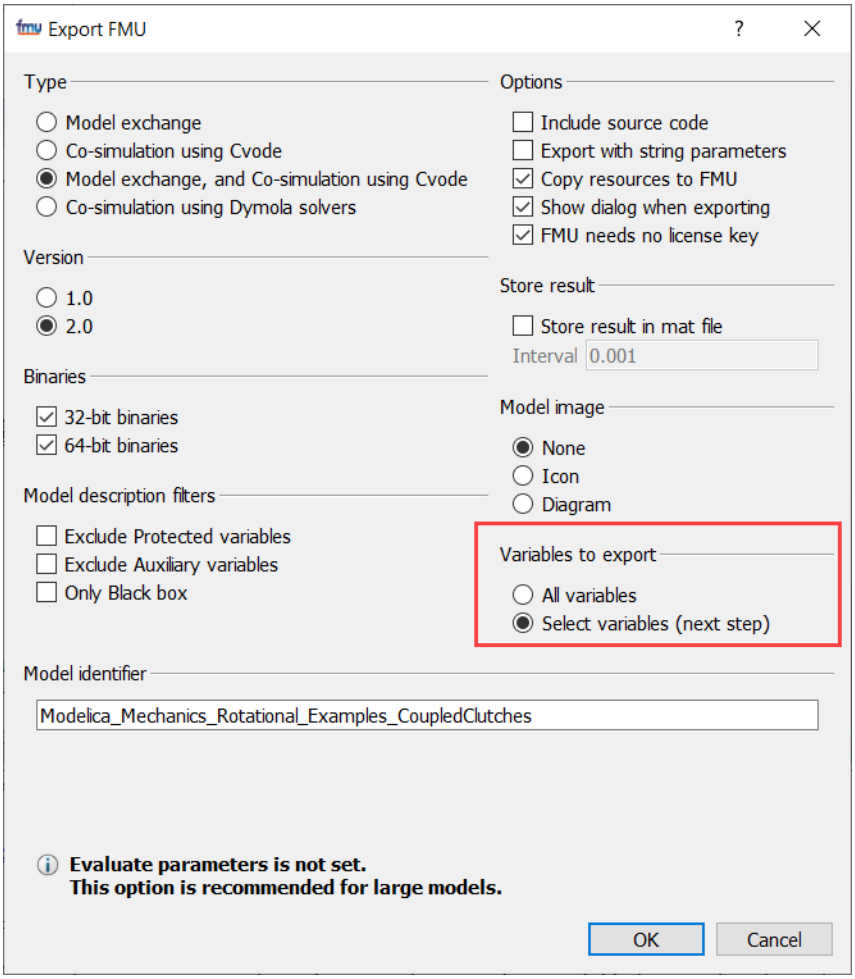
3.7.5 FMI Support in Dymola

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

FMU export

Filtering individual signals when exporting an FMU

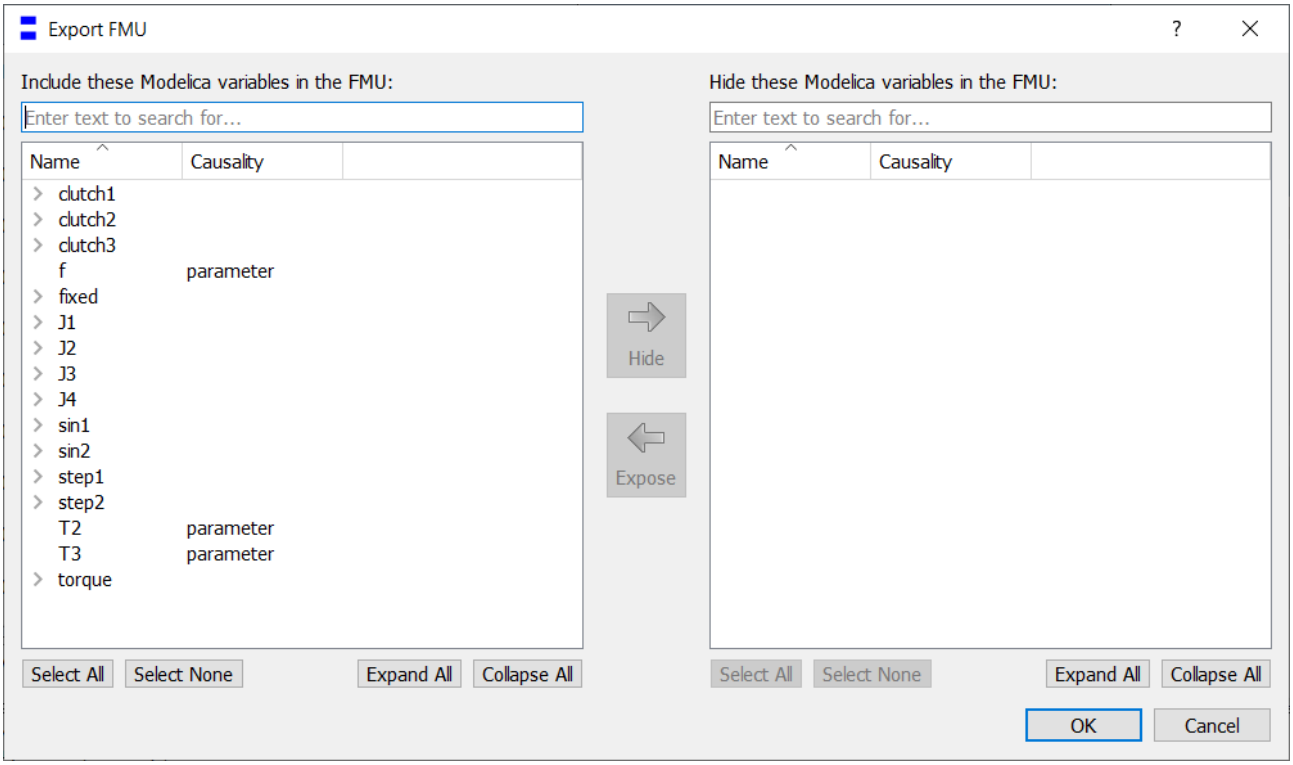
You can filter on individual signals when exporting an FMU. An example of a dialog to export an FMU from the demo Coupled Clutches:



There are two alternatives:

- **All variables**: This corresponds to the export possible in previous version of Dymola.
- **Select variables (next step)**: This is a new option in Dymola 2022, you can select what variables to expose or hide from the FMU in detail. Selecting this alternative and clicking **OK**, you will have a dialog for what variables to select.

An example of the dialog that appears when you select the second alternative and click **OK** may be (in this case the demo Coupled Clutches is exported as an FMU):



You select the variables you want to move to the other pane, and then click the arrow **Hide** or **Expose** depending on in which pane you want to have the variables, if they should be exposed or hidden.

For each pane, you have a search field on top and four buttons at the bottom: **Select All**, **Select None**, **Expand All**, and **Collapse All**. Note that it might be handy to, in the above figure, to click **Select All** and then click the **Hide** arrow, if you want to select which variables to expose rather than the ones to hide.

When this menu appears, it is by default adapted to what is selected in the **Model description filters** group in the previous menu.

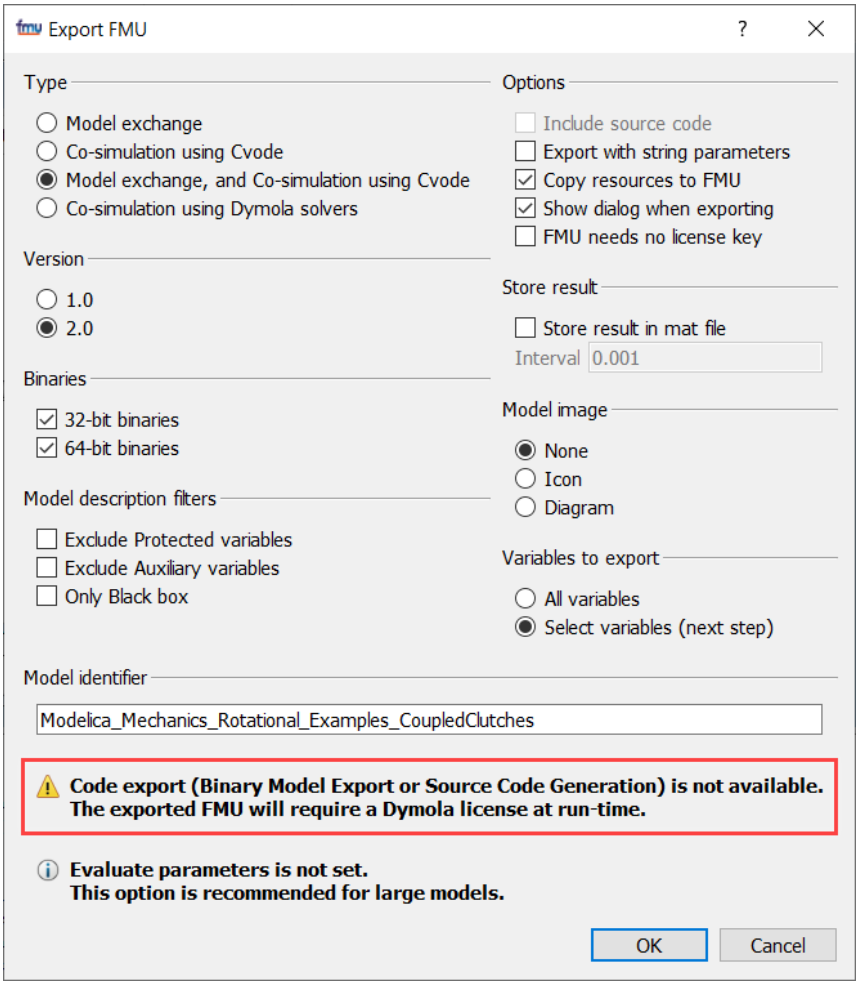
Note! Inputs cannot be deselected (hidden). They cannot be selected or moved; they are grayed out in the dialog above, if present.

This feature is also supported by scripting; see section "Improved built-in function translateModelFMU" starting on page 25.

Warning about missing code export license

If you export an FMU without any code export license, the usefulness of the generated FMU is greatly reduced. To export an FMU without any code license is easy to do if you have a small number of export licenses, because in that case people are usually required to disable code export while doing model development.

If you export an FMU without any code export license, you will get a warning:

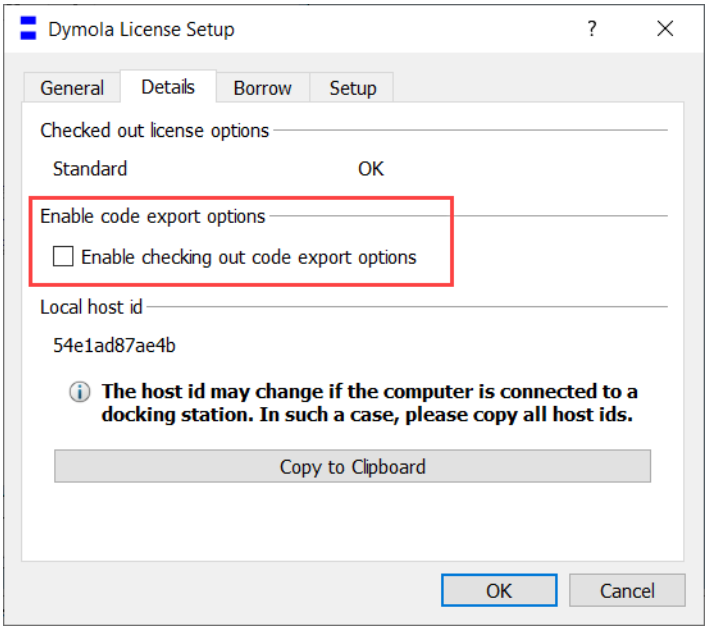


The warning is given under two conditions:

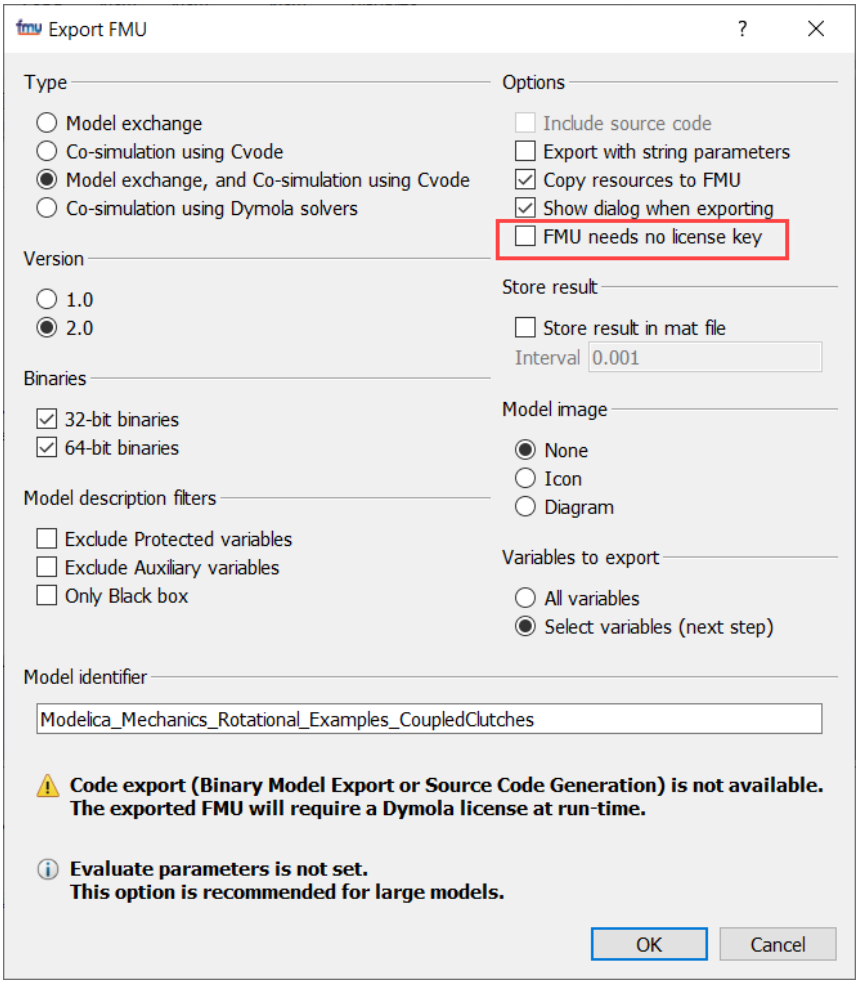
- The code export is disabled by selection
- If the code export is not disabled by selection, Dymola tries to check out an export option. If none is available (no such license, or license already used up by other users), the warning is shown.

Option to enable or disable the code export from the FMU export dialog

In previous versions of Dymola, the code export could be disabled by the dialog that was the result of the command **Tools > License Setup**, the **Details** tab. This is also the case in Dymola 2022, but the name of the setting has changed to **Enable checking out code export options**. (This also means that the default value has been changed to be activated):



Now also a corresponding option is also available in the dialog when exporting an FMU:



These options are working on the same feature, if you activate it in one dialog; it is activated also in the other dialog.

FMU export from multiple platforms

See section “Cross-compilation for Linux on Windows” starting on page 27.

3.8 Advanced Modelica Support

3.8.1 Support for Modelica Language version 3.5

Dymola 2022 is compliant with the new Modelica Language Specification version 3.5. The most important supported new features are:

- Public variables in functions must be input or output
- Hierarchical Evaluate
- Final for classes

Examples of other supported features are:

- Impure functions as function arguments
- Introduce ModelicaDuplicateString
- Element for convertElement
- Add SourceDirectory
- Test-case annotation for invalid models
- Dialog-annotation for classes
- Svg-bitmaps
- Unit for %par
- Deprecate byte-order mark
- Assertion-level should be a structural parameter

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

3.9 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.

3.10 New libraries

Below is a short description of new libraries. For a full description, please refer to the libraries documentation.

The libraries are presented in alphabetical order. If not stated as free, the library is commercial.

3.11 Documentation

New white paper: Exploring Model Structure with Equation Incidence

To investigate relationships between variables in the generated execution order of a translated model, “plot dependencies” if often used. From Dymola 2021x, the graphical complement “equation incidence” view is available.

The paper is available using the command **Tools > Help Documentation**, under the new section White Papers. It is also available here.

General note about the manuals

In the software distribution of Dymola 2022 Dymola User Manuals of version “March 2021” will be present; these manuals include all relevant features/improvements of Dymola 2022 presented in the Release Notes.

3.12 Appendix – Installation: Hardware and Software Requirements

Below the current hardware and software requirements for Dymola 2022 are listed.

3.12.1 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.

3.12.2 Software requirements

Microsoft Windows

Dymola versions on Windows and Windows operating systems versions

Dymola 2022 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 2022 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 2013 Express Edition for Windows Desktop (12.0)
- 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.

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 2013 (12.0)
- 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)

Intel compilers

Note!

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

GCC compilers

Dymola 2022 has limited support for the MinGW GCC compiler. The following versions have been tested and are supported:

- For 32-bit GCC: version 5.3, 6.3, and 8.2
- For 64-bit GCC: version 5.3, 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. **Note!** When migrating to Modelica Standard Library (MSL) version 4.0, MinGW gcc versions older than 5 are not guaranteed to work for source code export.
- 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.

Linux cross-compiler (WSL)

Dymola on window supports cross-compilation for Linux via the use of Windows Subsystem for Linux (WSL). 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.
- 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 11.14. This version is part of the Dymola distribution.)

As an alternative to FLEXnet, Dassault Systèmes License Server (DSLS) can be used.

Linux

Supported Linux versions and compilers

Dymola 2022 runs on openSUSE 42.2, 64-bit, with gcc version 5.3.1, 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 2022 it is Qt 5.15
- 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.

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

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 2022 application:

```
/opt/dymola-2022-x86_64/insert/dsbuild.sh
```

Dymola 2022 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 5.15.0 and thereby inherits the system requirements from Qt. This means:
    - Since Qt 5.15 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-5.15/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` and `libevent-2.0.so.5` 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 11.14.)

As an alternative to FLEXnet, Dassault Systèmes License Server (DSLS) can be used.