Integration with Dymola

On this page

Dymola overview

Dymola, an abbreviation for Dynamic Modeling Laboratory, is a complete tool for Modelica modeling and simulation of integrated complex systems used with automotive, aerospace, robotics, process and other applications. You can download and install Dymola from https://www.3ds.com/products-services/catia/products/dymola/ and set up your modeling tool to allow integration between Simulation (2021x and later) and Dymola , which can be specified as a language of Opaque expressions.

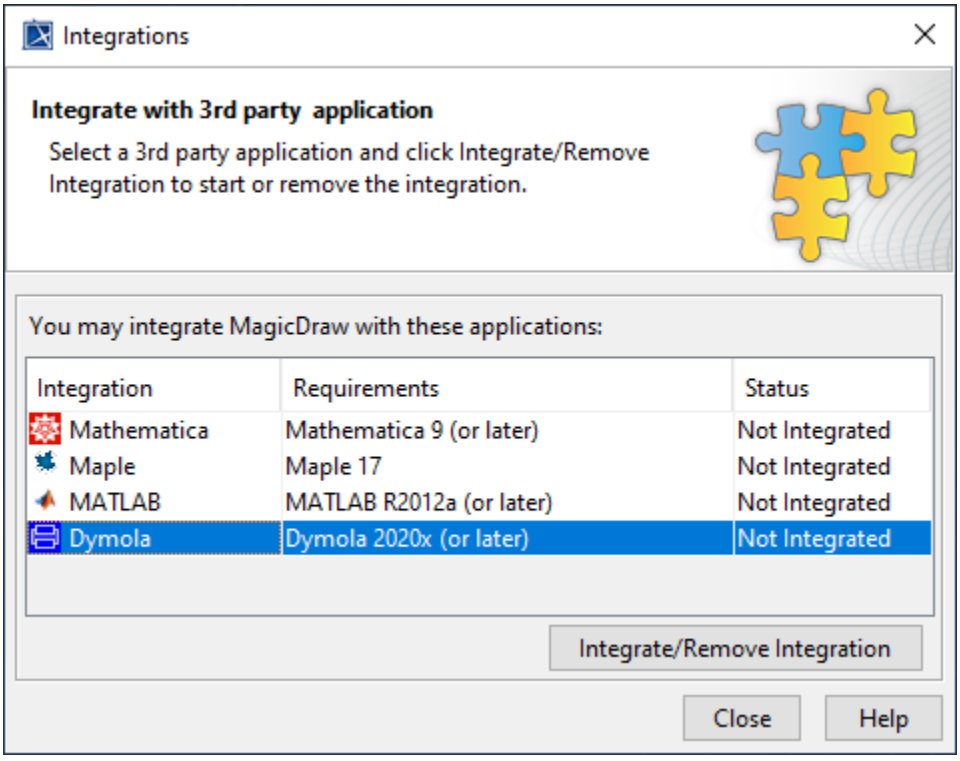
Integrating Simulation with Dymola

After successful integration, Dymola will be listed as available for language selection, e.g., through the language option in the **Simulation Console** pane. Simulation will then be able to process Modelica commands from the **Simulation Console** pane as shown in the image below.

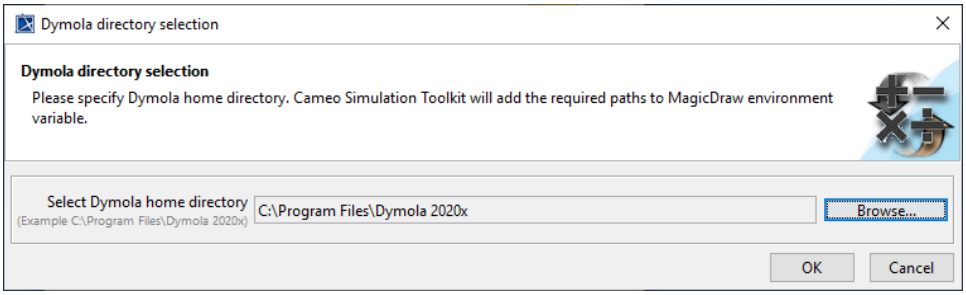
To integrate Simulation with Dymola

---

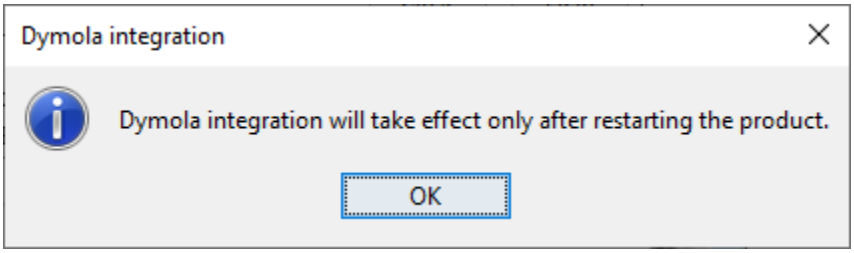
1. On the modeling tool main menu, select **Tools** > **Integrations**. The **Integrations** dialog opens.



2. Select **Dymola** (the **Not Integrated** status by default) and click **Integrate/Remove Integration**.
3. The **Dymola directory selection** dialog opens. Click **Browse** to select the Dymola home directory and click **OK**.



4. The **Dymola integration** dialog will be shown. Click **OK**. The Dymola integration status will change to **Integrated**.



5. Restart the modeling tool to complete the integration.

> **Note**
>
> - Dymola integration does not support Linux or Mac operating systems (only Windows).
> - For Teamwork Cloud projects, the model file must be attached only to the project so that the working directory of Dymola will be automatically set to the temp folder of the modeling tool, e.g., *C:\Users\ABC1\AppData\Local\.magicdraw \2021x\tmp\2020-11-23-13-32-34\attachedFiles*.

Using Simulation with Dymola through the scripting language

After installing Dymola, you can use Simulation with Dymola through the scripting language to call a Modelica model via two built-in Dymola commands: *simulateExtendedModel* and *simulateMultiExtendedModel*.

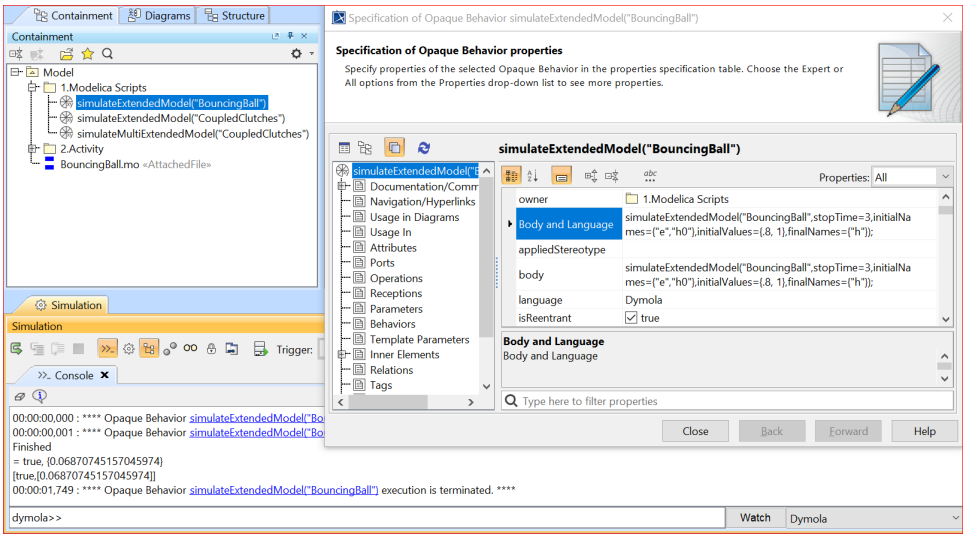
To use Simulation with Dymola through the scripting language (Steps 4 and 5 are optional)

---

1. Drag a Modelica model file, e.g., *BouncingBall.mo*, into the modeling tool to attach a Modelica model to the project.
2. In the **Console** pane, select the language list box and choose **Dymola**. The input prompt will be changed to *dymola>>*.
3. Enter the command below and press **ENTER**:

```
simulateExtendedModel("BouncingBall", stopTime=3, initialNames={"e", "h0"}, initialValues={.8, 1},
finalNames={"h"});
```

   The **Starting Math Engine** dialog appears shortly, and results will be shown in the **Console** pane.
4. Create an Opaque Behavior to call the model using the following tags:
   - *language*: Dymola
   - *name* (optional): e.g., *simulateExtendedModel("BouncingBall")*
   - *body*: Use the command from Step 3.
5. Run the Opaque Behavior. The results will be displayed in the **Console** pane as shown below.



Results from the simulateExtendedModel of Dymola called from an Opaque Behavior.

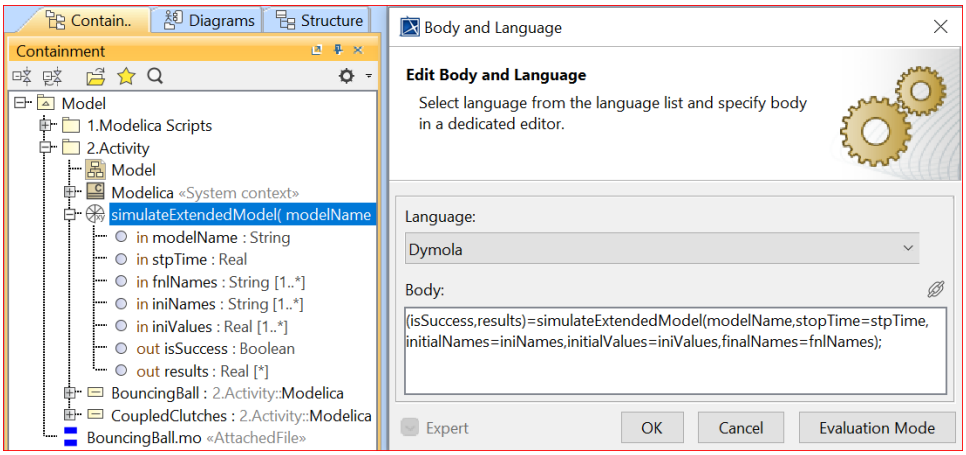
Using Simulation with Dymola through an Activity diagram

Alternatively, you can use Simulation with Dymola through an Activity diagram with parameters to call a Modelica model through Dymola built-in commands.

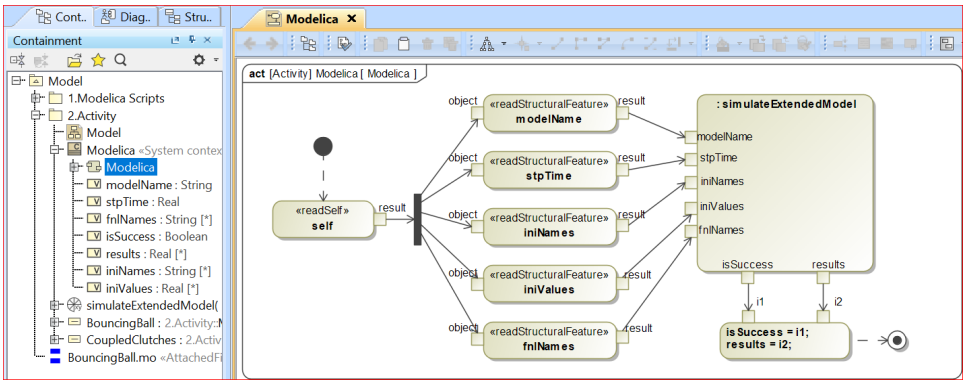
To use simulation with Dymola through an Activity diagram

---

1. Create an Opaque Behavior to call the model with the following tags:
   - *language*: Dymola
   - *name* (optional): e.g., *simulateExtendedModel*
   - *body*: *(isSuccess,results)=simulateExtendedModel(problem,stopTime=stpTime, initialNames=iniNames,initialValues=iniValues, finalNames=fnlNames);*
   - *parameters*: create parameters according to the command with valid Types and Multiplicities, e.g., *in modelName : String; in stpTime : Real; in fnlNames : String [1..*]; in iniNames : String [1..*]; in iniValues : Real [1..*]; out isSuccess : Boolean; out results : Real [*].*



2. Create a Call Behavior Action in an Activity diagram of a Block and create valid in/out Parameters through in/out Pins according to the parameters of the Opaque Behavior. You can also create a Block, e.g., *Modelica*, with value properties to keep values and set an Activity as the Classifier Behavior of the Block as shown below.



3. Create an Instance Specification to specify parameter values, e.g., *CoupledClutches*. Run the simulation through the instance to get results from Dymola back to Simulation as shown below.