

WORKSHOP 1.2

Building, simulating and analyzing a physical system

Contents

Introduction.....	1
Loading TrainingPack.....	1
Heating system	2
Building the model.....	2
Media propagation	4
Boundary Conditions	5
Simulate	6
Using stickies for model interaction.....	7
Creating and using views	8

Introduction

In this workshop, we will assemble a simple heating system and parametrize it to get to know the modelling interface. We will also be doing some plotting as well as using stickies for model input.

Loading TrainingPack

In this workshop, we will utilize a training package prepared in advance. It is called **TrainingPack.zip** and should have been provided to you by your course leader. Follow the instructions below to upload it.

- Start the import by clicking the upload button in the *Library Browser*:

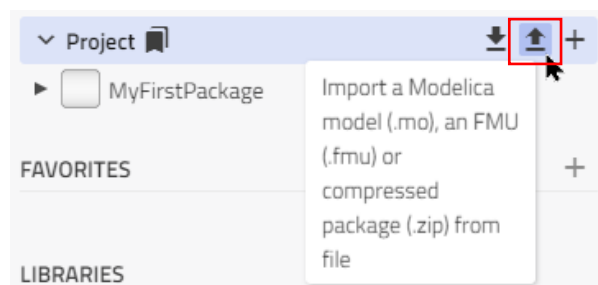


Figure 1 Import Modelica package

- Select the supplied **TrainingPack.zip** file.

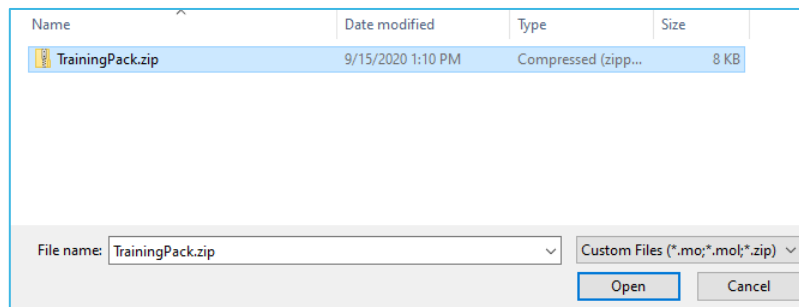


Figure 2 Load TrainingPack

- The package should now be ready to use.

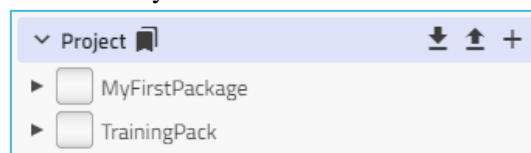


Figure 3 TrainingPack successfully loaded

Heating system

Building the model

In this exercise, a simple heating system will be implemented. To save the course participant some time, some of the components used in the exercise have been pre-parameterized and stored in the training package.

- Create a new sub-package called **Experiments** in **TrainingPack.W2.HeatingSystem**. There are two ways to do this from the UI:
 - Click on the **+** button to *Create a new class*, input **Experiments** as the *Name*, choose *Class specialization* as **package** and choose *Package* to be inserted in as **TrainingPack.W2.HeatingSystem**.

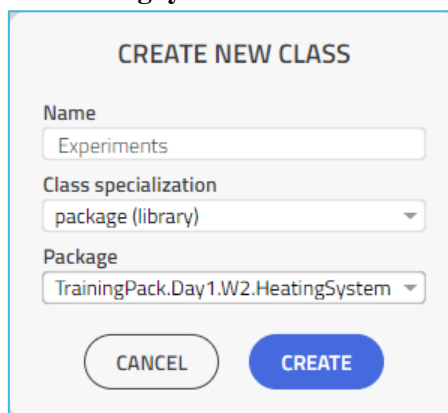


Figure 4 Create a new sub-package (Method 1)

- Right-click on **TrainingPack.W2.HeatingSystem**, select *New package* and provide the *Name* as **Experiments**.

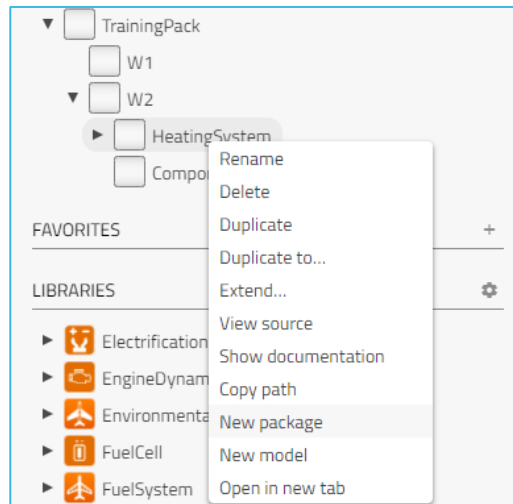


Figure 5 Create a new sub-package (Method 2)

2. Create a new model called **HeatingSystemExperiment** in **TrainingPack.W2.HeatingSystem.Experiments**.
3. Drag one instance of each class from **TrainingPack.W2.HeatingSystem.Components** onto the canvas.
4. Connect them according to the schema below in **Figure 6**:

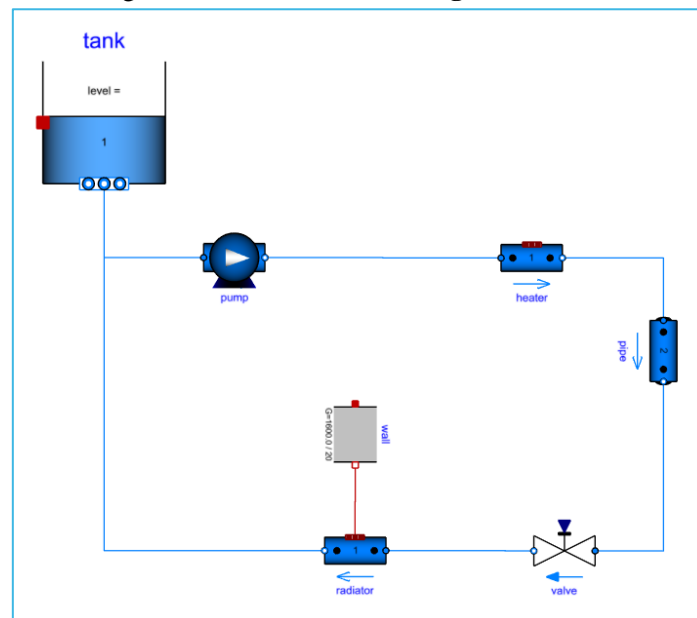


Figure 6 Coupling schema of heating system

5. To be able to easily access interesting results from the simulation, we will add sensors to the model. Drag the following sensors on to the canvas:
 - a. 2x **Modelica.Fluid.Sensors.Temperature** – name them **sensor_T_forward** and **sensor_T_return**
 - b. **Modelica.Fluid.Sensors.MassFlowRate** – name it **sensor_m_flow**
 - c. **Modelica.Fluid.System**
6. Connect them to the system according to the schema in **Figure 7**:

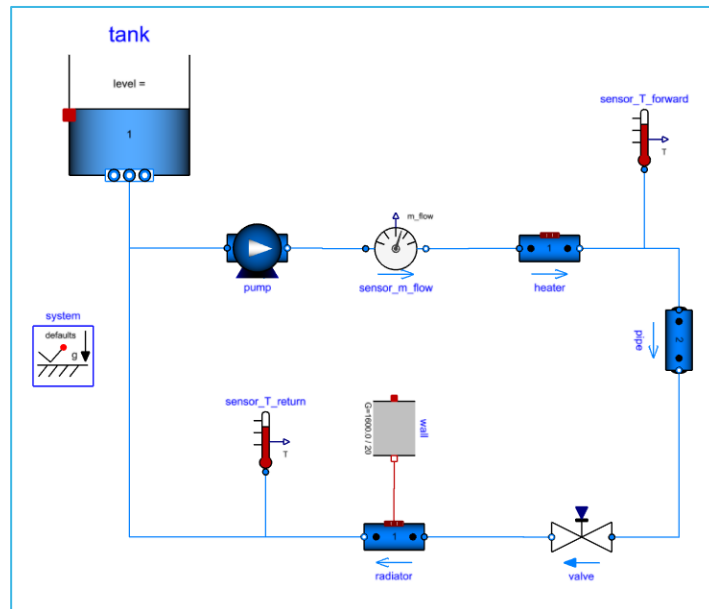


Figure 7 Connect sensors and add system component

Note: Components can be inserted between existing connections by dropping them on top of the connection line, see **Figure 8**.

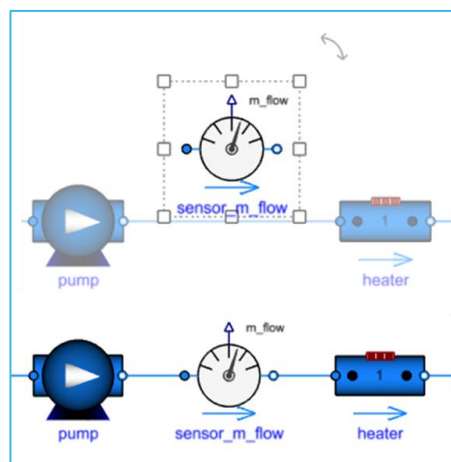


Figure 8 Insertion of component between existing connection

Media propagation

We need to define the media model to be used inside all components. It is important that all of the connected components are using the same medium in order to get consistent calculations. In Impact, there is a convenience function that simplifies this procedure.

7. Start by defining the media model to be used in one of the components, for example the **tank** model. Choose **Modelica.Media.Water.StandardWater** (StandardWater) as the medium.
Tip: You can quickly find the media model you want by starting to type in the field, see **Figure 9**.

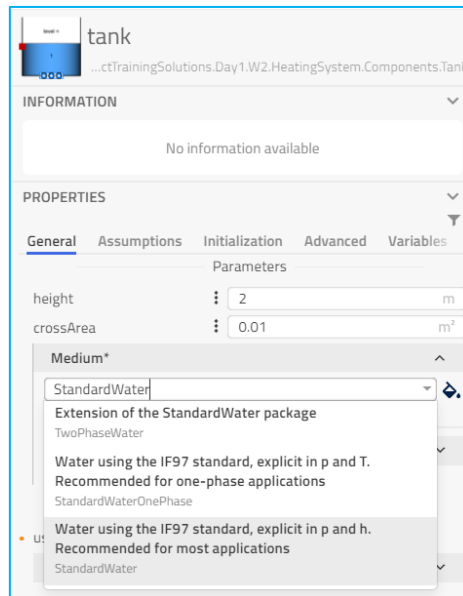



Figure 9 Redeclare media model

There is a setting in the tool settings called “**Enable automatic propagation**” which, if enabled, will make sure you are assigning the same media properties to each component in the fluid system. If you change the media in one component, Impact will automatically change it in all components connected to the particular component. You can turn off this behavior in the settings panel, and you will see the  button as in Figure 9 instead. If you click that, you will manually enforce all connected components to have the same media.

Boundary Conditions

The model still misses boundary conditions. In this last phase we will add appropriate boundary conditions. In addition to that, we also need to define a **system** object that is used in all fluid components in this model to keep track of global parameters.

8. Add the following components to the canvas:
 - a. *Modelica.Thermal.HeatTransfer.Sources.FixedHeatFlow*, name it **burner**
 - b. *Modelica.Thermal.HeatTransfer.Sources.FixedTemperature*, name it **T_ambient**
 - c. *Modelica.Blocks.Sources.Step*, name it **valve_handle**
9. Connect them according to the schema in **Figure 10**:

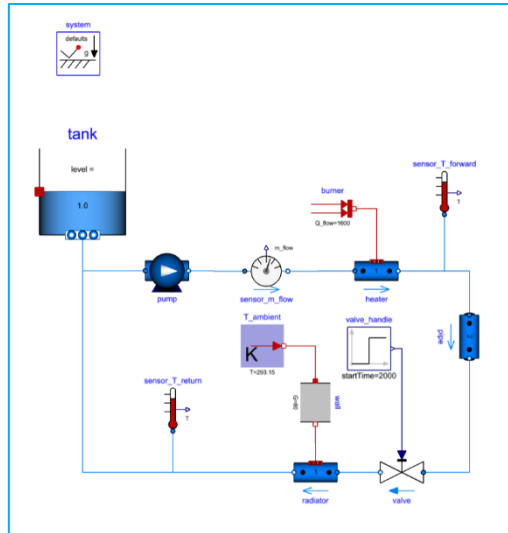


Figure 10 Connect the boundary conditions to the model

10. Parametrize the added components according to **Table 1**:

Component:	Parameter:	Tab:	Value:
burner	Q_{flow}	-	1600 [W]
burner	T_{ref}	-	343.15 [K]
burner	α	-	-0.5 [1/K]
T_ambient	T	-	system.T_ambient [K]
valve_handle	height	General	0.9
valve_handle	offset	General	0.1
valve_handle	startTime	General	2000 [s]
system	energyDynamics	Assumptions	SteadyStateInitial
system	m_{flow_small}	Advanced	0.0001 [kg/s]

Table 1 Parametrization

Simulate

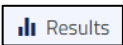
Now the model is ready to simulate.

11. Switch to the **Experimentation** mode.

12. Set **Stop Time** = **6000s**.

13. Hit the simulate button (▶).

Now we are ready to inspect the results.

14. Go to the **Results** mode  and create 3 plots:

- One with **sensor_T_forward.T** and **sensor_T_return.T** and **system.T_ambient**
- One with **tank.level**
- One with **sensor_m_flow.m_flow**

15. Rename them to something appropriate by clicking the name in *Details Panel* and typing in the new name.

16. The temperature unit can be converted to °C from the the **Units** tab in settings (⚙) – (Top Right corner of the mode selector toolbar), see **Figure 11**:

Unit	Conversion	Significant Digits	Scientific Notation
Default		6	Auto
1/K	1/K	6	Auto
A	A	6	Auto
C	C	6	Auto
F	F	6	Auto
kg	kg	6	Auto
J	J	6	Auto
J/kg	J/kg	6	Auto
J/(kg·K)	J/(kg·K)	6	Auto
K	K	6	Auto
kg/m³	kg/m³	6	Auto
kg/s	kg/s	6	Auto
m	m	6	Auto
m/s	m/s	6	Auto
m²	m²	6	Auto
m³/s	m³/s	6	Auto
m³	m³	6	Auto
m³/s	m³/s	6	Auto
N	N	6	Auto

CANCEL SAVE

Figure 11 Switch display unit

The plots should look like **Figure 12** now:

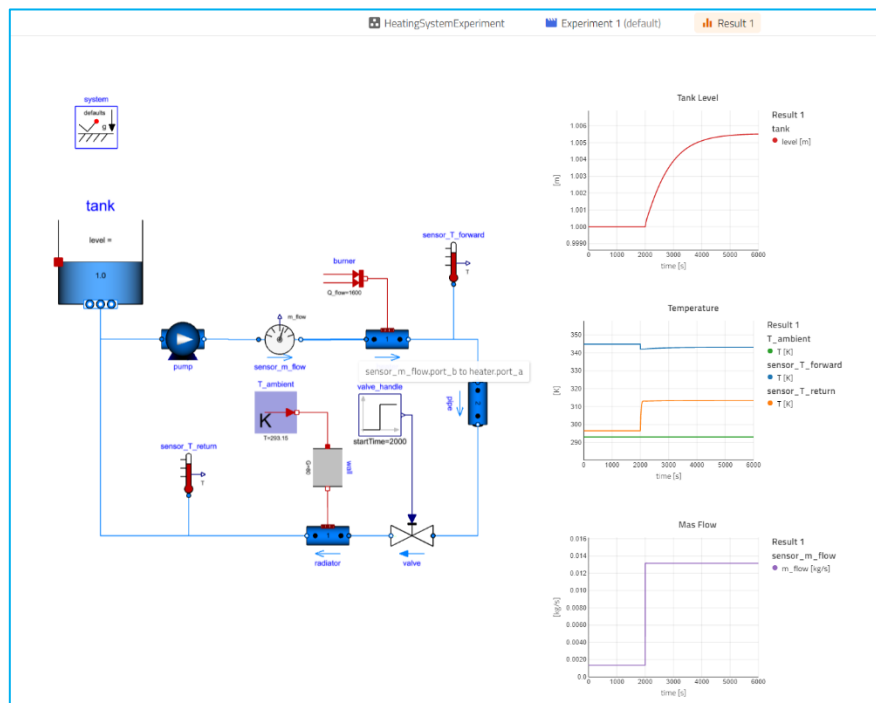


Figure 12 Plot of simulation of heating system

Using stickies for model interaction

In this section, we will investigate what happens if we change the ambient temperature of the system. To make it simple for the user to change a specific parameter, we will add a sticky so that the user can input the ambient temperature from the modelling canvas.

17. Select the system component and click the sticky-icon (📌) next to **T_ambient**. A sticky will appear under the system component on the canvas.
18. Change **T_ambient** to 0 °C in the sticky and re-simulate, the result should look similar to **Figure 13**:

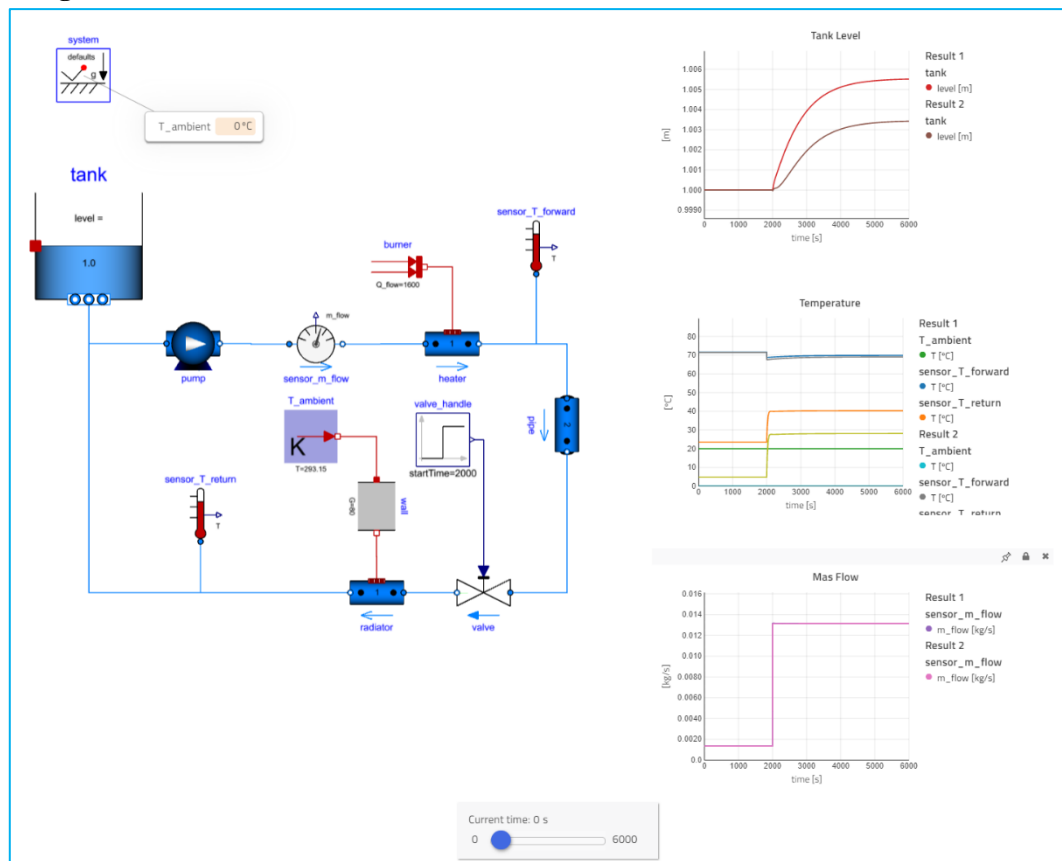


Figure 13 Lower the ambient temperature to 0°C

Creating and using views

Visualization of results is key to extracting value from the simulation of models. Impact allows users to create and save user-defined combinations of plots and stickies as views. There are two types of views in Impact – *Component views* and *System views*. Creating and using such views will become clear to you in this section.

19. **Figure 13** shows a combination of three plots and a sticky for our system model. To save this view for future use, hover the view button (👁) and save the view with **Save as...**, see **Figure 14**; let us name the view **MySystemView**. This is a *System view* as it includes variables or parameters from various components in the system.

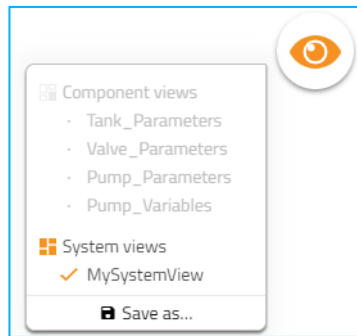


Figure 14 Saved views

20. It is possible to create multiple views and it is also possible to toggle views on or off. Active views will have a ✓ icon next to them. Let us turn off the view **MySystemView** by clicking on it.

21. Views can also be defined for individual components to input key parameters or monitor important variables. Some components (**Pump**, **Valve** and **Tank**) used in the **HeatingSystemExperiment** model were provided to you with pre-configured views.

Component views can be enabled in two ways:

- Hovering over the view button (👁) and toggling them on (see **Figure 14**)
- Selecting individual components with pre-defined component views, hovering over the view button for that component and toggling them on. See **Figure 15** below.

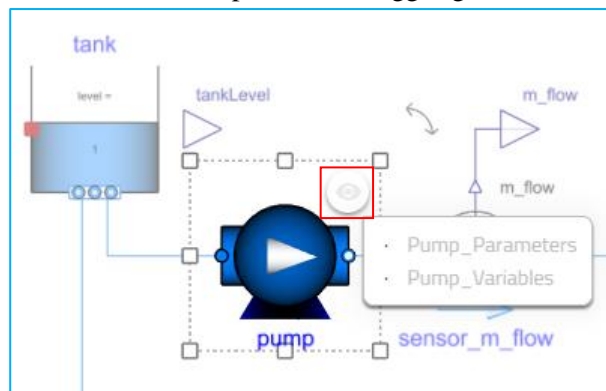


Figure 15 Component views

Turn on the four available component views. They should provide you with stickies for some important parameters/variables in the respective components.

22. It is possible to define component views for classes in **Global Libraries**, albeit only for the current workspace. Let us create a component view for *burner*.

- Right-click on the component *burner* in the model **HeatingSystemExperiment** and select **Open class**. This will open the class **Modelica.Thermal.HeatTransfer.Sources.FixedHeatFlow**.
- Expand the *Details Panel* and create stickies for the parameters Q_{flow} and α .
- Save the view as **Burner_Parameters**. See Figure 16 below.



Figure 16 Create component view

- d. Reopen the model **HeatingSystemExperiment**; it should have an additional *Component view* called **Burner_Parameters** now.
23. Toggle on the views **Pump_Parameters** and **Burner_Parameters** and toggle all other views off.

Note: **Burner_Parameters** is a *System view* in **FixedHeatFlow** but will be a *Component view* when the class is instantiated (for example, as *burner* in **HeatingSystemExperiment**).

This concludes workshop 1.2. Well done!