



Modelica™ Tutorial for Beginners

Exercises with Dymola

Bernhard Bachmann

University of Applied Sciences

Bielefeld

Outline

- Getting started with Dymola
- Exercises
 - Using the Modelica standard library
 - 1. Simple drive train
 - 2. Animation of a pendulum
 - Writing simple models (own equations)
 - 3. Acceleration of mass
 - 4. Different model variants of a pendulum
 - Definition of arrays of components
 - 5. 1-dim heat flow
 - Hybrid systems
 - 6. Automatic gear box
 - 7. Bouncing ball
 - 8. Hysteresis Block

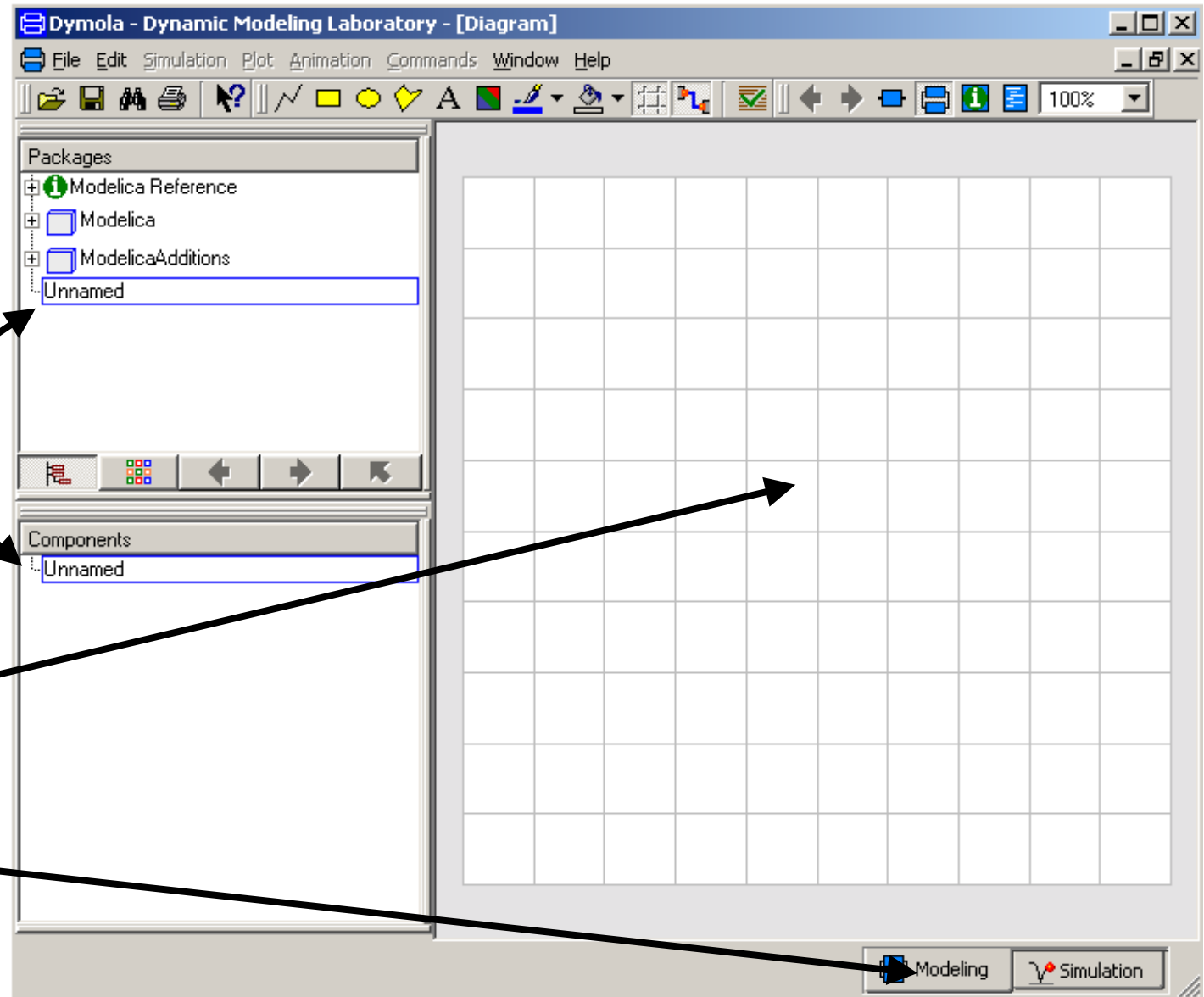
Getting Started with Dymola

1. Start Dymola

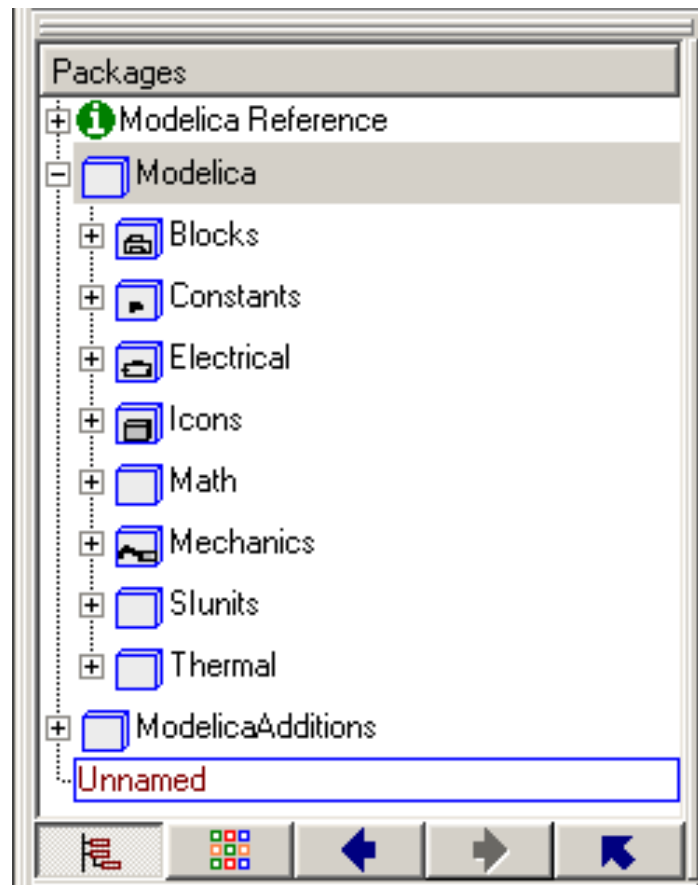
New model

Diagram layer

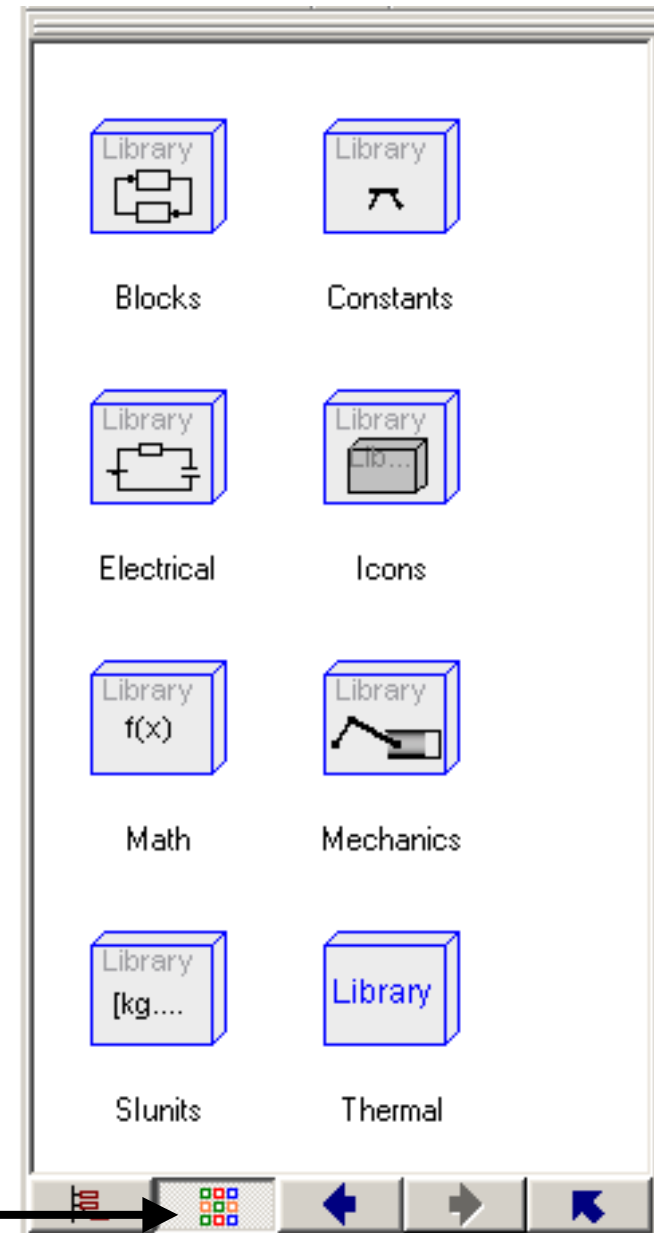
Modeling mode



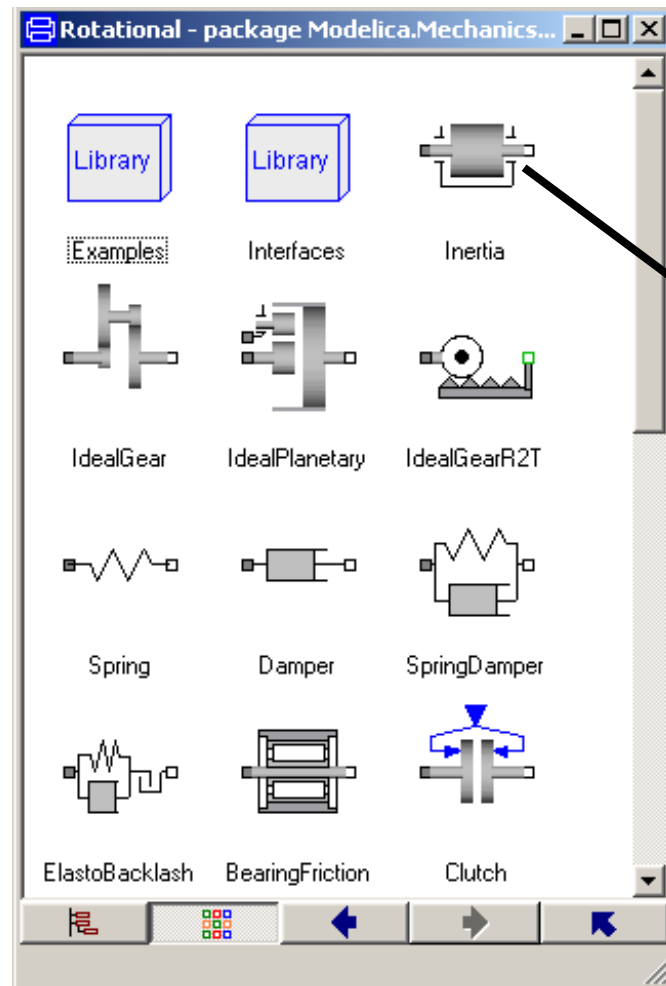
Open library windows



Hierarchical view

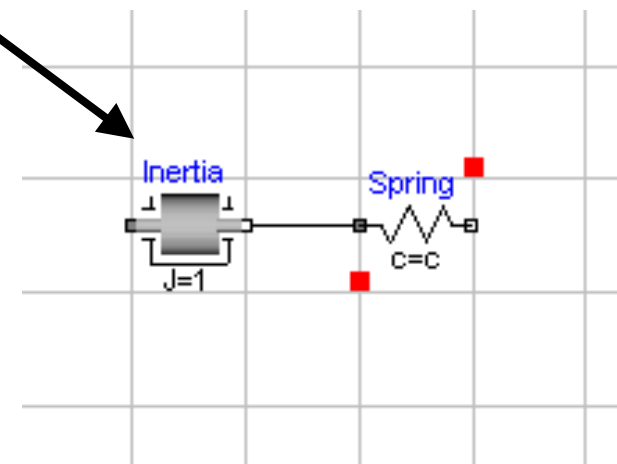


Package view



Drag & Drop modeling

Diagram layer



Modelica-Bibliothek:
Modelica.Mechanics.Rotational

Build up model using libraries

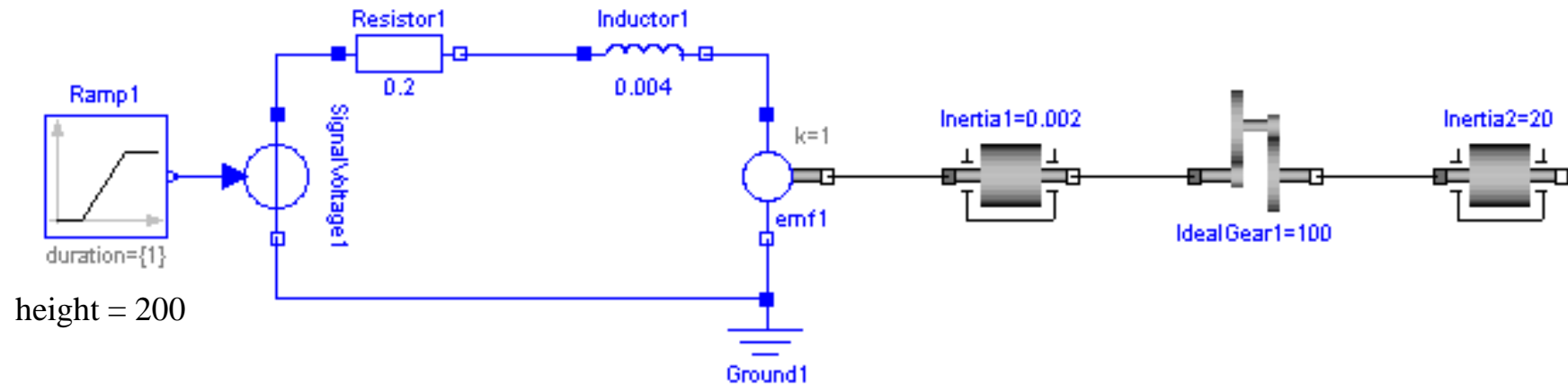
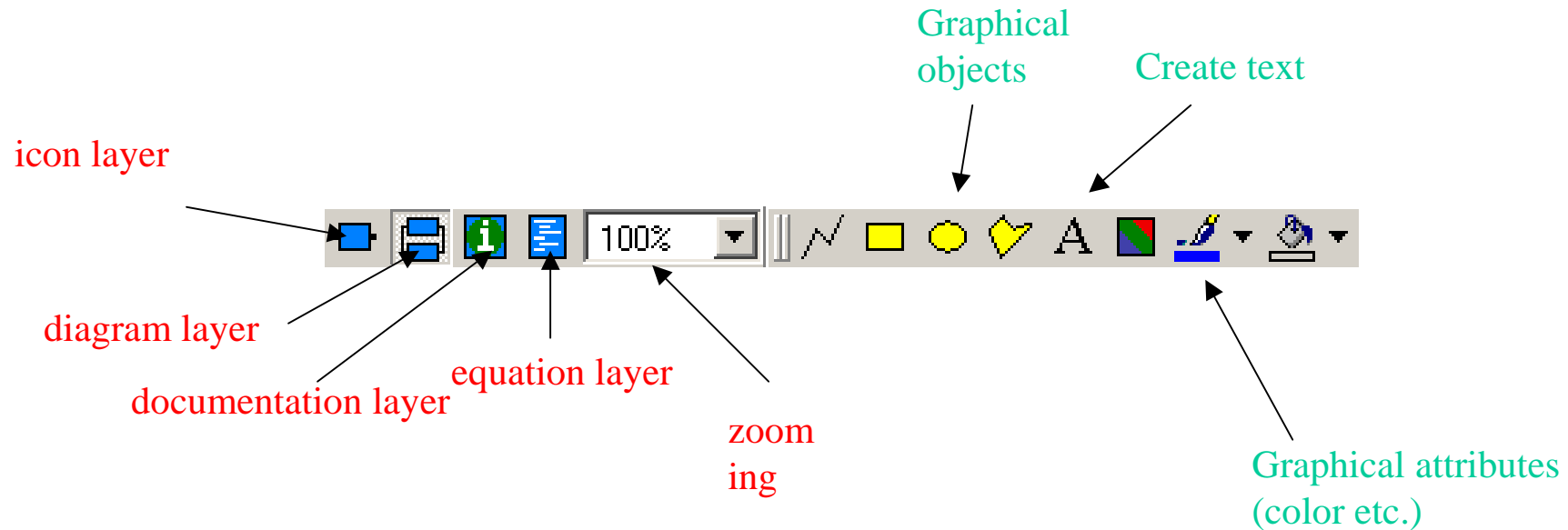
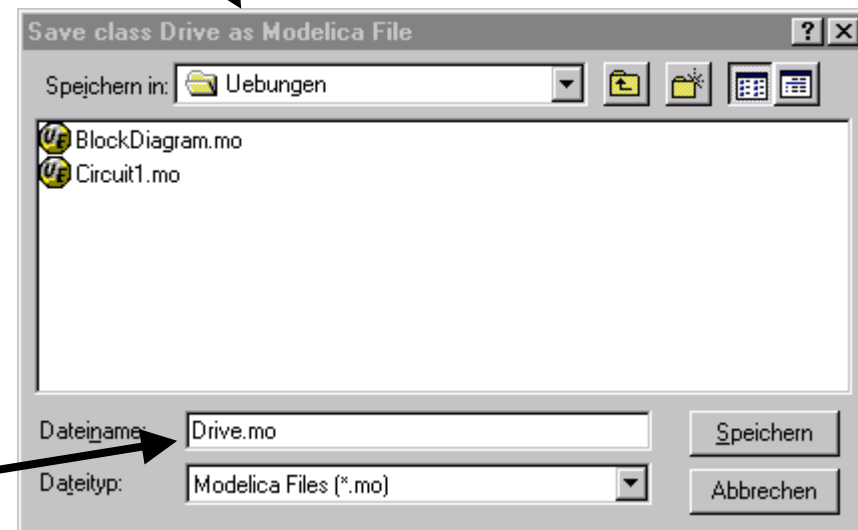
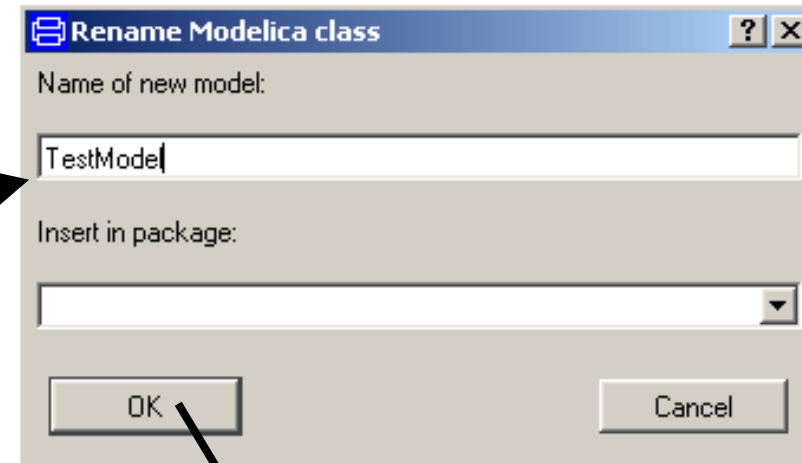
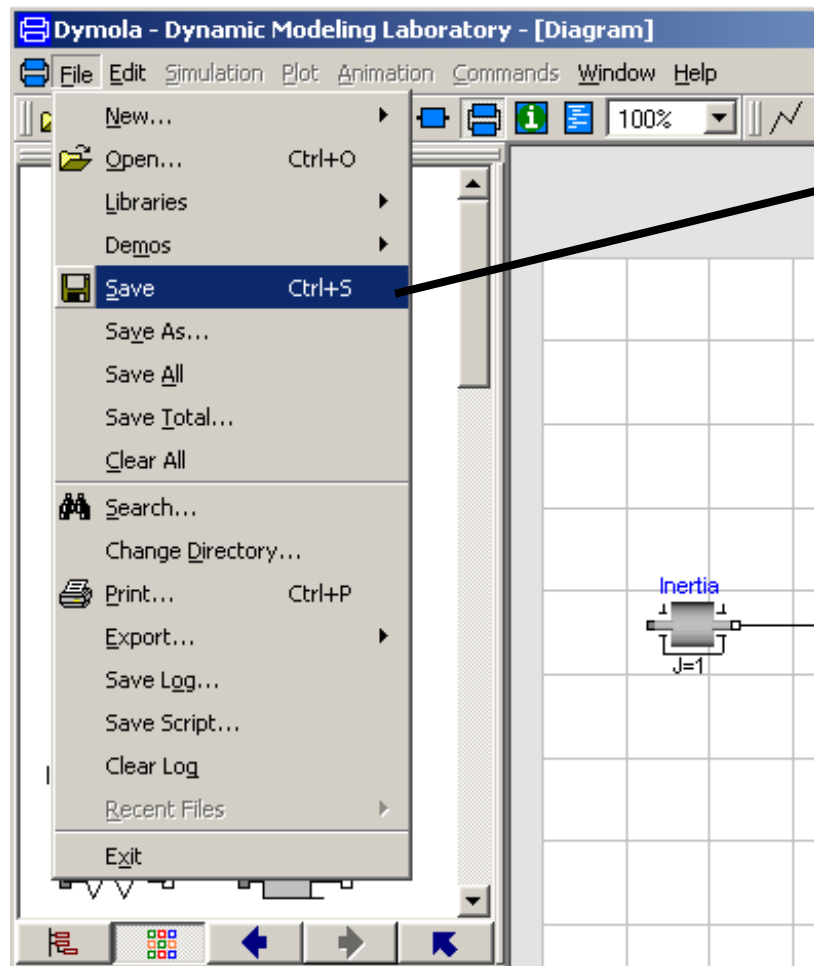


diagram layer

Save model

Model name (letters, digits, ...)



File name (Default: **<Model name>.mo**)

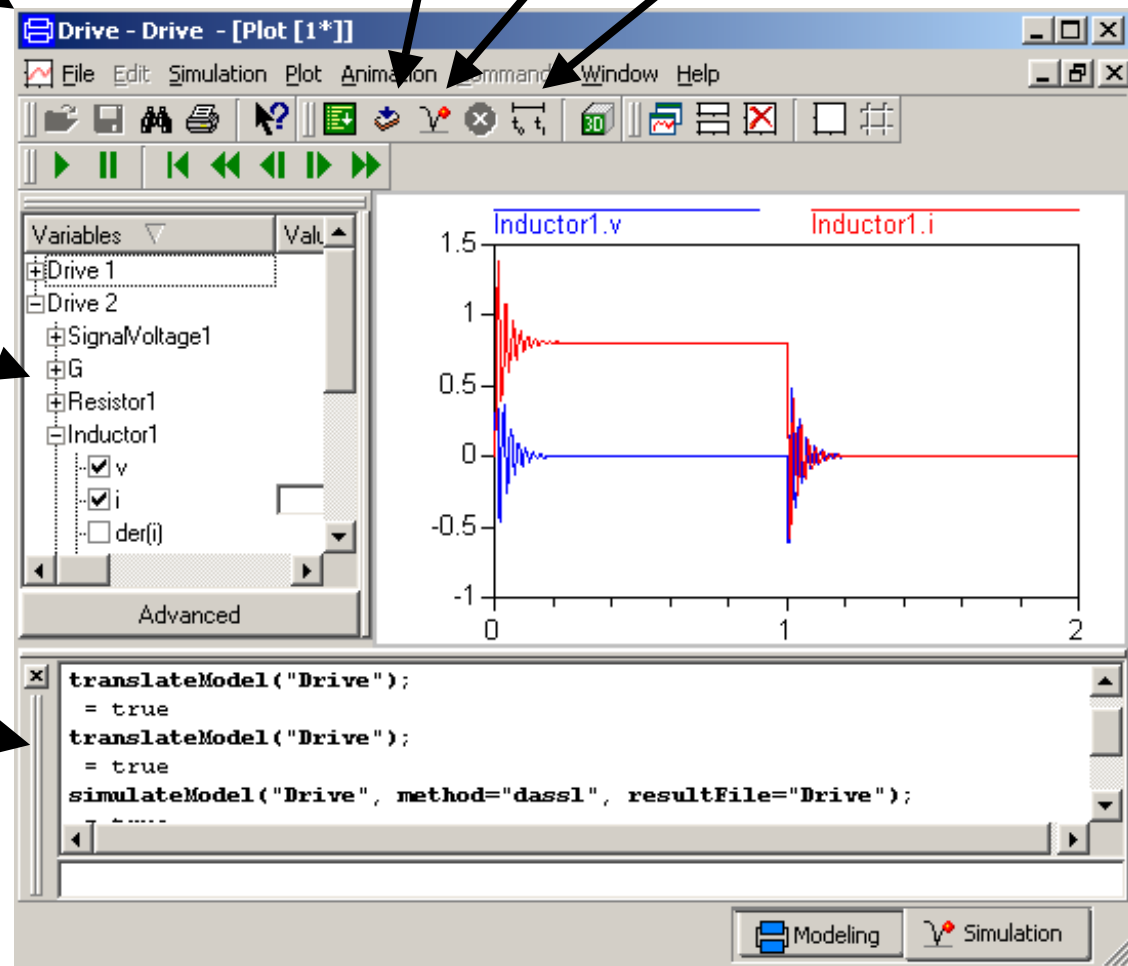
Simulate model



translate simulate experiment Setup

change parameter in translated model
and set initial condition for sates

Information on translation process



Exercise 1: Simple drive train

Goal:

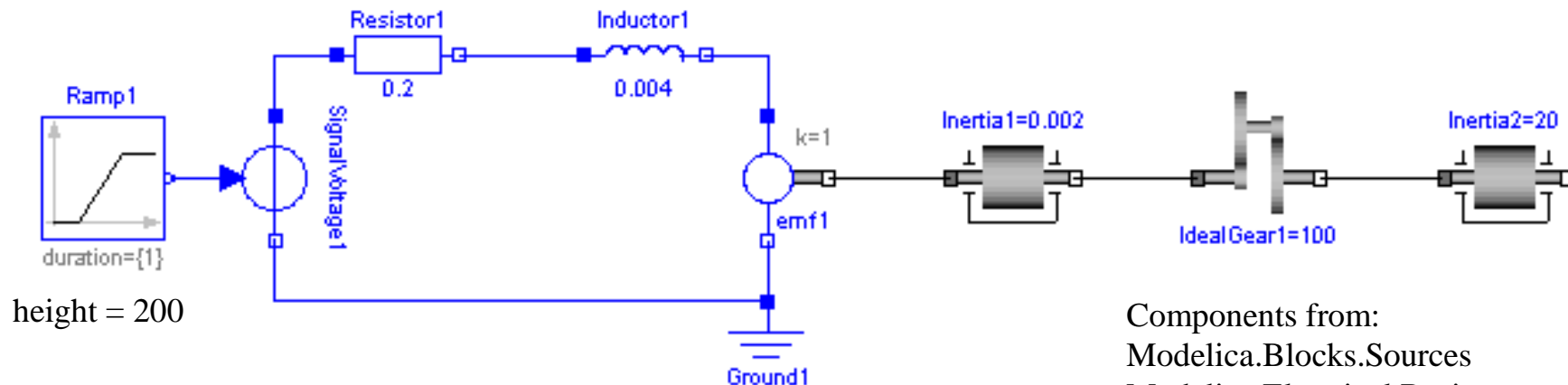
Build model and simulate using only components from available libraries
(useful for complete beginners)

Task:

Model and simulate the following simple drive train
(dc-motor with load and ideal gear-box)

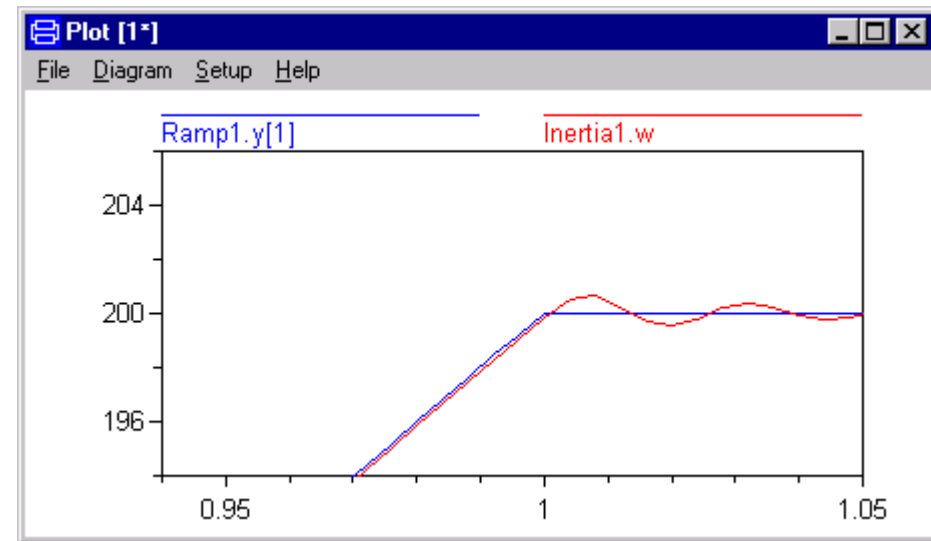
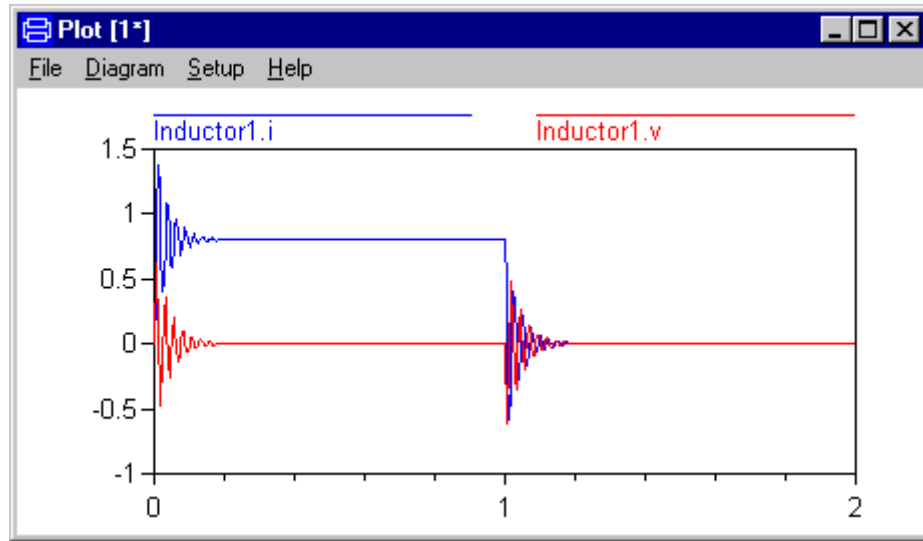
Simulate for 2 s

Try different values for the load inertia ($\text{inertia2} = 10 \dots 30$)



Components from:
Modelica.Blocks.Sources
Modelica.Electrical.Basic
Modelica.Mechanics.Rotational

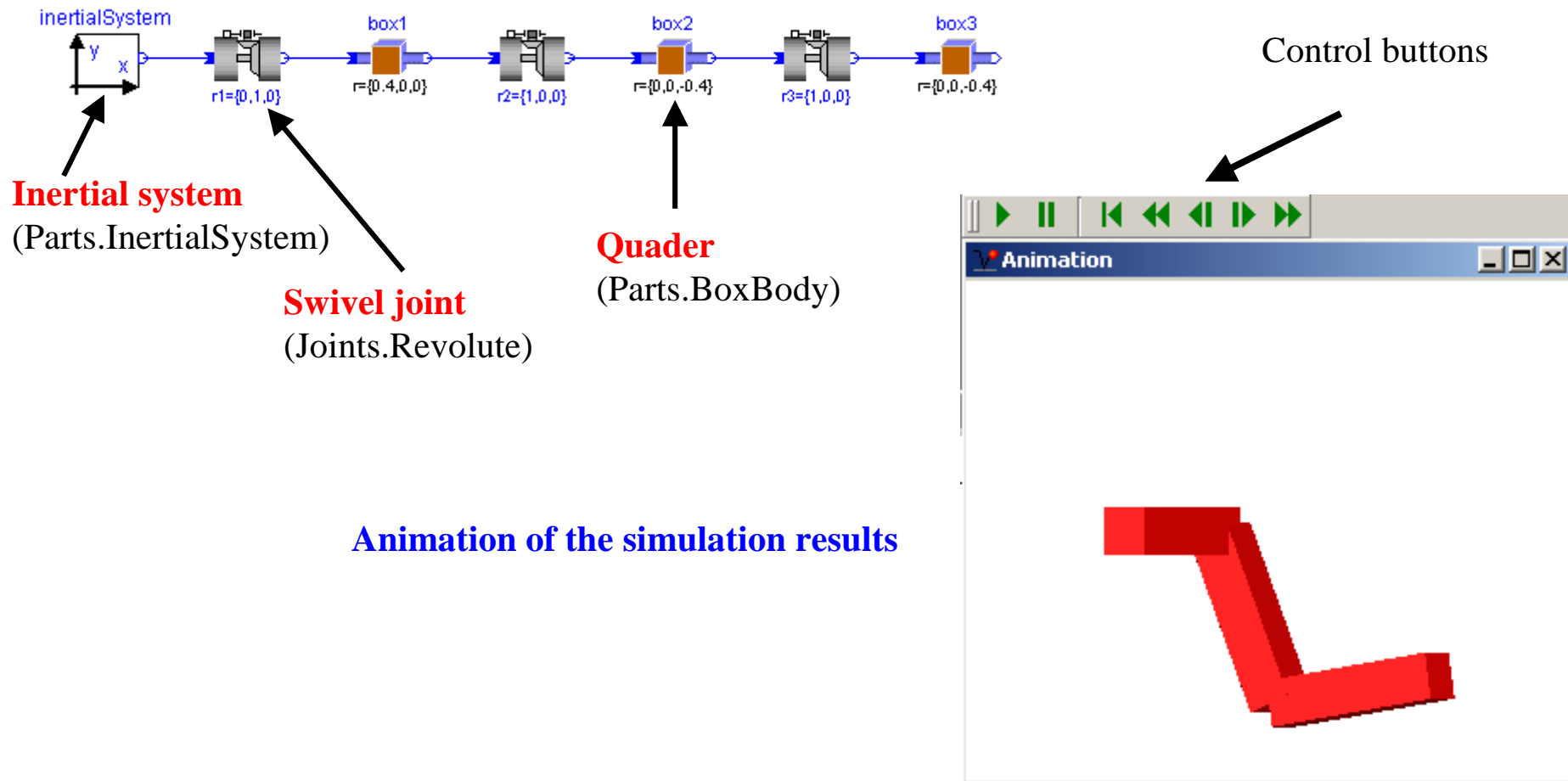
Exercise 1: Result plot



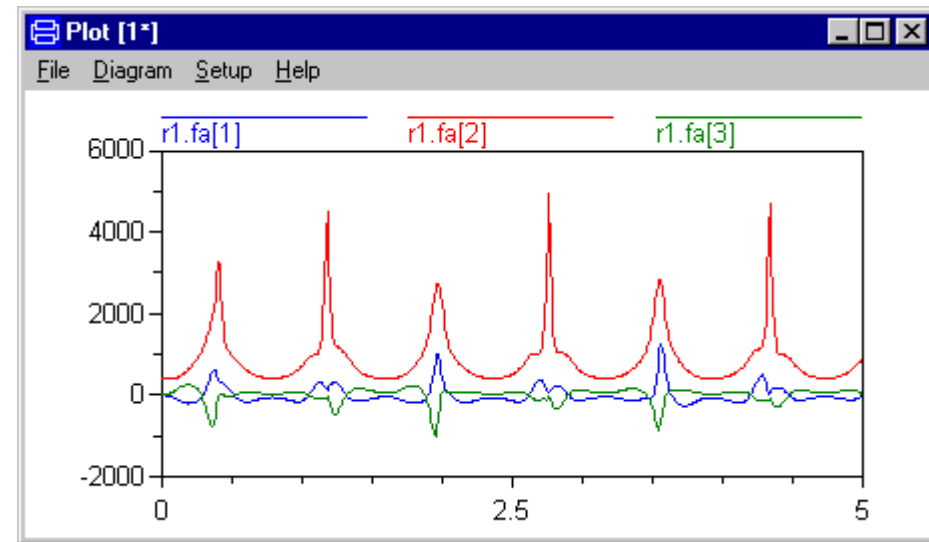
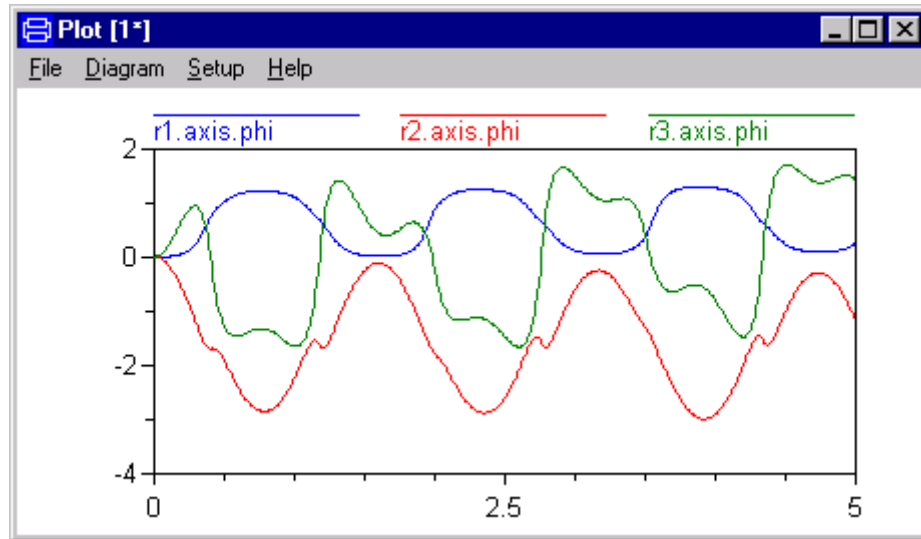
Exercise 2: Futura pendulum

Task:

Model, simulate and animate the futura pendulum example using the **ModelicaAdditions.MultiBody** library. Simulate for 5 s



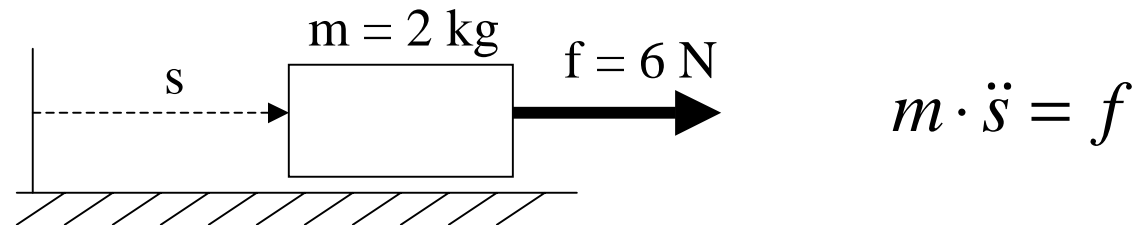
Exercise 2: Result plot



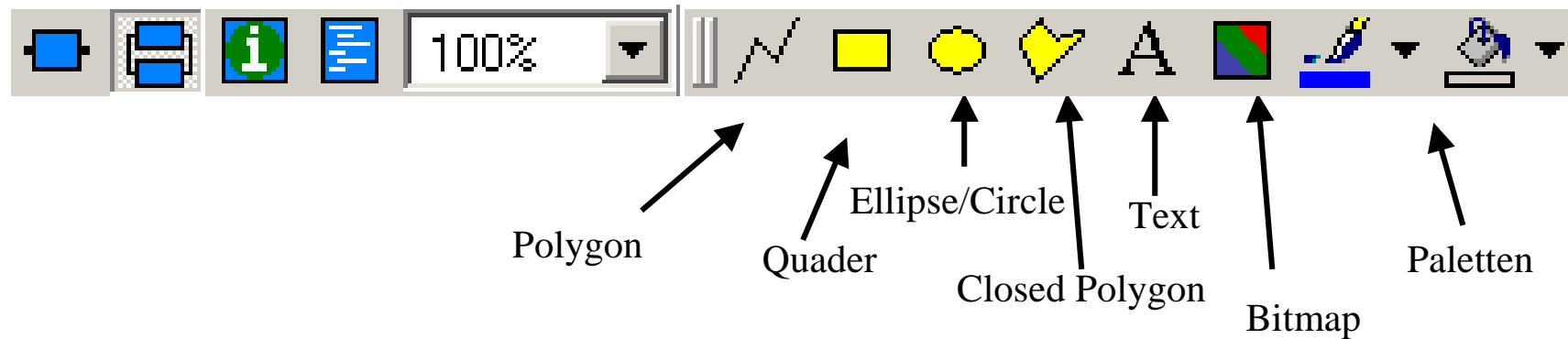
Exercise 3: Acceleration of mass

Task:

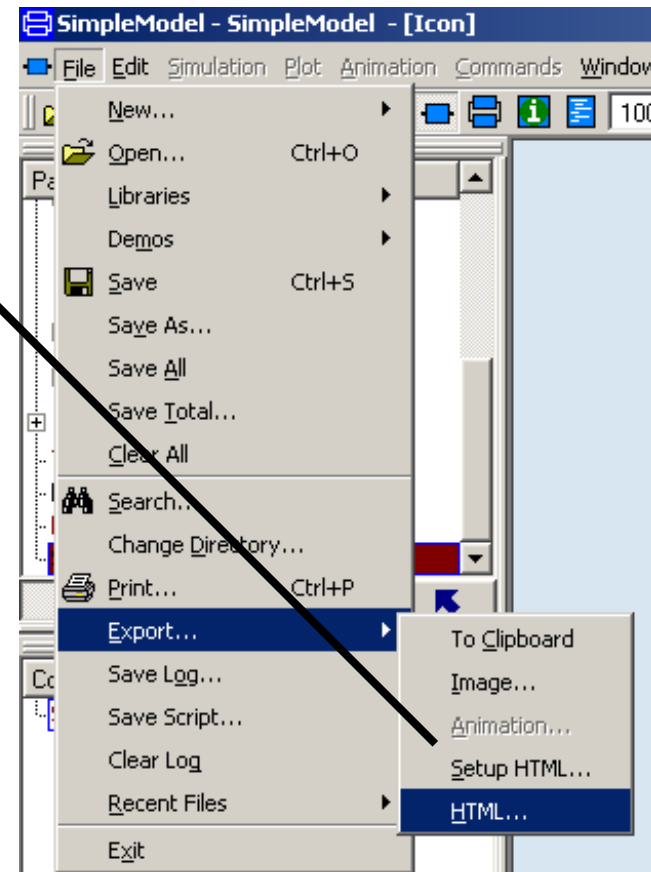
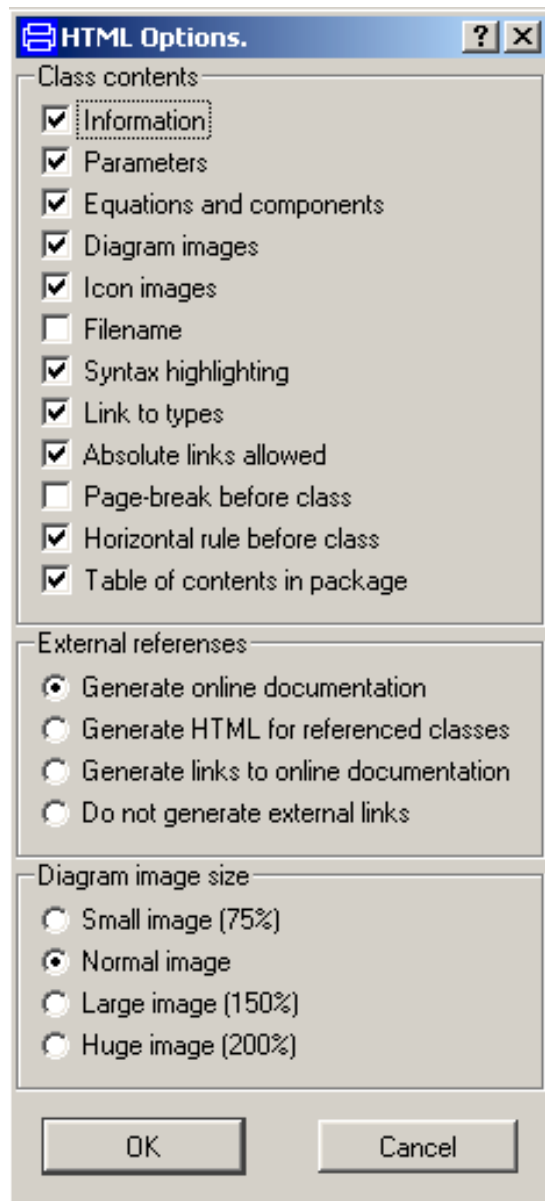
Model and simulate variant 3 of the accelerated mass example (see Tutorial)



create a **picture** of the model within the **icon layer** and
create an **HTML-documentation** of the complete model



Exercise 3: Create HTML-documentation



HTML-documentation is stored within the local help-directory.

Exercise 4: Different model variants of a pendulum

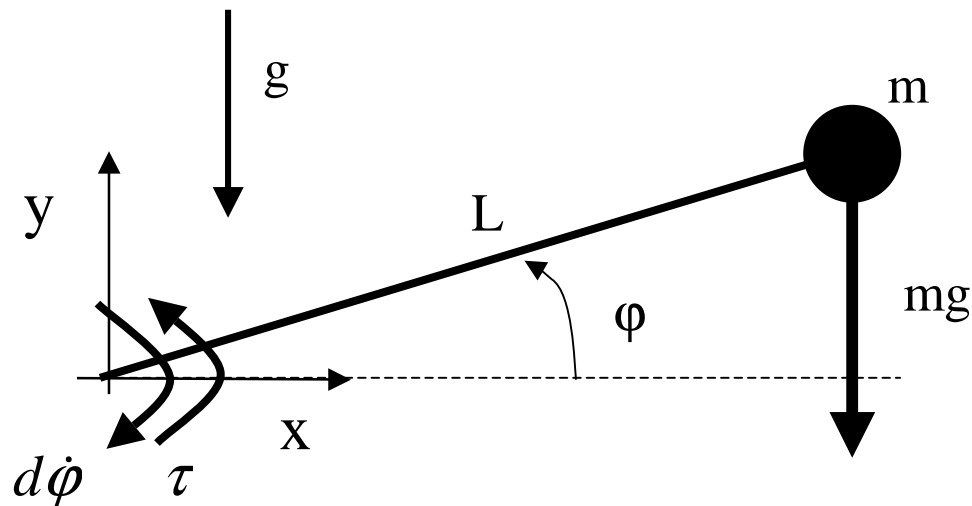
Task 1:

Create two different model versions of a robot arm (= Pendulum)

(a) using the component library [ModelicaAdditions.MultiBody](#)

(b) writing the **equations** by hand.

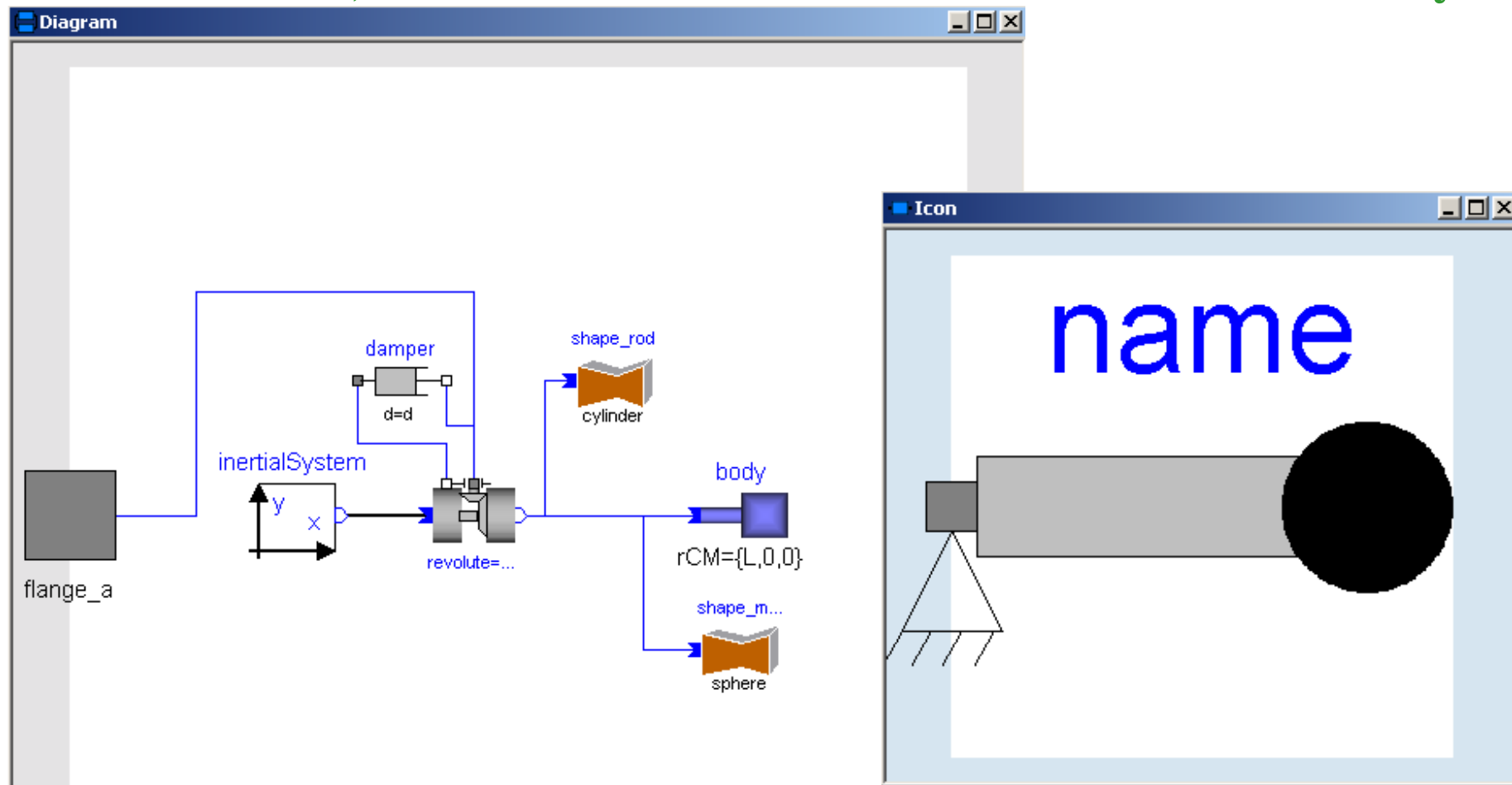
Simulate and check the results of both models!



Equation at the center of rotation:

$$J \cdot \ddot{\varphi} = -m \cdot g \cdot L \cdot \cos(\varphi) - d \cdot \dot{\varphi} + \tau$$

Exercise 4:a) Pendulum - with ModelicaAdditions.MultiBody



icon layer

diagram layer

Exercise 4:b) Pendulum – equations by hand

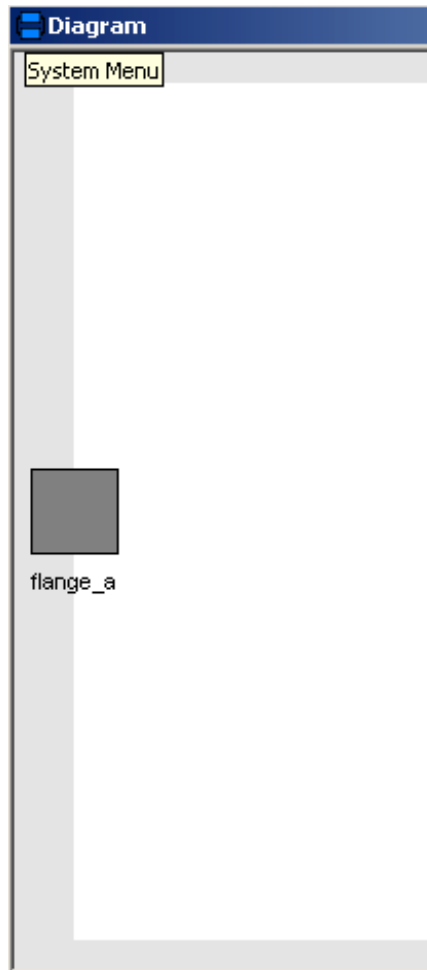


diagram layer

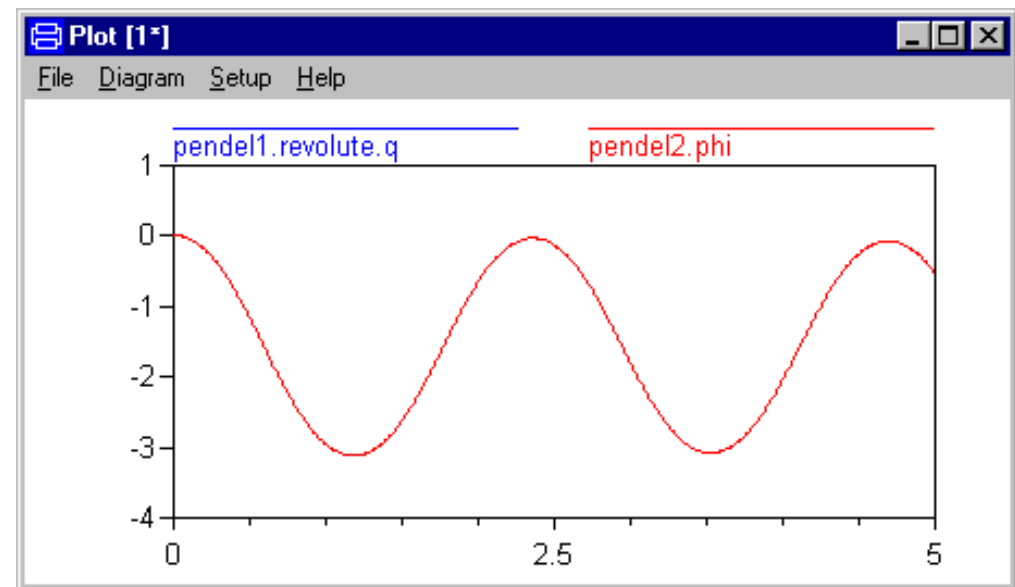
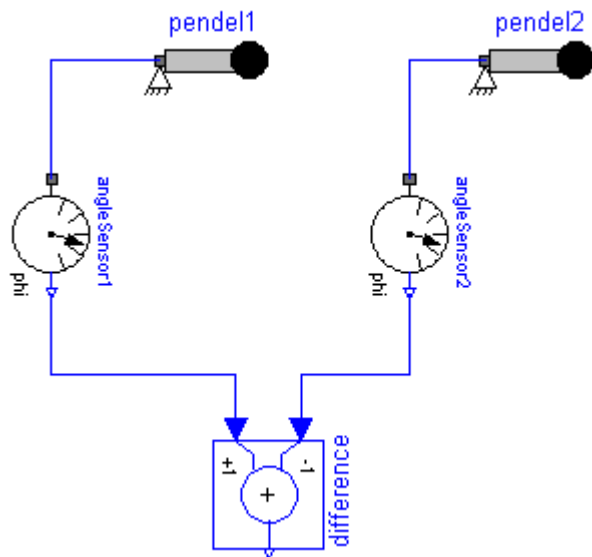
The Modelica text layer contains the following code:

```
model Pendel2
  package SI = Modelica.SIunits;
  parameter SI.Length L=1 "Length";
  parameter SI.Mass m=5 "Mass at the mass point";
  parameter SI.Damping d=0.1 "Damping at the joint";
  parameter SI.Acceleration g=9.81 "Gravitation acceleration";
  SI.Angle phi "Rotation angle of the Pendulum";
  SI.AngularVelocity w "Angle velocity of the pendulum";
equation
  phi = flange_a.phi;
  w = der(phi);
  m*L*L*der(w) + d*w + m*g*L*cos(phi) = flange_a.tau;
end Pendel2;
```

Modelica text layer

Exercise 4: Comparison of the two models

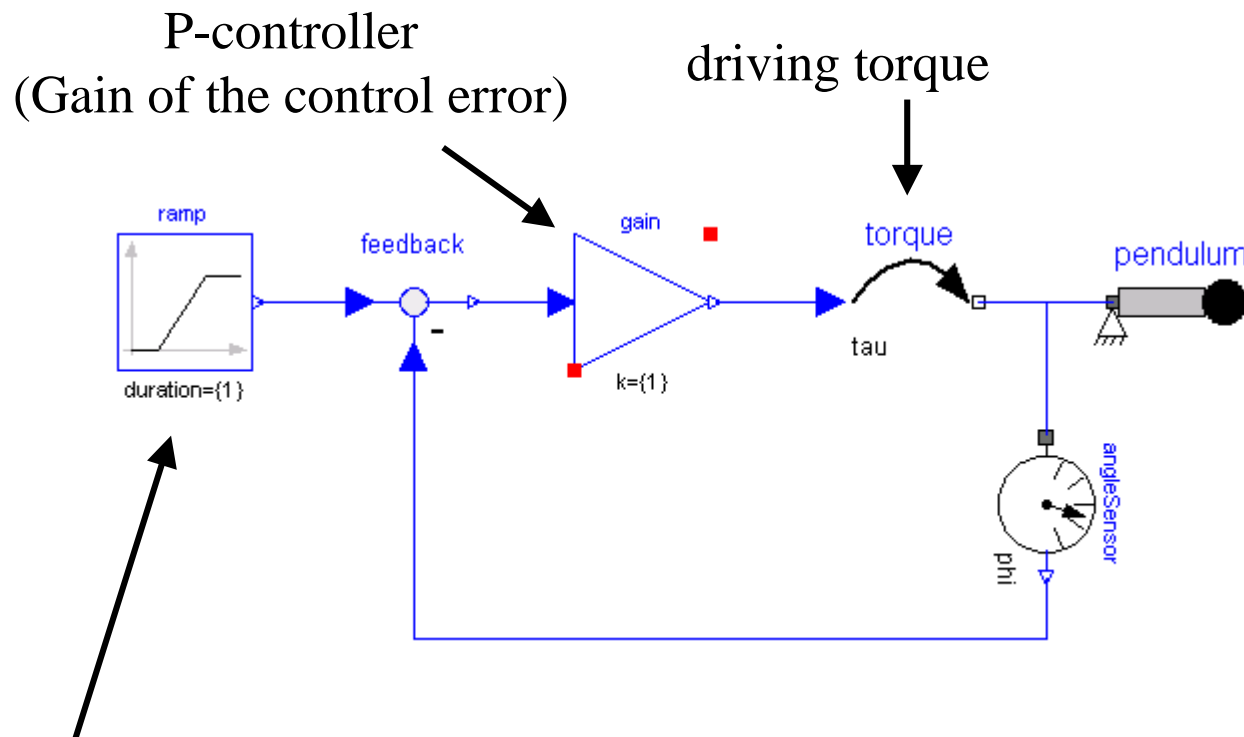
Compare the simulation results of both pendulum models (angle, etc.)



Exercise 4: Simple control system of the robot arm

Task 2:

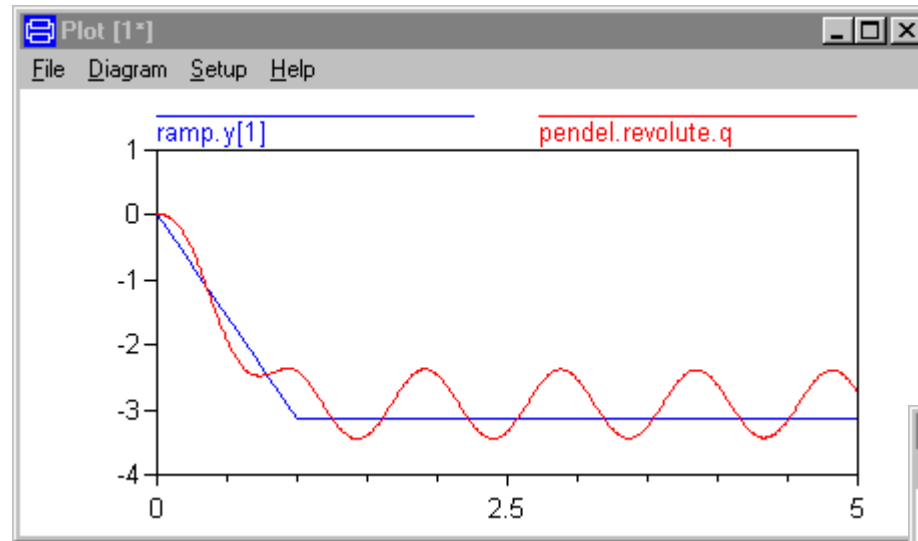
Build up a simple control system to easily change the angle of the robot arm:



target angle: change from 0 to -180 Grad

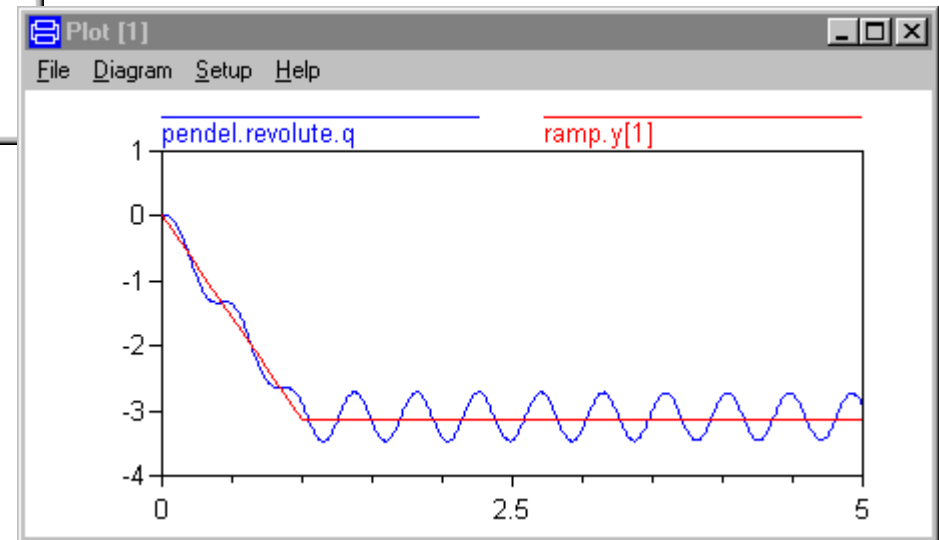
Exercise 4: Simple control system of the robot arm

Try different values for the gain of the controller
for example: $\text{gain.k} = 1, 10, 100, 1000, 200$



$k = 200$

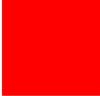
$k = 1000$



Control structure is too simple!


Exercise 5: 1-dimensional heat flow

Build up a library of components to model 1-dimensional heat flow
(package SI=Modelica.SIunits):



```
connector HeatCut_a
  SI.Temperature T;
  flow SI.HeatFLux q;
end HeatCut_a;
```

```
connector HeatCut_b
  SI.Temperature T;
  flow SI.HeatFLux q;
end HeatCut_b;
```



HeatResistance



```
model HeatResistance „only heat transport“
  parameter SI.Area A
  parameter SI.Length L
  parameter SI.ThermalConductivity lambda;
  HeatCut_a a; HeatCut_b b ;
equation
  a.q = lambda*A/L*(a.T - b.T);
  0 = a.q + b.q;
end HeatResistance
```

Exercise 5: HeatLib-library

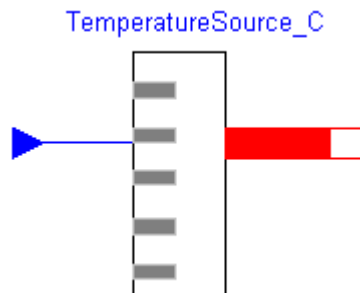


HeatCapacitance

```

model HeatCapacitance „only heat storage“
  parameter SI.Mass m;
  parameter SI.SpecificHeatCapacity c;

  parameter SI.Temp_C T0 "initial Temp.";
  HeatCut_a a(T(start=T0-Modelica.Constants.T_zero));
equation
  a.q = c*m*der(a.T);
end HeatCapacitance;
  
```

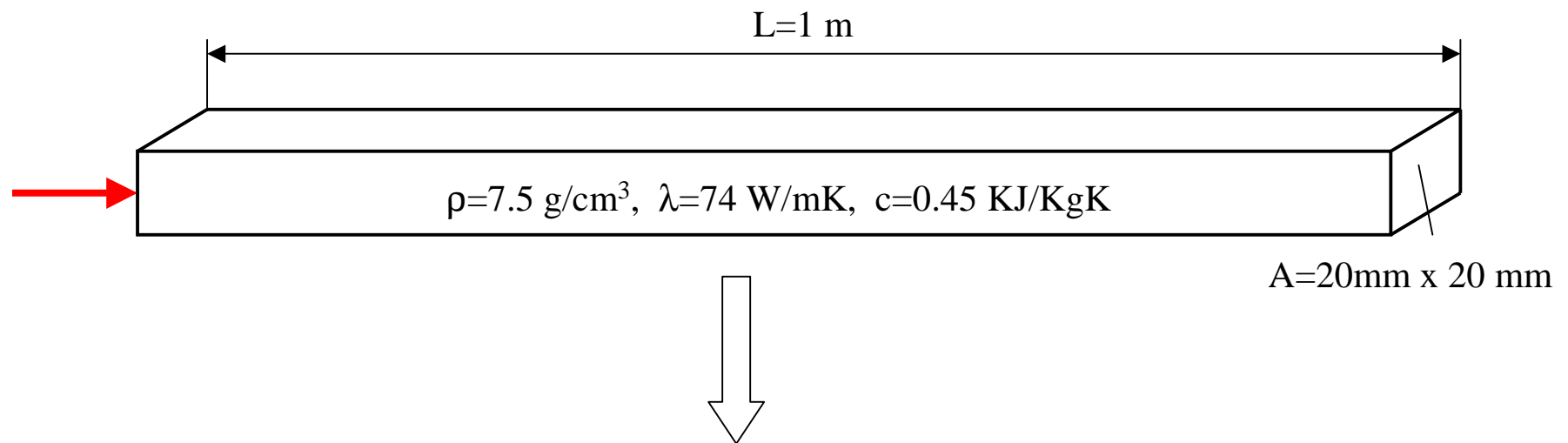


```

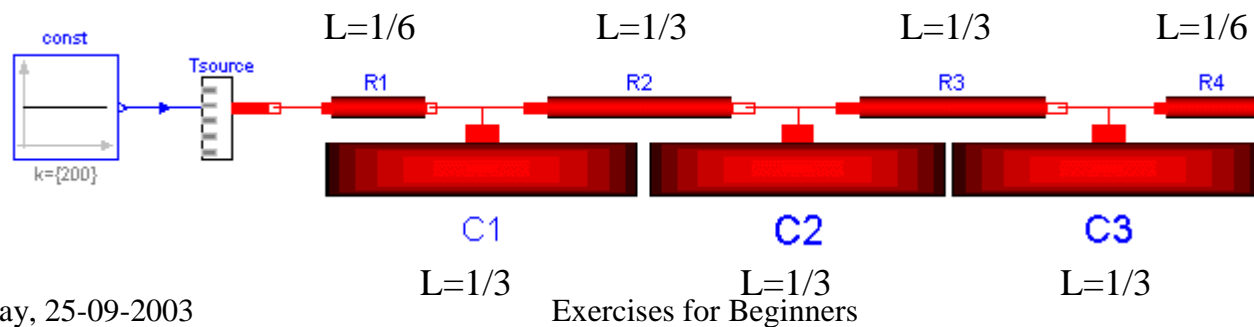
model TemperatureSource_C „fixed temperature“
  Modelica.Blocks.Interfaces.InPort inPort(n=1);
  HeatCut_b b;
equation
  b.T = inPort.signal[1]- Modelica.Constants.T_zero;
end TemperatureSource_C;
  
```

Exercise 5: Modeling problem

A completely isolated rod shall be modeled which has a temperature of 20⁰ Celsius everywhere. At the left end, suddenly the temperature is raised to a fixed temperature of 200⁰ Celsius:

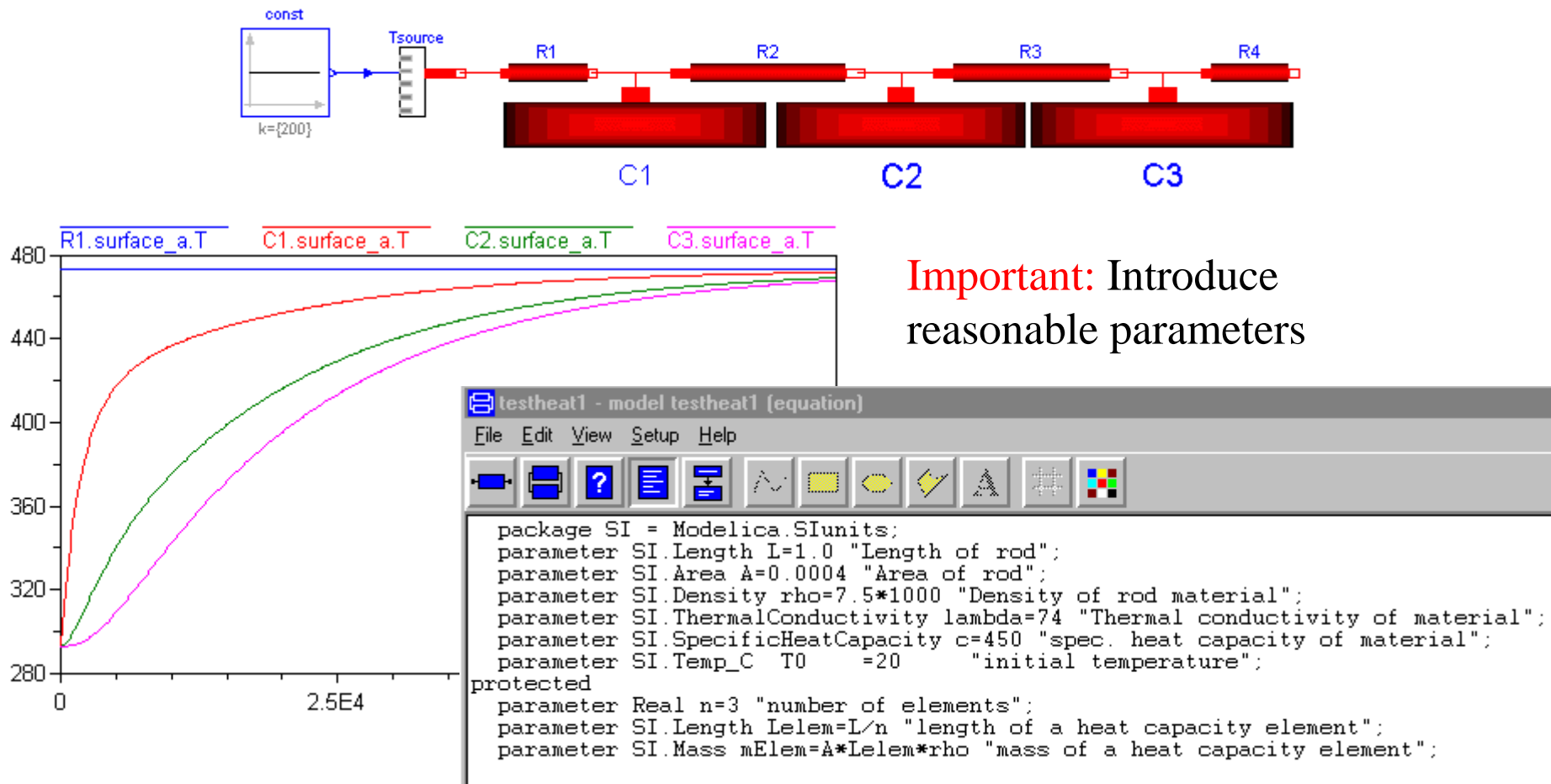


discretized rod in base elements of your library



Exercise 5: Modeling with single elements

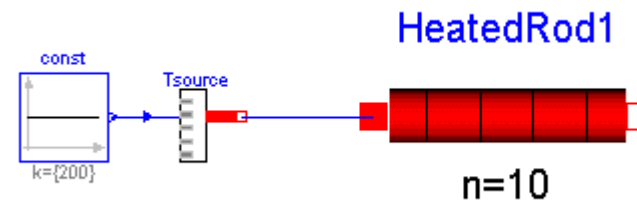
Task 1: Build up and simulate the rod with single elements ($n=3$).



Exercise 5: Modeling with component arrays

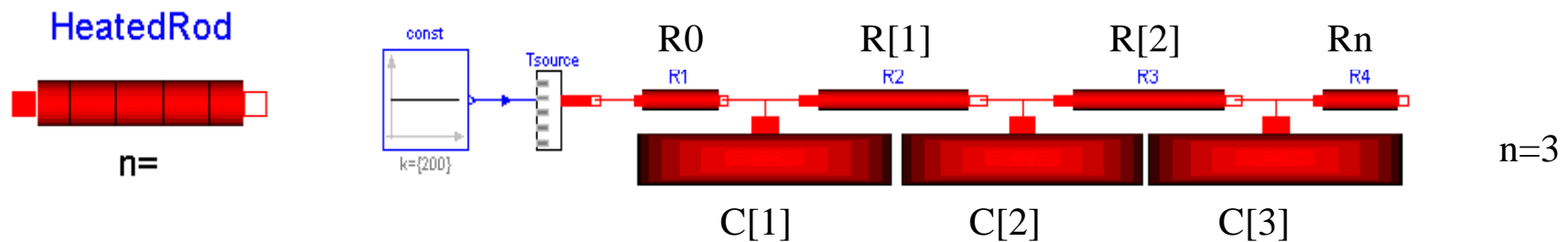
Usage of **single elements** has the **disadvantage**, that it is inconvenient to change to a **finer discretization**.

Better: use **component arrays**



Task 2:

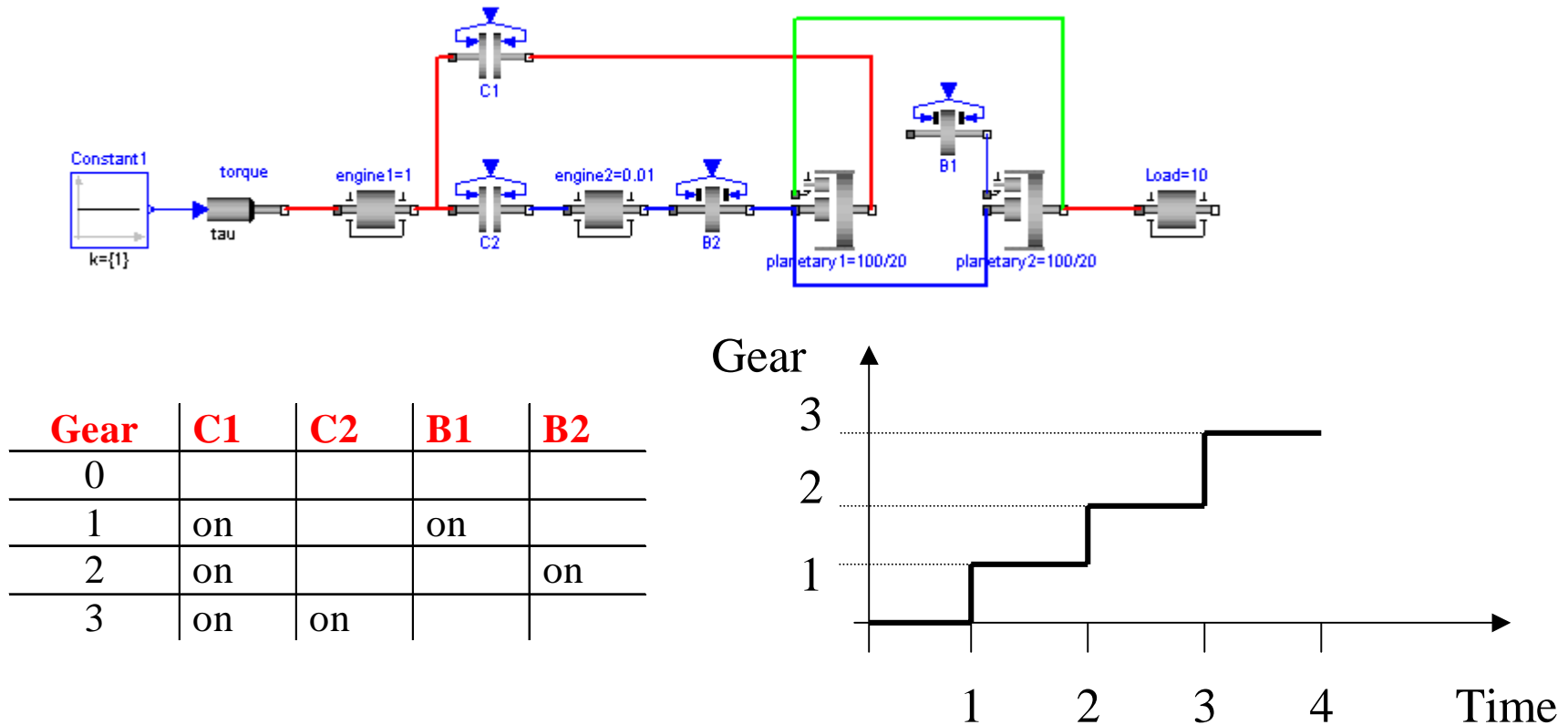
Model the rod with component arrays, such that the number n of discretized elements can be changed with one parameter. Compare simulations for $n=3$ and $n=10$. Simulate for $5e4$ s.



Exercise 6: Simple automatic gearbox

Task 1:

Build up the following model of a simple automatic gearbox and design a simple control for the clutches C1, C2 and brakes B1, B2, and simulate the gear shift from the figure at the lower, right part (shift from gear 1 to gear 3). Use the default value for the friction characteristics for all clutches and brakes.



Exercise 6: Fixing an appropriate contact pressure

Task 2: Determine an appropriate contact pressure of the Clutch.

Clutch.**mode**

= **2**: clutch is **not active**

= **1**: **forward sliding**

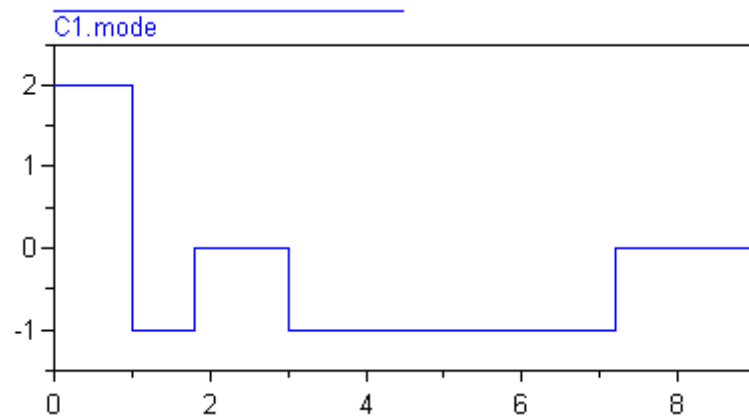
= **0**: **stuck** (no relative motion)

= **-1**: **backward sliding**

$k=200$

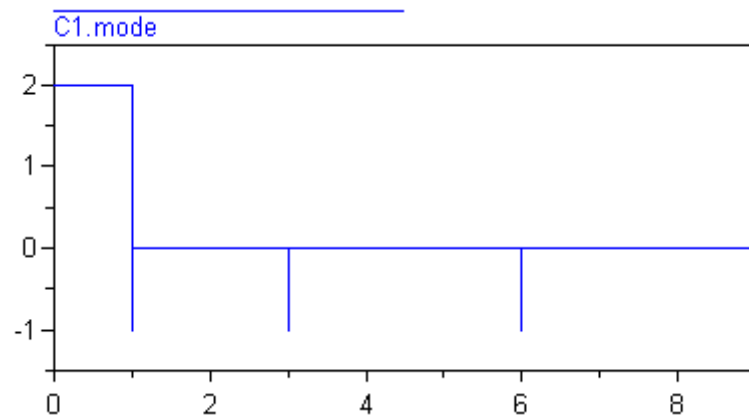
Clutch slips a lot

(heavy losses and wastage)

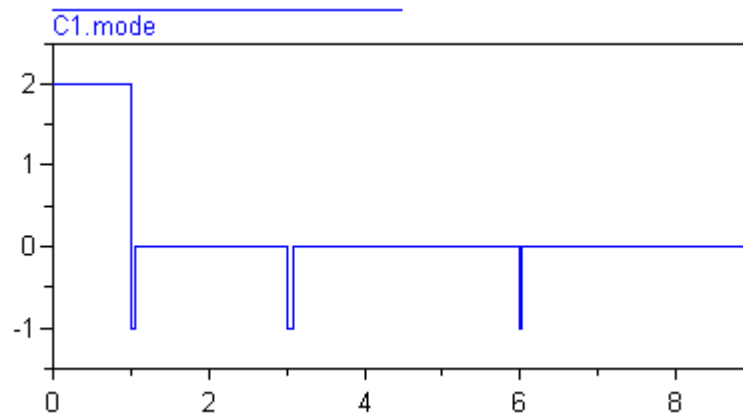


$k=10000$

Clutch does not slip at all
(jerky changing gears)

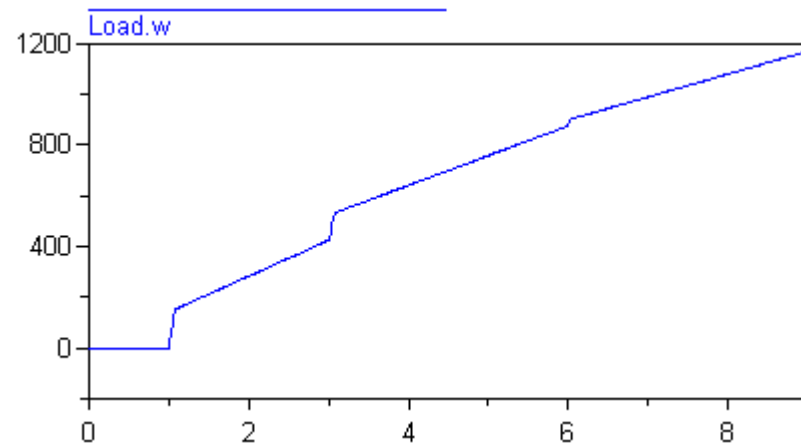


Exercise 6: Fixing an appropriate contact pressure



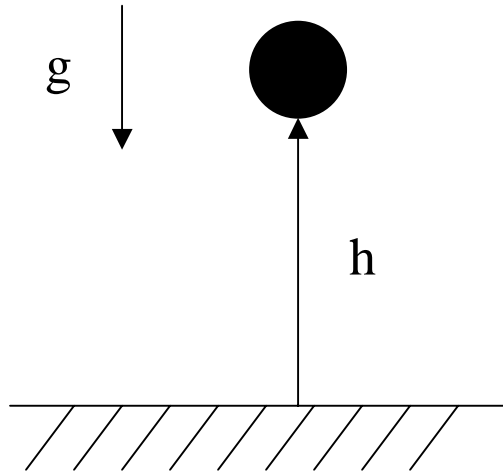
$$k = 2000$$

good compromise



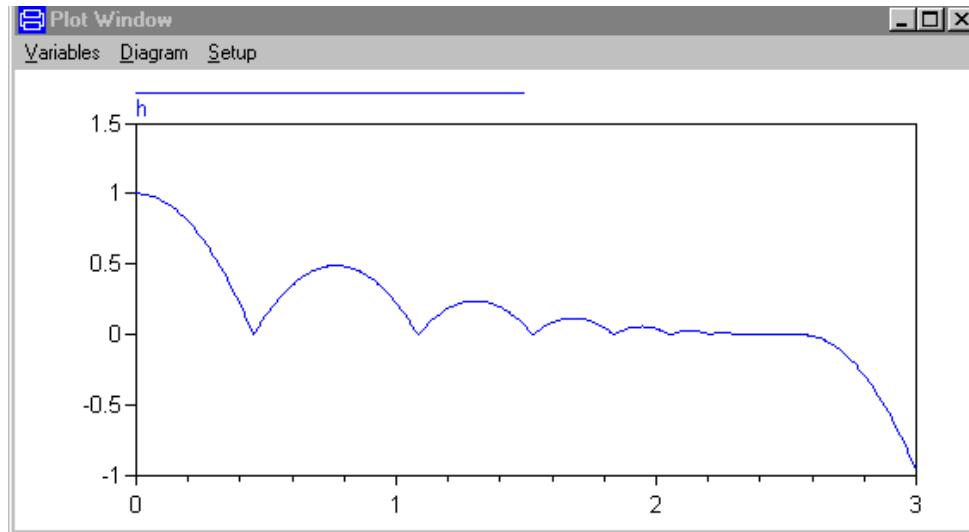
angular velocity of the load

Exercise 7: Bouncing ball



```
model bouncingBall
  parameter Real e=0.7;
  parameter Real g=9.81;
  Real h(start=1);
  Real v;
equation
  der(h) = v;
  der(v) = -g;

  when h <= 0 then
    reinit(v, -e*pre(v));
  end when;
end bouncingBall;
```

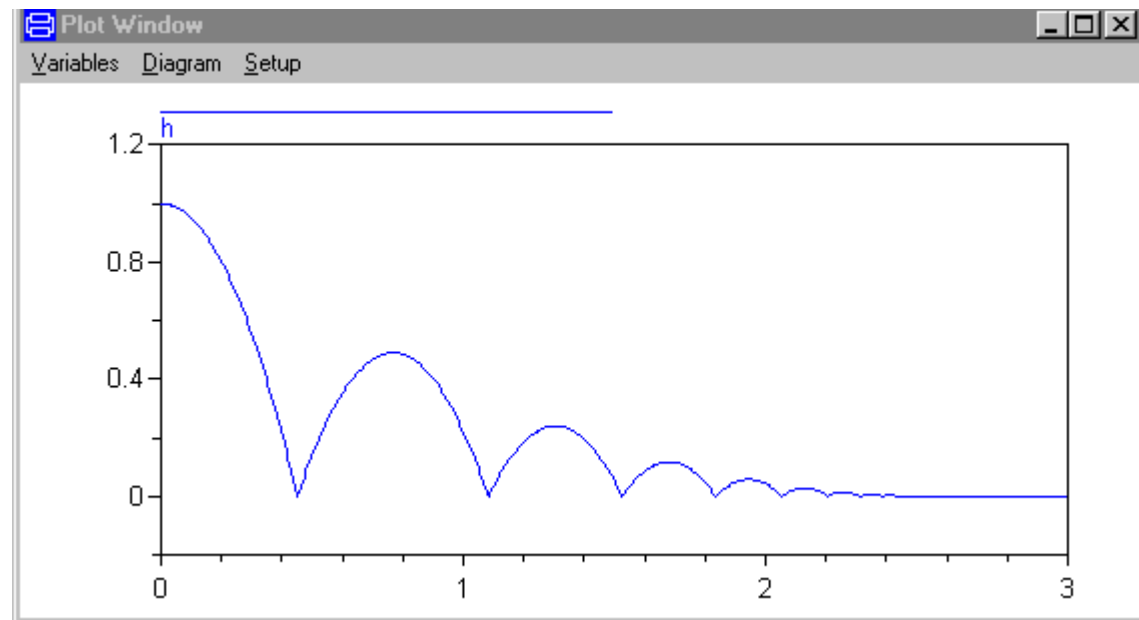


Exercise 7: Bouncing ball

Task:

- (a) Reproduce the simulation result of page 29.
- (b) Modify the model of the bouncing ball, so that the ball finally stays on the ground.

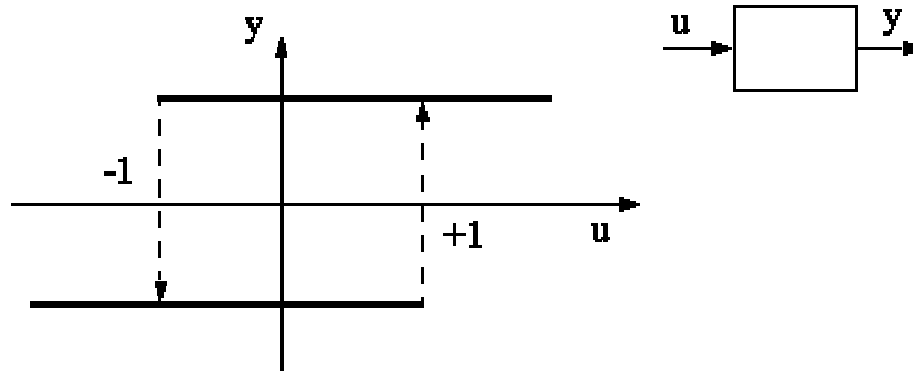
Idea: Change the model structure, if the velocity after the impulse is too small. (= Acceleration of the ball vanishes).



Exercise 8: Modeling of an Hysteresis

Task:

Build up a Modelica model of the following Hysteresis block. Use the **pre**-operator as memory to detect which branch of the characteristic curve has been calculated in the past.



Test the Hysteresis-block with a sinus-signal $u = \sin(w \cdot \text{time})$

