Tutorial Abstract Booklet

16th Modelica & FMI Conference

Lucerne, Switzerland

Tutorial Overview:

1. Beyond Simulation: Building Workflows and Web Applications with Modelica and Python
2. CasADi tutorial on dynamic optimization with FMI 3.0 Model Exchange
3. eFMI®: A beginner's overview and hands-on
4. Exporting and importing an FMU using C code
5. FMI Beginners Tutorial - Exporting, Simulating, and Co-Simulating FMUs
6. FMUGym: From Uncertainty-Aware Simulation to Learning-Based Control with FMI and Python
7. Introduction to Modeling, Simulation, Debugging, and Interoperability with Modelica and OpenModelica
8. Modelica in the Browser: Modeling, Simulation and Web App Integration for Custom UI
9. Modeling and Simulation of profitableness analyses in Modelica – industrial energy system meets variable energy prices
10. Modeling and Simulation of Robotic Arm Dynamics and Control in Modelica with MWORKS.
11. Modeling complex thermal architectures using the DLR ThermoFluid Stream Library
12. Regression Testing with Dymola and the Testing Library
13. System Structure and Parameterization
14. Tutorial on FMI3 co-simulation with UniFMU
15. Using SMArtInt+ for machine-learning and easy integration of artificial intelligence in Modelica

Beyond Simulation: Building Workflows and Web Applications with Modelica and Python

Daniel Rohde, Modelon

Duration: 2.5-3 hours

Abstract:

Modelica is a powerful language for modeling complex physical systems—but when combined with Python, its potential expands dramatically. This tutorial is designed to show attendees how to bridge the gap between simulation and practical application by building custom workflows and web applications based on Modelica models.

Through hands-on instruction, participants will learn how to integrate Python with Modelica to automate tasks, visualize results, and build interactive interfaces – all within Modelon Impact. Whether you're looking to streamline your simulation processes, create intuitive apps for sharing results with stakeholders, or create workflows that significantly cut down on post-processing time, this session offers a practical starting point.

By the end of the tutorial, each attendee will have successfully created a basic workflow and a functioning web app based on a simulation.

This is more than a coding session—it's a launchpad for turning your engineering models into accessible, usable tools that can drive innovation and collaboration across teams.

Expected experience of participants;

- Working knowledge of Modelica and Python

Software requirements:

- Laptop with Chrome or Microsoft Edge installed (Modelon Impact licenses will be provided by Modelon)
- Voila installed (https://voila.readthedocs.io/en/stable/install.html)

Link to further information:

Modelon blog about this topic in practice: https://modelon.com/blog/hybrid-modeling-with-a-reduced-order-model-and-neural-network-app/

16th modelica & fmi Conference

CasADi tutorial on dynamic optimization with FMI 3.0 Model Exchange

Joel Andersson, FMIOPT & CasADi
Joris Gillis, Yacoda & CasADi

Duration: 3 hours

Abstract:

CasADi is a versatile open-source framework for numerical optimization in general and numerical optimal control in particular. Over the past 12 years, it has been used for numerous applications in academia and industry.

In this tutorial, we provide a hands-on demonstration of FMI Model Exchange import and export in CasADi. Topics covered include validated first and second order derivative calculation, forward and adjoint sensitivity analysis, finding steady-state solutions and model predictive control.

Bring your laptop for the exercises, offered in Python and Matlab.

Expected experience of participants;

- Basic programming skills in Python or MATLAB.
- Interest in numerical optimization.

Software requirements:

- Laptop with Python or MATLAB.
- Python only: rights to install python wheels (e.g. pip, conda, …)
- Instructions on how to install necessary software will be provided live.

Link to further information:

See casadi.org for more information

16th Modelica & fmi Conference

eFMI®: A beginner's overview and hands-on

Christoff Bürger, Dassault Systèmes

Duration: 2h, 45min

Abstract:

This tutorial demonstrates the current state-of-the-art of available eFMI® tooling. Participants will get a very high-level overview of the eFMI® workflow from acausal physics models in Modelica® down to embedded target code and a *hands-on* experience of it for selected Modelica® examples. The generated eFMUs and their various intermediate model representations are investigated, focusing on the non-functional quality criteria satisfied by the generated solutions, like traceability within eFMUs, MISRA C:2023 compliance of generated production code and other code quality criteria like static memory allocation and error handling.

Compared to the tutorial at the last International Modelica Conference in 2023, we will investigate more advanced examples like hybrid physics and neural network (NN) models where unknown non-linear physics are modeled using NN surrogate models well integrated with known physics (so called physics-enhanced neural ordinary differential equations, PeN-ODEs). Also the latest eFMI® improvements will be presented, like tooling to import eFMI® Production Code in MATLAB®/Simulink®, the new standard library of GALEC built-in functions and others.

Expected experience of participants;

- No previous experience with eFMI®.
- Moderate knowledge of Modelica®.
- Good understanding of physics/equation-based modeling.
- Little/moderate knowledge of the embedded domain and its challenges.

Software requirements:

- Participants have to bring their own computer with Windows 10 or 11, 64-Bit, x86.
- Required software will be provided few days before via download and at the tutorial via USB stick.
- Time limited licenses (Dymola®, Software Production Engineering) valid throughout the conference week are provided.
- To provide the download links, participants have to register for the tutorial.
  Please write an e-mail to Christoff.Buerger@3ds.com (Modelica Association Project eFMI® project leader) that you like to participate.

Link to further information:

- Official eFMI® website: https://www.efmi-standard.org/
- Previous tutorial at the 15th International Modelica Conference: https://youtu.be/oCDH-8mXeNw

16th modelica & fmi Conference

Exporting and importing an FMU using C code

Torsten Sommer, Dassault Systèmes Deutschland GmbH
Andreas Nicolai, Umwelt- und Ingenieurtechnik GmbH Dresden

Duration: 3 hours

Abstract:

In this tutorial we show how to write a simple FMI 3.0 implementation for a Co-Simulation FMU in C code. Starting with a minimalistic code framework we explain the implementation of all required functions and give best-practice recommendations on implementing error handling. The first part of the tutorial session is completed by building the FMU and simulating it in an FMI master program.

Next we present the steps on how to import an FMU into a software. Again, focus lies on implementation of the necessary functions and implementation of robust error handling. The FMU created in the first part is imported and run through a prepared simulation loop. Using the debugger we follow the function calls and develop an understanding of how the FMI master interacts with the FMU and how data is being exchanged.

We show how to build the FMU and the master program using platform-specific compiler tool chains. Participants can use their own development environment or use a prepared online compiler and development environment.

Expected experience of participants;

- A basic understanding of the Functional Mock-up Interface and the C programming language

Software requirements:

- A laptop with Linux, macOS, or Windows

Link to further information:

https://github.com/modelica/fmi-import-export-tutorial-2025/ (currently under development)

FMI Beginners Tutorial - Exporting, Simulating, and Co-Simulating FMUs

Christian Bertsch, Bosch, Modelica Association Project FMI
Cláudio Gomes, Aarhus University, Denmark
Maurizio Palmieri, University of Pisa, Italy

Duration: 3 hours

Abstract:

The Functional Mock-up Interface (FMI) standard plays a central role in enabling interoperability between simulation tools, particularly in cyber-physical and embedded system development. This tutorial introduces the core concepts of FMI and provides hands-on experience in exporting, simulating, and co-simulating Functional Mock-up Units (FMUs). Designed for both beginners and intermediate users, the tutorial is divided into three main sessions.
The first session presents the motivation and history of FMI, key technical concepts, and an overview of FMI 3.0, followed by live demonstrations of FMU export using Dymola and Simulink. The second session dives into working with FMUs using Python and the FMPy library, including validation, simulation, and integration with Jupyter notebooks. The final session introduces participants to the INTO-CPS Application for building co-simulations involving multiple FMUs, highlighting both graphical and command-line workflows, and demonstrating the impact of co-simulation parameters on accuracy and stability.
The tutorial concludes with a joint session summarizing the role of FMI in digital twins and highlighting real-world applications. Throughout the tutorial, attendees will have opportunities to ask questions and follow along with guided examples and shared materials.

Expected experience of participants;

- Basic familiarity with simulation concepts
- Some programming experience (e.g., in Python)
- Optional: Experience with Simulink, Dymola, or co-simulation tools is beneficial but not required

Software requirements:

To follow the hands-on parts of the tutorial, participants should have the following installed:

- Python with fmpy[complete] (installation guide)
- Optional: Matlab/Simulink
- Optional: Java (version 11 recommended) and INTO-CPS Application
- Optional: a Modelica tool such as OpenModelica or Dymola to export FMUs.
- A Google account to use Google Colab for co-simulation notebooks

Link to further information:

More information and tutorial materials can be found on the FMI Tutorial GitHub Repository

16th modelica & fmi Conference

FMUGym: From Uncertainty-Aware Simulation to Learning-Based Control with FMI and Python

Konstantin Wrede, Fraunhofer Institute for Integrated Circuits
Christoph Steinmann, TU Dresden

Duration: 2 hours

Abstract:

Classical system identification often falls short when modeling nonlinear systems such as robot actuators or HVAC systems, especially under real-world conditions where measurement noise and input disturbances are unavoidable. These challenges highlight the need for **control strategies** that are not only accurate but **robust to uncertainties**.
This tutorial offers a hands-on introduction to **FMUGym** (https://github.com/Fraunhofer-IIS/fmugym), an open-source Python package for **system simulation and control design under uncertainties**. FMUGym leverages co-simulation of **Functional Mock-up Units (FMUs)** exported from tools such as OpenModelica or Dymola and provides interfaces compliant with the Gymnasium standard, enabling **reinforcement learning (RL)** environments where FMUs serve as the control plant.

Participants will work through three practical stages using FMUGym:

1. Conducting a **Monte Carlo simulation** on a nonlinear dynamic system to analyze the impact of **parameter uncertainties**,
2. Designing a **classical feedback controller** to stabilize the uncertain system, and
3. **Training** an **RL agent** — using libraries such as Stable-Baselines3 — to optimize control performance in the presence of **input/output noise and parameter variations**.

Attendees will leave with a functional **toolchain for uncertainty-aware control design** combining FMUs, Python, and RL libraries. This tutorial is ideal for practitioners and researchers familiar with Python who are interested in bridging system simulation with modern, learning-based control strategies.

Expected experience of participants;

- Basic knowledge of Python and Control Systems

Software requirements:

- Windows/Linux(/Mac) machine with Python >=3.10 installed, stable internet connection

Link to further information:

https://github.com/Fraunhofer-IIS/fmugym

16th Modelica & fmi Conference

Introduction to Modeling, Simulation, Debugging, and Interoperability with Modelica and OpenModelica

Peter Fritzson, Linköping University, Open Source Modelica Consortium
Lena Buffoni, Linköping University and St Anna IT Research Institute
Adeel Asghar, St Anna IT Research Institute

Duration: 3 hours

Abstract:

This tutorial gives an introduction to the Modelica language, the OpenModelica environment, and an overview of modeling and simulation in a number of application areas. Some advanced features of OpenModelica will be presented, including debugging, profiling, clocked synchronous support, real-time embedded code generation, OMPython, OMJulia, FMI export, etc. A number of hands-on exercises will be done during the tutorial, both graphical modeling using the Modelica Standard Library (MSL) and textual modeling. Bring your laptop for exercises.

Expected experience of participants;

- No prior Modelica experience needed.

Software requirements:

- OpenModelica supports Linux, Windows and Mac systems, Docker, VMBox and it is also possible to access Linux installations on the web via the browser.

Link to further information:

https://www.openmodelica.org

https://openmodelica.org/images/M_images/250204-ModelicaTutorial-PeterFritzson-AdrianPop-MODPROD-2025.pdf

16th Modelica & fmi Conference

Modelica in the Browser: Modeling, Simulation and Web App Integration for Custom UI

Hilding Elmqvist, Mogram AB, Sweden
Martin Otter, DLR-FK, Germany

Duration: 3 hours

Abstract:

Modiator (Modelica Instant Simulator) is a simulator for a subset of Modelica with some extensions. The compilations and simulations can run in the browser on desktops, laptops, tablets, mobiles, or using node.js to run on IoT devices or on the cloud. Modiator is based on pure JavaScript modules with the simulator based on Sundials compiled into WebAssembly. Modiator UI has a library browser, drag&drop Modelica diagram editing, Modelica text editor, plotting, Monte Carlo simulator and analyzer, 3D animation, etc.

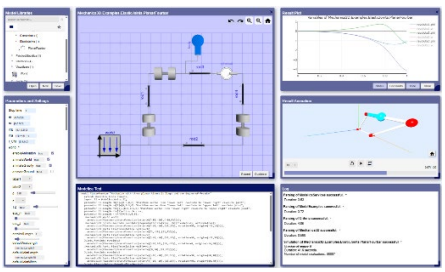
During the tutorial, the participants will be able to start develop a Modelica web app simulator with a specialized user interface for a certain model/domain. The modular architecture and API of Modiator will be explained. A development environment will be set up, for example, using VS Code for the web (i.e. no installation needed) and the participants will build a specialized user interface using the Modiator modules. The specific user Modelica model can be developed using the Modiator UI and tuned with the help of the Monte Carlo simulator and analyzer. A web app template will be provided which calls the Modiator API to compile and simulate Modelica models. The participants will adapt this template to get a desired user interface for experimenting with their specific model. Alternatively, AI tools, such as GitHub Copilot, can be used to produce a first Modelica model of the application and a first version of the HTML and JavaScript code of the web app.

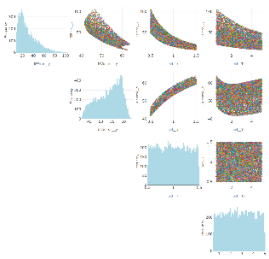
Expected experience of participants:

- Some know-how of Modelica, JavaScript and HTML is beneficial.

Software requirements:

- A Web browser (Chromium-based browser preferred, e.g. Chrome or Edge).





16th Modelica & fmi Conference

Modeling and Simulation of profitableness analyses in Modelica – industrial energy system meets variable energy prices

M.Sc. Martin Leuschke, EA Systems Dresden GmbH (martin.leuschke@ea-energie.de)

Duration: 1,5 hours

Abstract:

Modelica is now a very sophisticating modeling language that is used by engineers worldwide in multi-physical design processes. However, technical feasibility is only one part of the decision-making process, especially in industry. There, engineers need to consider both technical optimality and profitability in almost all design steps.

These already complex tasks are made even more complex by the variable pricing of energy consumption, grid-feed and peak power demand. Customized control algorithms can overcome these challenges by using available storage capacities and flexible system operation times. However, this requires an easy-to-use toolset with intuitive models and consistent accuracy and acceptable simulation speed.

This tutorial uses a typical industrial application with process and building heat demand as well as electricity consumption to demonstrate the benefits of modeling a complex system with controls and variable energy prices in a Modelica model. All necessary system components are modeled using the Modelica-based Blue Energy library. Modelon's Impact platform provides the flexible environment that engineers and simulation experts can use together to collaborate on evaluating the challenging issues in a consistent workflow. Modelica thus becomes an essential part of the design process, integrating seamlessly into companies' decision-making processes.

Expected experience of participants:

- Engineers and researchers with focus on energy system design and advanced system controls
- In-depth knowledge of the energy markets and the energy industry

Software requirements:

- Access to Modelon Impact Cloud with prepared Workspace
- Please send an e-mail to licensing@blue-energy-library.com so that we can provide you with access in advance

Link to further information:

https://www.blue-energy-library.com/

Modeling and Simulation of Robotic Arm Dynamics and Control in Modelica with MWORKS

Xueqi Ma, Suzhou Tongyuan Software&Control Technology Co., Ltd.

Duration: 2.5 hours

Abstract:

This tutorial introduces the complete development workflow for robotic arm systems using the Modelica language and the MWORKS platform. Participants will systematically explore methodologies to: model multi-domain physical systems, design control algorithms, execute simulations, perform design optimization, and generate embedded code—all within a cohesive model-based design (MBD) framework.

MWORKS, developed by Tongyuan, delivers an integrated environment for comprehensive system modeling, simulation, and deployment. Through practical hands-on examples, the tutorial demonstrates effective application of Modelica and MWORKS to co-develop physical and control systems, enabling accelerated design iterations with rigorous system-level verification and validation.

This session is ideally suited for engineers, researchers, and educators working in robotics, mechatronics, automotive, or any technical domain featuring intelligence embedded within physical systems.

Expected experience of participants;

- Basic knowledge of Modelica and Julia language syntax

Software requirements:

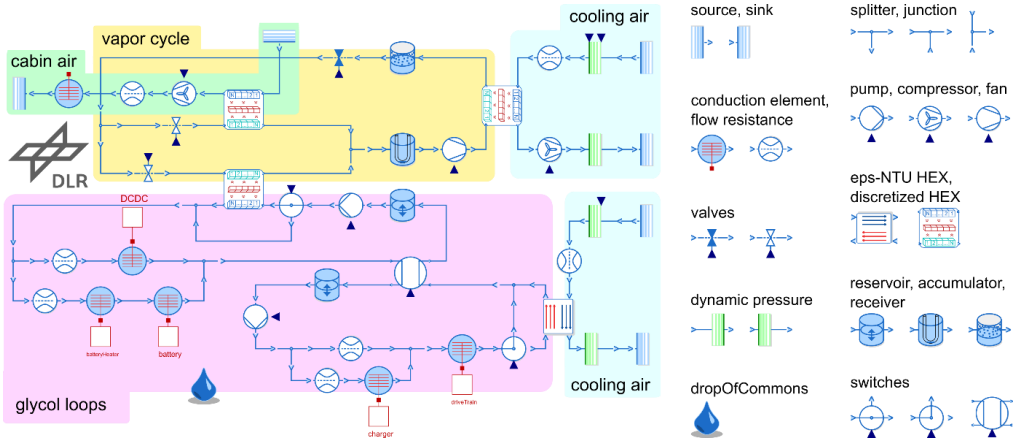
- Operating System: Windows 10 or later.
- Portable (USB drive) versions of MWORKS.Sysplorer and MWORKS.Syslab will be provided on-site; no installation required.

Link to further information:

https://en.tongyuan.cc (Available after August 15th)

Modeling complex thermal architectures using the DLR ThermoFluid Stream Library

Raphael Gebhart, DLR, Institute of System Architectures in Aeronautics
Dirk Zimmer, DLR, Institute of Robotics and Mechatronics



Duration: 3 hours

Abstract:

The DLR ThermoFluid Stream Library introduces a novel computational concept for simulating complex thermodynamic architectures. This concept is able to solve many well-known problems during initialization and simulation of thermo-fluid systems by avoiding algebraic loops and large nonlinear systems of equations.
In this tutorial, we will begin with an introduction to the core methodology behind the library, followed by an overview of its structure and the most essential components. This will help participants understand the modeling philosophy and capabilities of the library.
After the short introduction of the library, we will build up a thermal management system for an electric vehicle step-by-step in a hands-on example which will be easy to follow. The system will include multiple operating modes with reconfigurable fluid paths, as well as a fully integrated vapor-compression cycle. We will always provide intermediate results of the example system, to make sure all participants can stay engaged and can follow the tutorial until the end. Alongside the hands-on example, there will be enough room for discussions about modeling heuristics regarding the library and best practices in modeling thermal system architectures in general. We will also highlight recent updates to the library and invite participants to discuss future developments, including potential features and improvements. By the end of the tutorial, participants will have a solid understanding of how to work with the DLR ThermoFluid Stream Library to build up thermal system architectures.

Expected experience of participants;

- Used to Modelica. No further requirements needed.

Software requirements:

- All Open Source: Modelica Standard Library, ThermofluidStream Library, Tutorial Package
- Any Modelica IDE (Dymola, Open Modelica, ...)

Link to further information:

https://github.com/DLR-SR/ThermofluidStream

16th Modelica & fmi Conference

Regression Testing with Dymola and the Testing Library

Marco Keßler, Dassault Systèmes Austria GmbH

Duration: 2.5 hours

Abstract:

In this tutorial we use the Testing library (shipped with Dymola) to create unit tests and perform regression tests on Modelica classes. The tutorial addresses both, new and experienced users of the Testing library. New users learn how to get started and get and overview of the library's feature set. Participants which already use it, hear about the latest improvements, advanced features and our learnings on how to build reliable test models.

We elaborate on the benefits of test-driven development and how to do it in Modelica. For this purpose, various test methods in the Testing library are shown and explained, especially the novel clock-based comparison blocks. Together, we create test cases and execute them within Dymola. We discuss relevant signals for tests and recommend test methods and tolerances.

Finally, we run the tests in PowerShell with the new PSTesting package which is shipped with the Testing library. From there it's only a small step to run nightly tests on build automation servers like Jenkins and Gitlab CI. Users which are interested in this specific topic get guidance how to setup such a server and how to test Modelica libraries with it.

Expected experience of participants:

- Fundamental Modelica and Dymola knowledge
- Fundamental git knowledge

Software requirements:

- Dymola 2025x Refresh 1 Fixpack 1
  - Installer will be provided on USB sticks
  - Testing library is installed with Dymola and requires no extra license
  - Dymola license is not needed for small test models
- PowerShell 7.5 (for PSTesting):
  - Windows installation instructions
  - Linux installation instructions

Link to further information:

Testing Modelica classes on Jenkins, Gitlab and in local terminals (post in CATIA Systems Simulation Community)

Testing Library: A Look Inside (blog post by Claytex)

16th modelica & fmi Conference

System Structure and Parameterization

Dag Brück, Dassault Systèmes AB
Peter Lobner, eXXcellent solutions GmbH
Antoine Vandamme, Robert Bosch GmbH

Duration: 3 hours

Abstract:

System Structure and Parameterization (SSP) is a standard for combining simulation models, parameter sets, requirements, test and documentation into one coherent unit that can be processed by several complementary tools. From this starting point, the application of SSP has expanded to the definition of system architecture and to the scope of complete system verification and validation.

Key topics presented:

- What is SSP?
- Hands-on session with easySSP.
- Extending the scope to creadible modeling process and more advanced simulation.
- Closing and Q&A.

A large part of the tutorial will be focused on hands-on usage of SSP for system definition and simulation using easySSP.

Expected experience of participants;

- General knowledge of system modeling.
- FMI (optional).

Software requirements:

- Laptop with WiFi and a modern web browser. Please connect to the IMOC network in advance.
- Hands-on part will be run on a cloud server, no software installation required.

16th modelica & fmi Conference

Tutorial on FMI3 co-simulation with UniFMU

Santiago Gil, Aarhus University, Denmark
Cláudio Gomes, Aarhus University, Denmark
Yon Vanommeslaeghe, University of Antwerp, Belgium

Duration: 3 hours

Abstract:

UniFMU is an open-source command line tool, which provides a universal mechanism for implementing FMUs in popular languages that would otherwise not be able to produce C-compatible binaries. Currently, UniFMU has support for creating FMUs following FMI2 and FMI3 co-simulation in Python, Java, and C#. When used in combination with a co-simulation orchestration algorithm, UniFMU also has support for running distributed co-simulations.

In this tutorial, we want to showcase the usability of FMI3 co-simulation including clocks and clocked variables using UniFMU and its capabilities. We will have a general introduction to FMI3, and then, we will use a representative and simple case study, the incubator, where we will set up the incubator as a composition of FMI3 FMUs.

Attendees will be able to interact, update, and execute an FMI3 co-simulation that uses step- and event-mode. The co-simulation scenario replicates the functionality of the incubator with three coupled FMUs, namely, *Plant*, *Controller*, and *Supervisor*.

Expected experience of participants;

- Have a basic understanding of co-simulation.
- Have a basic understanding of Python.

Software requirements:

- Be able to execute commands from the command line.
- Have Python and pip installed on their machines with the corresponding path settings to run these from the terminal.
- Be able to create Python virtual environments and install new Python packages.

Link to further information:

https://github.com/INTO-CPS-Association/example-incubator-fmi3

Using SMArtInt+ for machine-learning and easy integration of artificial intelligence in Modelica

M. Sc. Tim Hanke, XRG Simulation GmbH

Duration: 3 hours

Abstract:

There is an increasing need for hybrid models consisting of physical and machine-learning models. XRG has developed the **SMArtInt+** Library, that enables the easy generation and integration of neural networks from different sources and types. This tutorial provides a hands-on introduction to the use of this library. The main use case is the creation of a hybrid model for a thermodynamic application.

Key topics include the generation of training data using SMArtInt+'s **DataGenerator** and the definition and training of neural networks using the **NetworkGenerator**. The DataGenerator allows for the generation of training data for physical Modelica models or model components by varying input parameters via Latin Hypercube Sampling (LHS) and recording the corresponding output values.

For the training process, the neural network architecture is defined directly in Modelica using predefined layer models. The library provides a range of useful layers for building neural networks, including dense layers, scalers, dropout layers as well as custom layers, for instance for constructing dimensionless neural networks. An overview of the various layer types and their role within neural network architectures is provided in this tutorial. The training is then performed automatically via a precompiled Python backend. Participants will explore strategies such as early stopping, learning rate scheduling, and hyperparameter tuning to improve training outcomes and assess model quality.

A central part of the tutorial is the integration of the trained neural networks into Modelica models using the SMArtInt+ **Interface-Blocks**, allowing seamless substitution of physical subcomponents with learned surrogates. To enhance understanding of the interface block, the integration of the trained network into Modelica is illustrated using both automated methods via the **ModelGenerator** as well as manual implementation.

Additionally, several concepts of hybrid modeling that combine physical and data-driven approaches are introduced, with a strong focus on the practical integration of trained neural networks within the Modelica environment.

Finally, model performance is evaluated through comparisons with the original physical model, including scenarios that trigger extrapolation and make use of SMArtInt+'s built-in **ExtrapolationWarnings**.

Expected experience of participants;

- No prior knowledge of neural networks/AI or Python required and basic Modelica knowledge

Software requirements:

- Windows 10 or 11 with Dymola 2025 Refresh 1 or OpenModelica 1.26 (backup)

Link to further information:

https://xrg-simulation.de/en/seiten/smartint