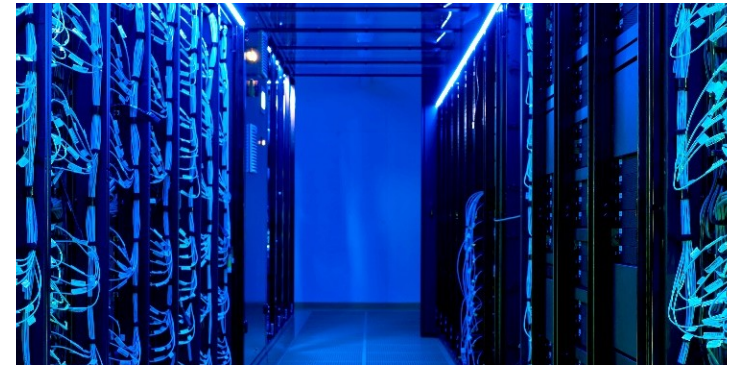
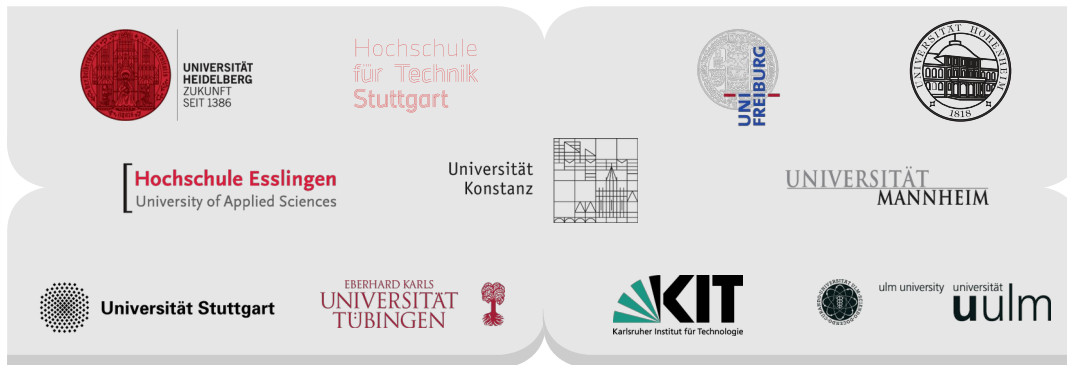


Introduction to Jupyter at NHR@KIT

Leon Pascal Schuhmacher, SCC, KIT



Outline

- Motivation
- Project Jupyter
- JupyterHub@HoreKa
- Software and Kernels
- Outlook
- Questions

Reference: Jupyter @ KIT

■ Most information can be found at

■ bwHPC Wiki:

https://wiki.bwhpc.de/e/Jupyter_at_SCC

■ NHR@KIT Wiki:

<https://www.nhr.kit.edu/userdocs/jupyter>

bwHPC Jupyter at SCC

Jupyter can be used as an alternative to accessing HPC resources via SSH. For this purpose only a web browser is required. Within the website source code of different programming languages can be edited and executed. Furthermore different user interfaces and terminals are available.

Contents (hide)

- 1 Short description of Jupyter
- 2 Access requirements
- 3 Login process
- 4 Selection of the compute resources
 - 4.1 Prioritized access to computing resources on bwUniCluster 2.0
- 5 JupyterLab
 - 5.1 Menu bar
 - 5.2 Left sidebar
 - 5.3 Main working area
 - 5.4 Classic Notebook
- 6 Log out
- 7 Selection of software
 - 7.1 Software Stacks for Jupyter
- 8 Installation of further software

1 Short description of Jupyter

Jupyter is a web application, central component of Jupyter is the **Jupyter Notebook**. It is a document, which can contain formatted text, executable code sections and (interactive) visualizations (image, sound, video, 3D views).

The Jupyter notebooks are executed in an interactive session on the compute nodes of the respective cluster. Access is via any modern web browser. Data is prepared and visualized on the server and therefore does not have to be transmitted over the network. Only the resulting text, image, sound and video data is transmitted. Starting point of a Jupyter session is the HOME directory of the user on the respective cluster.

JupyterLab is a modern user interface, within which one or more Jupyter notebooks can be opened, edited and executed. The individual notebooks can be arranged as tabs or tiled. JupyterLab is the standard user interface. Besides JupyterLab the classic notebook user interface is available, in which only one Jupyter notebook per browser tab can be opened at a time.

A **Jupyter Kernel** describes a separate process, in which one Jupyter Notebook is executed at a time. Different kernels are available for different programming languages or language versions.

NHR@KIT User Documentation

Start HoreKa and HAICORE Future Technologies Partition (FTP) Continuous Integration Jupyter

Jupyter

Overview

Getting access

Using Jupyter

Logging In/Resources

2-Factor Authentication

JupyterLab Basics

Advanced topics

Software Stacks

Overview

Jupyter enables interactive supercomputing and is an alternative to accessing HPC resources via SSH. It allows different programming languages and runtimes to be used within a web-based environment.

While the frontend runs in the browser on the client, the commands are executed on the HPC systems. Jupyter therefore allows users to use resources (e.g. GPUs) which are not present on the client side, and to use software without having to install it locally. Only a web browser is required.

Jupyter can be accessed using the following URLs, depending on which hardware resource should be used:

- HoreKa
- bwUniCluster 2.0
- HAICORE

Table of contents

Short description of Jupyter

Motivation

Why Jupyter?

HPC – “Classical”

- SSH
- **High Entry Hurdles**
 - Linux
 - Tools for connection and data transfer
 - Choice of resources
- Remote-Visualization
 - VNC? X11?
- State of the art for **advanced requirements!**

Why Jupyter?

HPC – Jupyter

- Web browser
 - No additional software
 - No data transfer for analysis
- Low Entry Hurdles
- Intermediate performance requirements
- Interactive visualization of data
- Prototyping

Project Jupyter

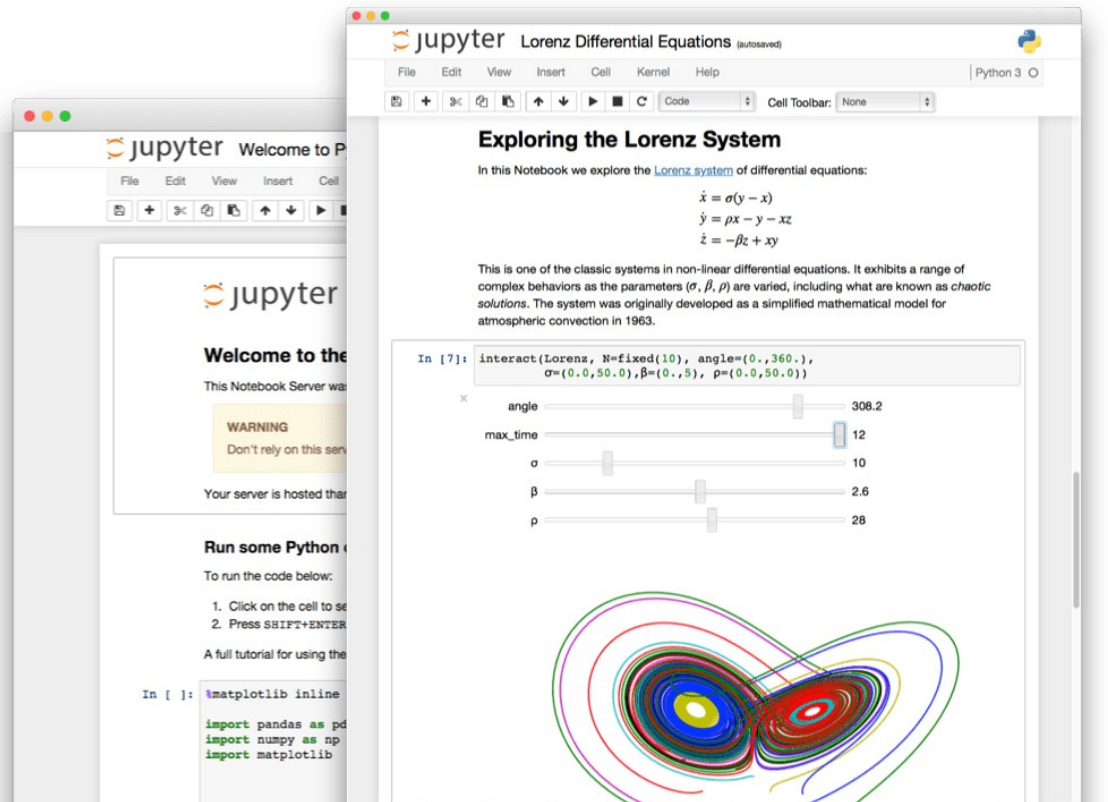
Project Jupyter

- Spin-off from project IPython
- Jupyter: Core languages
 - Julia
 - Python notebook
 - R
- Language agnostic
- Jupyter kernels
 - IPython
 - IJulia
 - IRKernel
 - >100 other kernels



Jupyter Notebook

- Open-source web application
- Create and share documents
 - Live code
 - Equations
 - Visualizations
 - Narrative text
- Execute code in browser
 - ... on HoreKa
- .ipynb file
 - JSON document

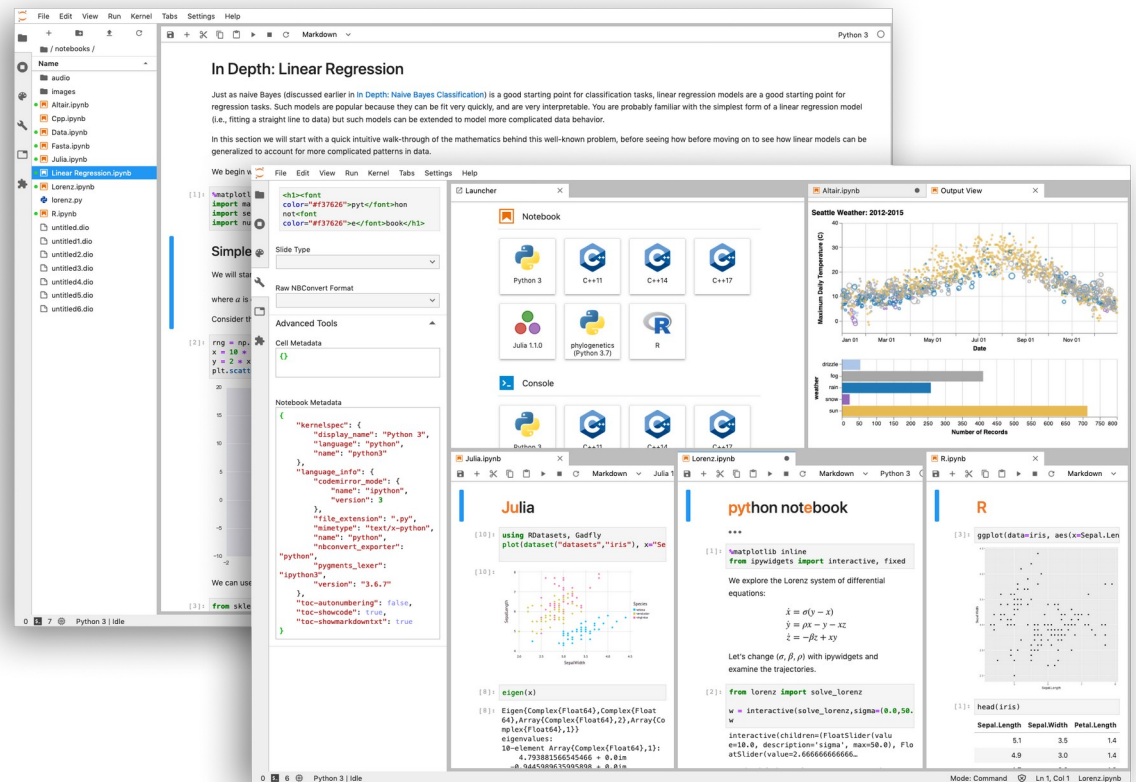


<https://jupyter.org/>

JupyterLab

- User interface for Project Jupyter
- Arrange documents/activities in tabs/blocks

- Notebook
- Terminal
- Text editor
- File browser
- Rich outputs
- ...



<https://jupyter.org/>

JupyterHub

- Multi-user server for Jupyter Notebooks
- User management and authentication
- Spawning and proxying
- HPC context
 - Choice of resources
 - Slurm integration
 - Authentication



Jupyter: How-To

Local Jupyter

■ Requirements:

- Python / Anaconda (Windows users)
- (nodejs + npm: JupyterLab extensions)

■ Install locally

```
python -m venv env  
source env/bin/activate  
pip install jupyterlab
```

■ Start

- Jupyter Notebook: `jupyter notebook`
- JupyterLab: `jupyter lab`

■ Use

- Open <http://127.0.0.1:8888> in web browser

Live Demo

■ Install

```
python -m venv env  
source env/bin/activate  
pip install jupyterlab
```

Additional Packages

```
pip install matplotlib ipyml  
jupyter labextension install jupyter-matplotlib
```

■ Start

■ Notebook

```
jupyter notebook
```

■ JupyterLab

```
jupyter lab
```

■ Get the notebook:

```
git clone https://github.com/pinae/BresenhamLidar
```

Jupyter@HoreKa

- Login to HoreKa, start **interactive job**

```
ssh <userID>@hk.scc.kit.edu  
salloc -p accelerated --gres=gpu:4 --time=30:00
```

- Wait till job is running, remember compute node hostname (nodeID) and install and/or **start** Jupyter Notebook or JupyterLab ...

```
module load jupyter/base  
jupyter notebook --no-browser --port=8888 --ip 0.0.0.0
```

- From your local terminal: Establish **SSH tunnel** to compute node

```
ssh -L 8888:<nodeID>:8888 hk.scc.kit.edu
```

- Open in web browser: <http://127.0.0.1:8888>

JupyterHub@HoreKa

Registration Process – HoreKa

■ Registration @ HoreKa

- [Online proposal form](#) (Jards)
- Peer reviewed proposal
- HoreKa access form for each coworker
- [Web registration](#)

■ Set **service password**

- FeLS → HoreKa → Set service password

■ Register a software or hardware token (alias **2FA**)

- [FeLS → My Tokens](#)
- KIT users: <https://my.scc.kit.edu/token>

Accessing HoreKa

- Only within **network**
 - ... of **KIT**
 - ... of your **home institution**
- ... otherwise establish **VPN** connection

- **SSH**
 - `ssh <userid>@hk.scc.kit.edu`
 - TOTP prompt (first)
 - Service password (second)

- **Jupyter**
 - (modern) Web browser: <https://hk-jupyter.scc.kit.edu>
 - **ONE successful login via SSH required**
 - ... otherwise there is no \$HOME
 - ... spawning will fail (timeout)

Selection of Resources – Normal

Select your resources

The grayed out fields contain a reasonable preselection of resources.
Other values can be selected in advanced mode.

Number of CPU-cores: 76 ▾

Number of GPUs: 0 ▾

Runtime: 0.5 hour ▾

Partition: cpuonly ▾

Amount of memory: 237GB ▾

JupyterLab-Basemodule: jupyter/tensorflow ▾

Advanced Mode: ☐

Spawn

■ “Normal” mode

- Number of CPU cores **OR** GPUs
- Runtime
- Jupyter Basemodule
- Grayed out fields: **Sane pre-selection** of resources

■ Spawn

- Starts JupyterLab in interactive Slurm session
- Connects/proxies to that session

Selection of Resources – Advanced

Select your resources

The grayed out fields contain a reasonable preselection of resources.
Other values can be selected in advanced mode.

Number of CPU-cores: 76 ▾

Number of GPUs: 0 ▾

Runtime: 0.5 hour ▾

Partition: cpuonly ▾

Amount of memory: 237GB ▾

JupyterLab-Basemodule: jupyter/tensorflow ▾

Advanced Mode: ☒

Reservation:

Account:

Mount LSDF: ☐

Use BEEOND: ☐

Spawn

- „Advanced“ mode
 - Free choice of resources
 - No grayed out fields
 - No auto reservation
- Reservation
- Account
- LSDF
- BEEOND

Jupyter Software Stacks

■ Lmod modules

- `jupyter/minimal`
- `jupyter/base`
- `jupyter/tensorflow`

■ JupyterLab lives inside venv

- `--system-site-packages` **enabled/visible**
- Possible **interference** with `pip --user` installs (!)

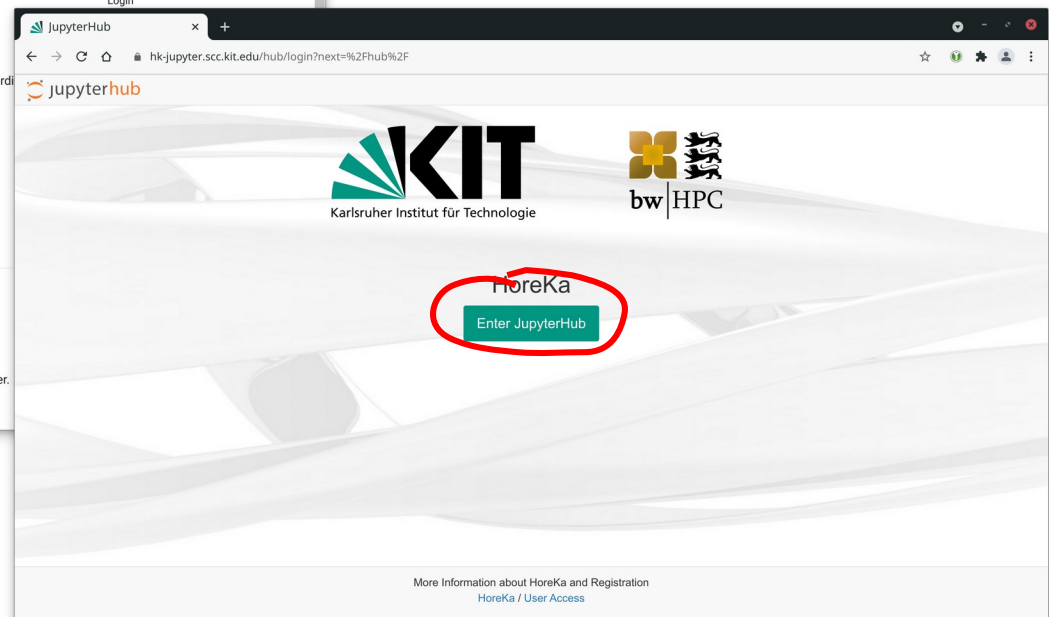
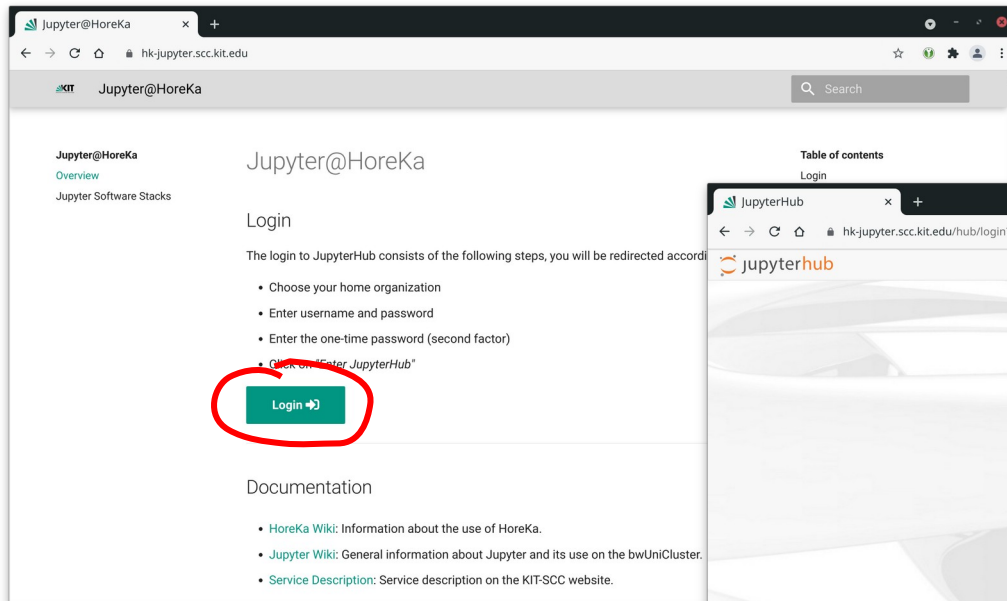
■ Access via

- Drop-down menu in JupyterHub: “**JupyterLab-Basemodule**”
- `module load jupyter/base` or `jupyter/tensorflow`

Login: Step-by-Step

Step-by-Step Jupyter@HoreKa (1/4)

- Go to <https://hk-jupyter.scc.kit.edu> and click on „Login“
 - ...or go directly to <https://hk-jupyter.scc.kit.edu/hub/login>
- Click “Enter JupyterHub”



Step-by-Step Jupyter@HoreKa (2/4)

- Choose your **home organization** and continue
- Login to your home organization
 - **Username + password**

FELS - Federated Login Service

Willkommen

Sie wurden von einem Dienst hierher weitergeleitet, um sich zu authentifizieren:
HoreKa Jupyter

Heimatorganisation merken: ☐

Föderation: Alle

Suchfilter:

Heimatorganisation:

- Albert-Ludwigs-Universität Freiburg
- Anno-Wegener-Institut, Helmholtz-Zentrum für Polar- und Meeresforschung (A)
- Badische Landesbibliothek
- Bibliotheksservice-Zentrum Baden-Württemberg
- DHBW Lörrach
- DHBW

Web Anmeldedienst

KIT
Karlsruher Institut für Technologie

Shibboleth Identity Provider

Anmelden

Sie wurden von dem Serviceprovider **Föderierte Dienste am KIT** hierher weitergeleitet und befinden sich nun auf einem Server des KIT. Bitte melden Sie sich mit Ihrem KIT-Account (z.B. ab1234 als Mitarbeiter oder uxxxx als Student) und Ihrem Passwort an.

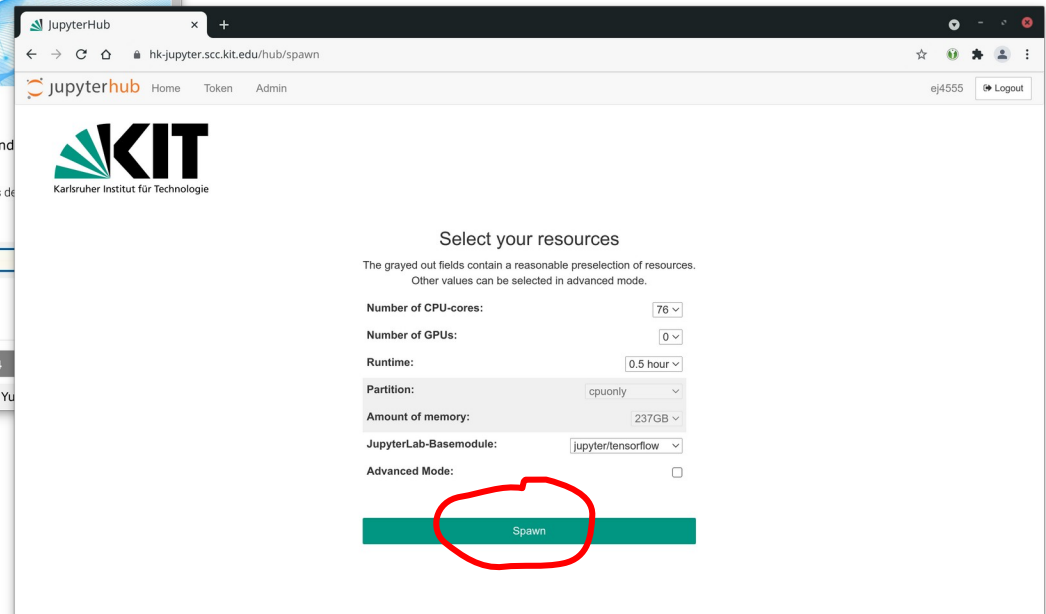
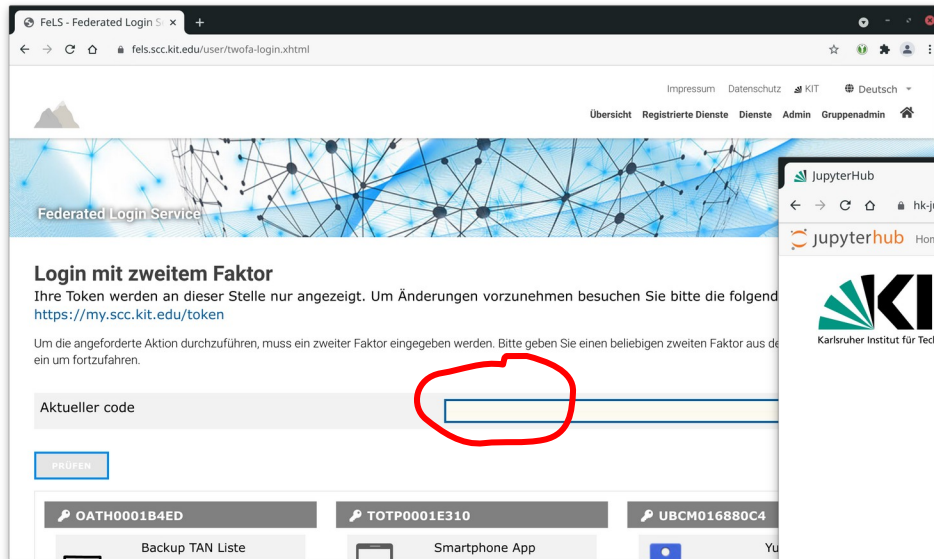
Benutzername:

Passwort:

WINDOWS LOGIN VERWENDEN

Step-by-Step Jupyter@HoreKa (3/4)

- Login to your home organization
 - 2FA
- Select resources and click “Spawn”



Step-by-Step Jupyter@HoreKa (4/4)

- Spawning **may take a while**
 - ... timeout after 10 minutes
- JupyterLab runs **on compute node** on HoreKa

The image shows two overlapping browser windows. The background window is JupyterHub, displaying a message: "Your server is starting up. You will be redirected automatically when ready." Below this, it says "Cluster job running... waiting to..." and shows an "Event log" section. The foreground window is JupyterLab, showing a file explorer on the left with "LOADED MODULES" (compiler/gnu/10, devel/cuda/11.4, dot, jupyter/tensorflow/..., mpl/openmpi/4.0, numlib/mkl/2020) and "AVAILABLE MODULES" (cae/openfoam/v1612+, cae/openfoam/v1712, cae/openfoam/v1806, cae/openfoam/v1812, cae/openfoam/v1906, cae/openfoam/v1912, cae/openfoam/v2006, cae/openfoam/v2012, cae/openfoam/v2106, cae/openfoam/6, cae/openfoam/7, cae/openfoam/9, cae/ansys/2020R1, cae/ansys/2020R2, cae/lisdyna/12.0.0, cae/lisdyna/931). The main area shows a code editor with Python code for TensorFlow, including imports, version printing, physical device listing, GPU configuration, and a reduce_sum operation. The output shows TensorFlow version 2.6.0 and a list of physical devices (CPU and GPUs).

Hands-On: Login

~10min

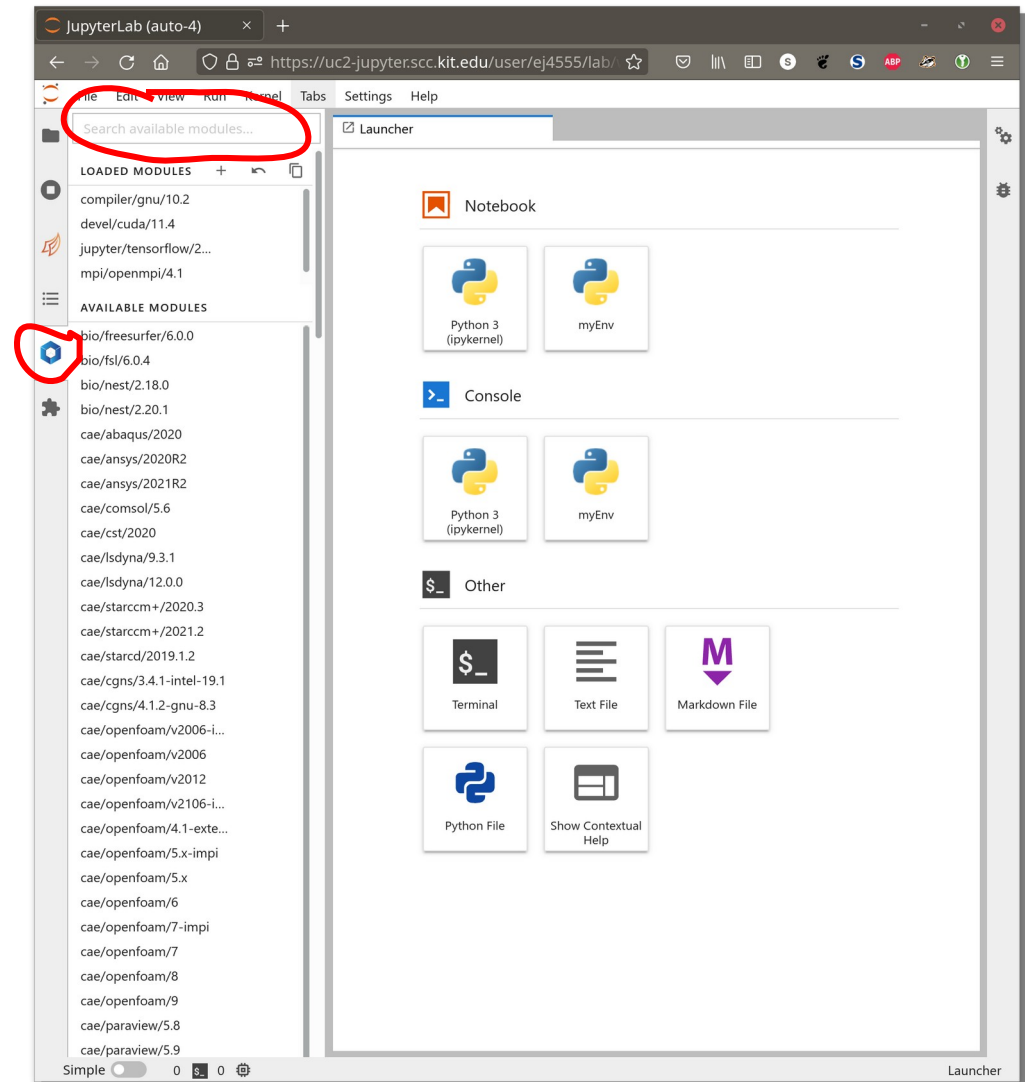
Hands-On: Login

- Start a session at <https://hk-jupyter.scc.kit.edu>
 - Hint:
choose “accelerated” or “dev_*” partitions

Software and Kernels

Select Software

- Activate Lmod software modules
 - blue button
 - search field
- Kernel restart required



Add Python Packages + Custom Kernel

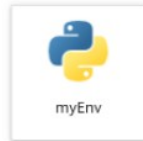
■ Python: Use virtual environments

```
python -m venv myEnv  
source myEnv/bin/activate  
pip install <myPackage>
```

■ Install kernel → [IPython docs](#)

```
python -m ipykernel install \  
    --user \  
    --name myEnv \  
    --display-name "Python (myEnv) "
```

■ You will get this:



R and Julia Kernel

You want to use R

- Load module
`math/R`
- Open terminal
- `R`
 - `install.packages('IRkernel')`
 - `IRkernel::installspec()`



You want to use Julia

- Load module
`devel/julia/1.6.2`
- Open terminal
- `julia`
 - `]`
 - `add IJulia`



Resetting everything

rm -r ...

■ Jupyter

■ ... ~/.local/share/jupyter/kernels

■ ... ~/.jupyter

■ Python packages

■ ... ~/.local/lib/python3.*

■ R

■ ... ~/R

■ Julia

■ ... ~/.julia

Check state

■ Bash

■ ~/.bashrc

Hands-On: First Steps

~10min

Hands-On: First Steps

- Run the c't example on HoreKa

- Hint 1:

- ```
git clone https://github.com/pinae/BresenhamLidar
```

- Hint 2:

- ```
Replace %matplotlib notebook by %matplotlib widget
```

- Install a Julia Kernel

- Compute `1+1` in a Julia Notebook

- Try out some examples, e.g.: <https://rosettacode.org/wiki/Factorial#Julia>

Outlook

WIP: BYO Jupyter Container

- Connect **containerized** Jupyter with JupyterHub@HoreKa
- **Docker images** from any registry

- For complicated/intrusive software stacks
- Optimized software stacks

- Intel, e.g.
`intel/intel-optimized-tensorflow`
- Nvidia, e.g.
`nvcr.io#nvidia/tensorflow:21.10-tf2-py3`
- AMD, e.g.
`rocm/tensorflow:rocm4.3.1-tf2.6-dev`

- Possible **root access** (sic!)

- **Yes, you can**

```
sudo apt-get install <myPackage>
```

Select your resources

The grayed out fields contain a reasonable preselection of resources.
Other values can be selected in advanced mode.

Number of CPU-cores: 1 ▾

Number of GPUs: 0 ▾

Runtime: 0.5 hour ▾

Partition: single ▾

Amount of memory: 4GB ▾

JupyterLab-Basemodule: Container Mode ▾

Advanced Mode: ☐

Container Mode: ☒

--container-image: jupyter/base-notebook

--container-name:

--container-mount-home: ☒

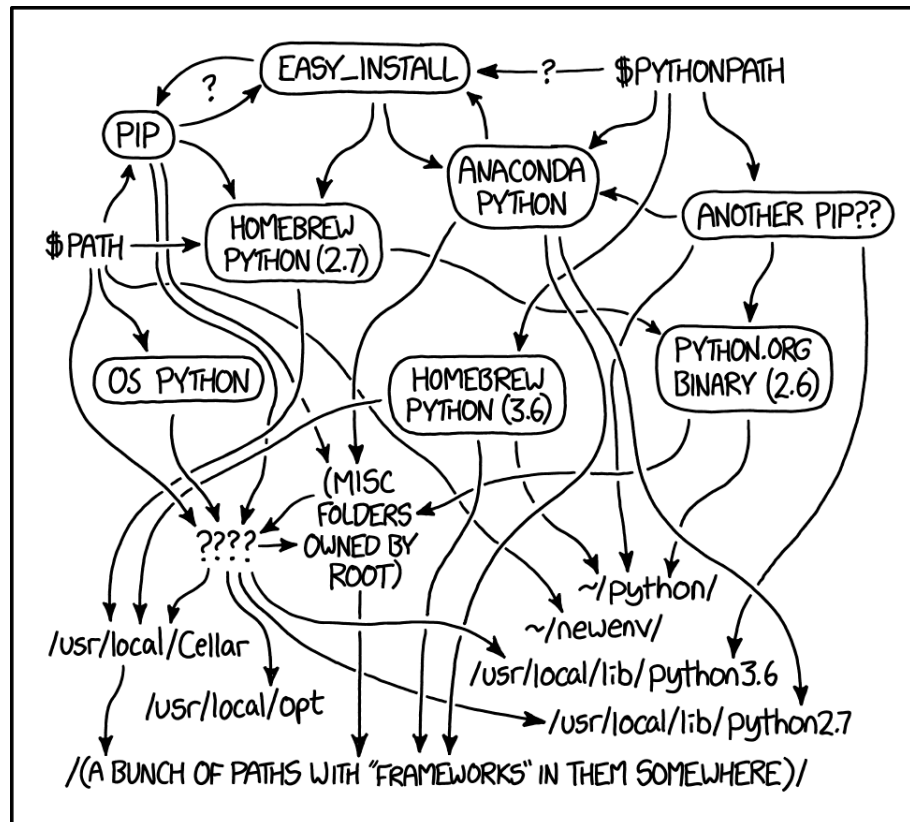
--container-mounts=<default mounts>: ☒

--no-container-remap-root: ☒

Spawn

Thank you for your attention!

Questions?



<https://xkcd.com/1987>

MY PYTHON ENVIRONMENT HAS BECOME SO DEGRADED
THAT MY LAPTOP HAS BEEN DECLARED A SUPERFUND SITE.