ScaDS.AI
DRESDEN LEIPZIG

CENTER FOR SCALABLE DATA ANALYTICS AND ARTIFICIAL INTELLIGENCE

JupyterHub Training

Mariela Sanchez, Adnan Haidar, Elias Werner

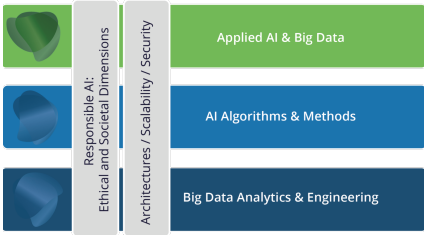
Dresden, 27th of October 2022

GEFÖRDERT VOM
Bundesministerium für Bildung und Forschung

SACHSEN
Diese Maßnahme wird gefördert durch die Bundesregierung aufgrund eines Beschlusses des Deutschen Bundestages.
Diese Maßnahme wird mitfinanziert durch Steuermittel auf der Grundlage des von den Abgeordneten des Sächsischen Landtags beschlossenen Haushaltes.

Presenters

ScaDS.AI
DRESDEN LEIPZIG



Mariela Sanchez,
Research Associate, ZIH
Data Science for synthetic data evaluation

Adnan Haidar,
Research Associate, ScaDS.AI
Data Science, Robotics Tactile Sense using Artificial Intelligence

Elias Werner,
Research Associate, ScaDS.AI
performance analysis of data science workflows

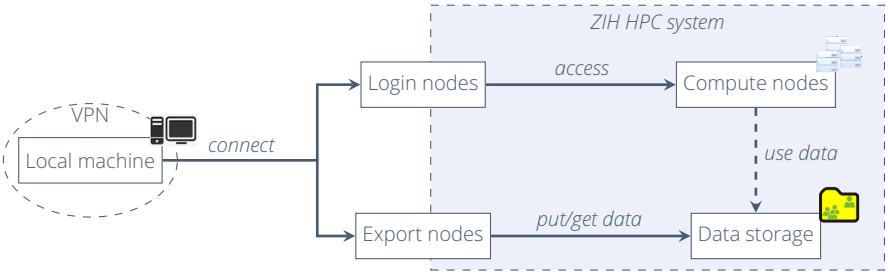
https://scads.ai/

1 Introduction – Agenda

1 Recap: What is an HPC machine?

- Cluster of many single nodes (single "computers") for pooling their resources (computation and memory)
- Job scheduler *Slurm* manages resource/hardware allocation
- Software environment can be set up easily via a module system
- Collaborative working spirit, shared machine
- HPC project required for interaction

Technical workflow:

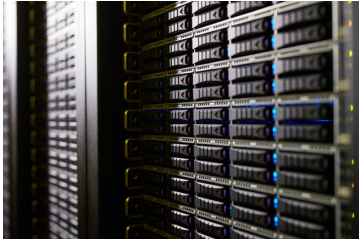


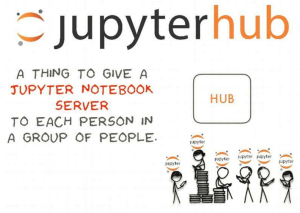
1 HPC system - Taurus

- **Name**: HPC-Cluster Taurus
- **Model**: The second high-performance computing/memory complex (HRSK-II)
- **Facility**: Center for Information Services and High Performance Computing (ZIH)
- **Details**:
  - More than 60,000 cores,
  - 720 GPUs,
  - Flexible storage hierarchy with about 16 PB total capacity,
  - Linux, shared login nodes, filesystems, batchsystem Slurm,
  - Perfect platform for highly scalable, data-intensive and compute-intensive applications.

1 JupyterHub - a Good Mean for Interacting with HPC

- Jupyter useful for data analysis
- Jupyter as an interface to an HPC system for many users
- Users access to complex computing infrastructures from the JupyterHub
- Jupyter provides remote access to Jupyter Notebooks





Taken from: geohackweek.github.io/Introductory/05-jupyter-tutorial/

1 Polls

Asking the audience:

1) What is your main discipline of work?
    A. Humanities and Social Sciences
    B. Natural Sciences
    C. Life Sciences
    D. Engineering Sciences
2) Which programming language you use mostly in Jupyter?
    A. Python
    B. R
    C. Julia
    D. Others
3) What is the OS you use to initialize and use Jupyter?
4) What are your main reasons for joining this training event?

1 Polls

Asking the audience:

1) What is your main discipline of work?
   A. Humanities and Social Sciences
   B. Natural Sciences
   C. Life Sciences
   D. Engineering Sciences

2) Which programming language you use mostly in Jupyter?
   A. Python
   B. R
   C. Julia
   D. Others

3) What is the OS you use to initialize and use Jupyter?

4) What are your main reasons for joining this training event?

1 Polls

Asking the audience:

1) What is your main discipline of work?
   A. Humanities and Social Sciences
   B. Natural Sciences
   C. Life Sciences
   D. Engineering Sciences

2) Which programming language you use mostly in Jupyter?
   A. Python
   B. R
   C. Julia
   D. Others

3) What is the OS you use to initialize and use Jupyter?

4) What are your main reasons for joining this training event?

1 Polls

Asking the audience:

1) What is your main discipline of work?
   A. Humanities and Social Sciences
   B. Natural Sciences
   C. Life Sciences
   D. Engineering Sciences

2) Which programming language you use mostly in Jupyter?
   A. Python
   B. R
   C. Julia
   D. Others

3) What is the OS you use to initialize and use Jupyter?

4) What are your main reasons for joining this training event?

1 Structure of the Presentation

- Jupyter
  - Features
  - Usability
  - Functionalities
- HPC basics
  - Access
  - Data management
  - Software management
- Collaborative working with HPC and Jupyter notebook
  - Shared system principles
  - Workflow
  - Useful features
- Hands-on session

1 Intention of the Presentation

- Making things understandable esp. for **frequent JH users** including teachers and researchers.
- Complete **workflow examples** are shown, here: based on **Python code** (can be used as blueprint for other software/tools).
- From the **user's perspective**: What are the most important things to work with JH on HPC?

Note

We do **not** want to show everything that is possible. We want to show, what is needed to get started

Note

We do **not** want to show *the* complete architecture and advanced properties of Jupyter. We want you to comprehend how Jupyter works and HPC principles behind JH tool with concrete examples

Hint

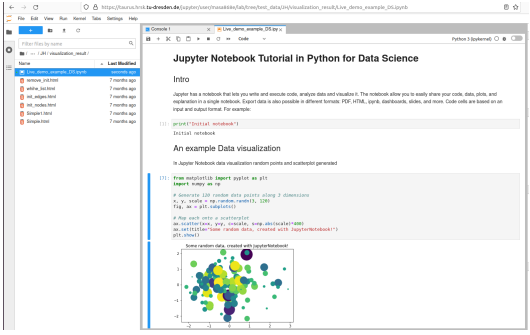
Please ask and interrupt our presentation immediately if something is not clear! Use your mic or the chat.

2 Jupyter overview – Agenda

2 Jupyter Overview - Why Jupyter is great

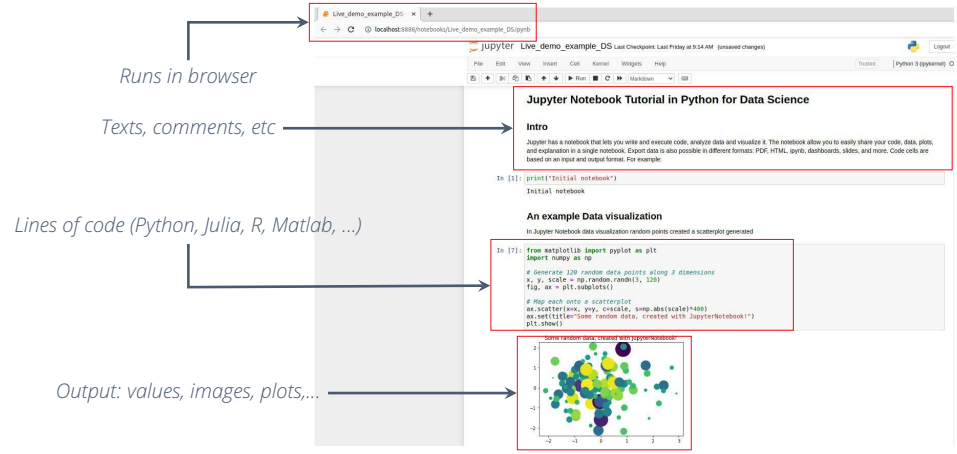
Jupyter Notebook documents:

- Dynamic notebook documents include:
  - Plots
  - Images
  - Video
  - Equations
  - Narrative text
  - ...
- Notebook web application enables to:
  - Run and edit code
  - Build hierarchical documents
  - Computations results as PDF, HTML, etc

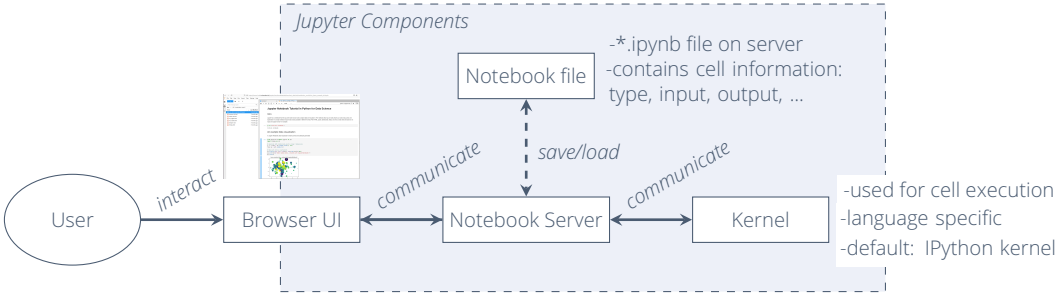


Jupyter Notebook and Web Interface

2 Jupyter Overview - Why Jupyter is great



2 Jupyter Overview - Why Jupyter is great



structure according to: jupyter.readthedocs.io/.../content-architecture

2 Jupyter Overview - Why Jupyter is great

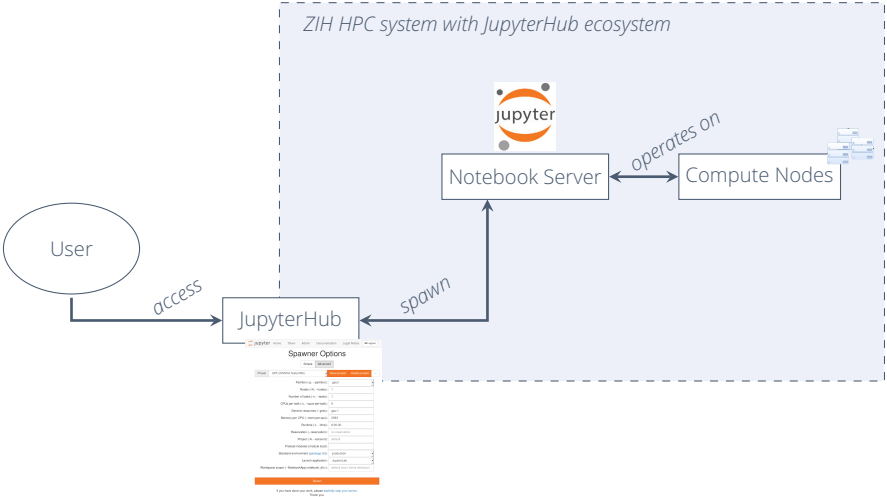
Jupyter Kernels:

- IPython: basic Python kernel with features for interactive computing
- IRKernel: basic R kernel
- IJulia: basic Julia kernel
- MATLAB Kernel: Basic kernel for Matlab language
- ITorch: IPython kernel with visualizations of images
- IPykernel: another Python kernel

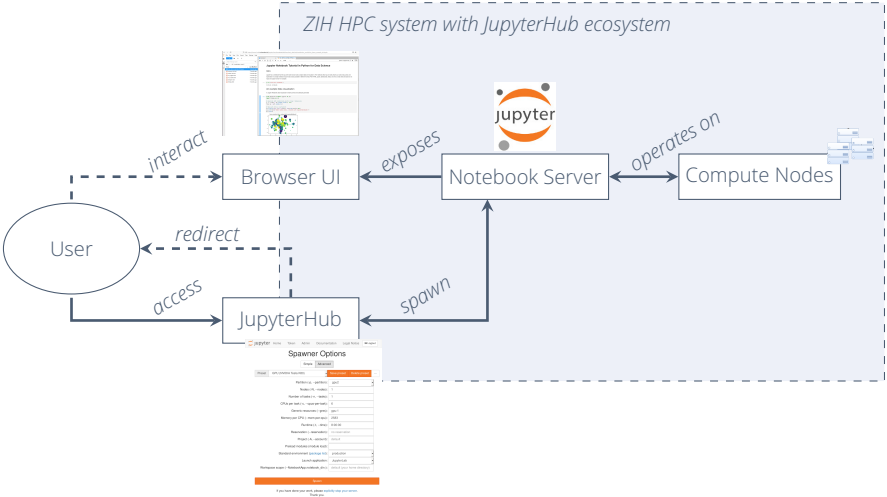
Jupyter kernels

> **Note**
>
> Jupyter supports over 100 programming languages. For more details: https://github.com/jupyter/jupyter/wiki/Jupyter-kernels

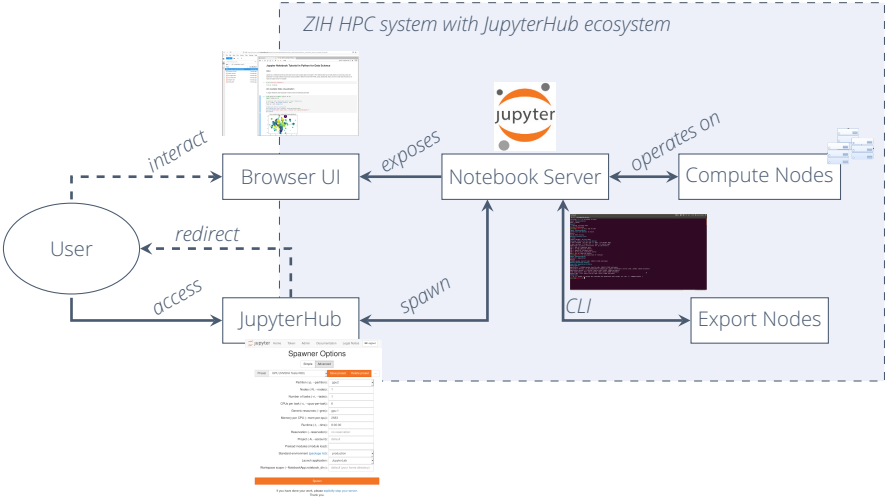
2 Access the ZIH System via JupyterHub



2 Access the ZIH System via JupyterHub



2 Access the ZIH System via JupyterHub



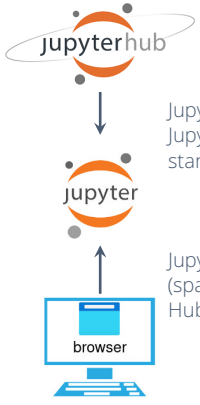
2 Access the ZIH System via JupyterHub

One single node of Jupyter:

- Each user a whole new instance of Jupyter software
- The instance has all necessary features to interact with the HPC:
  - Web browser
  - Jupyter Notebook
  - Kernel
  - others: Jupyter lab, etcetera

Note

Jupyter Hub has many interesting properties to explore for intermediate and basic users.

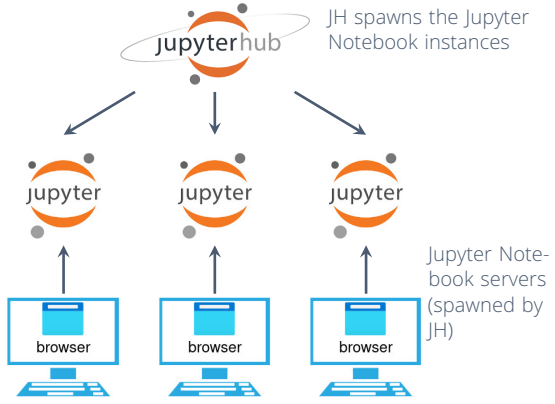


Adapted from: opencredo.com/blogs/writing-a-custom-jupyterhub-spawner/

2 Access the ZIH System via JupyterHub

The ecosystem of JH under HPC:

- A multi-user Hub with multiple instances of single-users
- Set up in HPC system
- Built for multiple user log in
- The instances are single Jupyter structures
- A interactive interaction with HPC in every instance



Adapted from:
opencredo.com/blogs/writing-a-custom-jupyterhub-spawner/

2 Jupyter Scenario: Research

Paulina is a researcher, she has some of her experiments ready

1

She wants to share them with other researchers

Adapted from:
Beg, Marijan, et al.
"Using Jupyter for reproducible scientific workflows."

2 Jupyter Scenario: Research

Paulina is a researcher, she has some of her experiments ready

1

She wants to share them with other researchers

2

She describes her experiments as a Jupyter Notebook

Her Jupyter Notebook is a mix of code, visualizations, annotations, among others

Adapted from:
Beg, Marijan, et al. "Using Jupyter for reproducible scientific workflows."

2 Jupyter Scenario: Research

Paulina is a researcher, she has some of her experiments ready

1

She wants to share them with other researchers

She describes her experiments as a Jupyter Notebook

2

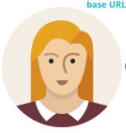
Her Jupyter Notebook is a mix of code, visualizations, annotations, among others

She sends the link to the other researchers

https://taurus.hrsk.tu-dresden.de/jupyter/hub/user-redirect/notebooks/demo.ipynb

base URL and page „user-redirect" path to file

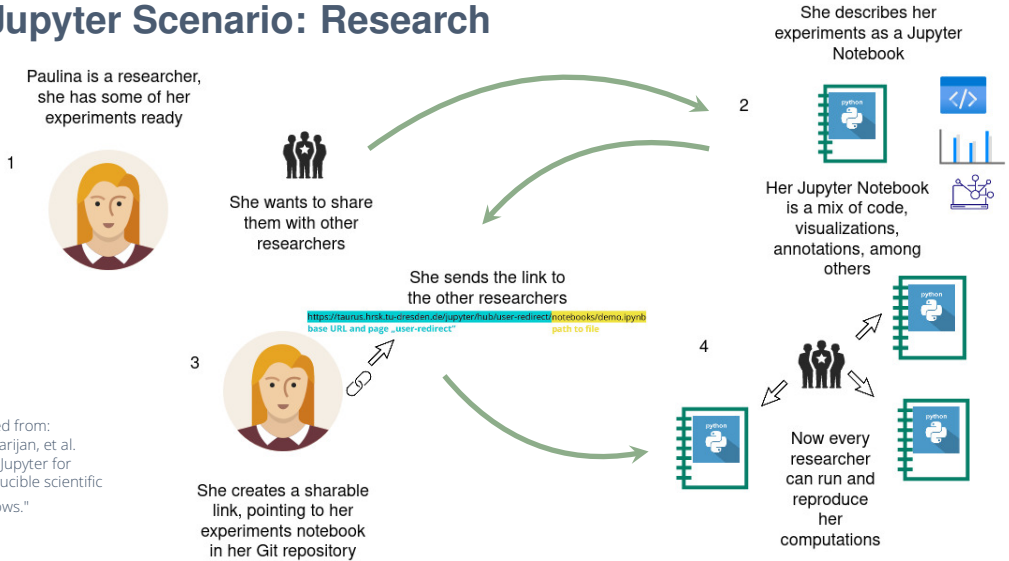
3



She creates a sharable link, pointing to her experiments notebook in her Git repository

Adapted from:
Beg, Marijan, et al. "Using Jupyter for reproducible scientific workflows."

2 Jupyter Scenario: Research

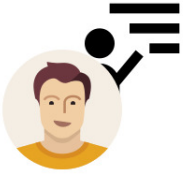


Adapted from:
Beg, Marijan, et al. "Using Jupyter for reproducible scientific workflows."

2 Jupyter Scenario: Teaching

John is a University Professor

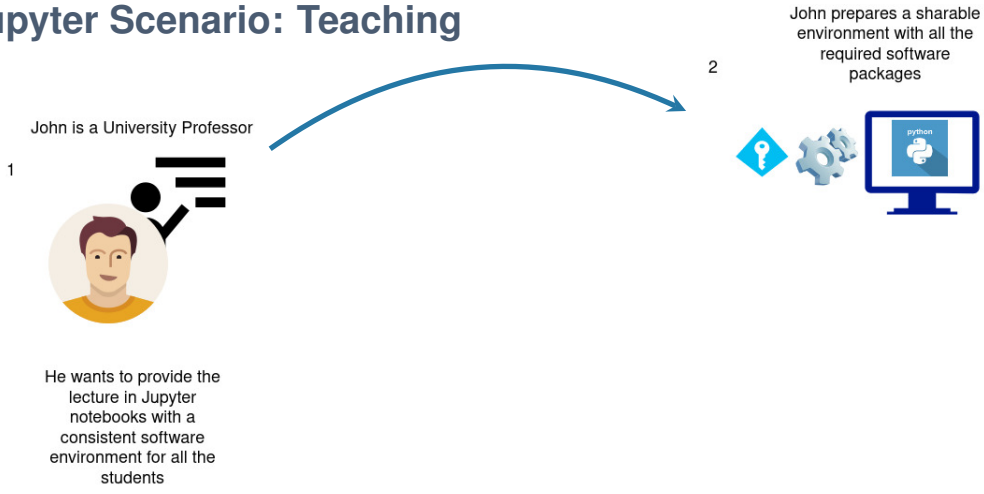
1



He wants to provide the lecture in Jupyter notebooks with a consistent software environment for all the students

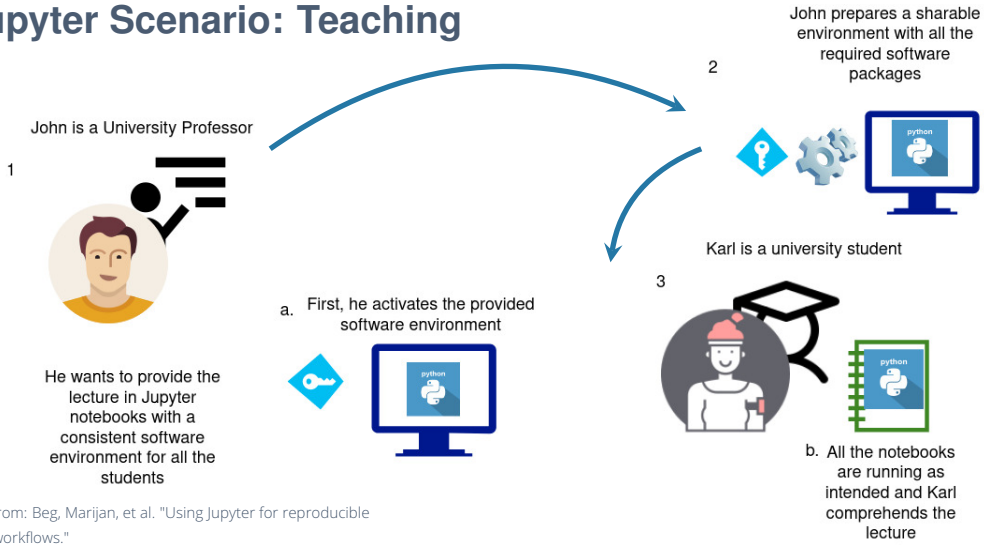
Adapted from: Beg, Marijan, et al. "Using Jupyter for reproducible scientific workflows."

2 Jupyter Scenario: Teaching



Adapted from: Beg, Marijan, et al. "Using Jupyter for reproducible scientific workflows."

2 Jupyter Scenario: Teaching



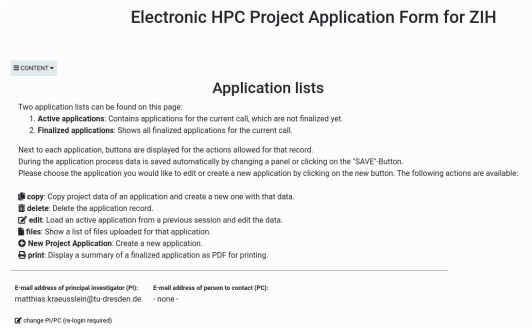
Adapted from: Beg, Marijan, et al. "Using Jupyter for reproducible scientific workflows."

3 Access and Basics of the ZIH HPC system – Agenda

3 JupyterHub Project on the ZIH System

Working on the JupyterHub system requires an HPC project. An HPC project on the ZIH system includes:

- project directory and login with existing or without existing ZIH Login
- project members (at least project leader and project administrator)
- resource quotas for compute time (CPU/GPU hours) and storage



3 JupyterHub Project on the ZIH System

There are different possibilities to work with HPC:

- create a new project
  - fill the application form: `https://hpcprojekte.zih.tu-dresden.de/`
  - find additional information in the wiki: `https://doc.zih..../application/project_request_form/`
- Join an existing project, new researchers in an existing project, teaching purposes

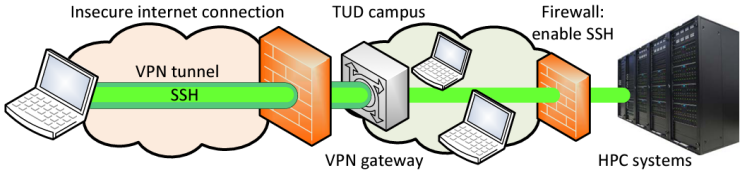
Hint

Kindly mention the HPC resource usage in the acknowledgment section of all publications `https://doc.zih.tu-dresden.de/application/acknowledgement/`

3 VPN to Access Services

To access ZIH services:

- from within the TU Dresden campus via secure shell (ssh).
- install VPN tool at your local machine.
  - OPENVPN client (recommended variant).
  - Cisco AnyConnect Secure Mobility Client

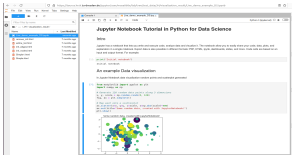


3 Accessing the JupyterHub

Different ways to access the ZIH system:

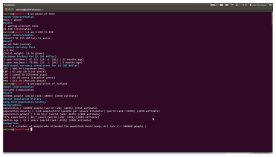
JupyterHub

- browser based approach
- most easiest way for beginners



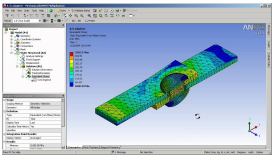
SSH connection

- "classical" approach
- command line interface (CLI) knowledge is necessary



desktop visualization

- esp. in case of using GUI-based software
- e.g. Ansys, Vampir,...



Hint

For getting in touch with the system and also development purposes the approach with JuypterHub is recommended.

3 Accessing the JupyterHub

To access JupyterHub to use:

1. use the browser link (https://taurus.hrsk.tu-dresden.de/jupyter)
2. Start session by clicking Start my server button

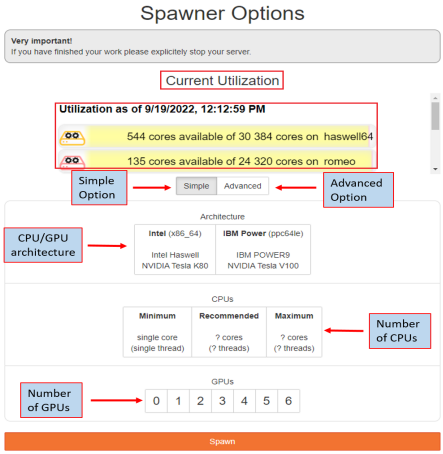
Allocation of resources via the GUI Spawner Option

- Simple Option
- Advanced Option

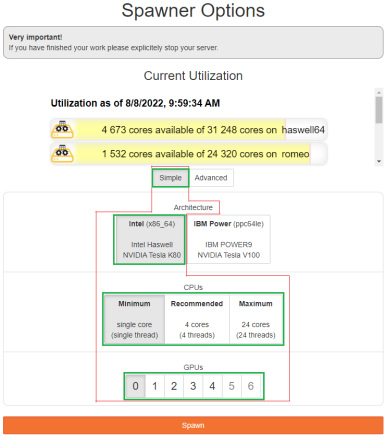
Utilization Bar

- Number of GPUs/CPUs available per node

More information in the wiki: `https://doc.zih..../access/jupyterhub`



3 Simple Option Resources



Simple form offers most important settings to start quickly:

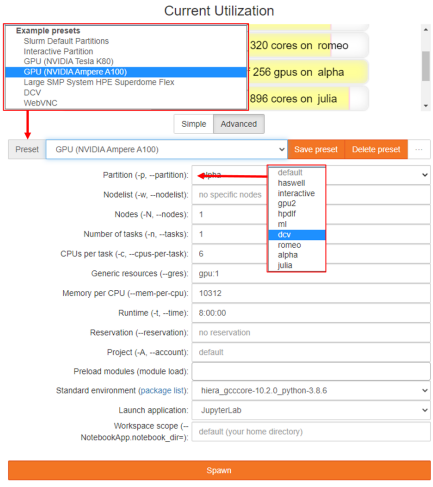
- Intel Hashwell or IBM Power9
- Numbers of CPUs cores and threads
- GPUs

Architecture	IBM Power9	Intel Haswell
CPUs per node	2 x IBM Power9 (22 cores@2.80 GHz)	2x Intel Xeon 2680v3 (12 cores@2.50GHz
Memory per node	256GB RAM DDR4 2666MHz	64GB RAM 128 GB SSD
GPUs per node	6x NVIDIA VOLTA V100	4x NVIDIA Tesla K80

3 Accessing the JupyterHub Advanced Option

Accessing the JupyterHub Advanced Option:

- Preset Option
  - Save own configurations as additional presets
  - Those are saved in browser or lost
  - Use import/export features to save as text
  - You can delete preset



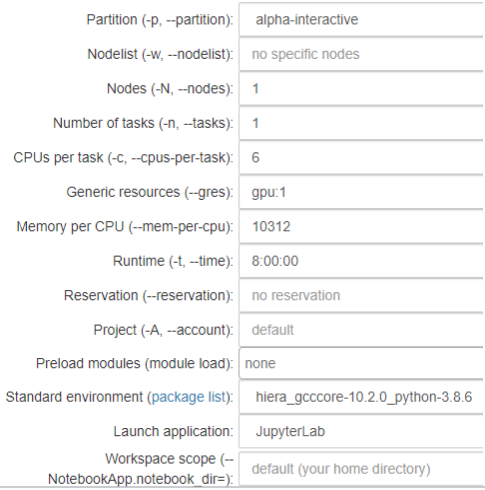
3 JupyterHub Advanced Option

Customise available resources via advanced form:

- Every parameters of advanced form can be set.
- Partitions, Nodes, Number of tasks.
- CPU, GPUs, Memory and Run-Time.
- Project, Modules.
- Standard environment, Workspace.

Hint

The resource parameters refer to the Slurm parameters of the `srun`, `salloc` and `sbatch` commands when using an ssh connection to the HPC system.



3 Software Management by Loading Modules

A module system:

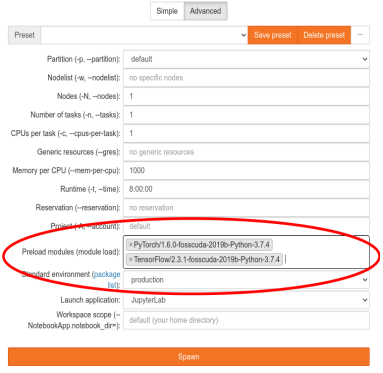
- is a user interface that provides utilities for the dynamic modification of a user's environment (e.g. $PATH variable)
- enables to smoothly switch between different versions of installed software packages and libraries
- in Jupyter, modules are available via `Preload modules` parameter

Hint

More information about module management: `https://doc.zih..../software/modules/`

3 Software Management - Preload Modules

- The `Preload modules` parameter for loading modules when spawning the JupyterHub server
- Preloaded modules will be available during the whole session
- After spawning, loading modules in the Jupyter terminal is not possible Only preloaded modules will be available in the Jupyter notebook files.



3 Module Commands - Example

A **module** usage is quite straight forward and the following table lists the basic commands.

Example

```
# Show all module options
marie@local$ module help

# List active modules in the user environment
marie@local$ module list

# Remove modules from the user environment
marie@local$ module purge

# Load module "modname" in the user environment
marie@local$ module load <modname>

# Remove module "modname" from the user environment
marie@local$ module unload <modname>
```

3 Standard Environments in JupyterHub

Standard environment through advanced option open a list with all included python-packages:

- To access the list of packages just select the "package list" link in the spawner
- Closely related to module environments

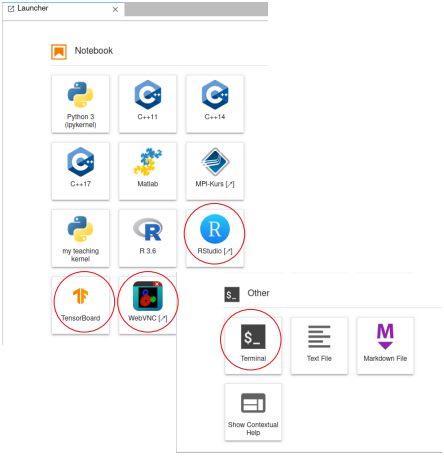
Hint

This list shows all packages of the currently selected conda environment. This depends on your settings for partition (CPU architecture) and standard environment. The default Python kernel uses conda environments.

List of available standard environments, namely:
`https://doc.zih..../access/jupyterhub/standard-environments`

3 Terminal, Tensorboard, WebVNC, RStudio

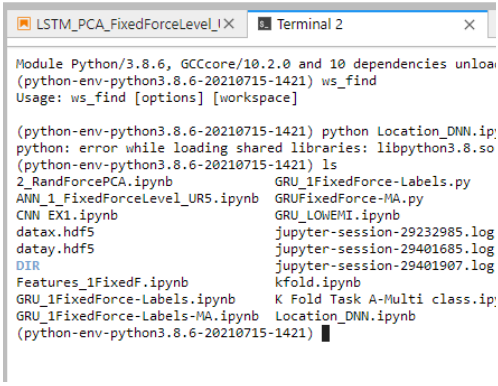
- JupyterHub launcher supports additional tools:
  - **Terminal:** access the compute node via CLI
  - **WebVNC:** spawn a graphical user interface on the compute node
  - **Tensorboard:** access the Tensorboard functionality
  - **RStudio:** connect to RStudio suite on the compute node
- Tools and features are updated regularly depending on the needs of the users
- Availability of features depending on the *standard environment*



3 CLI in JupyterHub

The terminal in the JupyterHub enables to:

- interact with ZIH system via CLI
- manage software and data with CLI commands



3 CLI Commands under the Hood: Jupyter Magic

The IPython Jupyter kernel supports *Jupyter magic commands* that:

- add additional functionality to Jupyter notebooks
- provide CLI functionality by the %%bash and ! magic commands

Hint

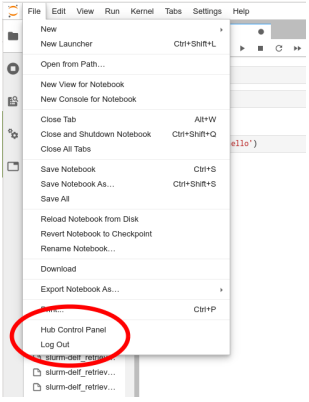
See the Jupyter documentation to learn more about magic commands: ipython.readthedocs.io/.../magics.html

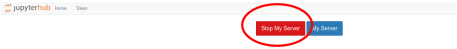
Example

```
[3]: %%bash
     cd /beegfs/ws/1/s4122485-jh_ws
     mkdir test_magic
     ls -l

total 42757
drwxr-xr-x 4 s4122485 p_scads        2 Oct 17 15:21 JPEGIm
-rw-r--r-- 1 s4122485 p_scads      789 Oct 17 13:22 LICENS
```

```
[5]: !cd /beegfs/ws/1/s4122485-jh_ws && ls -l

total 42757
drwxr-xr-x 4 s4122485 p_scads        2 Oct 17 15:21 JPEGIma
-rw-r--r-- 1 s4122485 p_scads      789 Oct 17 13:22 LICENSE
```

3 JupyterHub Stop a Session





Warning

Work finished do not forget to logout and stop your server! Otherwise, the resources will not be available to others and account for your **CPU quota**!

4 Data Management via Jupyter – Agenda

4 Data Management on HPC System

- many systems to store data that differ in capacity, bandwidth, IOPS rate etc
- organization of these storage systems in **files systems**
- many file systems, users, data require mechanisms for data management (duration, size, permission)
- the concept of work-spaces is used on the ZIH system for data management

Hint

for this presentation: a file system refers to a "place" to store data and a **workspace** refers to the "access" to that place

4 Filesystem Storage Overview

- Each filesystems serves as respective capacity, performance, duration, lifetime.
- User creates a workspace with defined expiration date.
- A workspace can be extended multiple times, depending on the filesystem.

scope	filesystem	speed	size	duration
local	/tmp	+++	---	---
Global temporary	/beegfs	++	--	--
Global temporary	/ssd	+	-	--
Global temporary	/scratch	-	+	+
Global permanent	/projects /home	--	++	++
archiving	/warm_archive	---	+++	+++

Note

/projects and /home are permanent filesystems without expiration date

4 Workspace management in JupyterHub

To manage workspaces in Jupyter, use dedicated `ws` commands via:

- the CLI from the JupyterHub
- Jupyter magic commands within a notebook cell to execute bash scripts

Description	Command
Find available workspace filesystems	ws_find – –list
Allocate workspace	ws_allocate
List your workspaces and get information (duration, path,...)	ws_list
Extend workspace	ws_extend
Delete workspaces	ws_release

Note

Find additional commands in the docs: https://doc.zih..../data/lifecycle/workspaces/

4 Workspace - Example

A **workspace** is a directory, with an associated expiration date, created on behalf of a user in a certain filesystem.

Example

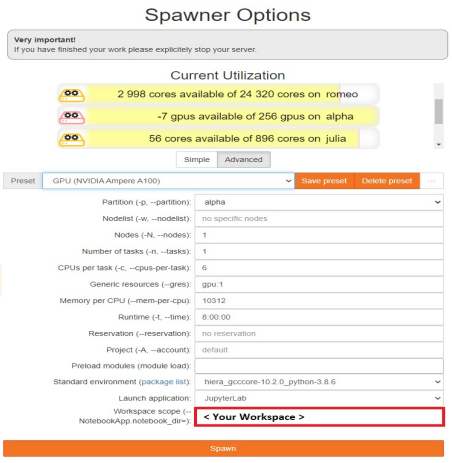
```
# Show available file systems for using workspaces
marie@compute$ ws_find -l

# List all allocated workspaces
marie@compute$ ws_list

# Allocate a workspace "test-workspace" in filesystem "scratch"
marie@compute$ ws_allocate -F scratch test-workspace

# Extend lifetime of a workspace called "test-workspace"
marie@compute$ ws_allocate -F scratch test-workspace 10

# Delete workspace
marie@compute$ ws_release test-workspace
```

4 Accessing Data: Linking Workspaces

- It might be valuable to links workspaces within a certain directory, e.g., your home directory.
- The command "ws_register" DIR to create link of workspace within the Home DIR directory
- via the Workspace scope parameter, set a workspace as the default directory for the notebook data

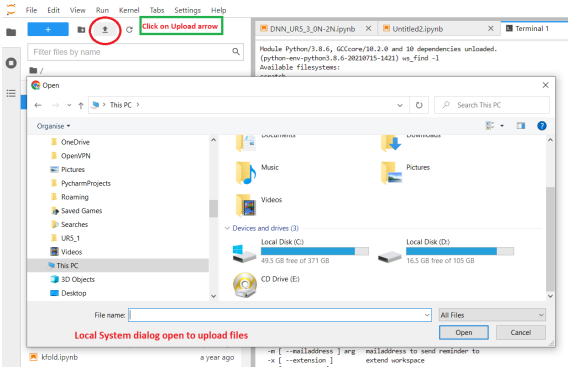
Warning

The **workspace scope** parameter is only for setting a specific working directory as default through the spawn menu. It is not related to the workspace principles on the system.



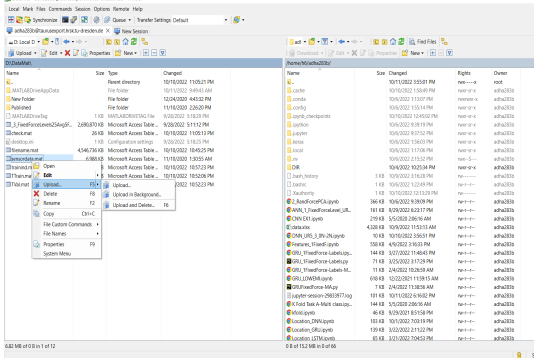
4 Data transfer to the JupyterHub

With JupyterHub GUI, data can be uploaded, downloaded with into workspace or home directory.



4 Data transfer to the ZIH System, Windows

With an WinScp software, data can be uploaded, downloaded with a GUI to workspace or home directory for windows



4 Data Transfer on HPC System

There are two methods for exchanging data on HPC:

- scp, rsync, and sftp
- Datamover, to move data faster inside the ZIH
- for datamover you have to use commands prefixed with dt: dtcp, dtwget, dtmv, dtrm, dtrsync, dttar, dtls

More information in the wiki: `https://doc.zih..../access/data/transfer/`

Example: Access From Linux using scp

Example

```
# Copy a file from your workstation to ZIH systems
marie@local$ scp <file> taurusexport:<target-location>
# Add -r to copy whole directory
marie@local$ scp -r <directory> taurusexport:<target-location>
# Copy a file from ZIH systems to your workstation
marie@local$ scp taurusexport:<file> <target-location>
```

4 Data Transfer on HPC System

Command examples prefixed with dt for transfer files from one filesystem to another filesystem

Example

```
# dtcp for copying data from one filesystem to another filesystem
marie@login$ dtcp -r <fs_origin>/results <fs_destination>/.

# dtmv for moving data from one filesystem to another filesystem
marie@login$ dtmv <fs_origin>/ws/results <fs_destination>/ws/ws-archive/.

# dttar for archive data "results.gz" from one filesystem to another filesystem
marie@login$ dttar -czf <fs_destination>/results.tgz <fs_origin>/ws/results
```

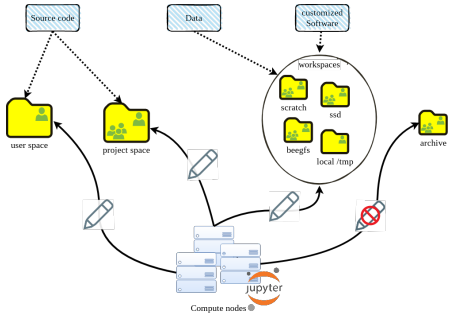
5 Collaborative Research and Teaching with Jupyter on HPC – Agenda

5 Collaborative Principles on HPC

- consider access/permission restrictions on the cluster
- level 1: permissions of files/directories need to be configured (esp. data and customized software environments)
- level 2: read/write connection between parts of the cluster (login nodes, compute nodes, storage)

Distinguish between: data, software environment, source code:

- data: in workspaces with group access
- software environment:
  - module system
  - in workspaces with group access
- source code:
  - user space or directory in project space
  - collaborative management via external version control, e.g. Git



5 Accessing Data and Sharing Software: Permissions Management on ZIH system

On the ZIH system, we use Linux permissions:

- distinguish **r**ead, **w**rite and e**x**ecute permissions
- assignable to **u**ser/owner, **g**roup (HPC project), **o**ther, **a**ll
- works on directory level and file level

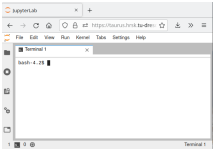
Hint

The `chown` and `chgrp` command change the owner and group of the directory/file

Reading permissions with `ls -l` command:

Example

```
1 # Check permissions of directory
2 marie@compute$ ls -l /scratch/ws/1/Marie-ml-ws/
3 # Output
4 -rwxr-xr-- 1 marie marie_group 10 11. Nov 11:11 textfile.txt
5 drwx------ 1 marie marie_group 4096 11. Nov 11:12 marie_dir
```



5 Accessing Data and Sharing Software: Permissions Management on ZIH system

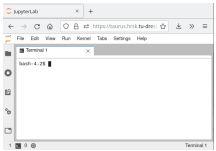
Permission management is done by the 'chmod' command via CLI. Syntax:

chmod [ugoa]*([-+=]([rwxXst]*|[ugo]))+|[-+=][0-7]+ myfile

- ugoa = user/owner, group, other, all
- -+= = remove(-), add(+), set(=) permissions
- rwx = read, write, execute
- for more information: `man chmod`

Example

```
1 # Give read and execute access to the group of the file
2 marie@compute$ chmod g+rx /scratch/ws/1/Marie-ml-ws/marie_dir -R
3 # Check permissions of directory
4 marie@compute$ ls -l /scratch/ws/1/Marie-ml-ws/
5 # Output
6 drwxr-x--- 1 marie marie_group 4096 11. Nov 11:12 marie_dir
```



5 Customized Software Environments with Python

- Python virtual environments create isolated working environments
- You can install packages, without interfere with other environments
- Multiple environments can be installed on one system simultaneously
- `conda` or `virtualenv` are state-of-the-art systems to create virtual environments

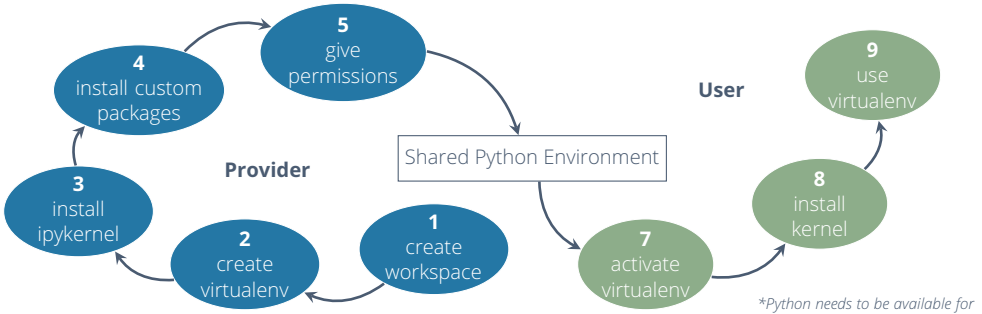
> **⚠ Warning**
>
> `conda` virtual environments may infer with the module system on HPC. Therefore we recommend using `virtualenv` and `pip` instead of `conda`

More information in the wiki: `https://doc.zih..../software/python_virtual_environments`

5 Shared Python Environment and Jupyter Kernels

We assume two roles for collaboration:

- **Provider**:
  - provide a consistent Python environment
  - share environment with collaborators
  - e.g. teacher, researcher
- **User**:
  - uses a Python environment to reproduce results or studying
  - e.g. pupil, research collaborator



*Python needs to be available for following the workflow

5 Shared Python Environment: Example - Provider

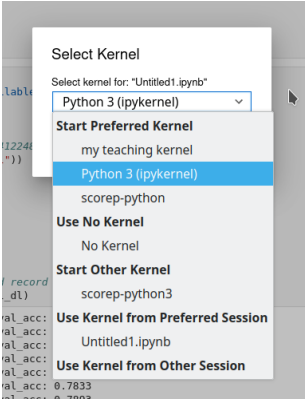
Example

```
# Create a virtual environment in the workspace
marie@compute$ virtualenv --system-site-packages jh_venv

# Activate the environment
marie@compute$ source jh_venv/bin/activate

# Install the ipykernel package to use the environment in JupyterHub
(jh_venv) marie@compute$ pip install ipykernel

# Install additional packages
(jh_venv) marie@compute$ pip install [...]

# Set permissions of the workspace to share environment with collaborators
marie@compute$ chmod g+rx /scratch/ws/1/jh_ws/ -R
```

5 Shared Python Environment: Example - User

Example

```
1 # Activate the environment
2 marie@compute$ source /scratch/ws/1/jh_ws/jh_venv/bin/activate
3
4 # Install the kernel for Jupyter
5 (jh_venv) marie@compute$ python -m ipykernel install --user --
      name my-teaching-kernel --display-name="my teaching kernel"
6
7 # Deactivate environment
8 (jh_venv) marie@compute$ deactivate
```



After installing the kernel, it is available in the drop down menu

5 Persistence of Environments

- Goal: long-term storage and distribution of a Python software environment
- Conda and pip provide mechanisms to persist environments via a file
- Permanently store files in Git repository, e.g. repository for lecture series
- Restore environments by creating new environments based on the file

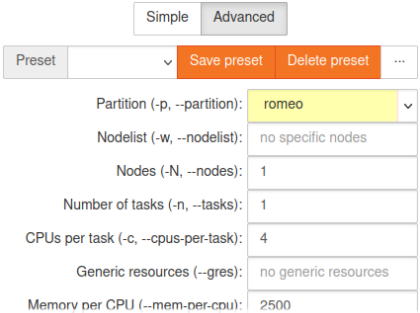
Example

pip:

```
1 # Store packages of "env_pre" in "req.txt" and restore in fresh environment "env_post"
2 (env_pre) marie@compute$ pip freeze > req.txt
3 (env_post) marie@compute$ pip install -r req.txt
```

5 Share a Spawn Link

- JupyterHub allows to share spawn options, e.g. Slurm parameters or work directory, via a link
- Every parameter of advanced form can be set as parameter
- If a parameter is not set, the default value will be loaded
- Setting the parameters in the form also sets the parameters in your browsers URL line:



Example

https://taurus.hrsk.tu-dresden.de/jupyter/hub/spawn#/~(partition~'romeo~cpuspertask~'4~mempercpu~'2500)

base URL and page "spawn" encoded spawn parameters

5 Git Pull Feature

- Clone a public git repository into users work directory via `nbgitpuller` extension
- Distribute notebooks, data and other material to students or colleagues
- Use **nbgitpuller link generator**

Example

https://taurus.hrsk.tu-dresden.de/jupyter/hub/user-redirect/...

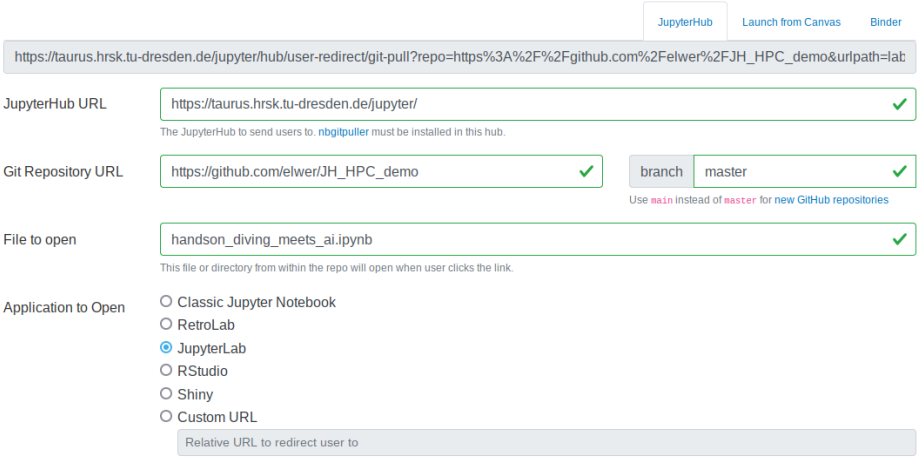
base URL and page "user-redirect"

...git-pull?repo=https://github.com/jdwittenauer/ipython-notebooks&urlpath=/tree/ipython-notebooks/notebooks/language/Intro.ipynb

call to nbgitpuller extension with "repo" and "urlpath" parameter

- clone repository *ipython-notebooks* into work directory
- open *Intro.ipynb* notebook

5 Git Pull Feature: NBGitpuller Link Generator



5 Git Pull Feature: Private Repository

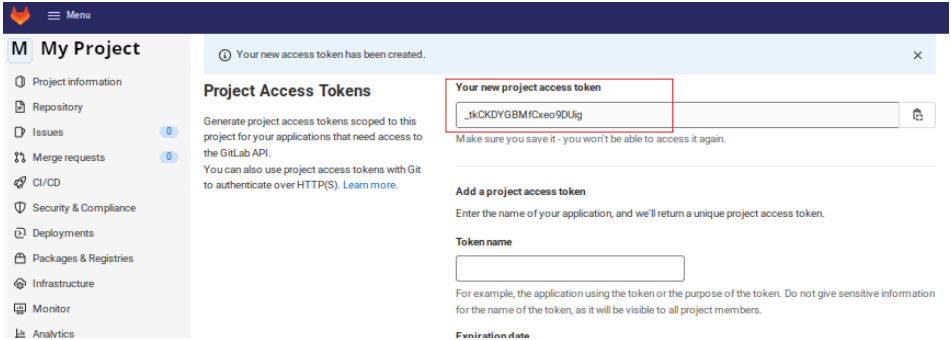
- Git pull feature also possible for private repositories, e.g. repository in GitLab Chemnitz
- Access tokens provide access to your repository for the "outer world"
- Permissions and expiration of tokens definable by yourself

Workflow:

1. Create access token in repository settings
2. Adapt link to repository for Git HTTPS cloning:
   `https://<username>:<access-token>@<repository-url>`
3. Use **nbgitpuller link generator**

> **Note**
>
> GitLab Chemnitz creates a "Bot user" when creating an access token. Use this user for the link generation. The username is created as follows: `project_project_id_bot`. You can find the `project_id` in your repository settings. More information: gitlab.hrz.tu-chemnitz.de/.../project_access_tokens

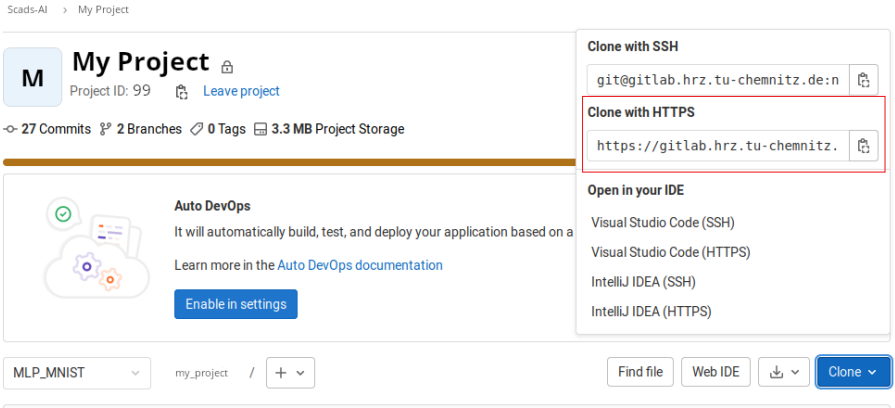
5 Git Pull Feature: Private Repository - Example(1)



Warning

The token is only visible at creation time. On the page you can find multiple options for configuring a specific token (access role, scopes, expiration date) as well as revoking access for existing tokens.

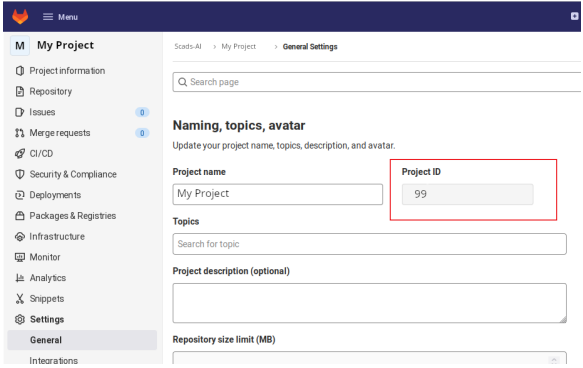
5 Git Pull Feature: Private Repository - Example(2)



5 Git Pull Feature: Private Repository - Example(3)

Workflow Example:

- get access token
- get git repository link
- get project id Bot username: `project_99_bot`



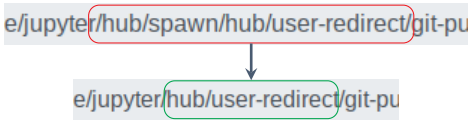
- now: create link for Git HTTPS cloning
  `https://project_99_bot:_tkagshd63Sd@gitlab.hrz.tu-chemnitz.de/scads-ai/my_project.git`
- use the link in the nbgitpuller link generator

5 Combine Parameter Sharing and Git Pull

- You can also combine the *parameter sharing* and the *git pull* feature via a single link
- Concatenation of links can be done manually or:
  1. create the *parameter sharing* link
  2. insert the link as *JupyterHub URL* in the nbgitpuller link generator and add *git pull* functionality

Warning

When **combining** the link via the nbgitpuller link generator, the link might be defective and needs to be fixed.



Example

https://taurus.hrsk.tu-dresden.de/jupyter/hub/user-redirect/...

base URL and page "user-redirect"

...git-pull?repo=https://github.com/jdwittenauer/ipython-notebooks&urlpath=/tree/ipython-notebooks/notebooks/language/Intro.ipynb...

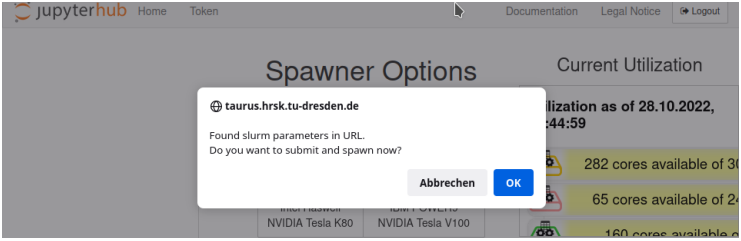
call to nbgitpuller extension with "repo" and "urlpath" parameter

...#/~(partition~'romeo~cpuspertask~'4~mempercpu~'2500)

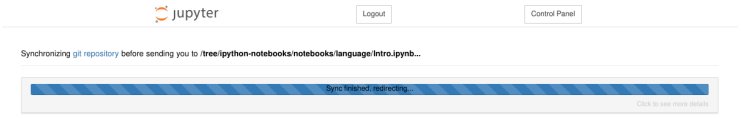
encoded spawn parameters

5 JupyterHub Link Feedback

Feedback of the parameter sharing feature:



Feedback of the git pull feature:



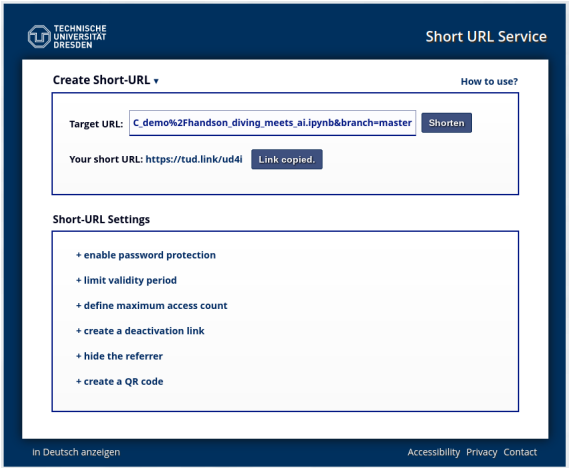
5 Short URLs

Hint

Short URL services shorten your links and may provide further functionality, e.g.

- expiration date
- maximum number of accesses

We recommend to use the TUD Short link service for that purposes: https://tud.link/create/

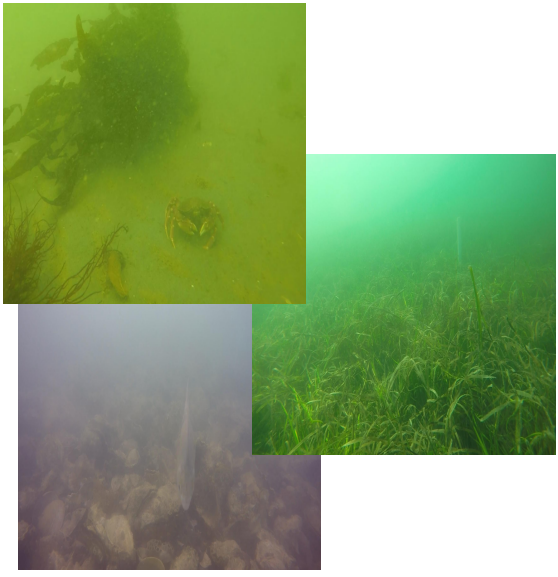


6 Hands On session: Prepare a Jupyter "environment" for collaborative working – Agenda

6 HandsOn: Diving meets AI

Dataset: NOAA Puget Sound Nearshore Fish 2017-2018

- 77.739 images from underwater video
- 30.384 images with fish and crustaceans, annotated as *animal*
- remaining images annotated as *empty*
- more information about the dataset: lila.science/.../noaa-puget-sound-nearshore-fish



⚠ Warning

Please use your personal ZIH login credentials we sent you for the following tasks! Use the reservation `p_scads_jupyterhub_773` for the `--reservation` parameter to reduce scheduling time.

6 Setting

Hint

We distinguish between *source code*, *data* and *software*. In this example, we already prepared the source code in a git repository.

- A Jupyter notebook with the source code is available in a Git repository:
  https://github.com/elwer/JH_HPC_demo
- The NOAA fish dataset is available in the workspace:
  `/beegfs/ws/1/s4122485-jh_ws/NOAA`
- **Aim:** Share your experiences with the dataset with your colleagues

- The example consists of two parts:
  1. Prepare and share your setup (data and software)
  2. Test the setup

6 Part 1a: Prepare the workspace and the data

Provider

Concrete tasks (data part):

1. Open the browser and spawn a Jupyter server:
   - Advanced option (see slide 30)
   - Partition: alpha
   - Runtime: 2hours
   - Reservation: `p_scads_jupyterhub_773`
2. Open a terminal after spawning (see slide 37)
3. Create a workspace `teaching_ws` on `beegfs` file system (see slide 44)
   - create a directory `data` for the data in `teaching_ws`
   - create a directory `sw` for the software in `teaching_ws`
   - note down/remember the workspace directory (it is required later)
4. Copy the data from `/beegfs/ws/1/s4122485-jh_ws/NOAA` to `[...]/teaching_ws/data` either via the datamover nodes (see slide 49) or simple `cp` command

6 Part 1b: Prepare the software

Concrete tasks:

Provider

1. Load the Python module: `module load modenv/scs5 Python/3.9.5`
2. Create a Python virtual environment in `[...]/teaching_ws/sw` and activate it (see slide 56)
3. Install the `ipykernel` package (see slide 56)
4. Install additional packages (see slide 56):
   - `pip3 install torch torchvision torchaudio --extra-index-url https://download.pytorch.org/whl/cu113`
   - `pip install tqdm matplotlib`
5. Give the group **r**ead and e**x**ecute permissions to `[...]/teaching_ws` and all subdirectories (see slide 53)
6. Log out from your session and stop it(see slide 39)

⚠ Warning

Important: This example is only for demonstration purposes. When interacting with the HPC, always prefer pre-installed Modules instead of installing the software on your own.

6 Part 1c: Prepare for sharing

Concrete tasks:

Provider

- Via the spawner advanced interface: Create a link for sharing the resources (see slide 59):
  - partition: alpha
  - nodes: 1
  - CPUs: 6
  - GPUs: 1
  - memory per CPU: 2500M
  - runtime: 2hours
  - reservation: `p_scads_jupyterhub_773`
  - preload modules: `Python/3.9.5`
  - standard environment: `production`
  - workspace_scope: your workspace directory
- Insert the link in the nbgitpuller website as *JupyterHub URL* (see slide 60/61)
- Add the *Git Repository URL* and the *File to open* to combine both links (see slide 61/62)
- Shorten the link (see slide 64) and communicate the link to the group via the chat

6 Part 2: Test the setup

User

Concrete task:

1. Use some link from the chat to start a JupyterHub (not your own)
2. Activate and install the virtual environment from the workspace of your group members either via CLI or Jupyter magic (see slide 38)
3. Select the installed Kernel
4. Execute the Jupyter notebook and explore the data set

> ⚠ **Warning**
>
> This part will only work, if the person that created your link, carefully followed all the steps in part 1.

6 Basic CLI Commands

Command	Description
ls	list files and/or directories
cd <path_destination>	change directory
mkdir <directory_name>	create a new directory
pwd	Show full path of the current directory
cp <path_source> <path_destination>	copy a file
cp -r <path_source> <path_destination>	copy a directory
mv <path_source> <path_destination>	move a file or directory
rm <path_source>	delete a file or directory
wget	get datasets from the web (using HTTP, HTTPS, FTP and FTPS)

6 Troubleshooting

- Problems when spawning:
  - delete cookies
  - delete access tokens
  - check whether your spawner is still running via `squeue --me` command in CLI (ssh connection)

- General:
- check the `jupyter-session-XXXXXXX.log` file in your HOME directory
- it gives a lot of insights of your notebook servers behaviour (e.g. if module (pre-)load was successful)
- Pika Job Monitoring tool shows you utilization statistics of your notebook server

- ssh connection to the node with the notebook server software (get name of the node via `squeue --me`), then `ssh taurusi_nodeID`, lets you investigate your computing node(s) further

Hint

Having a look in the compendium is always useful: doc.zih.tu-dresden.de

7 Conclusion, Q&A and Supplementary – Agenda

7 Conclusions

- The Jupyter universe is big (extensions, plugins, use cases, people)
- JupyterHub offers an easy access to HPC resources
- HPC is a complex task - JupyterHub abstracts some of the complexity
- Versatile features for collaborative working
- We proposed one approach but typically there is not only one solution

7 Conclusions

- In case of questions start at the official docs for the ZIH system: `https://doc.zih.tu-dresden.de/`
- Software ecosystems are changing dynamically, esp. in ML: with HPC systems it is hard to react immediately
- Contact us with your own ideas, experiences and wishes!

Note

logins (scads0xx) will work for further 10 days

- if your are interested afterwards please apply for your own HPC project
- for starting purposes we recommend a "Schnupperprojekt" without a detailed project description

ScaDS.AI
DRESDEN LEIPZIG

Thank you

Contact us on:
elias.werner@tu-dresden.de,
adnan.haidar@tu-dresden.de,
mariela_rossana.sanchez_figueroa@tu-dresden.de