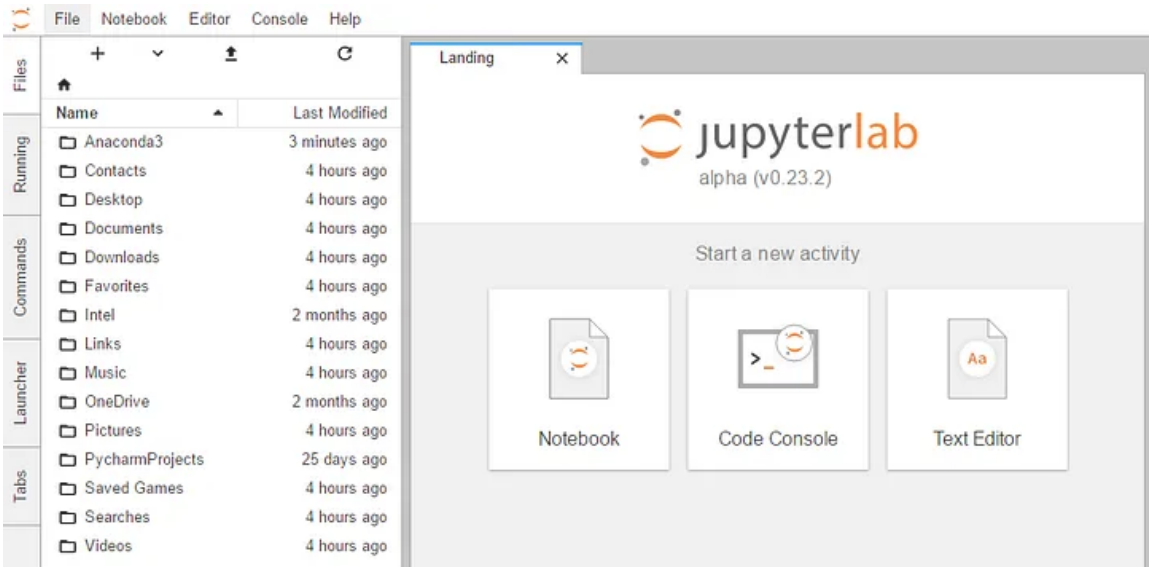
JupyterLab: The evolution of Jupyter Notebook

JupyterLab is a next-generation web-based user interface for Project Jupyter. It's a full featured IDE that has everything we ever wanted to be in Jupyter notebooks which enables you to work with documents and activities such as Jupyter notebooks, text editors, terminals, and custom components in a flexible, integrated, and extensible manner.



The main features of JupyterLab:

• *Drag and Drop*:

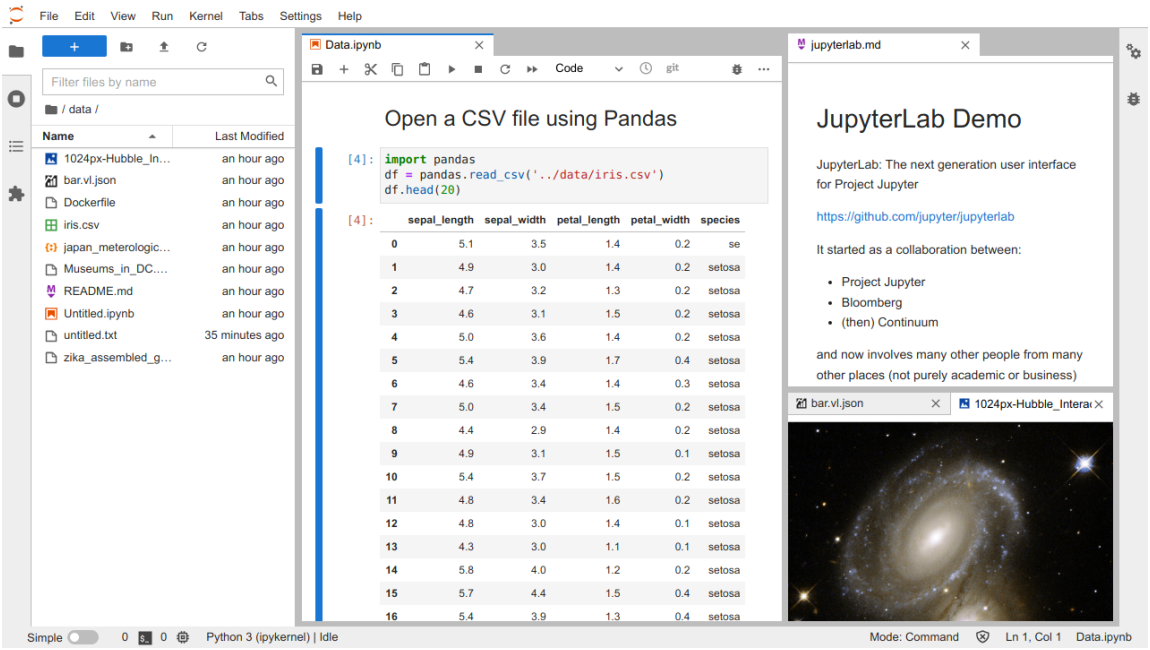
The ability to re order cells without cut and paste is powerful. It also feels more natural to do drag and drop given that code is organized in cells in notebooks

• *Multiple notebooks and kernels*:

Running multiple notebooks at the same time already exist with the jupyter notebooks. However, these notebooks had to be oppened in multiple browser windows. In JupyterLab, you can have multiple notebooks open at the same time and in the same browser window. Also, you can arrange your notebooks as you like which gives more flexibility. Another nice feature is it's possible to have each notebook running on it's own kernel, this is powerful when running multiple notebooks at the same time doing different things.

• *Real time markdown editor*

With this new feature, I can edit and see in real time the update of my markdown files in JupyterLab. This speeds-up the edit process and streamlines work.

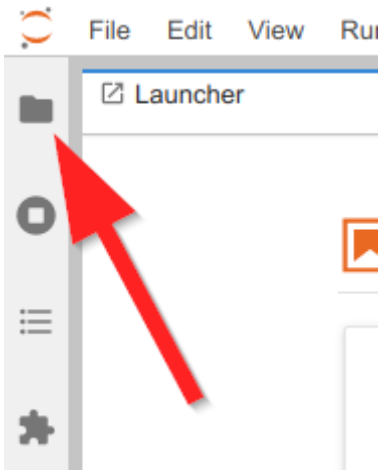


- *Multiple windows*

With multiple windows open at the same time, I can have multiple notebooks that I am working on and then use a terminal inside JupyterLab

- *Full file explorer*

The file browser and File menu enable you to work with files and directories on your system. This includes opening, creating, deleting, renaming, downloading, copying, and sharing files and directories.



- *Manage kernels and terminals*
- *Command search*
- *Fast CSV files viewer*
- ...

Running JupyterLab Remotely with Slurm

Just like Jupyter Notebook, JupyterLab can be launched locally and access local file systems, or they can be launched on a remote machine.

By default, **Labs is not secure, and potentially expose a users local files to unwanted users**. We recommend reading the section *Running Jupyter Notebook with Slurm* where you can find several advices to use JupyterLab.

Remember:

Do Not Run Jupyter on the Login Nodes

The reason here

Internet is Not Available on Compute Nodes

What is the solution?

You can't directly access the compute nodes

Why?

This deployment is not the multi-user server you are looking for

What is the meaning of this?

You can use JupiterLab in two different way:

- Create a Conda o pip environment
- Using as module

Create a conda environment to install JupyterLab

Create a conda environment

Here we have several recommendations that we recommend that you read. To simplify:

```
conda activate jupyter-labenv
conda install jupyterlab matplotlib ipykernel ipywidgets  ipympl --channel conda-forge -y
```

Using JupiterLab as module

You only need type:

```
module spider jupyter
```

```
----------------------------------------------------------------------------
  JupyterLab: JupyterLab/3.1.6
 ----------------------------------------------------------------------------
    Description:
      JupyterLab is the next-generation user interface for Project Jupyter offering all the familiar
building blocks of the classic Jupyter Notebook (notebook, terminal, text editor, file browser, rich
outputs, etc.) in a flexible and powerful user interface.
      JupyterLab will eventually replace the classic Jupyter Notebook.


    You will need to load all module(s) on any one of the lines below before the "JupyterLab/3.1.6"
module is available to load.

     GCCcore/11.2.0
...
```

How to start JupyterLab

If you prefer use JupyterLab as module you only need change in the following lines the line

```
conda activate jupyter-labenv
```

with

```
module load GCCcore/11.2.0 JupyterLab/3.1.6
```

Running JupyterLab on a Compute Node via *salloc*

```
```bash
salloc --nodes=1 --partition batch --ntasks=1 --mem=20G --time=12:00:00
# or salloc -N 1 -p batch
# or srun -p batch --pty bash

conda activate jupyter-labenv
jupyter lab --no-browser --port=8890 --ip=127.0.0.1
```
```

By default, JupyterLab are not secure, and potentially expose a users local files to unwanted users.

Running Labs as usual uses secure HTTP connections. In this guide, we present a guide to running it more securely. More info here

Tip

Note, the default is for JupyterLab to automatically open a browser – but we can't do that on a remote server, so we bypass that function with the *--no-browser* flag.)

Create a SSH Tunnel

Here we talk about the reasons why it is necessary to create the ssh tunnel and how to make this.

Start a second terminal session, connect to any login node and

```
ssh -N -L localhost:8889:localhost:8889 yourUser@nodeXXXX-Y.hpc.iter.es
```

On your local machine (e.g., laptop) setup the tunnel as follows:

```
ssh -N -L localhost:8890:localhost:8890 yourUser@loginX.hpc.iter.es
```

You can use the DNS of the node or the ip

Change *nodeXXXX-X.hpc.iter.es* by node ip.

Enter the address in your favorite browser

http://127.0.0.1:8890/lab?token=4c7eddd5770e27195808f4615d8e6f3de48b45c57169b69e
We recommend secure your Jupiter Lab

Running JupyterLab on a Compute Node via *sbatch*

```
#!/bin/bash
#SBATCH --nodes=1
#SBATCH --ntasks=1
#SBATCH --cpus-per-task=1
#SBATCH --mem=20G
#SBATCH --partition=batch
```

```
#SBATCH --constrains=sandy
#SBATCH --time=01:00:00
#SBATCH --job-name=jupyter-lab
#SBATCH --output=jupyter-lab-%j.out
#SBATCH --error=jupyter-lab-%j.err
############################################################
# Enable modules profile

# Enable conda on bash console
eval "$(conda shell.bash hook)"

# load modules or conda environments here
module load  Miniconda3/4.9.2
conda activate jupyter-labenv

# get tunneling info
XDG_RUNTIME_DIR=""
node=$(hostname -s)
user=$(whoami)
cluster="teide-hpc"
port=8890

# print tunneling instructions jupyter-log
echo -e "
Command to create ssh tunnel from login node to compute node:
ssh -N -L localhost:${port}:localhost:${port} ${user}@nodeXXXX-Y.hpc.iter.es

Command to create ssh tunnel from your local machine to any login node:
ssh -N -L localhost:${port}:localhost:${port} ${user}@login1(2).hpc.iter.es

Use a Browser on your local machine to go to:
localhost:${port}  (prefix w/ https:// if using password)
"

# Run Jupyter
jupyter lab --no-browser --port=${port} --ip=127.0.0.1
```

Run script with

```
sbatch jupyter-lab.sbatch
```

In order to access JupyterLab navigate to http://localhost:8890/

A few important notes

Set a strong password in JupyterLab

By default, you start a Jupyter server with token authentication enabled and this token is logged to the terminal, so that you can copy/paste the URL into your browser. **This token can be used only once**, and is used to set a cookie for your browser once it connects. After your browser has made its first request with this one-time-token, the token is discarded and a cookie is set in your browser.

Alternatives to token authentication you can set a password for your server. Jupyter server password will prompt you for a password, and store the hashed password in your *jupyter_server_config.json* which is usually located in ~/**.jupyter** typing:

```
jupyter server password
```

```
Enter password:
Verify password:
[JupyterPasswordApp] Wrote hashed password to /home/vdominguez/.jupyter/
jupyter_server_config.json
```

You can access now with password instead of token autentication on http://localhost:8890/lab

More security advice here

https://jupyter-server.readthedocs.io

How to change the JupyterLab start-up directory

If your notebook files are not in the current directory, you can pass your working directory path as argument when starting JupyterLab.

```
jupyter lab --notebook-dir=/home/yourUser/data/notebooks/ --preferred-dir /home/yourUser/data/
myapp
```

Keep in mind Lab sessions always reside in a workspace. The default workspace is the main */lab* URL:

```
bash http(s)://<server:port>/<lab-location>/lab
```

Make sure you shutdown your JupyterLab when you are done

To do this, if you are using *salloc mode* you can log back onto the *screen* session you started earlier where the jupyter notebook is running and use *ctrl-C* should shutdown the jupyter notebook and *exit* to close the session with the node.

If you are using *batch mode* you can use the command *scancel* .

JupyterLab Extensions Manager

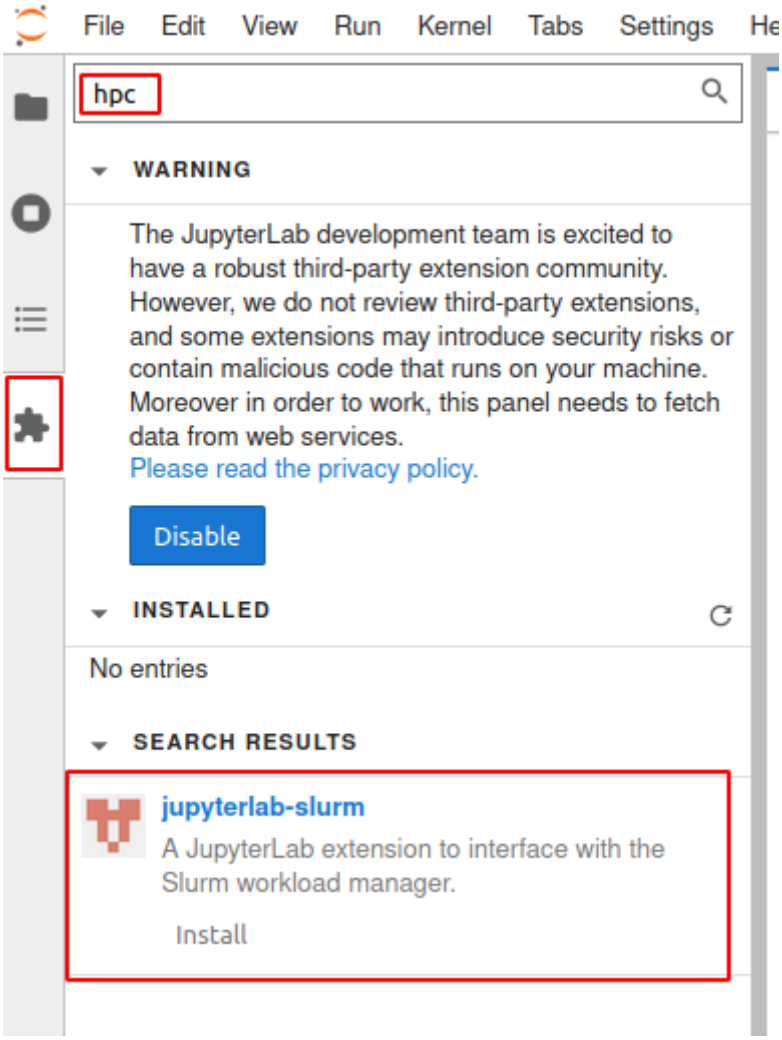
JupyterLab extension is simply a plug-and-play add-on that makes more of the things you need possible.

Technically JupyterLab extension is a JavaScript package that can add all sorts of interactive features to the JupyterLab interface

There are a ton of JupyterLab extensions that you may want to use. Among the best known we could highlight:

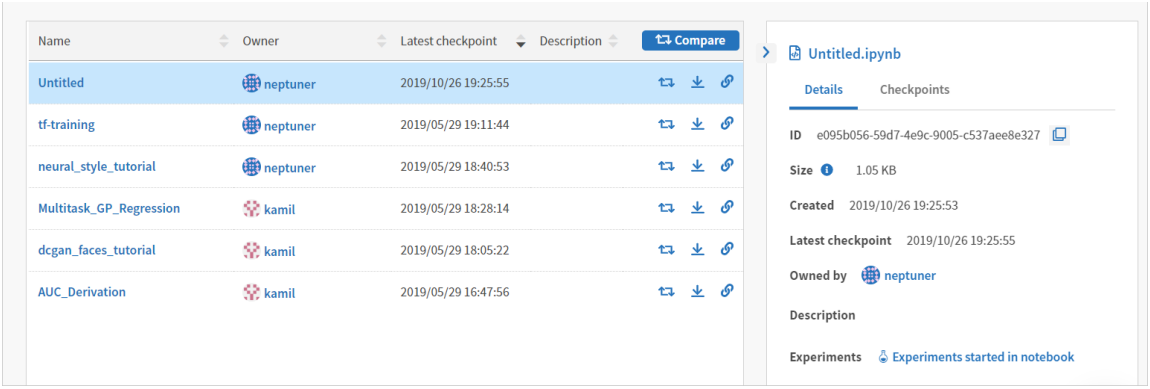
Jupyterlab-slurm

A JupyterLab extension to interface with the Slurm workload manager.



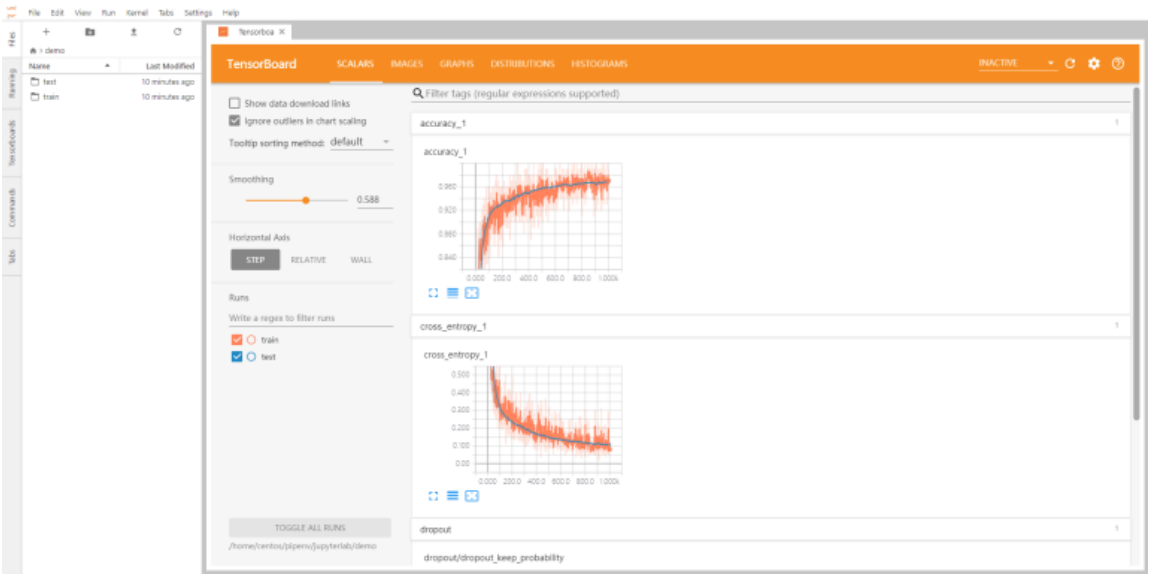
Neptune-notebooks

Neptune is a tool for experiment tracking, model registry, data versioning, and live model monitoring.



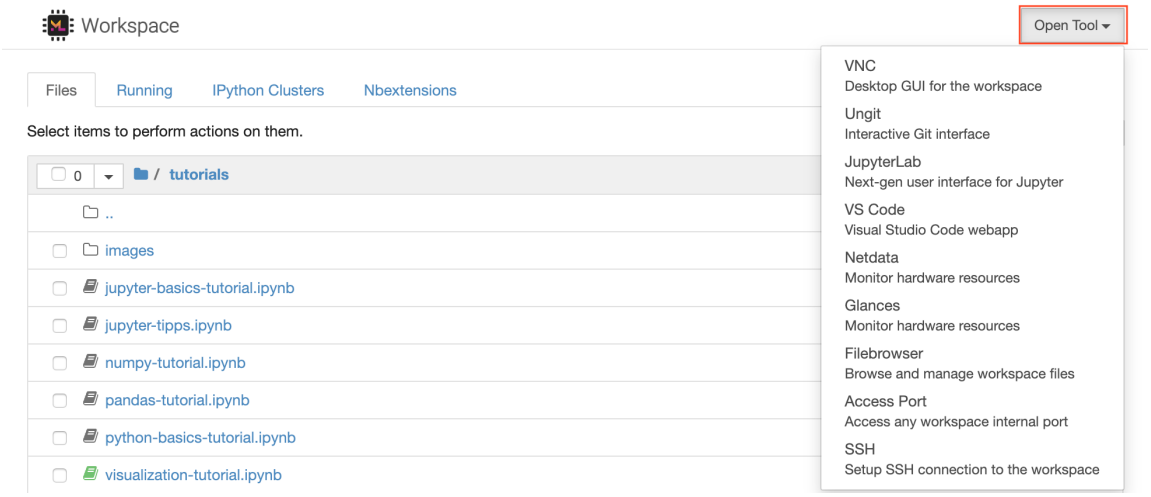
JupyterLab TensorBoard

JupyterLab TensorBoard is a frontend extension for tensorboard on jupyterlab. It helps to collaborate between jupyter notebook and tensorboard (a visualization tool for tensorflow) by providing a graphical user interface for tensorboard start, manage and stop in jupyter interface.



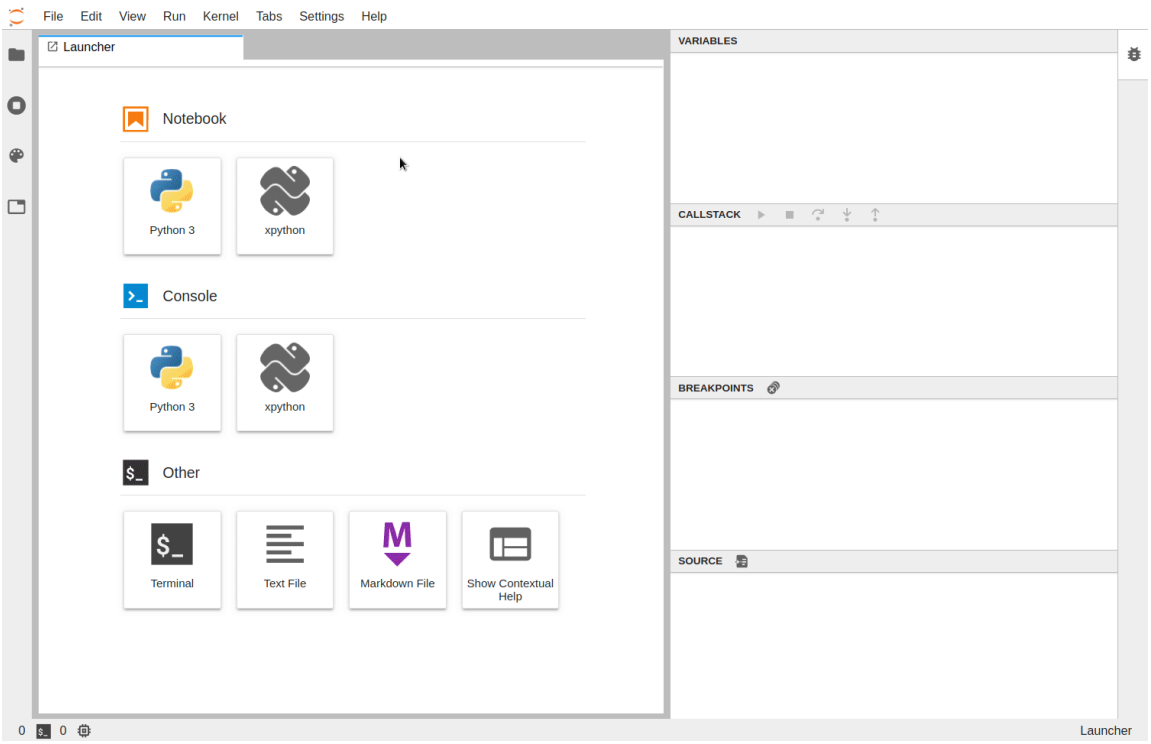
Jupyter ML-workspace

The ML workspace is an all-in-one web-based integrated development environment dedicated for machine learning and data science.



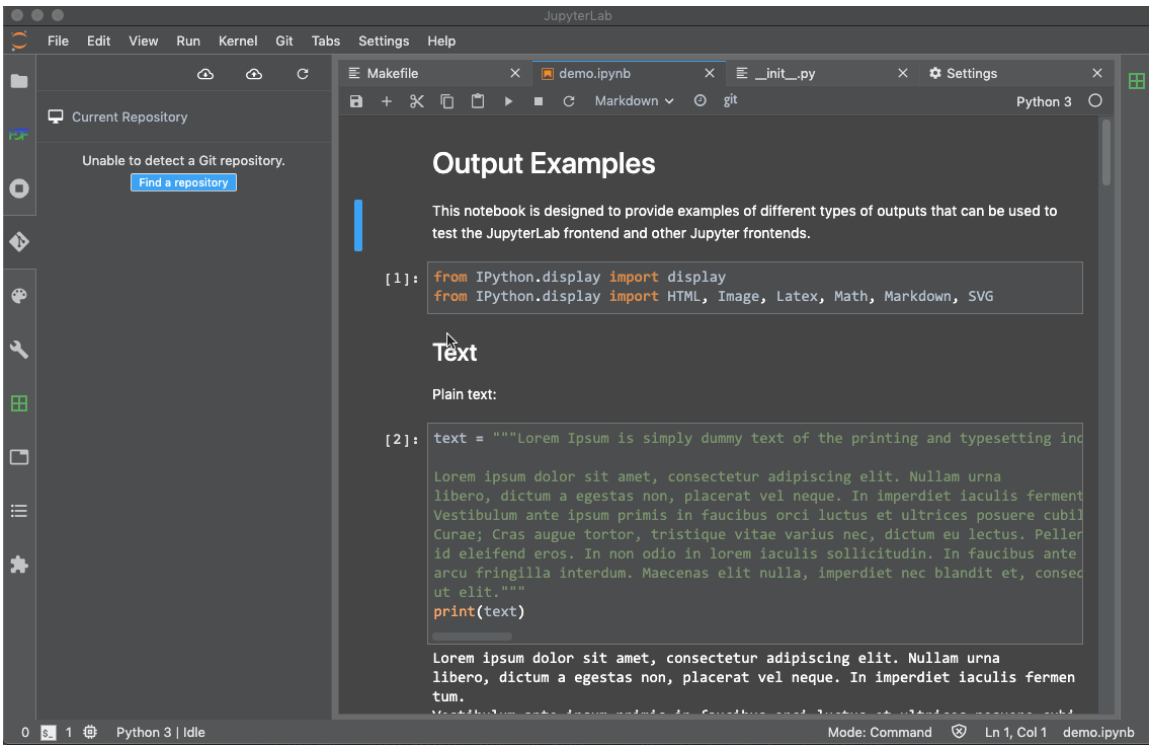
JupyterLab Debugger

Debugger is a JupyterLab extension that works as a visual debugger for Jupyter notebooks, consoles, and source files. It can help you identify and fix bugs.



JupyterLab Git

This one is a JupyterLab extension for Git free and open-source distributed version control system. It allows you for version controlling. You simply use it by opening the Git extension from the Git tab on the left panel.



Other notable extensions:

- JupyterLab LaTeX
- JupyterLab variableInspector
- JupyterLab plotly
- JupyterLab bokeh
- Jupyter Dash
- JupyterLab Table of Contents
- JupyterLab SQL