



Leibniz-Rechenzentrum
der Bayerischen Akademie der Wissenschaften



Jupyter and VS Code

Ferdinand.Jamitzky@LRZ.de



the python interactive command line interface was not very comfortable, so ipython was born. It evolved later on to a Web-Interface (jupyter). You can enter even shell commands.

```
$ ipython
```

```
Python 3.6.2 |Continuum Analytics, Inc.| (default, Jul 20 2017, 13:51:32)
```

```
Type 'copyright', 'credits' or 'license' for more information
```

```
IPython 6.1.0 -- An enhanced Interactive Python. Type '?' for help.
```

```
In [1]: pwd
```

```
Out[1]: '/home/hpc/pr28fa/a2815ah'
```

```
In [2]: import os; os.getcwd()
```

```
Out[2]: '/home/hpc/pr28fa/a2815ah'
```



ipython

ipython is a hybrid between the python cli, a bash shell and macros. It recognizes shell commands (ls, pwd, cp, ..) and macros (magic commands) can be defined by %name or %%name.

```
In [2]: %timeit sum(range(1000))
```

```
20.8 µs ± 412 ns per loop (mean ± std. dev. of 7 runs, 10000 loops each)
```

```
In [13]: %%timeit
```

```
...: x=sum(range(100))
```

```
...: y=x+1
```

```
...:
```

```
1.52 µs ± 5.34 ns per loop (mean ± std. dev. of 7 runs, 1000000 loops each)
```

help information can be retrieved by `?command` and more detailed information by `??command`

```
In [17]: ?pprint
```

```
Docstring: Toggle pretty printing on/off.
```

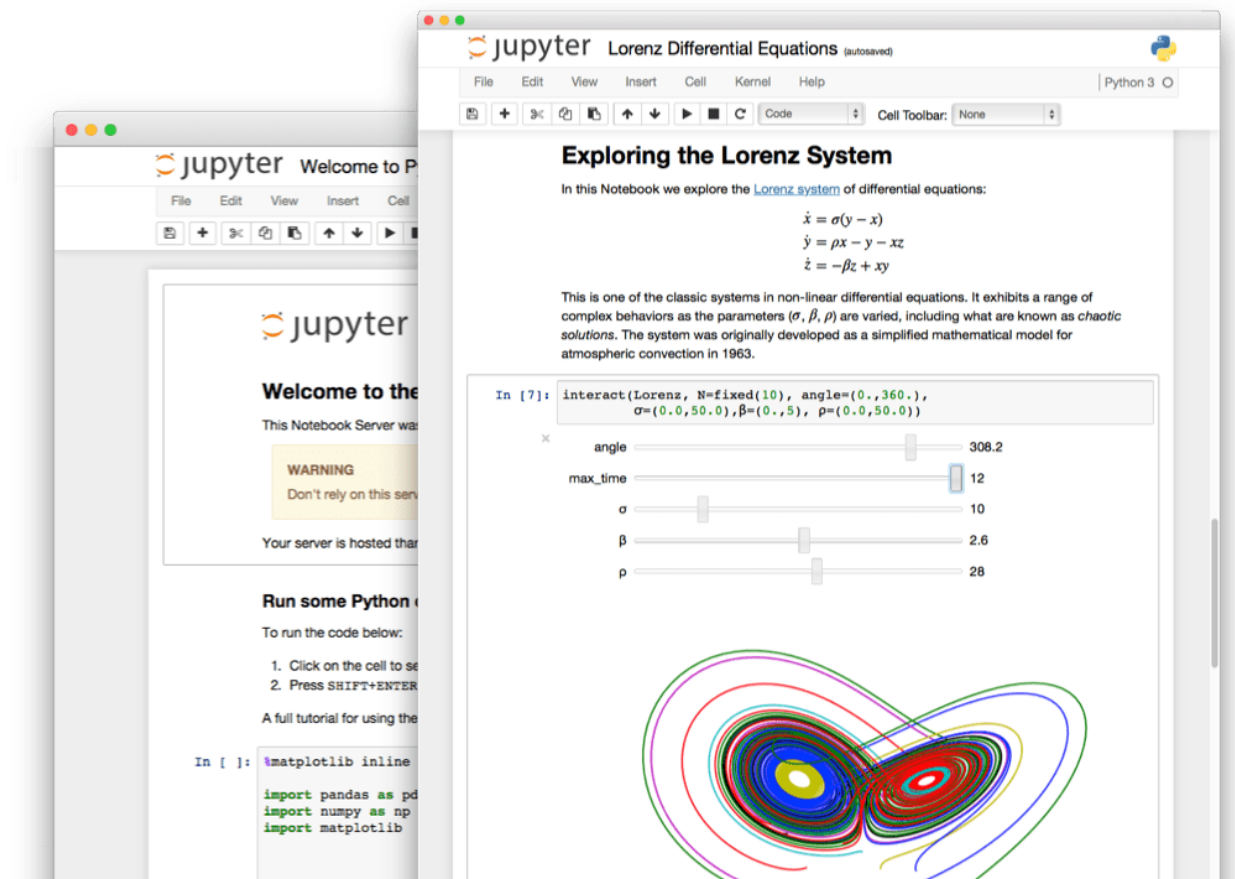
```
File:      ~/.conda/envs/py36/lib/python3.6/site-  
packages/IPython/core/magics/basic.py
```

```
In [16]: ??pprint
```

```
Source:
```

```
@line_magic  
def pprint(self, parameter_s=''):  
    """Toggle pretty printing on/off."""  
    ptformatter = self.shell.display_formatter.formatters['text/plain']  
    ptformatter.pprint = bool(1 - ptformatter.pprint)  
    print('Pretty printing has been turned',....
```

finally ipython evolved into a web-service where you can run any code through a browser interface and even plot.



The image shows a Jupyter Notebook interface with the title "Exploring the Lorenz System". The notebook content includes the following text and code:

Exploring the Lorenz System

In this Notebook we explore the [Lorenz system](#) of differential equations:

$$\begin{aligned}\dot{x} &= \sigma(y - x) \\ \dot{y} &= \rho x - y - xz \\ \dot{z} &= -\beta z + xy\end{aligned}$$

This is one of the classic systems in non-linear differential equations. It exhibits a range of complex behaviors as the parameters (σ, β, ρ) are varied, including what are known as *chaotic solutions*. The system was originally developed as a simplified mathematical model for atmospheric convection in 1963.

```
In [7]: interact(Lorenz, N=fixed(10), angle=(0., 360.),
                sigma=(0.0, 50.0), beta=(0., 5), rho=(0.0, 50.0))
```

The plot shows a 3D visualization of the Lorenz attractor, a chaotic system characterized by its butterfly-like shape. The plot is rendered with multiple colored trajectories (red, blue, green, yellow, purple) showing the complex, non-linear behavior of the system. The plot is titled "x" and has a vertical axis labeled "y".

Below the plot, there are five interactive sliders for the parameters:

- angle: 308.2
- max_time: 12
- σ : 10
- β : 2.6
- ρ : 28



Installing JupyterLab

conda

If you use conda, you can install it with:

```
$ conda install -c conda-forge jupyterlab
```

pip

If you use pip, you can install it with:

```
$ pip install jupyterlab
```



Starting JupyterLab

Start JupyterLab using:

```
$ jupyter lab --no-browser
```

If you leave out the "no-browser" flag JupyterLab will open automatically in your browser or you can enter an URL which is displayed:

Jupyter Notebook 6.3.0 is running at:

```
http://127.0.0.1:8888/?token=241cffc9c9f2732
```

File Edit View Run Kernel Tabs Settings Help

Files

Running

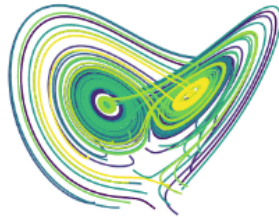
| Name | Last Modified |
|---------------------|--------------------|
| Data.ipynb | an hour ago |
| Fasta.ipynb | a day ago |
| Julia.ipynb | a day ago |
| Lorenz.ipynb | seconds ago |
| R.ipynb | a day ago |
| iris.csv | a day ago |
| lightning.json | 9 days ago |
| lorenz.py | 3 minutes ago |

Commands

Cell Tools

Output View

sigma 10.00
 beta 2.67
 rho 28.00



Terminal 1 Console 1 Data.ipynb README.md

Code Python 3

In this Notebook we explore the Lorenz system of differential equations:

$$\begin{aligned}\dot{x} &= \sigma(y - x) \\ \dot{y} &= \rho x - y - xz \\ \dot{z} &= -\beta z + xy\end{aligned}$$

Let's call the function once to view the solutions. For this set of parameters, we see the trajectories swirling around two points, called attractors.

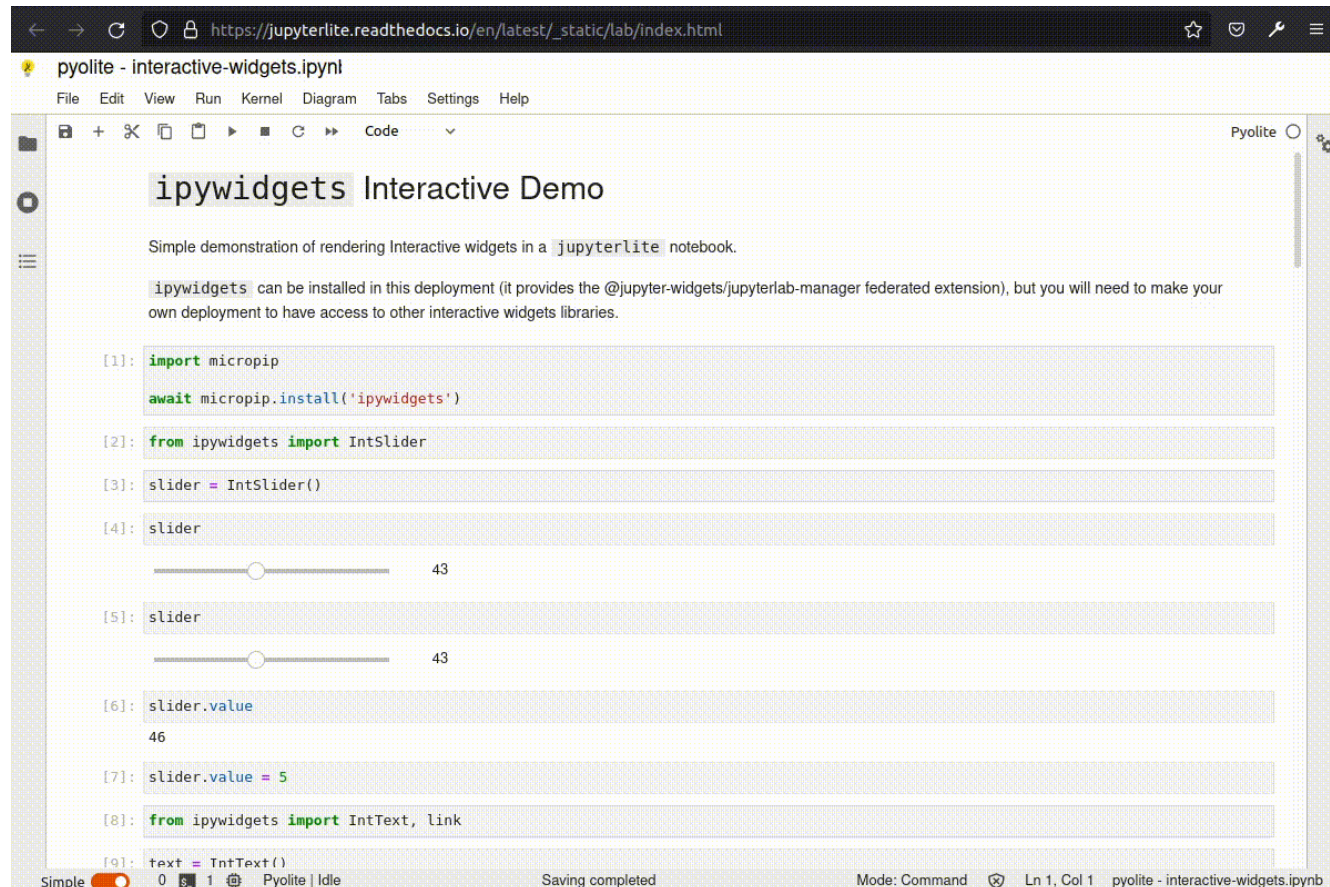
In [4]: `from lorenz import solve_lorenz
t, x_t = solve_lorenz(N=10)`

lorenz.py

```

9 def solve_lorenz(N=10, max_time=4.0, sigma=10.0, beta=8./3, rho=28.0):
10     """Plot a solution to the Lorenz differential equations."""
11     fig = plt.figure()
12     ax = fig.add_axes([0, 0, 1, 1], projection='3d')
13     ax.axis('off')
14
15     # prepare the axes limits
16     ax.set_xlim((-25, 25))
17     ax.set_ylim((-35, 35))
18     ax.set_zlim((5, 55))
19
20     def lorenz_deriv(x_y_z, t0, sigma=sigma, beta=beta, rho=rho):
21         """Compute the time-derivative of a Lorenz system."""
22         x, y, z = x_y_z
23         return [sigma * (y - x), x * (rho - z) - y, x * y - beta * z]
24
25     # Choose random starting points, uniformly distributed from -15 to 15
26     np.random.seed(1)
27     x0 = -15 + 30 * np.random.random((N, 3))
28
  
```


- JupyterLite is a JupyterLab distribution that **runs entirely in the browser** built from the ground-up using JupyterLab components and extensions.



The screenshot shows a web browser window displaying a JupyterLite notebook. The notebook title is "pyolite - interactive-widgets.ipynl". The interface includes a menu bar (File, Edit, View, Run, Kernel, Diagram, Tabs, Settings, Help) and a toolbar with various icons. The notebook content is as follows:

```
ipywidgets Interactive Demo
```

Simple demonstration of rendering Interactive widgets in a `jupyterlite` notebook.

`ipywidgets` can be installed in this deployment (it provides the `@jupyter-widgets/jupyterlab-manager` federated extension), but you will need to make your own deployment to have access to other interactive widgets libraries.

```
[1]: import micropip
    await micropip.install('ipywidgets')
```

```
[2]: from ipywidgets import IntSlider
```

```
[3]: slider = IntSlider()
```

```
[4]: slider
```

43

```
[5]: slider
```

43

```
[6]: slider.value
```

46

```
[7]: slider.value = 5
```

```
[8]: from ipywidgets import IntText, link
```

```
[9]: text = IntText()
```



Simple 0 1 Pyolite | Idle Saving completed Mode: Command Ln 1, Col 1 pyolite - interactive-widgets.ipynb




Start it now!




- goto: <https://jupyterlite.readthedocs.io>
and click the Lab Button:




Try  Lab  Retro

 User Guide Developer C

Search the docs ...

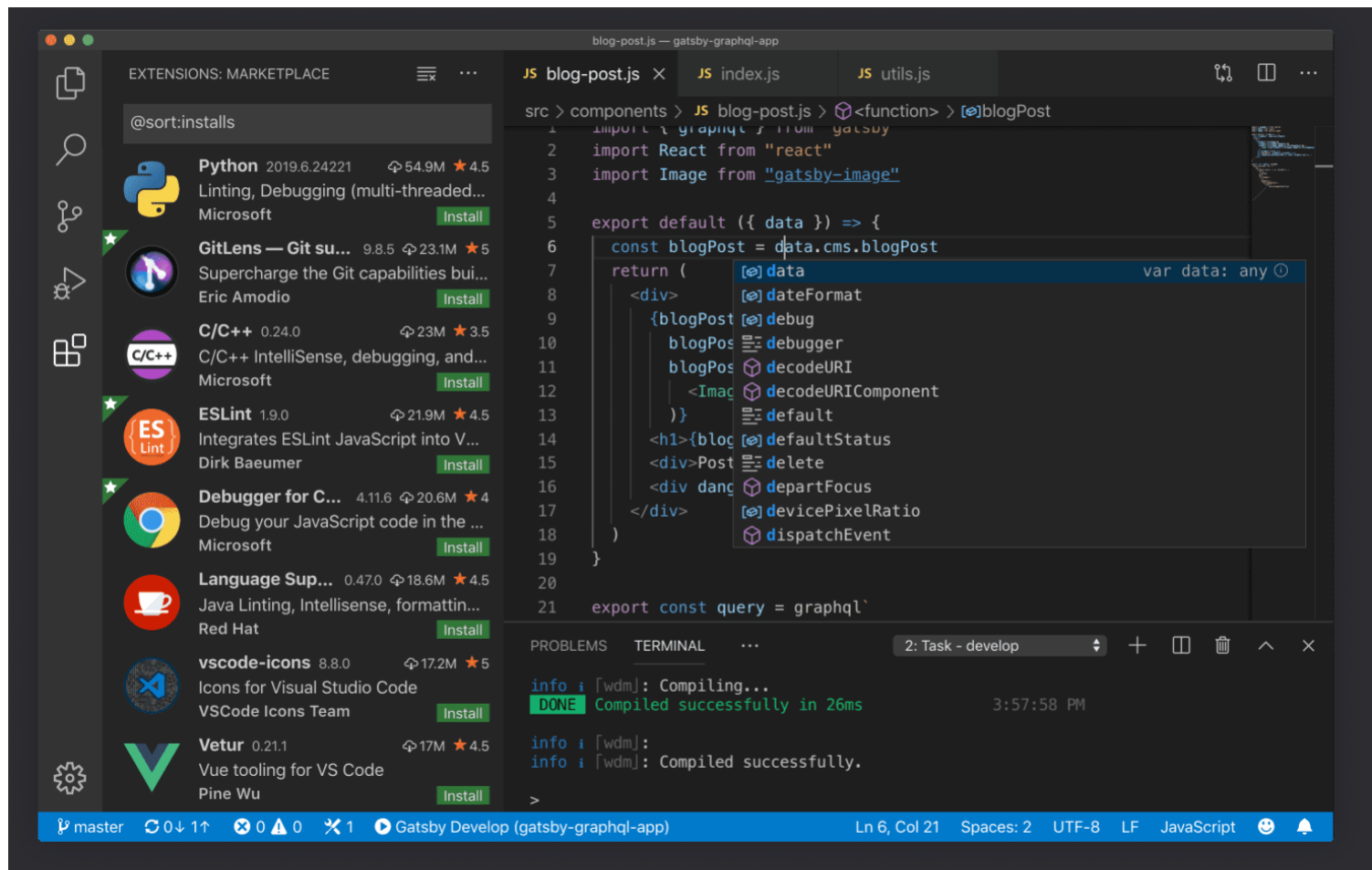


JupyterLite

 Build passing  launch binder  docs passing

visual studio code

- Visual Studio Code is a very popular IDE which is free and open source and it is possible to use it on the Linux Cluster for development.



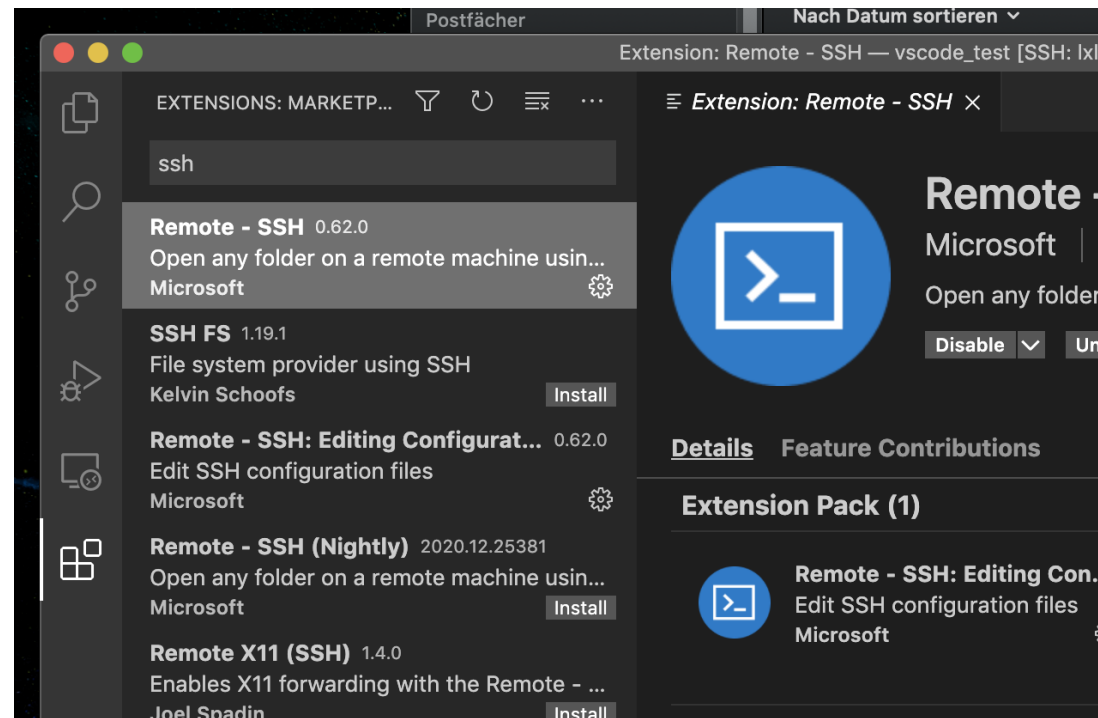


visual studio code installation

- See the following steps for getting it up and running
 - Download VS Code from the Microsoft Website <https://code.visualstudio.com/download>
 - Choose your local OS and run the local installation
 - Start it up the first time

visual studio code

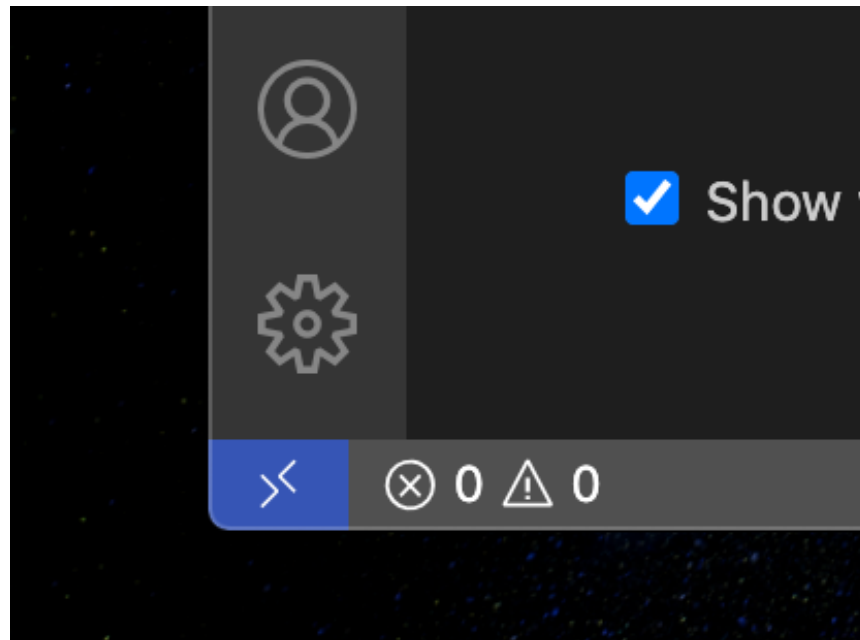
- Click on the lower icon for Extensions and search for ssh
- When Remote - ssh comes up, install it





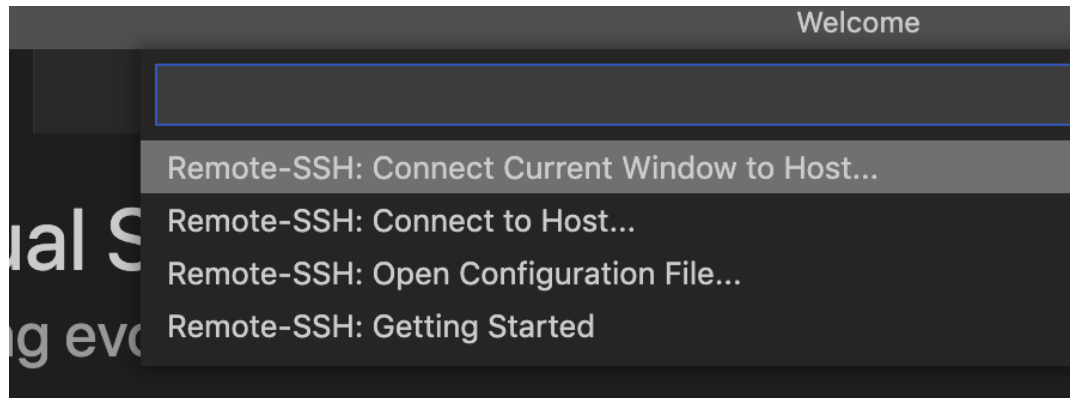
open ssh

- Now you have to connect to the Linux Cluster to edit and run the files.
- In the lower left corner you should now see the connection status for ssh:

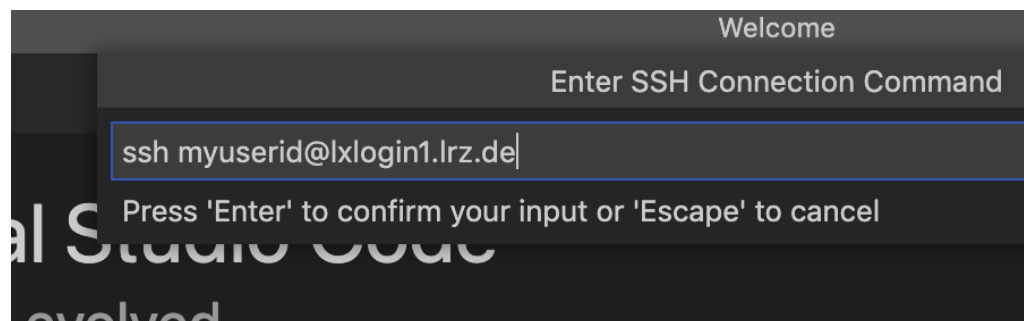


Connect to Host

- Click on it and choose "Connect to Host..."



- now select
- + Add new SSH Host
- enter your user id together with the address





Login with password

