

# Primer: Running JupyterLab on the ACES Portal

Presented by Siddhi Mittal  
November 7th, 2025

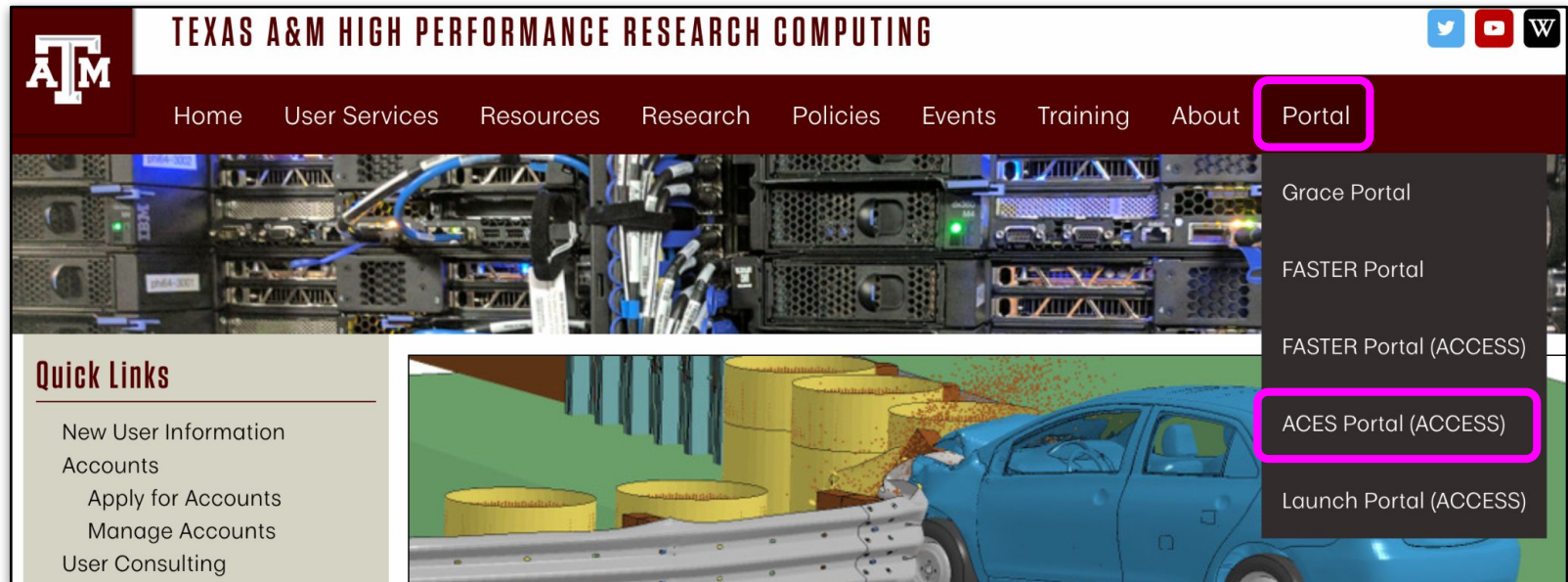


High Performance  
Research Computing  
DIVISION OF RESEARCH



# The ACES Portal

- Get your ACCESS ID ready!
- Head to HPRC webpage: [hprc.tamu.edu](http://hprc.tamu.edu) and navigate to Portal menu
- ACES portal shortcut - [portal-aces.hprc.tamu.edu](http://portal-aces.hprc.tamu.edu)



# JupyterLab Interactive App

ACES OnDemand Portal

Files ▾

Jobs ▾

Clusters ▾

Interactive Apps ▾

In the portal, navigate to:

**> Interactive Apps**

**>> Servers section:**

**>>> JupyterLab**

## Servers



Jupyter AI Assistant



Jupyter Notebook



JupyterLab



RStudio



TensorBoard



Tutorials OnDemand

# JupyterLab Interactive App

## Supported Environment Types

- TAMU Modulair
- Container (Singularity)
- Anaconda environment
- Module load + Python Virtualenv

ACES OnDemand Portal Files ▾ Jobs ▾ Clusters ▾ Interactive Apps ▾ Affinity Groups ▾ Chatbot ▾ Dashboard ▾ Training ▾

Home / My Interactive Sessions / JupyterLab

Interactive Apps

GUI

NextSilicon VNC

VNC

Imaging

ChimeraX

CryoSPARC

CryoSPARC 4.2.1

ImageJ

Jmol

Paraview

### JupyterLab

This app will launch a [JupyterLab](#) server on the [ACES](#) cluster.

**Type of environment**

Module load + Python virtualenv ▾

Select the type of environment in which Jupyter is installed.  
[Help me choose](#)

**Python module to be loaded**

Python/3.11.5

Select a Python module to load.

**Optional Python virtual environment**

Enter the full path to a python virtual environment

Next we have to specify the environment and resources for our JupyterLab session

# Python Environments on ACES

For now:

Choose these two:

Leave this blank:

## JupyterLab

This app will launch a [JupyterLab](#) server on the [ACES](#) cluster.

### Type of environment

Module load + Python virtualenv

Select the type of environment in which Jupyter is installed.

[Help me choose](#)

### Python module to be loaded

Python/3.11.5

Select a Python module to load.

### Optional Python virtual environment to be activated

Enter the full path to a python virtual environment "activate"

# Session Parameters on ACES

- Provide **node type** (CPU only, H100 GPU or A30 GPU)
  - CPU
- Provide **number of hours** [1-72] for running jupyter lab on the cluster
  - 1 core
- Specify **number of cores** [1-96] allocated on node on the cluster
  - 1 core
- Requested **total memory** [5-485 GB]
  - 5 GB

## Node type

CPU only

- [cpuavail](#) [gpuavail](#) select a non-CPU node type only if your software supports the Accelerator

## Number of hours (max 72)

1

## Number of cores (max 96)

1

## Total GB Memory (max 488)

5

# Optional Session Parameters on ACES

You can ignore these today, but you can also:

- Specify an account number to charge SUs
- Provide an email to be notified when session is ready to launch
- Choose to work in a group directory instead of user's \$SCRATCH
- Advanced Parameters:
  - Reservation name
  - Node name
  - Environment file to source (e.g. additional path for \$PATH or \$LD\_LIBRARY\_PATH)

☐ I would like to use an account other than my default

☐ I would like to receive an email when the session starts

☐ Select if you want to work in a group directory instead of your \$SCRATCH

☒ Advanced Params

**Reservation Name**

**Node Name**

**Environment file to source**

Select Path

Launch

# Interactive Sessions on ACES Portal

After clicking **Launch**, you are sent to the **My Interactive Sessions** page.

- Your job will first show as a blue box with “JupyterLab (<job ID>)” at the top
- ‘Session ID’ is link to a directory where you can find the jobs logs
- Click **Connect to JupyterLab** when it appears

(box will turn green; speed depends on cluster usage)

The screenshot displays a user interface for managing interactive sessions. At the top, a green header bar contains the text 'JupyterLab (1279167)' on the left and '1 node | 1 core | Running' on the right. Below this, the 'Host' is listed as '>\_ ac009' with a blue dropdown arrow, and a red 'Delete' button with a white 'x' icon is positioned to the right. The 'Created at' timestamp is '2025-10-16 10:15:18 CDT'. The 'Time Remaining' is '59 minutes'. The 'Session ID' is '90943b73-7221-49b3-aff3-2a23858ce830', which is highlighted with a pink rectangular box. Below the session ID, the 'Type of environment' is 'pythonvenv'. At the bottom of the card, a blue button with a white eye icon and the text 'Connect to JupyterLab' is highlighted with a pink rectangular box.



# Jupyter Notebook vs JupyterLab

## Jupyter Notebook

- Web-based interactive environment
- Simpler more lightweight interface
- Can only open Jupyter Notebooks
- Used for data analysis and visualization
- Supports languages like Python, Julia, R, etc.

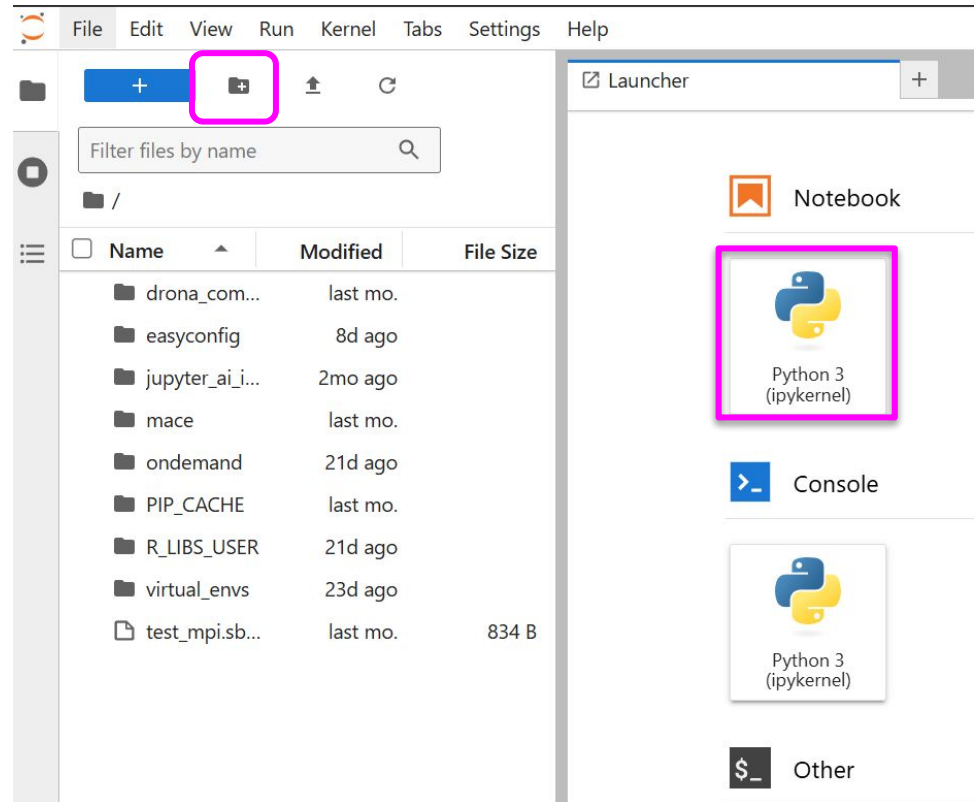
## JupyterLab

- IDE like experience
- Versatile and feature-rich interface
- Modular structure where you can open different file types:-
  - Jupyter Notebook
  - Terminal sessions
  - PDF, HTML, txt files
  - and more!
- Supports languages like Python, Julia, R, etc.

# Welcome to JupyterLab

## File Navigator in JupyterLab

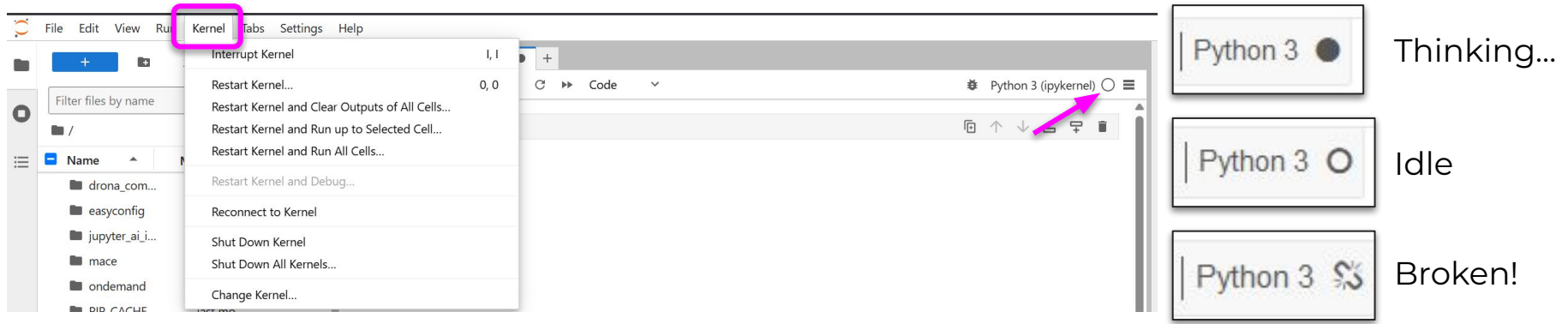
- Default directory: user's \$SCRATCH directory or optionally-provided group directory
- Create new directories using the New Folder icon
- Create a new Jupyter Notebook using the first icon in the Notebook section on the right panel
- If you are uploading an existing .ipynb file, ensure the Python version is compatible



# Exercise 1

1. In JupyterLab, create a new directory and go into it.
2. Create a new Jupyter Notebook file inside the directory.  
Save the file regularly to avoid losing changes.

# Jupyter Notebook Kernels



The “kernel” is like the brain of your notebook. It determines what language you are thinking in—and if it is not working, you cannot process anything.

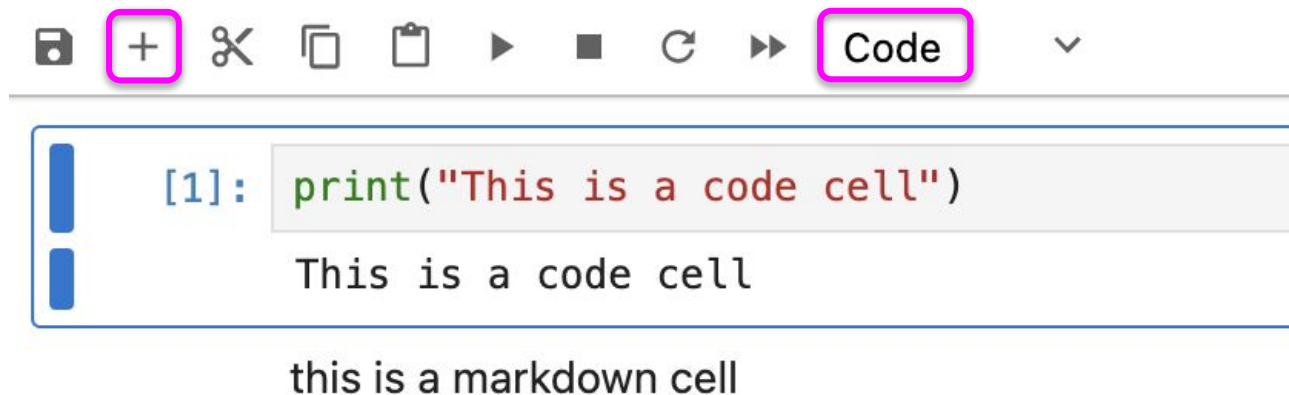
In the top-right of your window is the kernel status, showing what language you are using and what the kernel is doing.

When you first start a notebook, it will take a moment for the kernel to start up. If something goes wrong, you may have to restart it.

# Cells in Jupyter Notebook

Notebooks are organized into Cells.

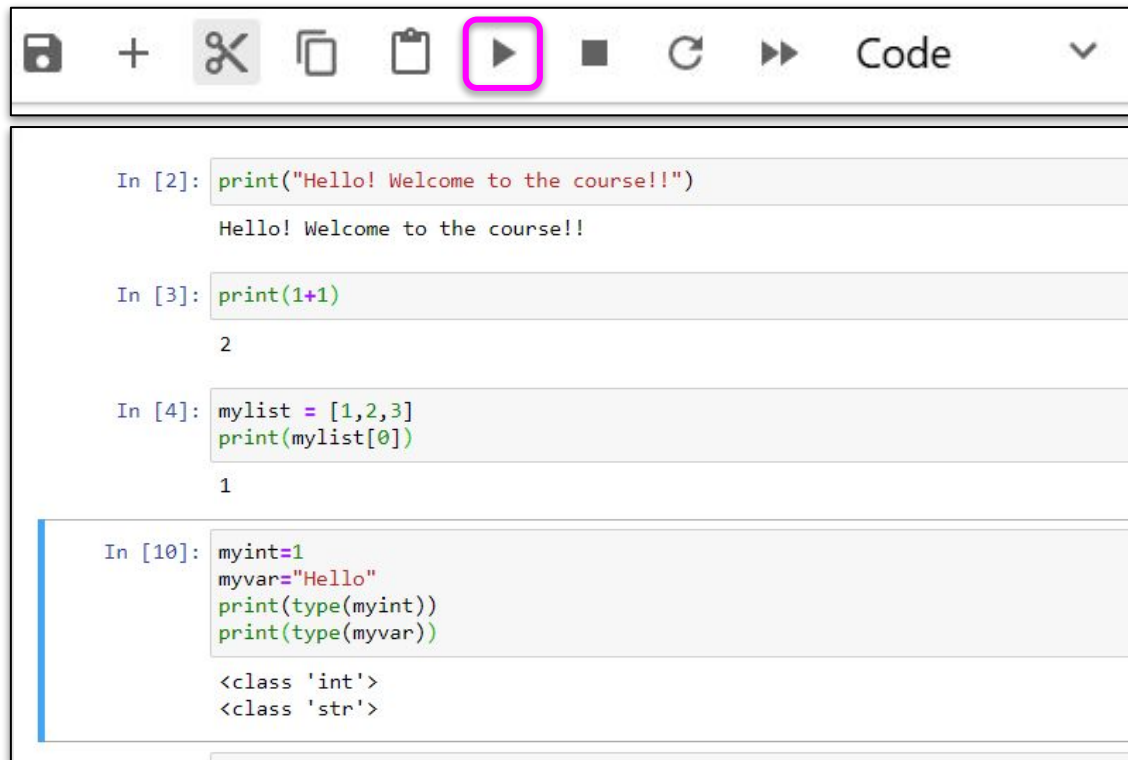
- These can be either *code* cells or *markdown* cells (we'll start with code).
- Click to the left of a cell to highlight it (blue, for “command mode”)
- Click inside the body of the cell to edit it
- Add new cells using the “+” icon



# Executing Code in Jupyter Notebooks

**ctrl+enter** or click the **Run** button to run a cell.

The brackets to the left will show [\*] if the cell is running and change to a number (showing what order the cells were executed) once it's done.



```
In [2]: print("Hello! Welcome to the course!!")
Hello! Welcome to the course!!

In [3]: print(1+1)
2

In [4]: mylist = [1,2,3]
print(mylist[0])
1

In [10]: myint=1
myvar="Hello"
print(type(myint))
print(type(myvar))

<class 'int'>
<class 'str'>
```

# Python in Jupyter Notebooks

Cells are just blocks of code you choose to execute all at once. You can execute them in any order, and any defined objects persist across them.

All that matters is what order you send code to the kernel.



```
In [2]: #I'm going to execute this cell second!  
x = 5 * 20  
  
In [3]: #I'm going to execute this cell last!  
print(x)  
  
100  
  
In [1]: #I'm going to execute this cell first!  
print("Hello world!")  
  
Hello world!
```

# Plotting in Jupyter Notebooks

Making plots in Jupyter depends somewhat on your environment.

You may need a “notebook magic” to make images show up.

*(magics are a big, kernel-dependent topic we won't cover here!)*

```
[6]: # A simple plot with matplotlib
from matplotlib import pyplot as plt

x = [0, 1, 2, 3, 4, 5, 6, 7, 8, 9]
y = [2, 4, 7, 11, 6, 2, 0, -4, -1, 2]

plt.plot(x, y)
plt.xlabel("x")
plt.ylabel("y")
plt.show()
```

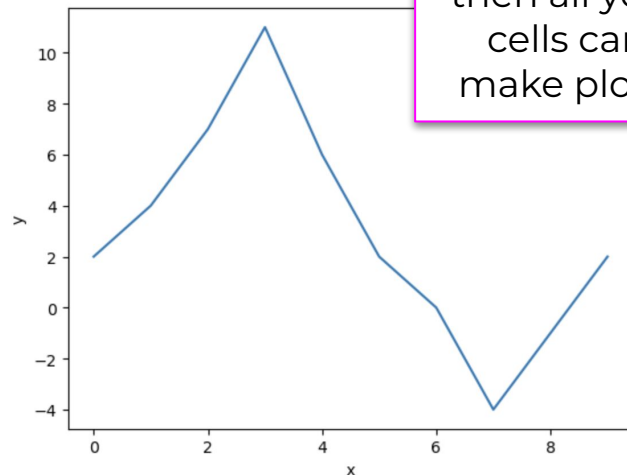
No magic,  
no plot!

```
[15]: %matplotlib inline

[16]: # A simple plot with matplotlib
from matplotlib import pyplot as plt

x = [0, 1, 2, 3, 4, 5, 6, 7, 8, 9]
y = [2, 4, 7, 11, 6, 2, 0, -4, -1, 2]

plt.plot(x, y)
plt.xlabel("x")
plt.ylabel("y")
plt.show()
```



Magic first,  
then all your  
cells can  
make plots!



# Exercise 2

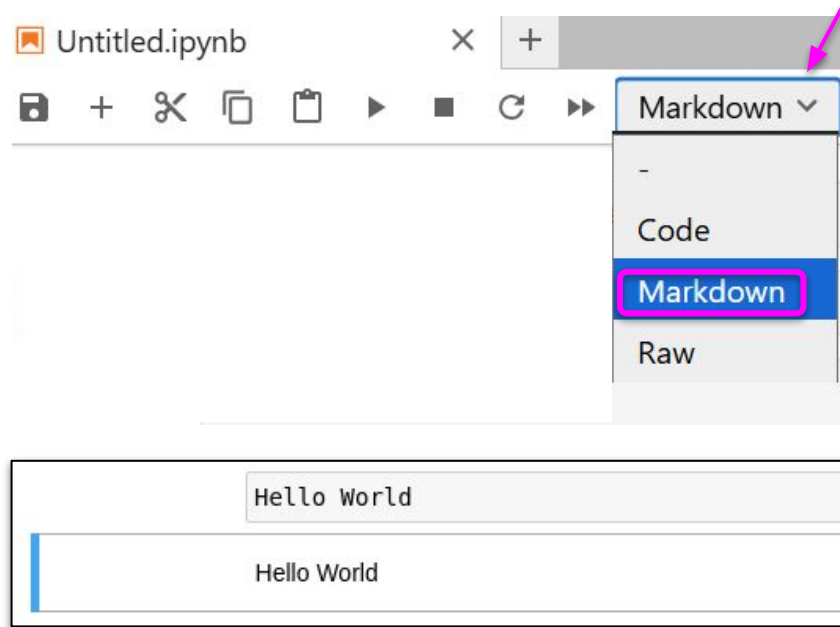
1. Enter some Python code in a cell. Print a message: “Welcome to HPRC”.
2. Try to make a simple plot, like on the previous slide.

# Markdown in Jupyter Notebooks

Markdown cells are used for annotation, documentation, organization, etc.

- Convert a cell from drop down menu in the toolbar (can also be found in the Cell menu)
- As with code cells, **Ctrl+enter** or **Run** to execute and format Text.

Font changes from monospace to something more *beautiful*.



# Markdown in Jupyter Notebooks

- *Headers*  
# (Header 1, title) stands for html code  
<h1>Header 1,title</h1>  
Add more '#' for smaller headers
- *Line Break*  
Insert line breaks using <br>
- *Inline Formatting*  
Bold: **HPRC**  
Italics: *HPRC*  
Horizontal Line: \*\*\*
- *Embed external link*  
Works like html:  
<a href="url" >link text</a>

## # Look! Markdown!

Normal text<br>  
Normal text on a new line

**HPRC**

*HPRC*

\*\*\*

<https://hprc.tamu.edu/>Link to HPRC

## Look! Markdown!

Normal text  
Normal text on a new line

**HPRC** *HPRC*

---

[Link to HPRC](https://hprc.tamu.edu/)

# Exercise 3: Markdown

1. Create a Markdown Cell and add some text.
2. Change the formatting as indicated below:
  - Bold
  - Italics
  - Header 1 format
  - Include a link (<https://hprc.tamu.edu/>) next to the text

# More Text Formats in Jupyter Notebooks

## LaTeX: for mathematical operations

- In a Markdown cell:
- `$ math expression $`

```
$\sqrt{k}$
```

$\sqrt{k}$

## Embed Code in Text:

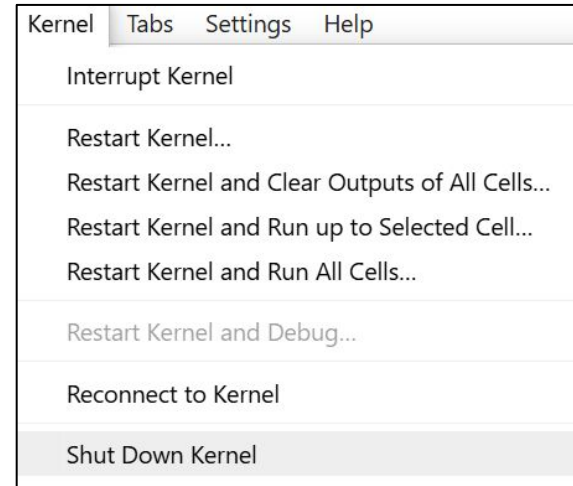
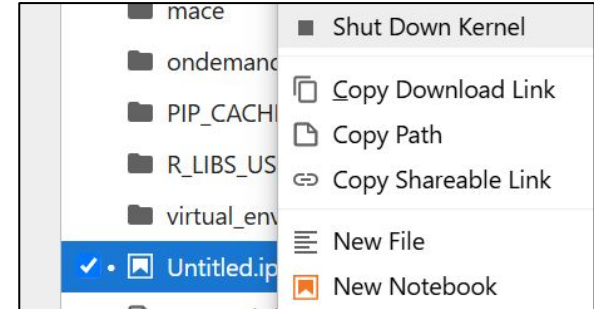
- In a Markdown cell:
- use the triple backticks ````` followed by a language name.

```
```Python
str = "This is block level code"
print(str)
```
```

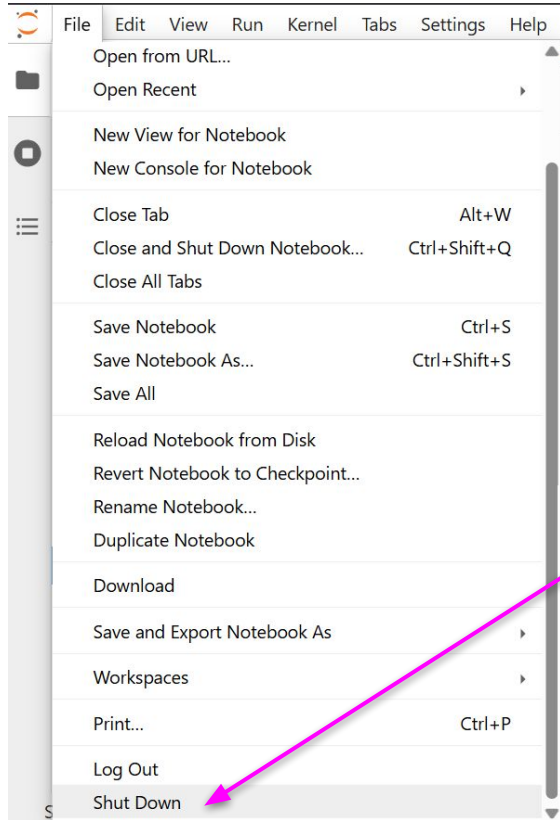
```
str = "This is block level code"
print(str)
```

# Shutting Down

- Notebooks (i.e. their kernels) keep running even if you exit their browser tab
- To shut one down, find it in the Jupyter file browser
- Right click and choose “Shutdown Kernel”
- Can also shutdown kernel in the “Kernel” menu at the top and shut them down there.



# Ending the Jupyter Session



As long as your Jupyter session is running (not just notebooks!) you're taking up compute resources!

To quit before the time you specified at the beginning, go to File → Shut Down

Back in “My Interactive Session” in the HPRC Portal, the box for your session will turn grey to indicate it is done.

JupyterLab (1279167)

Completed |  | 

**Created at:** 2025-10-16 10:15:18 CDT

**Session ID:** 90943b73-7221-49b3-aff3-2a23858ce830

**Type of environment:** pythonvenv

For debugging purposes, this card will be retained for 6 more

 Delete

You can click on the Session ID to go to the log files if you want to debug!

# Exercise 4: Shutting Down

1. Shut down your Jupyter session
2. Find the log files for your recent session.  
(Hint: the Session ID **link** from the My Interactive Sessions is a shortcut; you should end up in:  
`$SCRATCH/ondemand/data/sys/dashboard/batch_connect/sys/jupyter/output/`  
(You may need these some day to diagnose errors.)



# Need Help?

First check the FAQ <https://hprc.tamu.edu/kb/FAQ/Accounts/>

- ACES User Guide <https://hprc.tamu.edu/kb/User-Guides/ACES/>
- Email your questions to [help@hprc.tamu.edu](mailto:help@hprc.tamu.edu)

Help us help you -- when you contact us, tell us:

- Which Cluster you're using
- Your Username
- Job id(s) if any
- Location of your jobfile, input/output files
- Application used if any
- Module(s) loaded if any
- Error messages
- Steps you have taken, so we can reproduce the problem

# Continued Learning

[Intro to HPRC YouTube Channel](#)

[HPRC's Knowledge Base](#)



High Performance  
Research Computing  
DIVISION OF RESEARCH

Give us feedback on the class with this survey:  
[https://u.tamu.edu/hprc\\_shortcourse\\_survey](https://u.tamu.edu/hprc_shortcourse_survey)

Thank you  
*Questions?*



HPRC Survey