

Week 1 Tutorial: Introduction to Jupyter, CLI, Git

POP77001 Computer Programming for Social Scientists

Module website: tinyurl.com/POP77001

Integrated Development Enviroments (IDEs)

- There is a number of integrated development environments (*IDEs*) available for:
 - R (RStudio) and
 - Python (Spyder, PyCharm)
- As well as text editors with R/Python-specific extensions (Visual Studio Code, Atom, Sublime Text, Vim)
- Try different ones and choose what works best for you!

Jupyter Notebook

- [Jupyter Notebook](#) is a language-agnostic web-based interactive computational environment.
- It is available with backends (*kernels*) for different programming languages (**Julia**, **Python**, **R** = **Jupyter**)
- Can be used both locally and remotely.
- Good for ad-hoc data analysis and visualization.

Jupyter Notebook Cells

- Notebooks allow writing, executing and viewing the output of Python code within the same environment
- All notebook files have `.ipynb` extension for **interactive python notebook**
- The main unit of notebook is *cell*, a text input field (Python, Markdown, HTML)
- Output of a cell can include text, table or figure

Jupyter Notebook Installation

- There are two main ways to install Jupyter Notebook locally:
 - [pip](#) and
 - [conda](#)
- Unless you have prior experience with Python, I recommend installing [Anaconda](#) distribution, which contains all the packages required for this course.
- Alternatively, you can try using [Kaggle Code](#) or [Google Colab](#), online platforms for hosting Jupyter Notebooks.
- Their interfaces are slightly different and you need to register on Kaggle or have a Google account, but it does not require any local installations.
- However, for this module and course more broadly I recommend installing toolchain for working with Jupyter Notebooks locally.

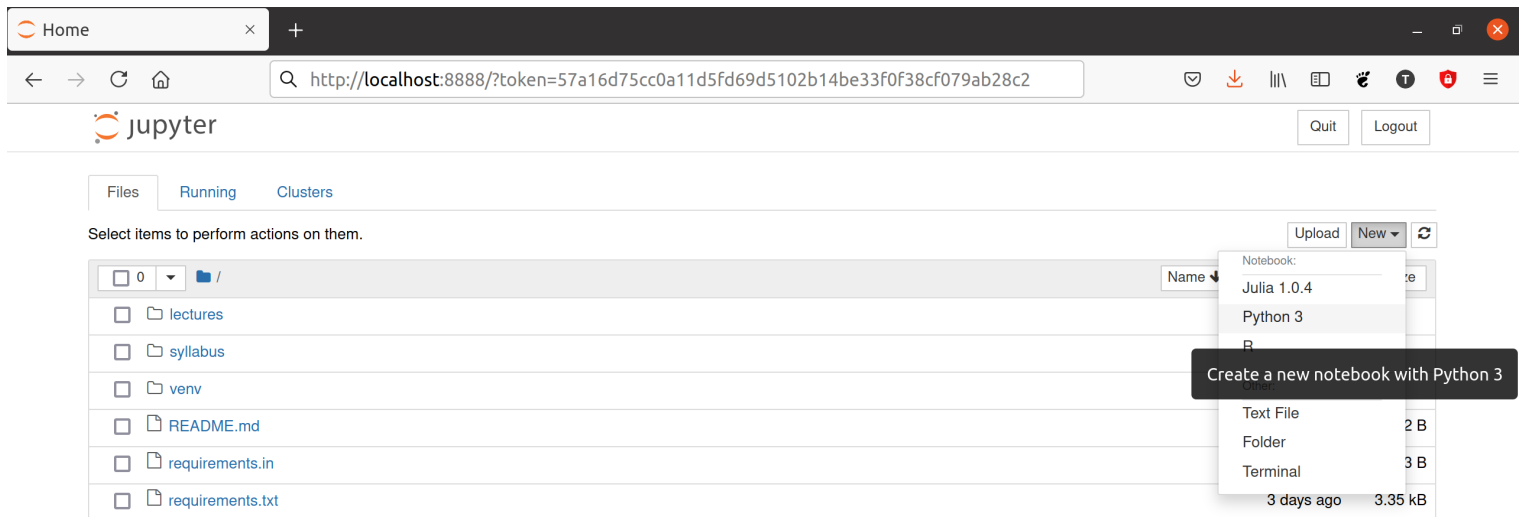
Starting Jupyter

- To start Jupyter, open CLI/Terminal and type `jupyter notebook`
- This will open a browser window with Jupyter Notebook displaying the directory, in which you executed the command above.
- To create a new notebook press `New` and select `Python` from the drop-down menu

Using Jupyter

- In order to run a Python command, create a new cell:
 - Press **+** in the toolbar or click **Insert**, **Insert Cell Below**
 - Make sure that in the drop-down menu on the toolbar you select **Code**
 - Press CTRL+ENTER to run a command
- Rather than running a Python command, you can also write Markdown in the cell (e.g. to create slides)
 - Select **Markdown** in the drop-down menu on the toolbar
 - Write Markdown (check [Markdown Cheatsheet](#))
 - Press CTRL+ENTER to render Markdown cell

Jupyter Notebook Demonstration



The screenshot displays the Jupyter Notebook web interface in a browser window. The address bar shows the URL `http://localhost:8888/?token=57a16d75cc0a11d5fd69d5102b14be33f0f38cf079ab28c2`. The interface includes a top navigation bar with the Jupyter logo and 'Quit' and 'Logout' buttons. Below this, there are tabs for 'Files', 'Running', and 'Clusters'. The 'Files' tab is active, showing a file browser with a list of files and folders: `/`, `lectures`, `syllabus`, `venv`, `README.md`, `requirements.in`, and `requirements.txt`. A 'New' button in the top right corner of the file browser is open, showing a dropdown menu with options: 'Notebook:', 'Julia 1.0.4', 'Python 3', 'Text File', 'Folder', and 'Terminal'. A dark tooltip message 'Create a new notebook with Python 3' is visible over the 'Python 3' option. The file browser also shows file sizes and modification dates, such as '3 days ago' and '3.35 kB'.

Home x +

← → ↻ 🏠 🔍 `http://localhost:8888/?token=57a16d75cc0a11d5fd69d5102b14be33f0f38cf079ab28c2` 📁 📄 📌 📧 🔒 ⋮

jupyter Quit Logout

Files Running Clusters

Select items to perform actions on them.

0 ▾ /

lectures

syllabus

venv

README.md

requirements.in

requirements.txt

Upload New ↕

Notebook:

Julia 1.0.4

Python 3

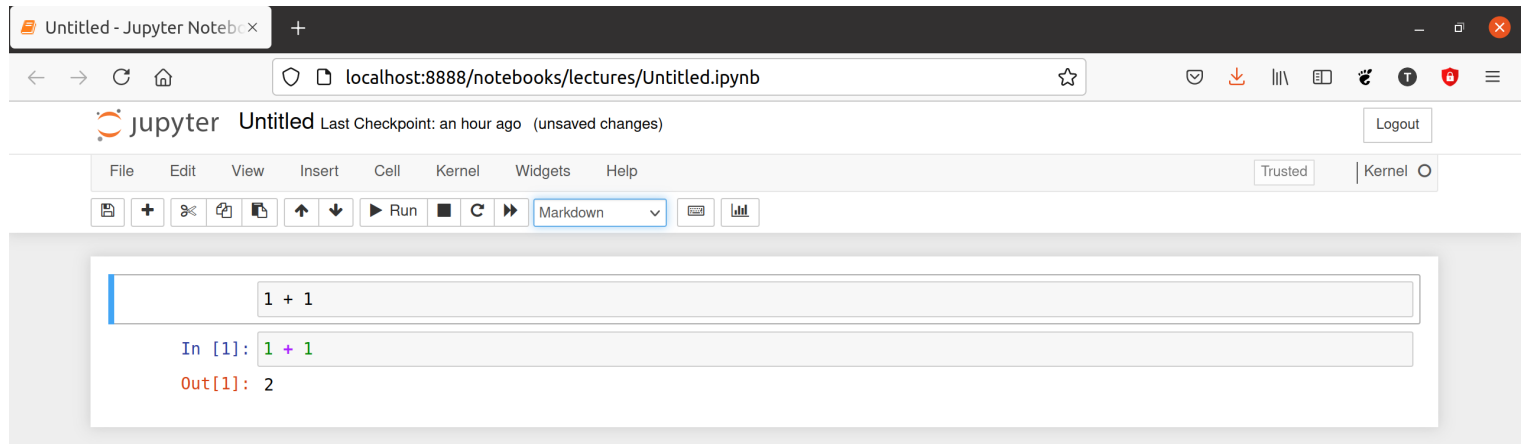
Text File 2 B

Folder 3 B

Terminal 3 days ago 3.35 kB

Create a new notebook with Python 3

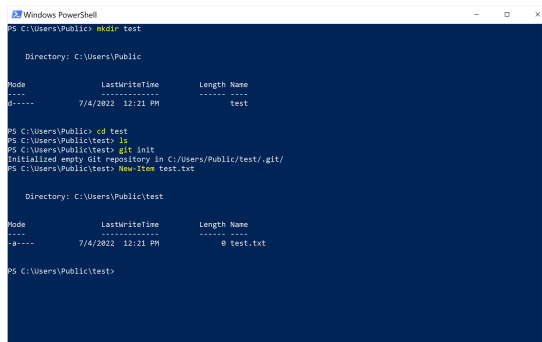
Jupyter Notebook Demonstration



Stopping Jupyter Notebook

- First, make sure you saved your work (!) by pressing Command+S / CTRL+S
- You can close the running notebook by clicking **File** and then **Close and Halt**
- Jupyter Notebook runs as a server
- Which means that closing its tabs/web browser does not stop it
- You need to press **Quit** in the upper right corner of your main Jupyter tab (located at <http://localhost:8888/>)
- Alternatively, you can press CTRL+C in the terminal window

CLI Examples



```
Windows PowerShell
PS C:\Users\Public> mkdir test

Directory: C:\Users\Public

Mode                LastWriteTime         Length Name
----                -
d-----          7/4/2022 12:21 PM             test

PS C:\Users\Public> cd test
PS C:\Users\Public\test> ls
PS C:\Users\Public\test> git init
Initialized empty Git repository in C:\Users\Public\test\.git\
PS C:\Users\Public\test> New-Item test.txt

Directory: C:\Users\Public\test

Mode                LastWriteTime         Length Name
----                -
-a-----          7/4/2022 12:21 PM             0 test.txt

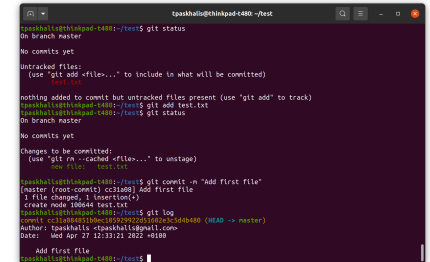
PS C:\Users\Public\test>
```

Microsoft PowerShell
(Windows)



```
zsh: macOS: Fri Sep 2 02:22:11 AM console
(tash) tash@Macintosh-MBP ~ %
```

Z shell, zsh
(macOS)



```
tashkhalis@tashkhalis-pc:~/test$ git status
On branch master
No commits yet
Untracked files:
  (use "git add <file>..." to include in what will be committed)
  test.txt
nothing added to commit but untracked files present (use "git add" to track)
tashkhalis@tashkhalis-pc:~/test$ git add test.txt
tashkhalis@tashkhalis-pc:~/test$ git status
On branch master
No commits yet
Changes to be committed:
  (use "git rm --cached <file>..." to unstage)
  new file:   test.txt
tashkhalis@tashkhalis-pc:~/test$ git commit -m "Add first file"
[master (root commit) cca08b] Add first file
1 file changed, 1 insertion(+)
create mode 100644 test.txt
tashkhalis@tashkhalis-pc:~/test$ git log
commit cca08b3c9e0777777777777777777777 (HEAD -> master)
Author: tashkhalis <tashkhalis@gmail.com>
Date:   Wed Sep 21 12:31:23 2022 +0300
    Add first file
tashkhalis@tashkhalis-pc:~/test$
```

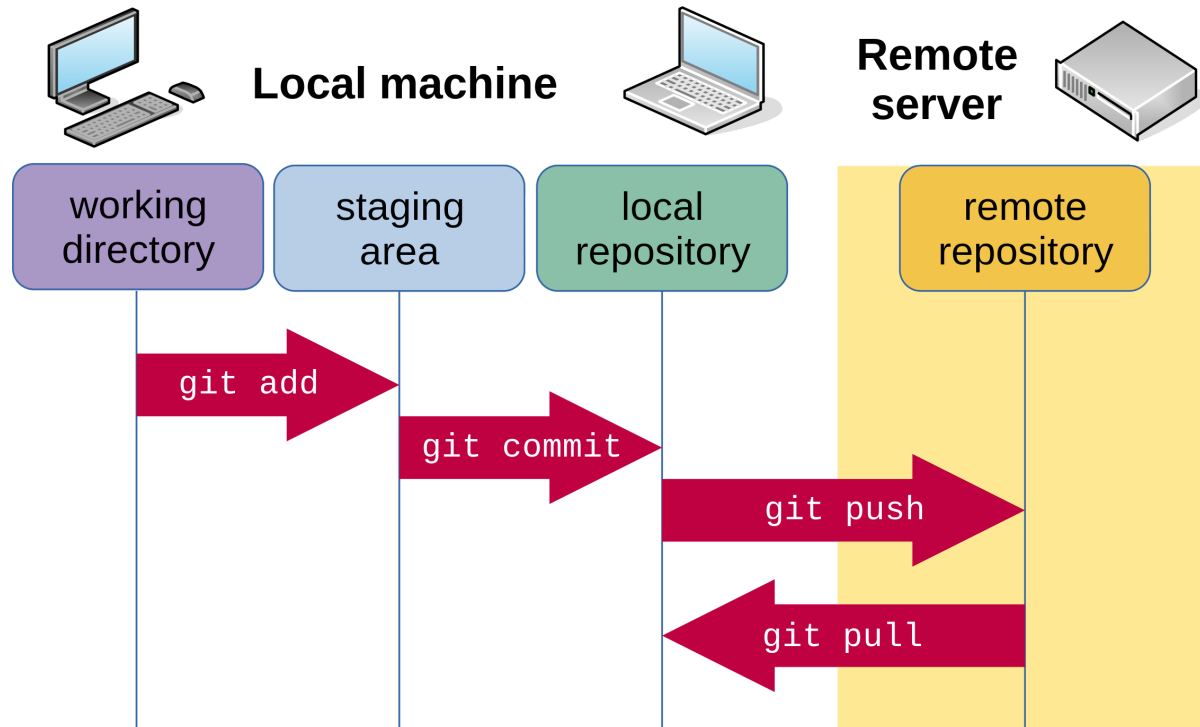
bash (Linux/UNIX)

Some Useful CLI Commands

Command (Windows)	Command (macOS/Linux)	Description
<code>exit</code>	<code>exit</code>	close the window
<code>cd</code>	<code>cd</code>	change directory
<code>cd</code>	<code>pwd</code>	show current directory
<code>dir</code>	<code>ls</code>	list directories/files
<code>copy</code>	<code>cp</code>	copy file
<code>move</code>	<code>mv</code>	move/rename file
<code>mkdir</code>	<code>mkdir</code>	create a new directory
<code>del</code>	<code>rm</code>	delete a file

Extra: [Introduction to CLI](#)

Git/GitHub Workflow



Some Useful Git Commands

Command	Description
<code>git init <project name></code>	Create a new <i>local repository</i>
<code>git clone <project url></code>	Download a project from <i>remote repository</i>
<code>git status</code>	Check project status
<code>git diff <file></code>	Show changes between <i>working directory</i> and <i>staging area</i>
<code>git add <file></code>	Add a file to the <i>staging area</i>
<code>git commit -m "<commit message>"</code>	Create a new <i>commit</i> from changes added to the <i>staging area</i>
<code>git pull <remote> <branch></code>	Fetch changes from <i>remote</i> and merge into <i>merge</i>
<code>git push <remote> <branch></code>	Push local branch to <i>remote repository</i>

Extra: [Git Cheatsheet](#)

Creating local Git repository


- Let's create a test project and track changes in it
- Create a test directory by typing `mkdir test` in your CLI/Terminal
- Go into the newly created directory with `cd test` command
- To make Git track changes run `git init` command in this directory
- Congratulations! You now have a local repository for your test project

Making a commit

- Open your text editor of choice (Notepad, Sublime Text, Atom, Visual Studio Code, Vim, Emacs, ...)
- Create a file called `test.txt` in your local test repository
- Type whatever you like in this file
- Add this file to your staging area (make Git aware of its existence) by running `git add test.txt` command
- Commit this file to your local repository by running `git commit -m "Added first file"`
- Note that all files that were added at the previous stage with `git add <file>` would be committed
- Check the status of your repository by running `git status` (it should say 'nothing to commit, working tree clean')
- Check the history of your repository by running `git log` and make sure that you see your commit

Remote Git repository: GitHub



- Hosting platform for projects that rely on Git for version control
- Bought by Microsoft in 2018
- Provides extensive tools for collaborative development and search functionality
- Helpful for troubleshooting more narrow problems (check [GitHub Issues](#) of the package/library that you have a problem with)
- GitHub is far from the only platform for hosting Git projects
- Popular alternatives to GitHub include [GitLab](#) () , [SourceForge](#) , ...

Creating remote repository on GitHub

- Register and login into your account on GitHub
- Create a [new GitHub repository](#) (choose private repository)
- You should see a similar page with the project URL of the form:

```
https://github.com/<username>/<repository_name>.git
```

Quick setup — if you've done this kind of thing before

or

HTTPS

SSH

https://github.com/tpaskhalis/test.git

Get started by [creating a new file](#) or [uploading an existing file](#). We recommend every repository include a [README](#), [LICENSE](#), and [.gitignore](#).

...or create a new repository on the command line

```
echo "# test" >> README.md
git init
git add README.md
git commit -m "first commit"
git branch -M main
git remote add origin https://github.com/tpaskhalis/test.git
git push -u origin main
```

...or push an existing repository from the command line

```
git remote add origin https://github.com/tpaskhalis/test.git
git branch -M main
git push -u origin main
```

...or import code from another repository

You can initialize this repository with code from a Subversion, Mercurial, or TFS project.

Import code

Synchronising local Git repository with GitHub

- Go to your local Git repository (the one created in the previous step)
- Add link from your local Git repository to remote repository on GitHub by running:

```
git remote add origin <project_url>
```

- where:
 - `git remote add` is the command,
 - `origin` is the name given to this link (`<remote>`), and
 - `<project_url>` is the URL of the repository on GitHub
- Check the status of links between your local Git repository and remotes by running `git remote -v`
 - where:
 - `git remote` is the command, and
 - `-v` is the argument 'verbose'

Pushing local Git changes to GitHub

- Your local Git repository is now linked to the remote repository hosted on GitHub.
- Let's bring the changes made locally to the remote repository.
- We will use the `git push` command for that.
- One last thing to check before doing so is which branch we are currently on.
- Run `git branch` to see the name of the branch you are on (it would be 'master' or 'main')
- Finally, run `git push <remote> <branch>` (e.g. `git push origin master`)
 - where:
 - `git push` is the command,
 - `<remote>` is the name of the remote link, and
 - `<branch>` is the name of the branch.
- Visit your GitHub repository to check that your commit is reflected there.

Cloning module repository

- All module materials are hosted on GitHub in this [repo](#)
- You can clone this repository to your local machine by running:

```
git clone https://github.com/ASDS-TC/POP77001_Computer_Programming_2022
```

- This will create a folder called `POP77001_Computer_Programming_2022` within the directory where you ran this command
- To keep up to date with changes in the remote repository you can run:

```
git pull origin main
```

- Where
 - `origin` is the remote address of the repository - `https://github.com/ASDS-TC/POP77001_Computer_Programming_2022`
 - `main` is the name of the branch (recall the discussion about `main/master` change from the lecture)

Week 1 Exercise (Unassessed)

- Create a Jupyter notebook in your local repository
- Commit it to your local repository in the same way as `test.txt` file