CYPLAN 255

Urban Informatics and Visualization

HIT RECORD

Lecture 03 – GitHub Cont'd, Python at the Command-Line

January 26, 2022

Agenda

1. Announcements
2. Getting started with GitHub (cont'd)
3. Python at the command-line
4. For next time
5. Questions

1. Announcements

Announcements

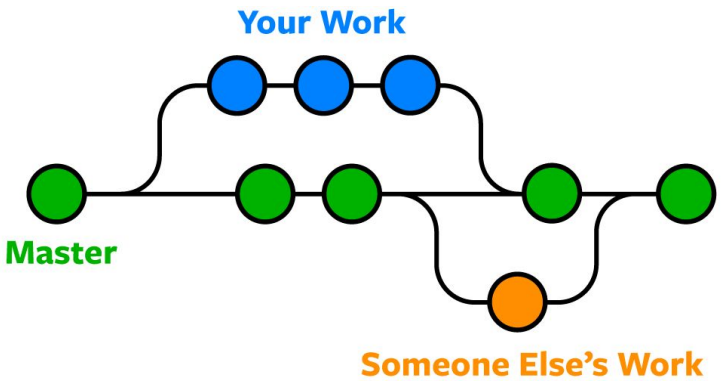
1. Readings
2. Exercises
3. Datasets

2. Git + GitHub (cont'd)

https://docs.github.com/en/get-started/getting-started-with-git

Git

- Git is a tool for *distributed version control*
  - Track changes
  - Log history of changes
  - Merge changes and histories from multiple contributors



[2]

GitHub

- GitHub is a website/service for hosting Git-based projects
- Stores a **remote** copy of a project along with commit history
- Tracks **branches** and **forks** of the main copy (repository)
- Provides a user interface for communicating with collaborators, tracking bugs, viewing commit history, and executing specific Git operations

[3]

Terminology

- **repository** – a folder of files constituting a project
- **remote** (*noun/adjective*) – the copy of your project stored on GitHub
- **clone** (*verb*) – copy a remote repository to your local machine
- **commit** (*noun/verb*) – a set of changes entered into the tracking system
- **push** (*verb*) – upload commits from your local work space to GitHub
- **pull** (*verb*) - download commits from remote to local
- **branch** (*noun*) – a separate working copy of the files, accumulating changes that will be merged into the main version later on
- **fork** (*noun/verb*) – a third-party copy of a repository, only loosely connected to the original
- **pull request** (*noun*) – a request to merge commits from one branch/fork to another

Configuring Git (LIVE DEMO)

- `git config --global user.name "Mona Lisa"`
- `git config --global user.email "mlisa@berkeley.edu"`
- `git config --list`

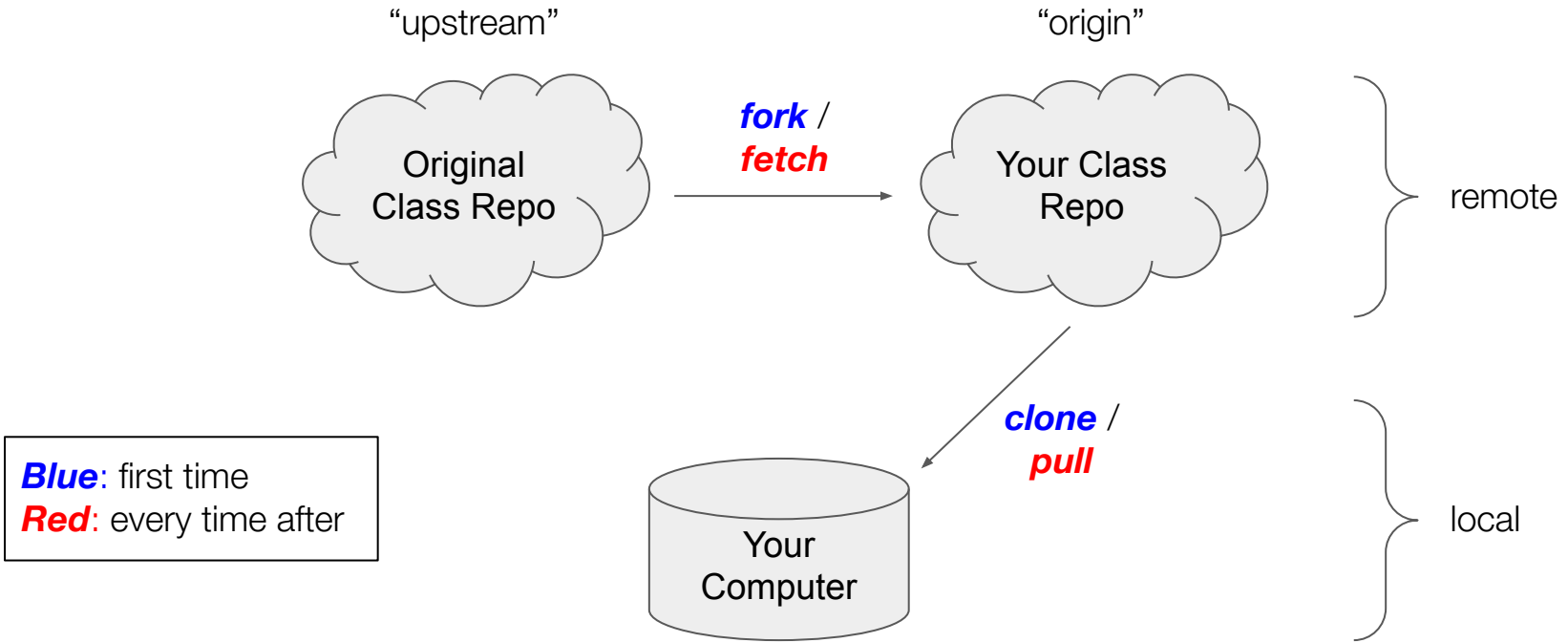
Typical Git Workflow on local

1. Do some work (make local changes)
2. Make sure you're on the right **branch**
   - `git branch` or `git checkout <branch name>`
3. Tell Git which changes you want to **commit**
   - `git add <filename>`
4. **Commit** your changes and describe them
   - `git commit -m "this is my first git commit"`
5. **Push** your local changes to remote
   - `git push`

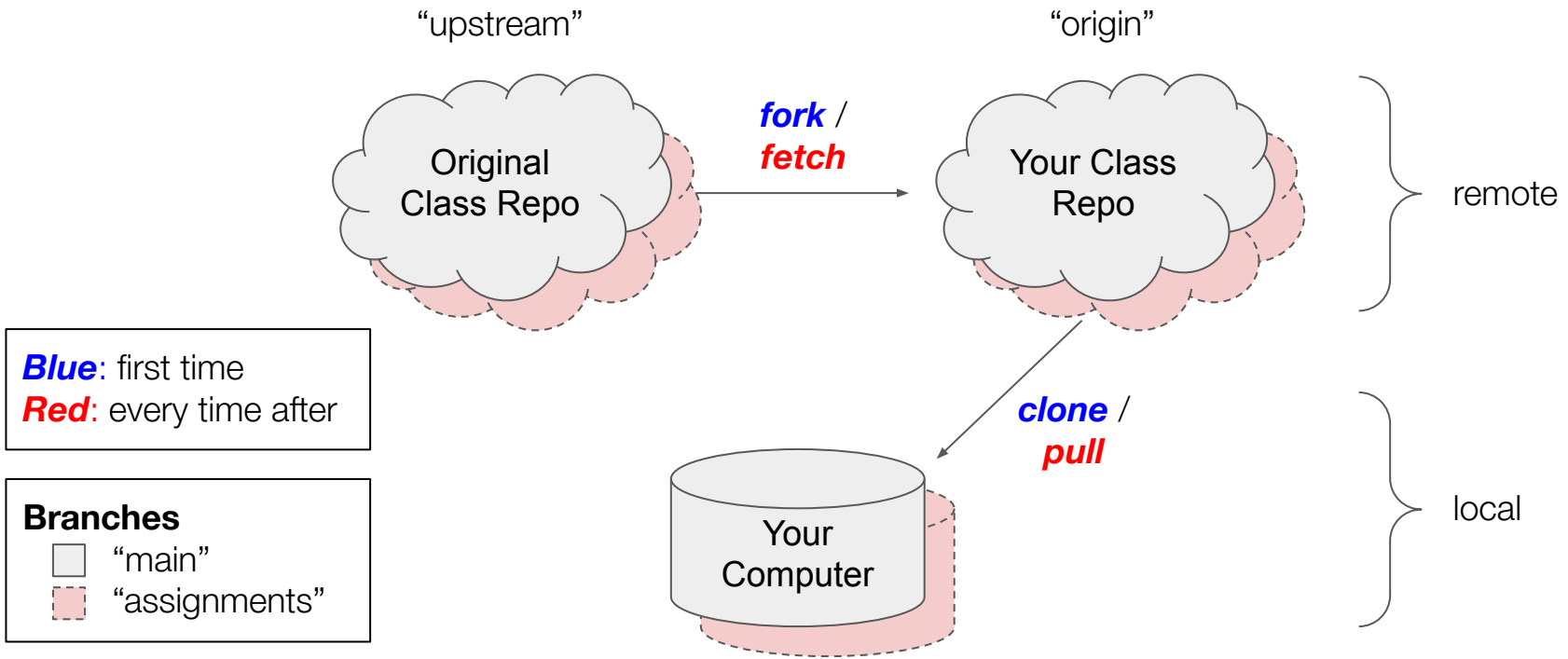
Still not sure why you're learning Git? Making a GitHub page?

- Industry standard for open source software projects
- GitHub projects as portfolio/CV/résumé
- GitHub Pages site as actual portfolio/CV/résumé

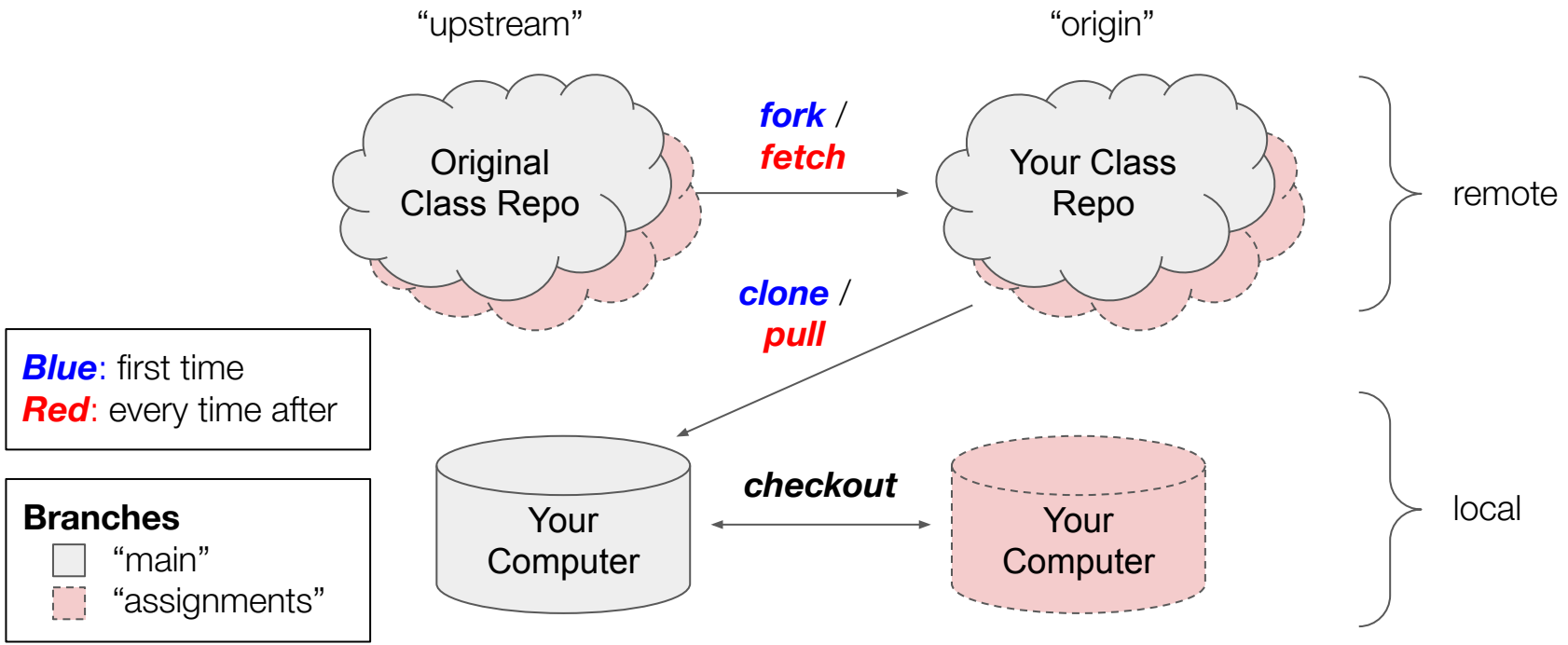
Creating (and maintaining) your class GitHub Repository



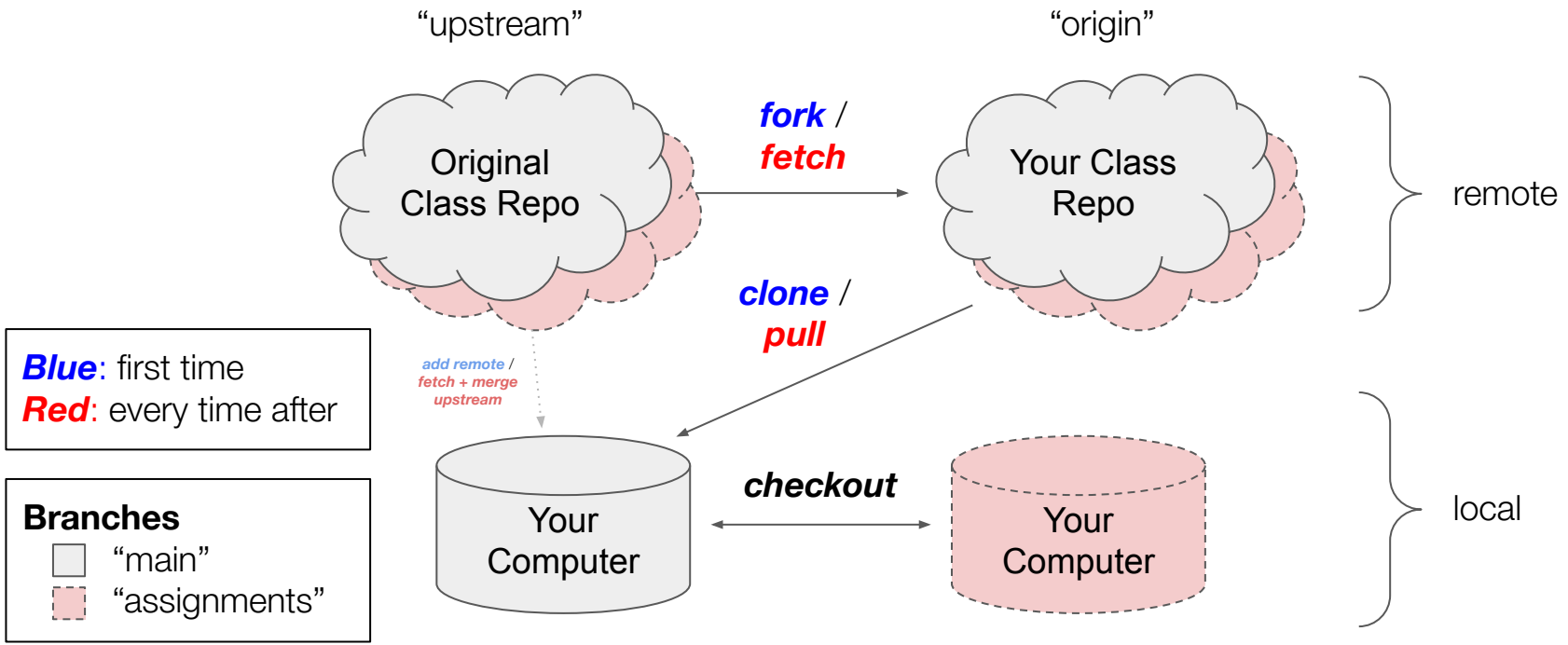
Creating (and maintaining) your class GitHub Repository



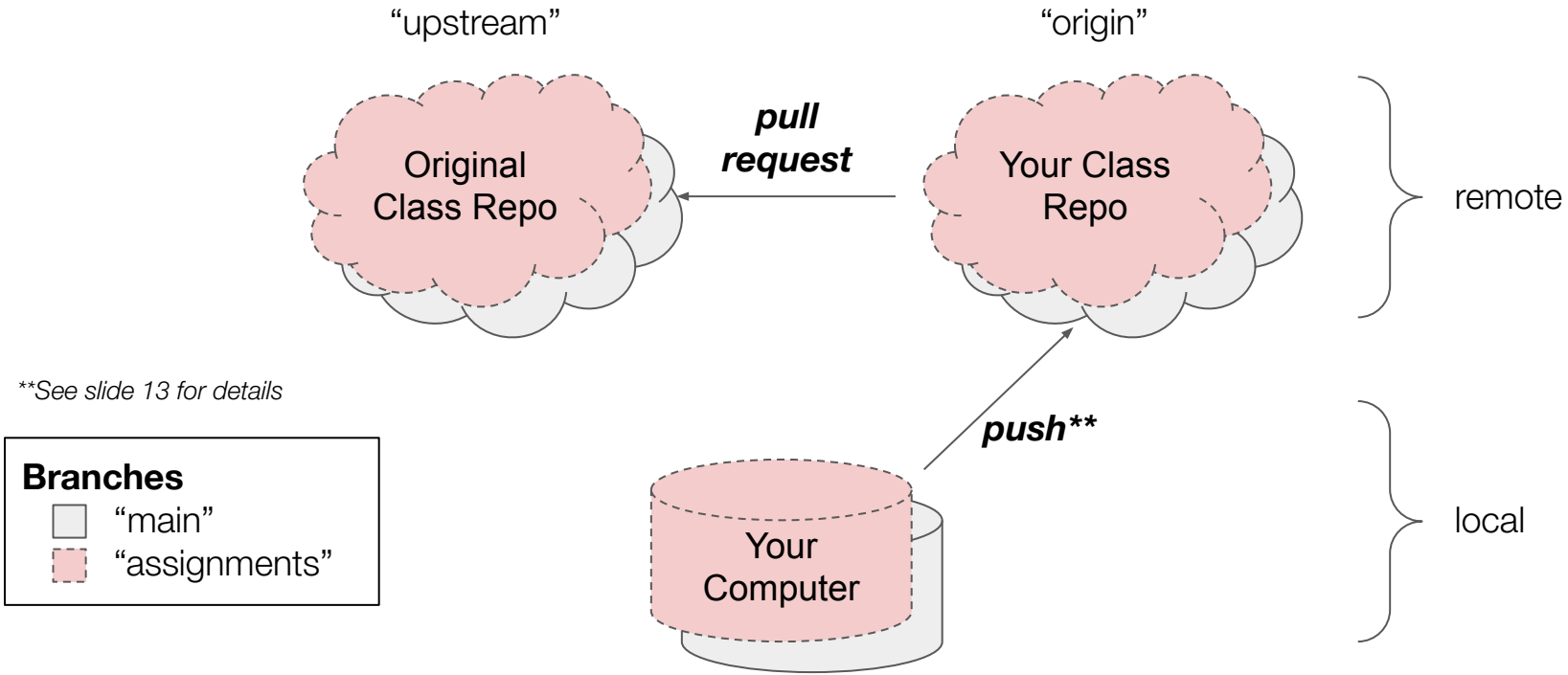
Creating (and maintaining) your class GitHub Repository



Creating (and maintaining) your class GitHub Repository



Submitting your assignments



Forking the class GitHub repo (LIVE DEMO)

- Breakout Rooms Part I
  1. Open a browser and go to https://github.com/mxndrwgrdnr/CYPLAN255
  2. Create your own fork
  3. Clone your fork
- 5 MINUTE BREAK: Instructor will create Assignment 0
- Breakout Rooms Part II
  1. Sync your fork
  2. `git pull` the changes to you local (cloned) copy
  3. `git checkout` the “assignments” branch
  4. Create/copy a new file WITH A UNIQUE NAME and make some changes to it
  5. Add, commit, and push your changes (see previous slide)
  6. Open a pull request

3. Python at the Command-line

Python vs. IPython vs. Jupyter

- Python - an **interpreted**, **high-level** programming language
- IPython - “interactive” Python interpreter
  - `In [1]:` instead of `>>>`
- Jupyter Notebooks - web-based GUI for IPython
  - .ipynb = “IPython notebook”

[5]

[4]

[6]

Options for Running Python

- `python my_script.py` — execute a Python script
- `python` — launch the default Python interpreter
  - `exit()` — exit
  - `<ctrl> + d` (Mac/Linux) — exit
  - `<ctrl> + z` (Windows) — exit
- `ipython` — launch the interactive Python interpreter
  - `exit` — exit
  - `<ctrl> + d` (Mac/Linux) — exit
  - `<ctrl> + z` (Windows) — exit
- `jupyter notebook` — launch a notebook server and dashboard
  - `<quit>` (notebook dashboard) — exit
  - `<ctrl> + c` (Mac/Linux) — exit

Managing Packages and Virtual Environments

- Anaconda – a Python **distribution**
- Conda – a Python **package manager** and **environment manager**
  - Created by the Anaconda folks
  - As a package manager
    - Installs Python libraries (packages) from package **repositories** (e.g. conda-forge)
    - Manages dependencies and resolves conflicts
    - Other examples: “pip”
  - As an environment manager
    - Manages Python **virtual environments** (sandboxes)
    - Other examples: “virtualenv”

[7]

ANACONDA®

Max’s Tips for Creating a Conda Environment

1. `conda create -n my-first-env`
2. `conda activate my-first-env`
3. `conda config --add channels conda-forge`
4. `conda config --set channel_priority strict`
5. `conda install python ipython notebook nb_conda_kernels jupyter_contrib_nbextensions`
6. `jupyter contrib nbextension install --user`

Intro to Python (LIVE DEMO)

- SLIDES ⇔ NOTEBOOK
- Options for following along:
  a. Start a Notebook server (`jupyter notebook`) and open the notebook named "lecture_03_intro_python_jupyter.ipynb"
  b. Open a static copy of the rendered notebook on the class GitHub repo here
  c. Sit back and enjoy the demo. You can (and should) explore the notebook on your own time afterwards.

4. For next time

For next time (“homework”)

1. Finish the GitHub exercise which includes:
    a. Forking the class repo
    b. Cloning your fork
    c. Syncing your fork
    d. Submitting Assignment 0
2. Work through lecture_03_intro_python_jupyter.ipynb on your own
3. Try creating a conda environment and accessing it from a notebook

5. Questions?

Image attribution

[2] https://www.nobledesktop.com/blog/what-is-git-and-why-should-you-use-it
[3] https://github.com/logos
[4] https://www.python.org/static/community_logos/python-logo-master-v3-TM.png
[5] https://upload.wikimedia.org/wikipedia/commons/3/3c/IPython_Logo.png
[6] https://commons.wikimedia.org/wiki/File:Jupyter_logo.svg
[7] https://en.wikipedia.org/wiki/Anaconda_(Python_distribution)#/media/File:Anaconda_Logo.png

Bonus Material

GitHub Authentication with SSH

1. `ssh-keygen -t ed25519 -C "your_email@example.com"`
2. “Enter file in which to save the key (/home/jovyan/.ssh/id_ed25519):”
   - `<return>`
3. “Enter passphrase (empty for no passphrase):”
   - `<return>`
4. “Enter passphrase again:”
   - `<return>`
5. Add your ssh key to your GitHub account
   - https://docs.github.com/en/authentication/connecting-to-github-with-ssh/adding-a-new-ssh-key-to-your-github-account