Python Orientation

Course setup PF0

Learning Objectives

1 What are GitHub Codespaces
2 Common errors and how to fix them
3 How to work locally
4 How to connect to GitHub through Git

Introduction

Before we begin coding, we'll get you set up with the right tools to ensure a smooth and productive workflow. Throughout this course, we make use of modern **development environments** that are widely used in both academic ar industry settings, across biological and medical sciences.

While you don't need to install anything on your machine to get started, there are options to do so, depending on how you want to work. Understanding how the setup works will help you become a more confident and independent coder as the course progresses.

GitHub & Codespaces

All your learning materials, assignments and notebooks are hosted on **GitHub**: the world's leading platform for sharing and versioning code. You'll receive an individual GitHub repository for each lesson via a GitHub Classroom invitation, sent through our Lesson Portal. These repositories are like digital folders in the cloud that store your assignments, data and lecture materials.

To complete your assignments, you'll launch a GitHub **Codespace** -- a full coding environment that runs in your browser. Codespaces use **Microsoft Visual Studio Code** behind the scenes and come pre-installed with everything you need: **Python**, **Jupyter** support, required libraries, and version control via Git. This eliminates the need to install anything locally and ensures that all students are working in an identical, error-free environment.

Codespaces also help us ensure fairness and consistency in grading. Once your work is complete, you'll submit it directly through Git's version control system -- all from inside the Codespace interface. Please refer to the **Student Handbook**, for a step-by-step tutorial on how to use Codespaces within this course.

Common errors with GitHub Codespaces

Sometimes (hopefully not often) Codespaces will generate an error when you try to *push* back your assignment. This occurs when the main repository is different to the **Codespace** repository, a merge conflict that we discussed earlier.

To fix this error you can use this workflow:

1. Open the terminal pane in your Codespace (usually cmd + J on a Mac, or ctrl + J on a Windows machine). This should toggle the Terminal pane at the bottom of the window.

2. Type in the following two commands:

```
git pull origin main
```

Note: If this doesn't work, try: git pull origin master

```
git config pull.rebase false
```

You should then be able to submit the assignment, as usual.

This information is to be used in tandem with 'how to submit' in the student handbook.

VSCode

If you prefer to work locally, you can download and install VSCode on your machine, see **Essential tools** for more information about the it and a guide to downloading. Often this route may be a requirement, if you are using an institutional computer that has certain access privileges on it.

Whether working online through Codespaces or offline in VSCode, you'll be using the same files, notebooks and Git workflows.

Git & GitHub: Version control and collaboration

You'll use Git to track and save changes to your code, and GitHub to store and share those changes online. We employ these mechanisms in assignment submission, so that you leave the course fully fluent in a professional, industry-standard way of working.

When you finish an assignment, you'll commit your changes and sync them to your repository using Git -- all of which you'll do through VSCode. Your feedback and marks will be returned through GitHub Pull Requests, and you'll learn how to revise and resubmit work by branching, committing and pushing your code.

All of this may feel unfamiliar at first, but you'll be walked through the process carefully and repeatedly during the course.

Connecting your computer to GitHub

When you want to clone a repository from GitHub it will give you two options, **HTTPS** and **SSH**. Clicking either will give you a link that you copy and paste into your terminal after `git clone`.

Each method needs a different route to authentication to allow the connection to take place. There are differing benefits to each, however this is normally only relevant for developers. Use which ever one looks easiest to you to setup.

Here is the official guide to setting up Git from GitHub themselves. This should be your primary source for setting up. Although, we will run through it quickly here too.

WARNING

Before starting on either of these setups, please make sure you have set your username and email address in Git, as set out in **Basic Git**.

Via HTTPS

If cloning a repository using HTTPS, it is recommended to use **GitHub CLI**, which is GitHUb on the command line (terminal).

GitHub CLI will automatically store you Git credentials for you, so that every time you call `git clone` or `git pull` this information is passed to GitHub, allowing you access.

To setup:

- Install GitHub CLI on your computer
- In the command line, enter `gh auth login` and follow the prompts:
  - When prompted remember to select `HTTPS` as your preferred protocol
  - Enter `Y` (yes) when asked if you want to authenticate

- This should take you through to a web portal, where you are asked to login

Once set up, you should no longer need to enter credential every time you make a request.This means any changes you make locally can be saved, tracked and pushed to GitHub with a single command. You can also pull down updates mad to your repo from another machine or location. This workflow mirrors ways of working commonly used in research, software development and collaborative coding environments.

If you run into any issues, head to GitHubs official guides and attempt alternative methods.

Via SSH

An alternative to **HTTPS** is **SSH** (Secure Shell Protocol), which is a typical method in computing for having two computers securely connect.

SSH works by creating a **pair of cryptographic keys**:

- A **private** key, which stays safely *stored on your computer*.
- A **public** key, which you *upload to your GitHub account*.

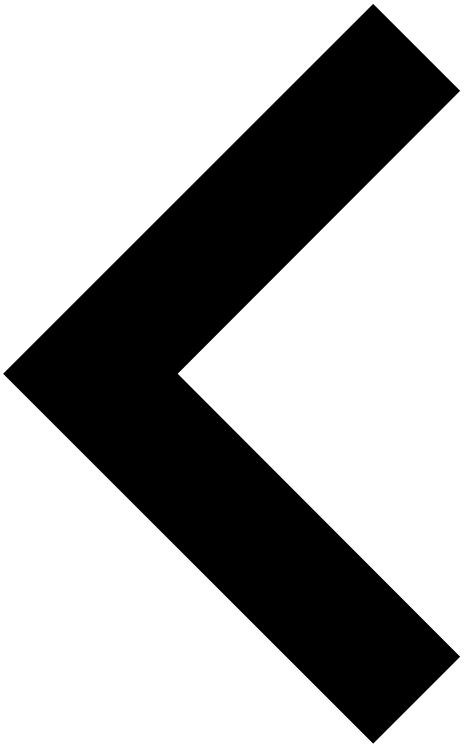
Once your SSH key is configured and linked to your GitHub account, it provides a secure **bridge** between:

- Your **local IDE** (Visual Studio Code), where you write and edit code
- Your **GitHub repository**, where your files are stored and versioned online

To setup:

- **Generate a new SSH key** and add it to your **SSH agent** (your system's secure key manager): Click here for GitHubs guide
- **Add the SSH key to your GitHub account** to authorise your device: Click here for GitHubs guide

Once this is set up, your Git commands in VS Code (including pull, commit, push) will automatically use SSH in the background to verify your identity, allowing for secure and seamless integration between your desktop code editor and GitHub.



Previous