Basic Git Workflow

Overview

Git is a system that lets you:

- Store versions of your files at key points that you want to be able to get back to
- Revert to previous versions of your files if need be
- Collaborate with other people making changes to the same files you are working on
- Share your work with the world

Your code (or other work) is kept in a **repository**. There are typically at least two copies of this repository:

1. One on a server on the internet; in this class we will keep a copy of our code on GitHub
2. One on the computer where you are editing the files; in this class I will suggest that you keep these files on Gryd.
3. If other people are also working on these files, they may have their own copies of the respository.

There are two main steps to using Git. Each has sub-steps that we'll outline below.

1. Initialize a git repository. In this class, I will set up starter repositories for you on GitHub, and you will **clone** these to the computer where you will be working on the files.
2. Ongoing updates. Make changes to files, **add** them to a staging area, **commit** them to a local repository, and **push** them to a remote repository

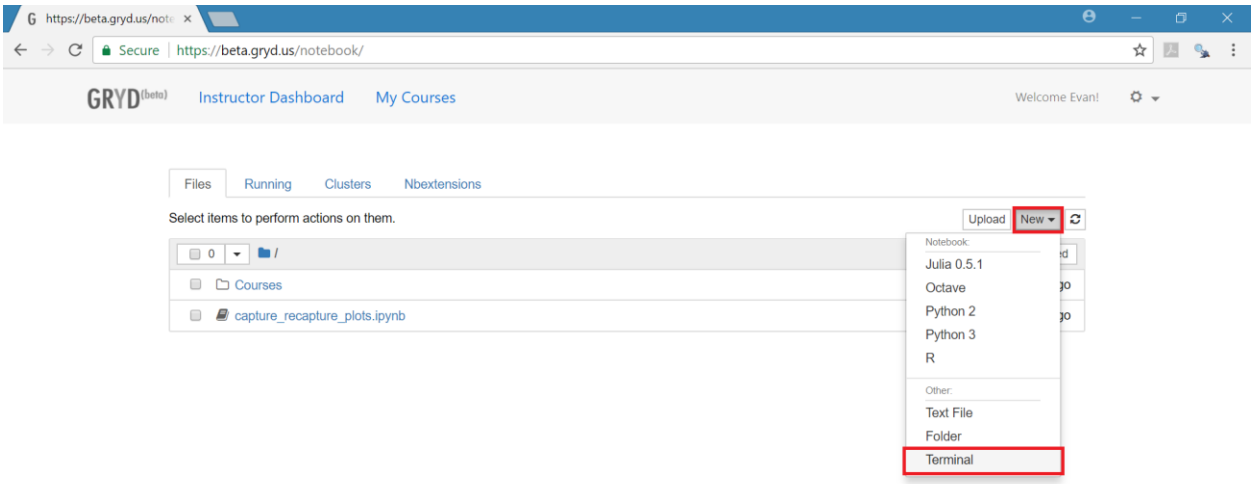
Note that these steps work if you are the only person working on a project. The workflow for collaborating on projects and sharing code via Git is more involved; we will get there later in the semester.

Cloning a Repository from GitHub

In this class, I will set up repositories with starter code on GitHub. Your first step is to **clone** that repository, which downloads it from GitHub to the computer where you will be editing the files.

To do that on Gryd, do the following:

1. Sign in to Gryd at gryd.us
2. Click on the "New" button in the top right, and then select "Terminal". These are outlined in red in the screenshot below:



3. In the terminal, navigate to a directory where you want the files to be stored (See the handout about Unix navigation).
4. Obtain the web address of the repository from GitHub.
5. Enter the following command into the terminal:

git clone <repository url>

Ongoing Updates

The graphic below illustrates the four phases involved in committing changes you make to your files to your local repository and pushing them to github:

When you create a new file or edit an existing file on your local computer (or, in our class, on Gryd), those changes are not backed up and are not available for other people to see. At this point, these files are **unstaged**.

When you have reached a point where you are ready to save some changes you have made, you will **add** them to a staging area, **commit** them to a local repository, and **push** that commit to a remote repository. You will usually want do this after you have finished a particular programming task that is more or less self-contained. These steps are outlined in more detail below:

1. To **add** a file to the staging area, in your terminal navigate to the directory where your local repository is stored. Then enter a command like the following:

git add file_name

   You can add multiple files to the staging area if you have made changes to several files that you want to save together.

2. To **commit** the files in the staging area to your local repository, enter the following:

git commit –m “description of the changes you have made”

3. To push the commits in your local repository to the remote repository, enter the following:

git push origin master

In this command, “origin” is a key word that has been configured to tell git where the remote repository is, and how to communicate it. “master” is the name of the branch where your changes have been made. A branch in git is a way to organize development of new features or code that you’re not sure you want to keep permanently yet. We will talk about branches in git later in the semester. The default is to work on the “master” branch, so for now the exact command above will work.