Data 115: GitHub Intro

Daryl DeFord

Fall 2021

1 What is GitHub?

GitHub is a web interface for the git version control and file tracking system. It is an invaluable tool for collaborative projects and has become a particularly common way to publicly share data sets. The syntax can feel a little overwhelming and complicated but we are mostly going to be using just a small collection of the available functionality this term. The main purpose is to allow multiple people to work on the same codebase without conflicting changes while making all of the code available to everyone at once.

This guide mostly explores techniques that are more advanced than we will need for our class but might serve as a helpful resource if you decide to do more.

2 Basic Usage

For our course it will probably be sufficient to work solely through the web interface of GitHub and not need to interact with git through a terminal. In week 5 of the course I showed some examples of this usage, so those videos might be helpful to review. Basically, repositories can be created, files can be uploaded, and the readme can be edited directly through the GitHub web interface on your browser - this will be enough flexibility to complete your personal dataset project.

2.1 Creating a Data Repository

Once you have created an account, you can create a new repository by clicking the repositories tab at the top of your homepage and then clicking the green 'New' button in the upper right. On the next screen, you can choose a name for your site and add a brief description if you want. Select the checkbox near the bottom of the page that says ' Add a README file' and then you can click the green 'Create repository' button at the bottom of the page and you should be good to go!

- From the homepage of your new repo, you can edit the readme (the text that shows down below the file structure) by clicking the edit pencil in the upper right.
- The readme file uses a markdown syntax similar to the Rmd syntax we have been using in class. A nice guide to the version supported by GitHub is at this link. You can also use html formatting directly in the readme if you prefer that style (I added an example to the repo we made in class: github.com/drdeford/115_git_practice)
- To upload a new file, you can click the gray 'Add file' button next to the green 'Code' button at the top of the page and either create the file directly using the text editor or choose a file on your computer to upload.

For the purposes of this class, specifically the personal data set project, that should actually be enough to do everything you need.

3 Advanced Git Workflow

As an overview of the git workflow, consider a collection of people contributing to the same project. They each want to make a contribution but are working in separate locations at separate times. Anyone who has tried to collaborate over Dropbox or even OneDrive/GoogleDocs has probably had the experience of accidentally overwriting someone else's work. Git provides a tool for managing these situations effectively.

This walkthrough provides a really useful starting point for getting a feel for GitHub.

3.1 Note for Windows users

You will probably need to install gitbash from here in order to have a working terminal to enter commands.

3.2 Cloning

To clone a repository you start by clicking the green "Clone or Download" button on the front page of the repo. This provides you a link that you should then copy to your clipboard. Navigate your command line to the directory you would like to extract the repository and enter:

```
>>> git clone [paste the text from your clipboard here]
```

3.3 Forking

Forking a repo creates a separate copy of the repository under your name that you can use to test out larger scale changes and use the pull-request system. To fork a repo you can just click the fork tab at the top of the repo mainpage. Then you can clone your fork and keep your changes separate from the main branch until you are ready to merge them into the main repo.

Once you have a fork you need to connect your version to the main version, which is usually called "upstream" in this context:

```
>>>git remote add upstream [url for main repo goes here]
```

To update your fork with the latest commits to the main repo, you can use:

```
>>> git pull upstream master
```

then to push it to the online copy of your repo

```
>>> git push origin master
```

3.4 Branches

Branches are how git keeps your edits separate from the main code until you are ready to incorporate them. You can make a branch through the web interface by clicking the branches button or you can create one through the command line with:

```
>>> git branch my_branch
>>> git checkout my_branch
```

Now you should see the branch listed to the right of your cursor or directory.

3.5 Adding Files

Once you have edited files in the directory you can tell git to track the changes you've made using:

```
>>> git add [name of the file you changed here]
```

To see that git is tracking the changes you have made you can type:

```
>>> git status
```

and see which files you have modified since your last commits.

3.6 Commits

Once we have made the changes we want to, we next need to tell github that we are ready to incorporate them into our fork. The command for this is commit which:

```
>>> commit -m ‘‘[Your message about the changes goes here!]’’
```

3.7 Pushing and Pulling

The syntax for moving files from your local copy of the repo after committing them is [push]ing and for updating your local copy with more recent updates to the online version of the repo is [pull]ing. Once you have committed changes to your branch, you can push your changes to the web interface with:

```
>>> git push origin [name of branch]
```

3.8 Pull Requests

Once you have pushed your branches changes to your copy of master, it is time to get them added to the main repository. This is easiest to do using the web interface. After pushing to origin, you should go to github.com/your-fork and click the open a pull request button. A new interface will open that allows you to add additional comments and details about the code that you are adding and then your proposed changes will show up under the pull request tab on the main repository.

At that point, an owner of the main repo can look over your changes and decide whether to add them to the main branch. Once your changes have been merged you can delete your branch with

```
>>> git branch -d [your branch name]
```

4 Pull Requests

4.1 Introduction

The next section walks you through cloning and committing for the first time, while the remainder of this guide provides the steps of making your first pull request into a repo.

4.2 Cloning

1. Start by signing in to your GitHub account and going to the cloning practice repo.
2. Click the green “Clone or Download” button and copy the text to your clipboard.
3. Open your terminal[1] and navigate to the directory you want to use.
4. Type:

   ```
   >>> git clone https://github.com/drdeford/115_git_practice.git
   ```

5. This will make a new directory called Cloning Practice on your computer.
6. Go into the directory and make a new directory called [your name]
7. Inside that directory use your favorite text editor to create a txt file called init.txt
8. In your terminal type:

[1] or git bash, for Windows users

```
>>> cd Cloning_Practice
>>> git add [your name]
>>> git commit -m "Added my directory"
>>> git push origin master
```

9. If the last line returns an error[2], saying:

   ```
   error: failed to push some refs to 'https://github.com/drdeford/115_git_practice.git'
   ```

   type:

   ```
   >>> git pull origin master
   >>> git push origin master
   ```

   and close the text file that open in the middle.

10. Congratulations!!! You made your first commit.

4.3 Pull Request Walkthrough

1. Start by signing in to your GitHub account and going to the PR practice repo.
2. Click the fork button at the top right of the page and wait until the interface creates your new
3. After the process completes, click the green "Clone or Download" button and copy the text to your clipboard.
4. Open your terminal and navigate to the directory you want to use.
5. Type:

   ```
   >>> git clone [pasted text here]
   >>> cd 115_git_practice
   >>> git branch add_name
   >>> git checkout add_name
   ```

6. The previous commands placed a copy of the repo on your computer, openend a new branch for you to make edits and checked out that branch
7. In the new directory there is a text file called Students/favorite_icecream.txt. Open the file and add your name in the row next to your state.
8. In the terminal type:

   ```
   >>> git add favorite_icecream.txt
   >>> git commit -m "Added my flavor!"
   ```

9. This tells git to track your changes to the file and prepares them to be merged with the main version of the repo
10. Type:

    ```
    >>> git push origin add_name
    ```

11. Go back to your browser tab with your fork, where you copied the clone text. Click the "Compare & Pull Request" button

[2] which it probably will, since everyone is trying this at once

12. You can leave additional comments in the text field and then click the green create pull request button.

13. Have someone sitting next to you click the merge pull request button on the vrdi copy of the repo.

14. Once they are done click the green "Clone or Download" button on the vrdi page and copy the text to your clipboard.

15. Type:

```
>>> git checkout master
>>> git remote add upstream https://github.com/drdeford/115_git_practice.git
>>> git pull upstream master
>>> git push origin master
>>> git branch -d add_name
```

16. This tells your computer about my version of the repo and updates your local and online copies with the new version of the .txt file.

17. From now on, to update your local version to match the official version you can use:

```
>>> git pull upstream master
>>> git push origin master
```

18. Congratulations!!! You made your first pull request.