Introduction to Git and Github: Tutorial 4 Miscellaneous features of git and GitHub

1 Back to Tutorial 1

2 Back to Tutorial 2

3 Back to Tutorial 3

4 Introduction

4.1 Untracking files

The `reset` command will remove the file from the repository, but it won't delete the file from your system.

```
> git reset plants.txt
```

4.2 Avoid adding specific files to a repository

Let's say that you don't want to track the changes to specific files in your repository. For instance, you might have personal information that you don't want to make public.

Add a text file to the top directory called `.gitignore` (note the dot at the start, no '.txt' at the end), and add the files you want to avoid adding on a seperate line. e.g.:

```
data/participantNames.csv
consentForms/*.pdf
```

This applies to the file "data/participantNames.csv" and all files in the "consentForms" folder ending in ".pdf". These won't be added to the repository.

But be careful! This will only stop files being **added**. If they are already in the repository, this won't remove them, so you need to add a gitigore file before your first commit.

If you've added files to a repository, but you shouldn't have, there are ways of removing them. But the safest and simplest thing to do is to copy the files to another location, delete the old repository and then make a new repository with a .gitignore file.

4.3 Forking GitHub repositories and pull requests

If you aren't a collaborator on a GitHub project, you can still contribute to it by forking the repository, making changes, then making a pull request.

First you "fork" the existing repository. This is actually just making a branch of the repository on the GitHub server. Clone this fork to your local machine with `git clone`. You can then make changes, make a commit and push the commit to your GitHub fork. Then you synch your repository, and make a "pull request". This sends a reqest to the owner of the project for them to merge your fork/branch with the master branch of the project.

There are more details here

4.4 Caching github password

You can store your password secuerly on your machine to avoid having to type it in at every puhs. There are a number of ways of doing this, see https://help.github.com/articles/caching-your-github-password-in-git/.

4.5 Viewing GitHub content directly

You can use services like htmlpreview to view html files in a repository. e.g.:

https://htmlpreview.github.io/?https://github.com/seannyD/SeansGitHubTutorial-Collaboration/blob/master/results/MainResults.html

4.6 GitHub licences

By default, GitHub projects do not specify a license. When you create a GitHub repository, there's an option to add a license from a list of candidates. All that this does is add a file called LICENCE.txt into your project with the legal terms of the licence. The suggested options limit people's ability to take and modify your work. A better option for freer distribution which allows people to copy and modify as long as they attribute you is the "GNU AGPLv3" (GNU Affero General Public License v3.0) see here fore more details. Note that most Creative Commons licences are not suitable for software. More options can be found on this site.

4.7 fetch versus pull

The command `git pull` actually does two things: it downloads stuff from GitHub, then performs a `git merge`. If you just want to download stuff, use `git fetch`.

4.8 Avoid tracking mac .DS_Store files

.DS_Store files are files that macs use to keep track of how you view folders. They're not really important to a project, but can cause conflicts because different users will modify the files in different ways. There's one of these files in every folder, so a simple line in .gitignore won't work. You can tell git to always ignore .DS_Store files in all repositories:

```
> echo .DS_Store >> ~/.gitignore_global
> git config --global core.excludesfile ~/.gitignore_global
```

Note that this is not retroactive. To remove .DS_Store files from an existing repository, you can use the following line. But be careful - this uses `git rm` which can delete files from your system.

```
> find . -name .DS_Store -print0 | xargs -0 git rm --ignore-unmatch
```