Git and GitHub Notes

Fall 2022

These notes are written for using Git and GitHub on Linux Mint.

Token: `ghp_E5a4GDlk9bTBzrGPawUzpCafUr7RlD1YJEj0` good until November 3ish, 2022

RESOURCES

- Notes taken from Kevin Stravert: https://www.youtube.com/watch?v=tRZGeaHPoaw
- Official Git web site: https://www.git-scm.com/
- Official GitHub.com web site: https://github.com/
- Git and GitHub.com cheat sheet: https://education.github.com/git-cheat-sheet-education.pdf
- Git Reference Manual: https://git-scm.com/docs
- Git Overview Book: https://git-scm.com/book/en/v2
- Sample ignore files: https://github.com/github/gitignore
- Hyper.is Terminal: https://hyper.is/

1 Get Git

Determine if you have Git installed. Type `git --version` in terminal and press Enter.

If `Command 'git' is not found`, install with the following:

```
sudo apt install git
```

2 Configure Git

Specify your name:

```
git config --global user.name "James Morgan"
```

Specify your email address:

```
git config --global user.email jamorgan75@protonmail.com
```

Set default branch name to "main":

```
git config --global init.default branch main
```

3 Get help

Ask git for help:

```
git config -h
```

If you want more detailed information,

```
git help config
```

This follows the format `git help <command>`

4 Initialize Repository

Within terminal, change directory to the folder for which you wish to create a repository. Type the following and the press Enter:

```
git init
```

This creates a hidden folder `.git` within the directory that contains the repository files.

5 Git status

We can find the status of our repository with the following command:

```
git status
```

This command will return the following information:

- branch name
- commits
- tracked (green) or untracked (red) files

6 Track and untrack files

To track a file:

```
git add index.htm
```

To untrack (unstage) a file:

```
git rm --cached index.htm
git add notes_on_git_and_github.tex
```

To untrack (unstage) a file:

```
git rm --cached notes_on_git_and_github.tex
```

7 Ignore files with `.gitignore`

Create a `.txt` file in the directory and add the filetype. Comment with the hashtag symbol.

```
# ignore all .txt files
*.txt
```

Ignoring files is useful if you have files with sensitive information within the directory.

To see a comprehensive list of all the ways to ignore files, visit https://github.com/github/gitignore

8 Track all files / add to staging

To track all files within the directory, use one of the following:

```
git add --all
git add -A
git add .
```

9 Commit

To commit is to take a snap shot of what your repository looks like at this point in time.

```
git commit -m "first commit - committing all fules to the repository"
```

10 Change files and view differences

If we change and save a file, the file is now categorized as “modified.”

We can compare the differences in the modified file with the following command:

```
git diff
```

The original text will be in red, and the changed text will be in green.

We can now use the add command, and the file will be sitting in “staging”.

Within Git, we have three different environments: working files, staging, and commit.

WORKING FILES	Files are edited here
STAGING	Files are held to be committed
COMMIT	A record of change is created for a file

To remove files from staging:

```
git restore --staged index.htm
```

If we commit now, the `index.htm` file will not be included as it is in the working environment.

11 Bypass staging and commit

The following skips the staging steps:

```
git commit -a -m "updated text to free range"
```

12 Delete / remove files

```
git rm "secret recipe.htm"
```

If Git is tracking the deleted file, status will show that the file has been deleted.

13 Restore files

```
git restore "secret recipe.htm"
```

This recovers the file.

14 Rename files

```
git mv "KCC Logo.png" "Primary Logo.png"
git commit -m "chagnnnged the file name of an image"
```

15 View commit history with git log

```
git log
```

To get an abbreviated view:

```
git log --oneline
```

16 Amend commit

```
git commit -m "changed file name to Primary Logo.png" --amend
```

17 View changes in commits

```
git log -p
```

To find the different capabilities of `git log` is to use the help feature:

```
git help log
```

18 Reset to previous commit

To jump back to a previous commit:

```
git reset c193894
```

19 Rebase git repository

To modify your commits:

```
git rebase -i --root
```

To exit this menu, press :X and then Enter.

20 Branches

To create a new branch, from the directory we enter:

```
git branch FixTemp
```

To see branches, type:

```
git branch
```

We are currently in the branch with the asterisk.

To switch into another branch, we type:

```
git switch FixTemp
```

Now any changes made to the files in the directory will be associated with the current branch. Our commits will be applied to that branch.

Switching back to main,

```
git switch main
```

our files in the directory no reflect the changes made in the FixTemp branch. To bring the changes to the main branch, we need to merge the changes in.

21 Merge branches

We will need to specify which branch we want to merge with main.

```
git merge -m "Merge fixtemp back to main" FixTemp
```

22 Delete branch

Now that we've merged the FixTemp branch with main, we no longer need the FixTemp branch.

```
git branch -d FixTemp
```

23 Merge conflicts

If changes in main are committed before a branch is merged, we will have a merge conflict.

The following line will create a new branch and switch to that branch:

```
git switch -c UpdateText
```

24 Typical Git flow

You have some feature or bug to work on, so you create a new branch.

1. Create new branch
2. Make all changes
3. Merge into main
4. Delete branch that you were working on

25 Set up GitHub account

Your GitHub account can be created at https://github.com/

26 Create new cloud repository

On the left-hand side of the GitHub web page, you should see the option to create a new repository (a.k.a. repo). Alternatively, you can visit github.new, and this will drop you on the new repo page.

Enter a name for the repo. If you make the repo private, you can assign different individuals access to the repo.

27 Push local repo to GitHub

28 Working with files

29 Edit repo details

30 Issues

31 Pull requests

32 Actions, Projects, Wiki, Security, Insights, Settings

33 Releases

34 Fetch and pull

35 Wrap up