**Git Notes**

**What is Git**

* Removes the need to copy files to and from the class share
* Is like using a camera to take a snapshot of your files at a specific point in time that you can magically go back to if terrible things happen
* Checkpoint for your files
* Was built for code collaboration
* Secures in the knowledge that you can not ruin your work *too badly*
* Git is a collaboration tool that allows different people to work
* Git is a tool that protects yourself and others from yourself and others

**The Local Workflow**

* Tell Git to watch this folder by initializing git init
* Files will exist in three states
  + Modified - Files that are new or have changes not yet saved by GIT
  + Staged - The current version of a file
  + Commit - files that are safely stored in Git
* Use git status to see the status
* Use git add to add the file, neatly packs a copy of the specific file changes into a box
* Use git commit -m “description” adds a “checkpoint” moves a copy not the original file
* Use git log to see your changes and it includes the date and time and importantly, the description

Outcomes

* Set up a Remote repository
* Learn how to push our local files to the remote server
* Understand Branches and how to use them
* Understand how to Merge different branches
* Learn what a Merge Conflict is
* A remote repository is a copy of our project that is stored in the cloud
* It is where we backup our work and share it with others
* It is accessible where there is an internet connection

Set up our local repository

* Locate your project in File Explorer
* Initialize your local repository
* Add Commit your existing files
  + Git add.
  + Git commit -m “initial commit”
* Tell git about the remote repository (once per computer):
  + Remote add origin https://<YOUR URL>.git
* Push your changes to the remote server (as needed):
  + Git push -u origin master
  + After the first push it can simply be: git push
* Git push tells git to upload all your changes to the server. It does not need to be done after every commit, because it will upload all commits since last push.

Branches

* Branches allow us to work on code fixes and features without breaking what we already have (presumably) working.
* Fixes and new features should always start on a branch.
* The master branch is the “trunk” of your code tree and should only contain clean code ready for deployment (use on the web).
* Git branch <name> tells git to maintain a new copy of our code with the given name.
* Git branch on its own will list the branches available and display an asterisk next to the one we are currently working on.
* Git checkout <branch> tells git to switch our working folder to the branch name specified.
* Git merge <branch> combines the file changes in branch we name into our current working branch

Merge Conflict

* A merge conflict is when a file has changed in both of the branches you are trying to combine.

Questions

Today's lesson showed us how github works with collaboration. It seems to me that github will be necessary in our future projects. Github makes it very easy to share our code with other people using the command prompt. This will be very important for the next collaboration projects.

On a scale of 1 to 4 I’s say i'm at a 3

I do not have any questions about today's lesson

The best part of my thanksgiving break was being able to just relax and have to worry about anything for a whole week.