Week 1 Day 1 Notes:

• Welcome

• Syllabus Review

• Understanding GitHub and GitHub Classroom

* Link creates a root directory and then gives Matt access to it at the same time
* Adding on just what 2910 works on
* This is what you submit to employers, a GitHub link
* Instead of having a media to store code
  + Using GitHub which is made for multiple people working on code
  + Conflict resolution
  + Branching
    - Doesn’t screw up main environment
    - Takes down test environment not main
* Thumb drive has pieces of code that need to be worked on
  + Everyone works on something different
  + How do you get that to other people and how do you synch it all up?
  + Send email to person who sent out code
* GitHub is a cloud file system
* Git is how to access that file system
* VCS
  + Git is an implementation of VCS
  + All downloaded a new something for our devices and it implements a version control
  + Way to manage what release is
  + The number like iOS 13.432 has meaning
  + Track changes over time
    - Recall previous versions
      * When pushes and reverts back
      * Get it back to when it was ok
      * Fix what is wrong, push back up, and try again
* What is git?
  + Cross-platform
  + Almost all companies use git for version control
  + Ma and Pa shops may not implement this, but main companies are using it
* Git is a Distributed Version Control
  + Daisy chain computers together and then some bit of codes would be on every computer
  + Distribute code across many machines
  + Control allowed no single point failure
    - No longer the thumb drive that can crash and burn
    - Project would be killed
* Repositories
  + Used to organize a single “Project”
    - Can use it for none project based thing
    - Has a repo that controls all his code
    - Can contain many things
      * Uses repos for his PhD
      * Everything can be stored in it
    - Nothing special about code, glorified secretarial units
      * Know how to put it in Java format or C# format
      * Organize it in a certain way
    - Nothing more than 1s and 0s in certain way to organize something
* Branching
  + Top line is base line, Main.
    - Everyone baseline is kind of different
    - If start a new project
      * Get all the basics before logical dev
      * Get it to where you can start coding
      * Before working on the code
        + Should baseline the project
        + Got to get it to where you want to start working
        + Then start making small little changes at a time
        + Doesn’t work? Revert and keep going
  + We should keep making revisions until it fits what we wanted
    - Code is so easy to change
    - Branching lets you do this
  + Left to right is time for visual
    - Go and grab a copy of the mask
    - The box is the baseline
      * Do whatever you want to said mask
      * Everyone grabs a copy of the mask
        + Baseline never gets changed
        + Everyone just gets a copy of it
    - As we go left to right
      * They are working on the code
      * They are in the blue series of working on the code
* Next Branching
  + In the branch, the box never changes
    - Box can represent anything, amazon, etc.
    - Breaks something and realizes it sucks
      * Can just delete it as if nothing had happened
      * Throwing away the sucky idea doesn’t affect the main box
    - You can branch off a branch where you like your idea a lot
      * Baseline it for yourself and continue working
      * See a solution and then an enhanced solution
* Main(~~master~~)
  + Would have 2 baselines
    - Used to be Master and Slave
    - Then moved to Master and Dev
    - Now moved to Main and Dev branch
  + Need to make the change in a company that still using the master branch
  + There is only a main branch when first making a repo
  + Want Main branch to have the least amount of pull and request as much as possible
    - Dev will be touched all the time by many things
    - Bossman owns the main branch
    - Project leads will push to dev branch and deconflict with the other leads
      * Once fully pushed, bossman tells everyone to break it
      * Then it goes live once its good
      * Then they rebranch the Dev branch from the main branch
  + Seems hard in the concept but Git has tools to help us
* Commits
  + When you commit to branch you do not get pulled in
  + Just commit the code
    - When committing you don’t work on the code again
    - Gives it to team lead and then submit a code request
      * Then commits to the branch team lead pulled from
  + Change logs are from commit messages
    - Clear Concise commit messages
    - WHAT DID YOU DO??
    - If you can’t do that then you are doing something wrong
* Pull Request
  + Hierarchy of who can create and do pull requests
    - Someone needs to be in charge
    - Person in charge gets fired lol
  + Put protections in place to make sure
    - Code Reviews
    - Review committees
    - Requesting your code gets pulled onto the branch
* Merge and Deploy
  + I have 3 changes on the same line
    - How do I keep it?
    - Who dies?
    - Can keep 1 or 2 or rewrite type thing
  + We keep the one everyone decided on
* GitHub
  + Everything above was the Git Process
  + Implementation of git
  + Cloud based version of git
* Visualization
  + Forking = Branching
  + Local Machine at the body
  + LM said get me a copy of that branch
  + Pull/fetch me a copy of that