# GitHub README

Author: Nathan Iltis

Date: 10.10.19

Please read entire document straight through regardless of previous GitHub experience. Watch the youtube video and look at the GitHub command cheatsheet (steps 2,3) if you have never used GitHub before. If you have never used the terminal command line on your computer for anything of substance, please have someone walk you through using that as it will be much faster than reading about it online.

1. General - account and naming scheme
   1. I have had Laken Allen create a git\_alias@beyondefficiency.us alias email account that forwards all messages from our github.com account to [Peter Grant and Nathan Iltis] or anyone else interested or who feels a need to receive these account updates. In this way the account is completely general and does not need to be managed by any one person.
   2. Each project will have a repository with the words git\_repo in it and some descriptive project text that will keep every project name unambiguous and relatively short. They will all end with git\_repo. This is the folder name that will be seen in our account on GitHub.com, in our team dropbox folder, and on your personal machine in a (preferably cloud-synced) place of your choosing. It is necessary to keep the folder name in these three places identical because that is what the git software uses as identification.
   3. The point of using GitHub is that we keep track of when, why, how, where changes were made, especially changes to computer code. This requires that everyone remembers to “push” their changes to the team GitHub at the end of every working session and subsequently “pull” those changes back down to our dropbox project folder. It is necessary to set it up this way because otherwise there would be a constant syncing conflict between GitHub and dropbox. See the mind map below for a visual representation for how this is set up.
   4. You have to check the online GitHub account or the dropbox folder to find out what the repository name is for your project. Thy all end in git\_repo which stands for GitHub repository (a repository is basically just a fancy name for a directory)
   5. GitHub Account details
      1. Username: beyondefficiency
      2. email - [git\_alias@beyondefficiency.us](mailto:git_alias@beyondefficiency.us)
      3. password - beyeFF09
2. Watch video <https://www.google.com/search?q=how+to+use+github&rlz=1C5CHFA_enUS867US867&oq=how+to+use+github&aqs=chrome..69i57j0l5.2534j1j7&sourceid=chrome&ie=UTF-8#kpvalbx=_IMyXXYyPEsfz-gS64YSoCg31>
3. Use command cheat sheet: <https://github.github.com/training-kit/downloads/github-git-cheat-sheet.pdf>
4. Some essential points: Each of the essential commands below starts with “git”, which is the software being used to perform all of the operations. You do not need to install it.
   1. The whole point of using GitHub is to keep track of file changes over time so that collaborators can understand what has been done and ensure no repeat work or conflicting simultaneous work exists. Two collaborators can work on code simultaneously and GitHub is smart enough to sync both of their changes to GitHub without any error messages appearing as long as they were working on significantly different portions of the file.
   2. Like any collaboration software, there is no way for it to work perfectly because humans are involved. There may be times when file conflicts have to be reconciled. Git has a way of annotating this within code scripts to point out where manual reconciliation must be done. Also, this is the purposes of branches on GitHub - they track the work when multiple people have “branched off” and taken the work in their own directions. Branches can be merged to combine all work back into one main branch - the upstream/master branch. We should basically avoid branching but if it does happen then there may be some administrative intervention that needs to happen to repair the repository. If you don’t know what you are doing, don’t merge branches.
5. Essential commands (for your first time):
   1. Copy the clone URL for the project from github.com. Go to desired desktop directory and use "git clone [copied URL]" to put the git repository there - this will create a new folder with the exact name of the repository in the directory you are currently in. This will put all of the project files there.
   2. Use “git status” to understand whether there are changes to be committed or files to add or other discrepancies between your local repository and the shared online repository.
   3. Use “ git commit -m [descriptive message] to record version history as you go. A “commit” stages the changes and is basically a checkpoint you create during your work that records what changes were made and any other details you feel are pertinent. Commits should be made frequently / as necessary. A good guideline is a once an hour and more if you are working on something simultaneously.
   4. You should also frequently use “git push” to push your work to the online shared git repository location so that others can “pull” the changes you have made. Before you use “git push”, all changes are just saved on your local machine (git knows which files are in git repositories and is constantly tracking changes locally)
   5. If you have created new files (probably using your code, you can double check what new files there are using “git status”, and then use “git add [file\_name]” to add certain files to the staged commit. Use “git add -A” to add all new files.
6. Essential Commands for every other time:
   1. After any repository is on your local machine - you should start any working session with a “git pull” to pull all of the changes others have made onto your machine before moving forward.
   2. After you have used “git push” to move all your changes to the shared git repo, cd to our shared dropbox repo and “git pull” the online repo into that to keep it current.
   3. NEVER CHANGE IMPORTANT FILES WITIHIN THE DROPBOX VERSION OF THE GIT REPOSITORY. YOU SHOULD DO SUBSTATNTIVE WORK IN YOUR LOCAL LOCATION AND THEN TRANSFER THOSE CHANGES TO THE DROPBOX LOCATION.