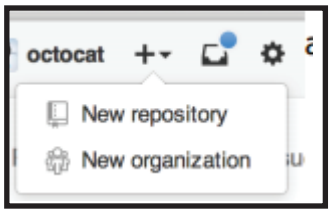
GITHUB CHEAT SHEET

Getting Started!

Github is the software program used to record changes in your code.

Windows: https://windows.github.com
Mac: https://mac.github.com

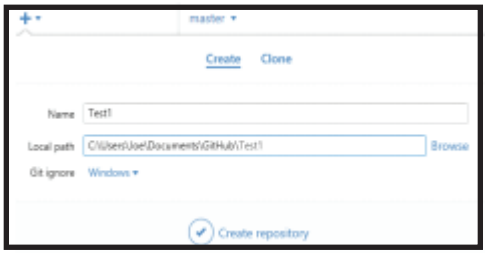
Creating a New Repository:



A repository is a folder that stores all of your code for a project. Go to github.com and click the "+" to create a new repository. Add a ".gitignore" based on the language you're programming in. Add a License.

Pro Tip: The MIT Liscense is a good default!

Adding them to Github GUI:



In the GitHub GUI, make a copy of the repository on your computer by clicking "clone".

Pro Tip: Add your repository to the default directory called "Github", usually located in the Documents folder.

Commiting!

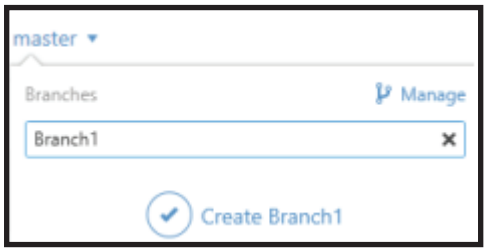
Each version of your code is stored as a commit in GitHub. Any time you make a change, you should commit to GitHub. When commiting, add a short but meaning ful description of the changes you made.

Syncing:

After every commit, you should sync so that the updates can be accessed from other computers.

Pro Tip: ALWAYS sync when things are going well, and NEVER if experiencing issues.

Branches!



Branches are different "folders" within your repository. Each one stores specific code. Your default is the "Master"

Pro Tip: An example of different branches includes one for Teleop, and another for Autonomous.

Multiple Branches:

Different files will live in different branches, depending on use. When working with many branches:

-ALWAYS commit before switching branches
-Do NOT try to access fi les that aren't located the branch you're currently in.
-NEVER have more that one person working in the same branch at once.

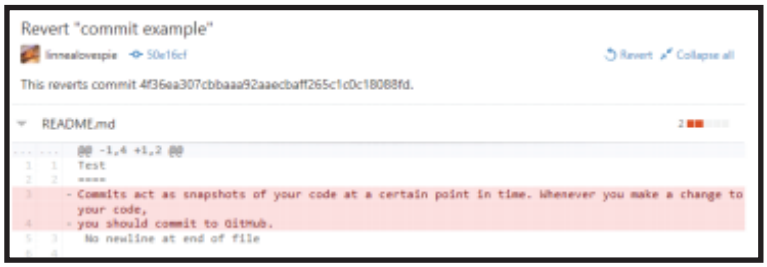
Merging Branches:

Whenever you make significant progress within a certain branch, you will want to "merge" it to Master, where all of your most current working code will be.

In the branch selector, click "Manage" to merge the two branches together.

Disaster Recovery!

If something goes wrong when committing and one needs to undo the immediate changes, revert back to the previous commit.



Pro Tip: The Git Shell can be used to "roll back" to even older commits.

If its REALLY bad...

If reverting the commit won't solve the problem, delete the local repository in the directory and reclone it using the GitHub GUI.

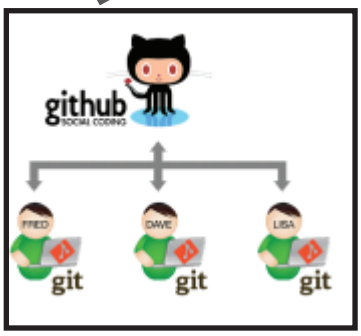
Common Destructive Scenarios:

-"Backing up" syncs, when one person forgets to sync their commits and another tries to sync changes from a separate computer

-"Merge confl icts," when one file is edited in two different branches.

.

Git vs. GitHub



GitHub is the website where one can interact with a repository. Git is the actual software that stores versions of code, and what is used on one's computer when commiting

Using the Git Shell!

While using the GUI has its benefits, the shell can provide advanced options in manipulating the repository.To change the default shell, go to Settings > Options.

Pro Tip: GitBash is recommended due to its compatibility and ease of use.

Basic Git Commands

$ cd
Change your current directory to the folder with your repository.
$ git status
Show the status of your local repository- if you need to commit, and if it needs to be pushed to the server.
$ git log
Shows the previous commits in the branch.
$ git checkout <branch>
Switch to specified branch.
$ git pull --rebase
Gets updates from GitHub.
$ git commit -a
Opens a text file to log commit messages
$ git push origin <branch>
The Git equivalent of "sync", sends your local updates to the server.
$ git revert --no-commit <head>
Revert to a previous commit, specified by the commit's "head."