

Suggested Items

- **A laptop with the following:**
 - Github desktop
 - Vscode
 - Git-Bash (Advanced)

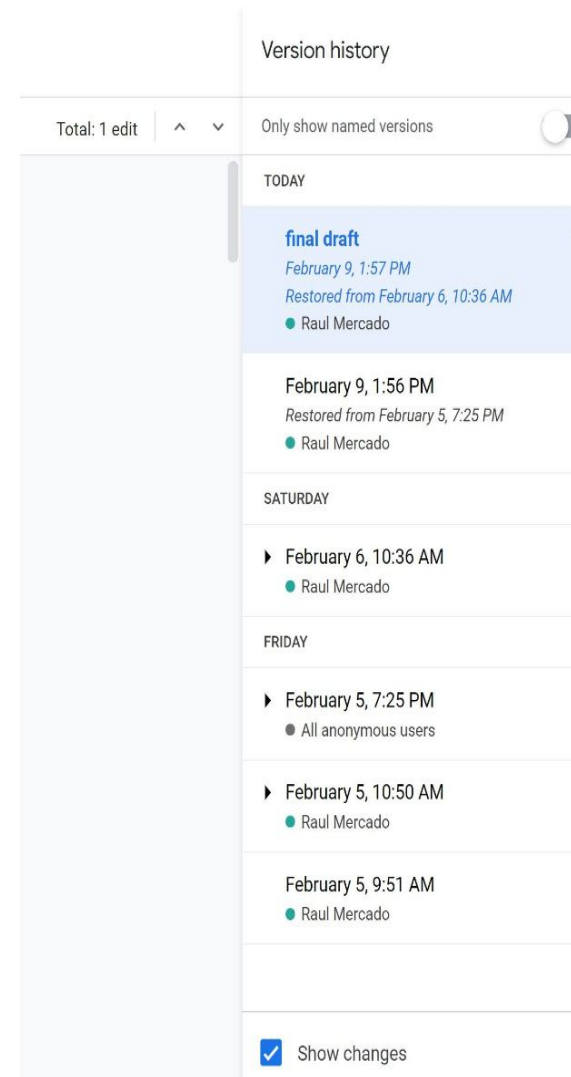
How to *Git* Going with *GitHub*



Presenter:
Ace Hathaway &
Sophia Seidman

What is Version Control Software?

- Designed to control the different updates or versions of a project
- Main goal: to keep clear points at which the project was changed
- Common well known example of a Version Control Software is Google Docs “Version History” →



What is Git?

- A free and open source Version Control System (VCS)
- Created by Linus Torvalds (creator of the Linux kernel)
- Facilitates seamless collaboration on software projects
- Unlike traditional Version Control Systems, Git allows for completely offline functionality
- You can make local changes as well as have completely local repositories
- Well liked by software developers for its simplicity, and efficiency

<https://initialcommit.com/blog/How-Did-Git-Get-Its-Name>



Basic Git Concepts

- Repository: A collection of branches and commits within a folder
- Commit: A snapshot of changes made in a repository at a point in time
- Branches: Different ‘versions’ of the same code
 - Usually used to create a different workspace to write new features
- Local Repository: A term used to describe if a version of the repository is on only your computer, or onto the main branch.
- Pull request: A request to merge to branches of code together.

More on these later!



Interfacing with Git

- Git is primarily a command-line controlled system. Multiple Graphical User Interfaces exist for Git, but they all just wrap around the basic Git interface
- Knowing how to use the underlying Git system is important in the event of breaking changes, as most GUIs do not expose the more dangerous actions of Git.
- We can utilize git by using a tool known as Git-Bash, which is a version of the Bash terminal from Linux with the git extension.



Basics Git Commands

git add

Adds files to be tracked by Git

Add specific file: git add <filename>

Add all files: git add *

git commit

Take a snapshot of all files tracked by Git, and 'commits' them

git commit -m "Commit Message"

Basics Git Commands

git push

Pushes local commits to a remote server/repository

```
git push origin <branch>
```

git checkout

Switch to a different branch, or create a new one

Create a branch: `git checkout -b <new_branch>`

Switch to existing branch: `git checkout <existing_branch>`

Delete a branch: `git checkout -b <existing_branch>`



Basics Git Commands

git pull

Pulls commits from a remote server/repository into your local repo

git pull

git stash

“Stashes” incomplete changes away

Store files: git stash

Restore latest stash: git stash pop

View stashes: git stash list

Discard latest stash: git stash drop

Basics Git Commands

Replacing Local Changes

Revert changes to last commit: `git checkout -- <filename>`

Revert to state of remote branch:

```
git fetch origin
```

```
git reset --hard origin/master
```



What is GitHub?

- A company that allows for free cloud-based Git repository hosting
- Allows for individuals and other organisations, such as FRC teams, to easily create, manage and collaborate on projects whilst using Git
- Designed to be User-Friendly
- Equipped with a GUI (Graphical User Interface)



Types of Repositories

- **Public**

- Group Membership not required to view
- Allows you to publish previous year's code in order to comply with FIRST regulations
- Anyone can view, fork (clone), and make changes to the repository
 - Changes from outside the organization must be approved

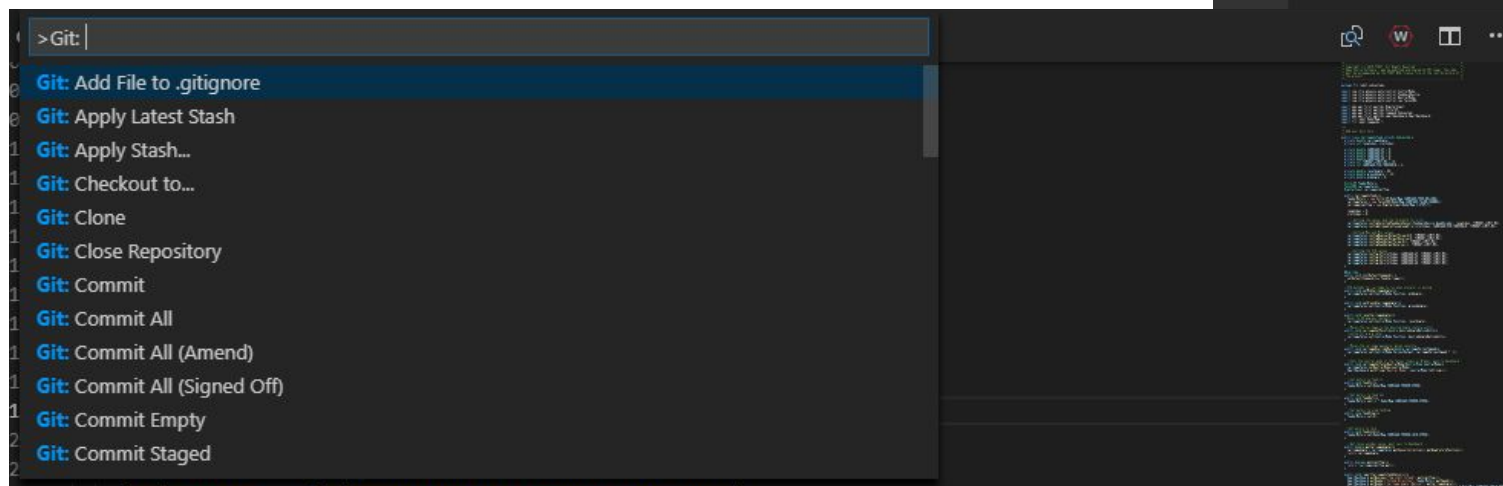
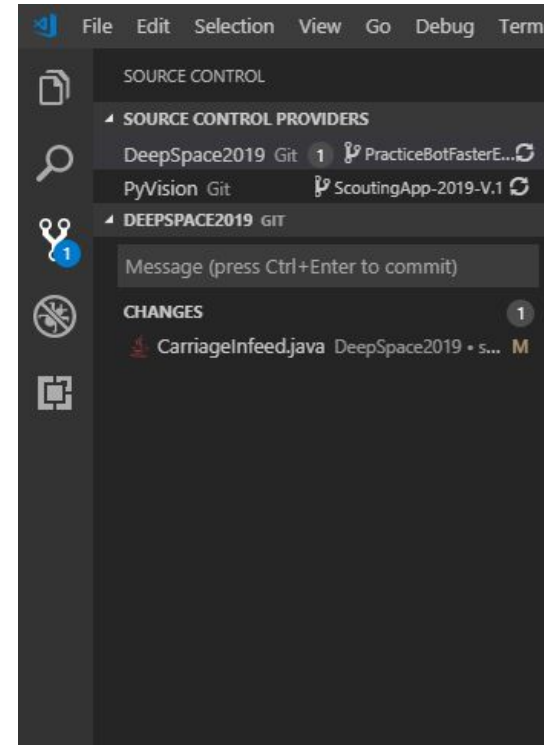
- **Private**

- Group Membership required to view
 - Guests can be invited
- Allows you to keep the current season's code private while still allowing for collaboration
- Easily converted into a Public repository at the end of the season



Git in VS Code

- You can commit your changes to robot code directly in VS Code
 - Available in the Source Control menu
- Access to all Git commands inside VS Code
 - Click the WPILib button and type >Git:
 - Supports all Git commands
 - Simpler interface than through terminal



GitHub Desktop

<https://desktop.github.com>



GitHub provides a desktop client for Git and GitHub, allows for easy to creation and management of repositories all from a single GUI based application. It is available on all OS's (Operating Systems)



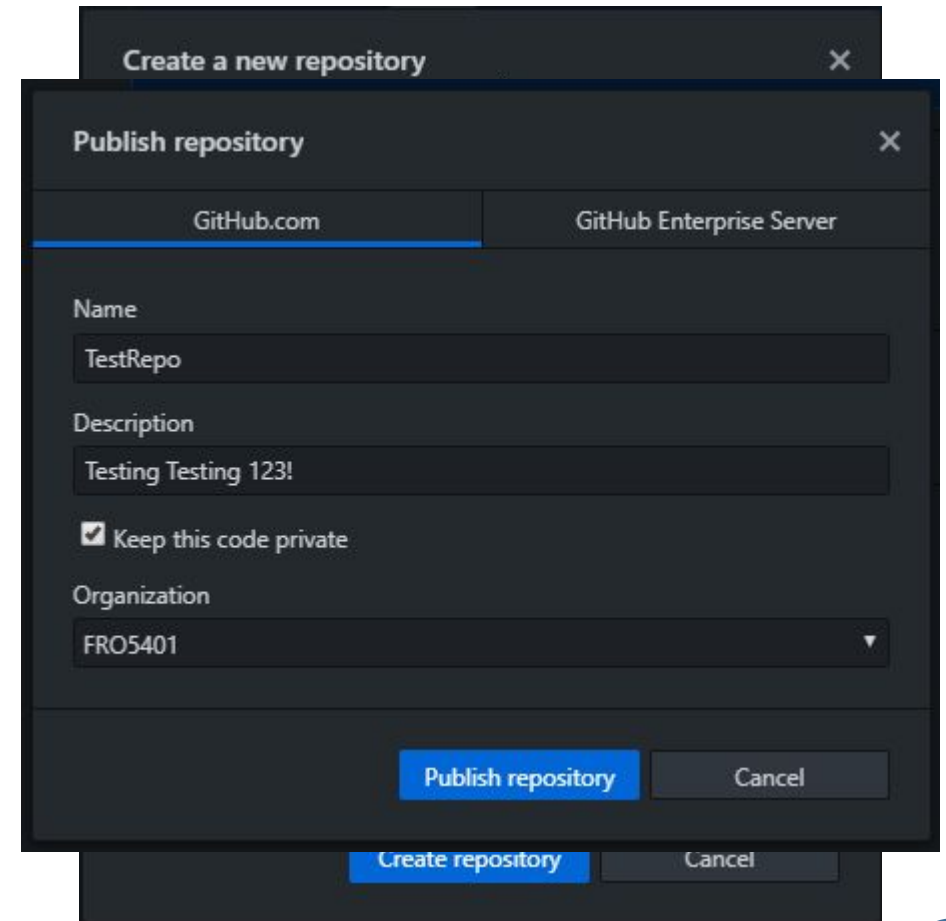
Creating a Repository on GitHub

In GitHub Desktop

- File -> New repository (Ctrl+N)
- Fill in a name and description
- FRC utilizes the BSD 3-Clause license

Once created, publish the repository

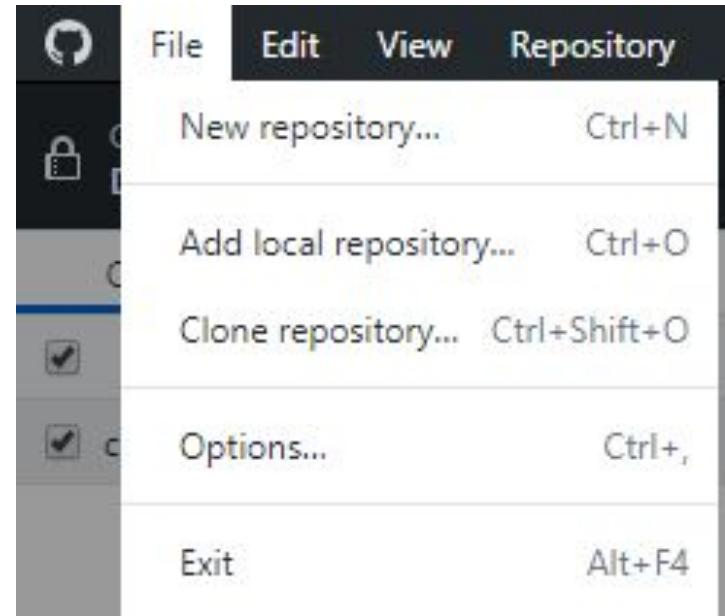
- Repository -> Push
- Fill in a name and description
 - Typically the same as set before
- Choose to make the repo public or private
- Choose an organization to publish to



Cloning a Repository from GitHub

In GitHub Desktop

- File -> Clone Repository (Ctrl+Shift+O)
- Choose a repository from the list
- Confirm the directory where the repository will be cloned to

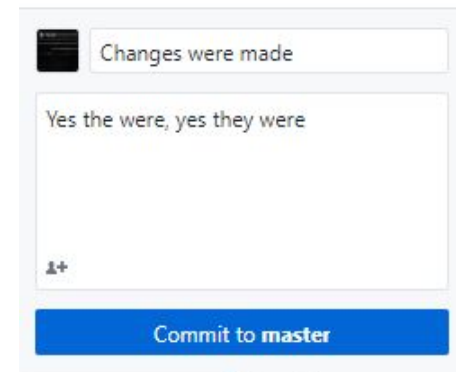
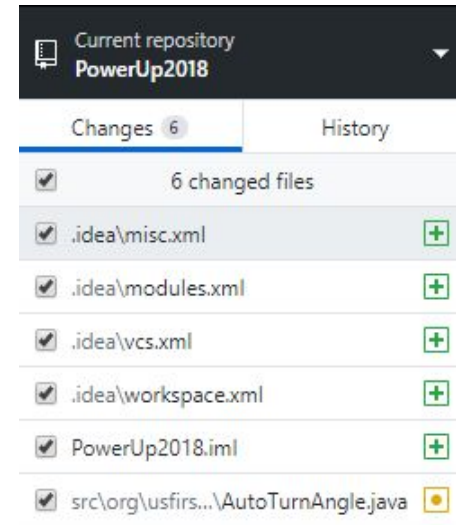


Commits

- A commit is a snapshot of a branch at a single point
- Only contains changes made at that point in time
- Can be easily reverted in the event of a breaking change
 - Individual files or entire commits

To create a commit in GitHub Desktop

- Change/create files in the repository
- Write a short summary (one sentence)
- Optionally, add a full description

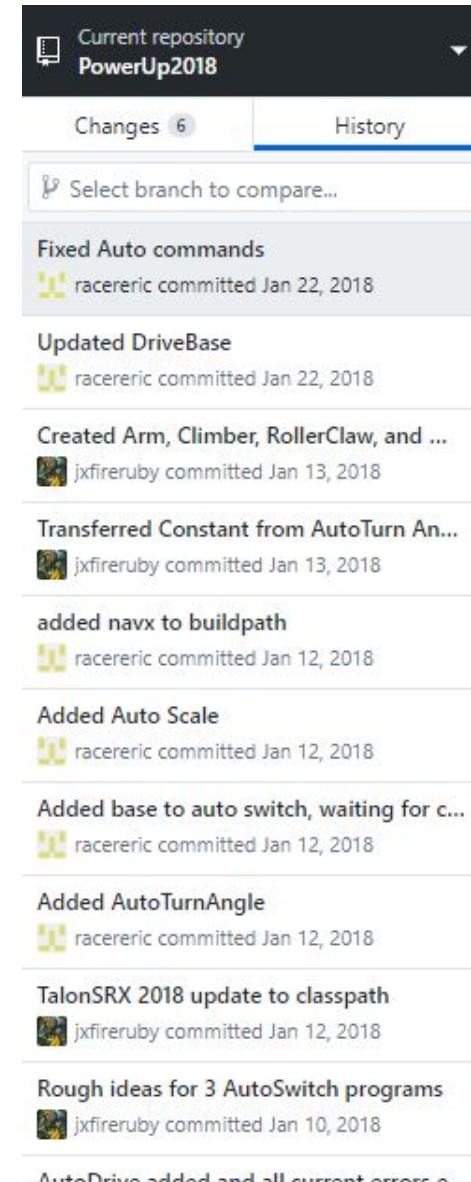


Pushing Commits

- Commits only exist locally
- To make your commits viewable to the rest of the organization, you must push them to the 'remote'

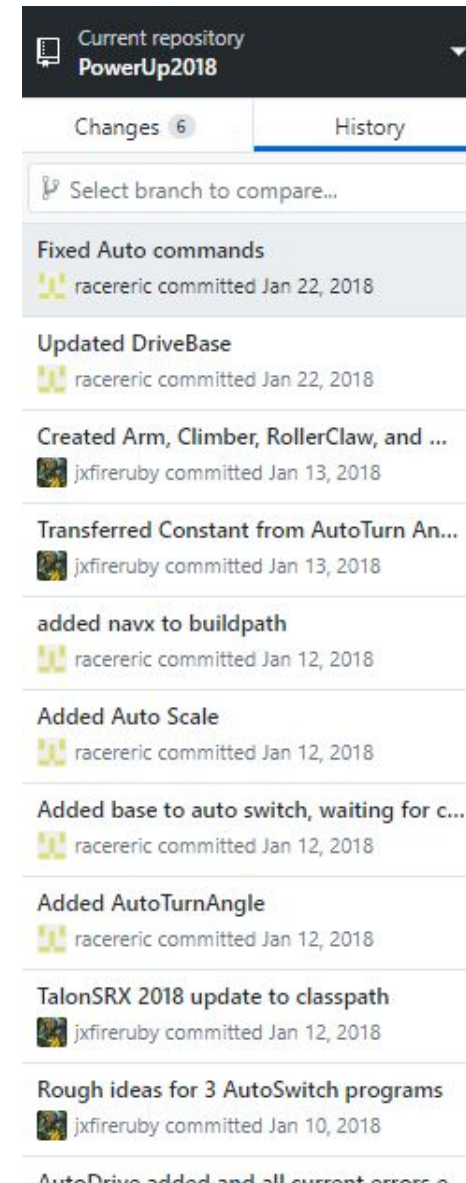
In GitHub Desktop

- Create Commits
- Repository -> Push (Ctrl+P)



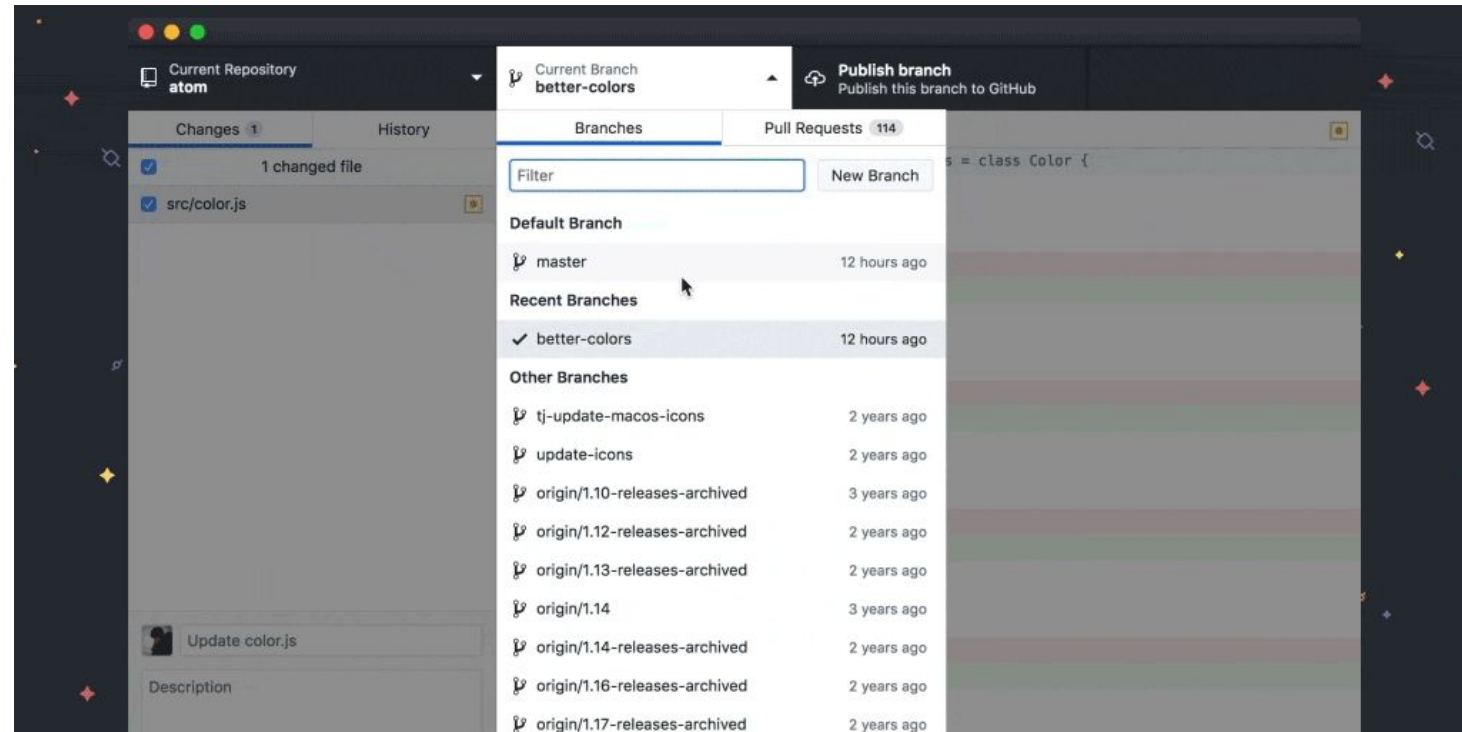
Reverting Commits

- Fixes project breaking errors
- Allows you to track changes as reverting problems, so you don't accidentally make the same mistake again
- Reverts can also be reverted
 - Brings files back to a point before the revert

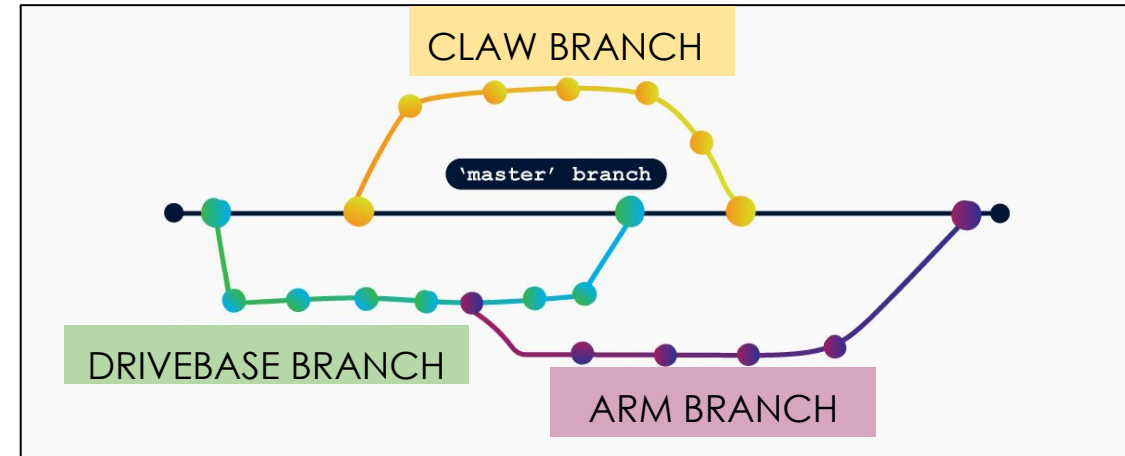


Stashing Commits/Changes

- Incomplete changes can be “stashed” away to go back to later
- Stashes are local only
 - Not synced to GitHub
- Useful when
 - You are not ready to commit changes just yet
 - You are switching branches to work on another feature



Branches

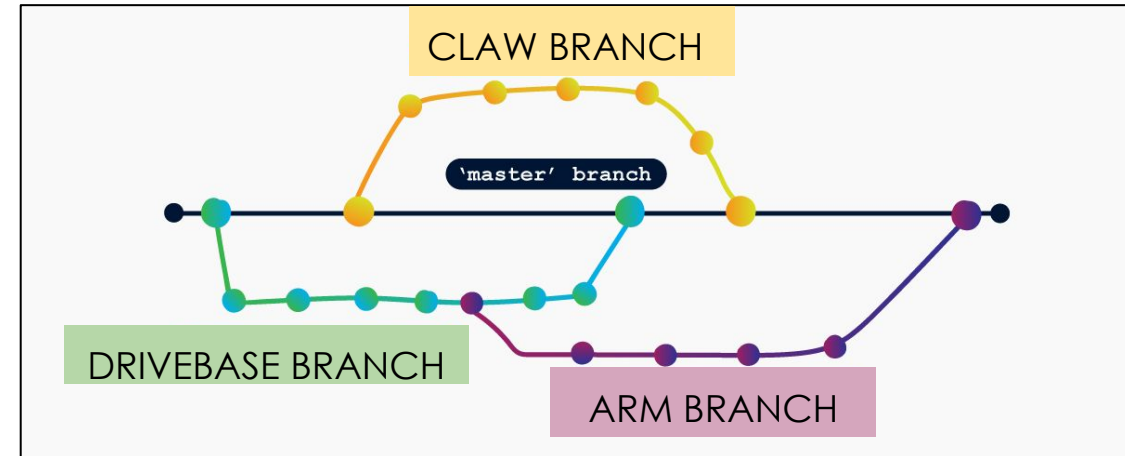


- Different versions of the same repository with different features
- Allows for multiple different features to be added at the same time without the risk of damaging or other harming the “Master” or main branch (Your public and most used branch)



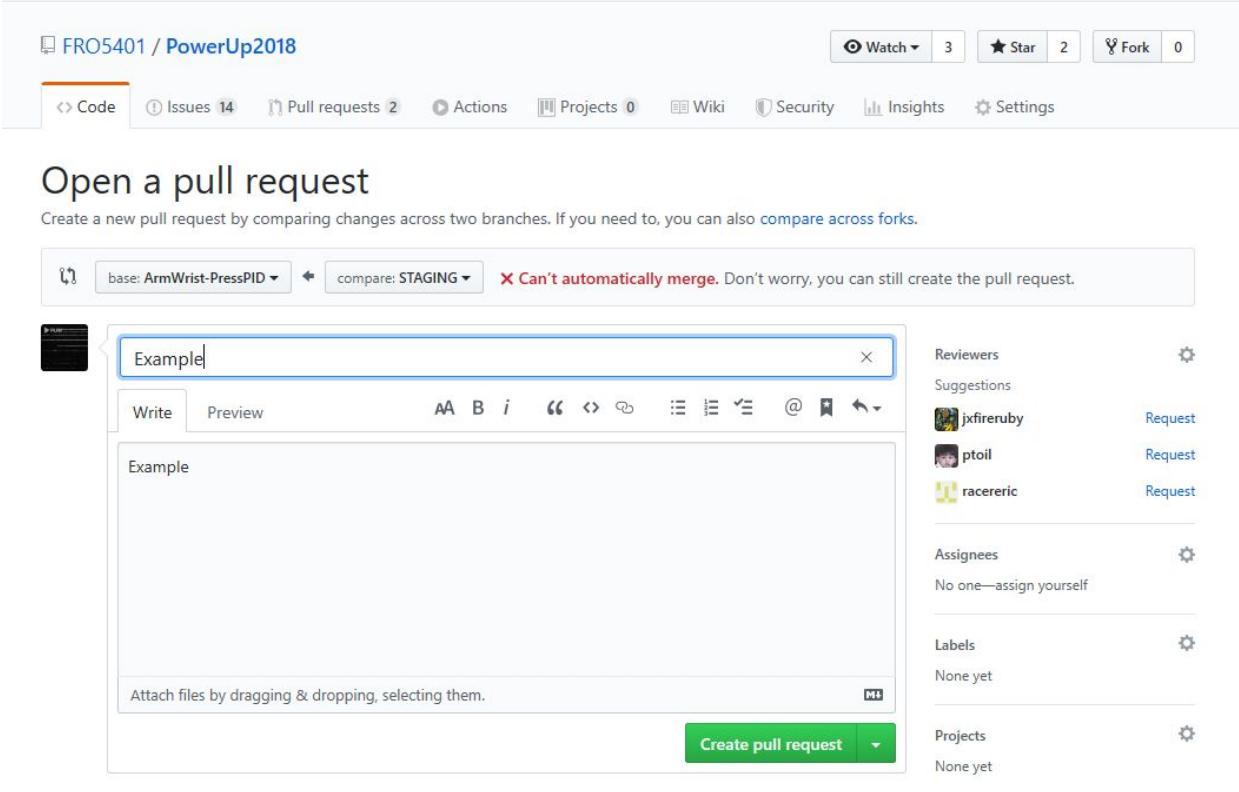
Uses For Branches

- Individual branches for each subsystem
 - Drivebase
 - Subsystem(s)
- Individual branches for multiple students working
 - When teaching a new topic, students can create a branch for themselves to demonstrate knowledge, rather than cluttering the organization with multiple repositories
- Branches for competitions
 - Allows you to make on-the-fly changes during competitions, and review them after, without directly modifying master
- Allows you to keep a working copy of code in master, with changes being done in different branches



Pull Requests

- A Pull Request (PR) is a request to merge two branches
 - Typically a feature branch being merged into staging/master
- Provides a chance to review code for quality and functionality before marking it as stable
- Can result in merge conflicts
 - The same piece of code being edited by multiple people in a way that cannot be automatically solved
 - These conflicts must be merged manually



FRO5401 / PowerUp2018

Watch 3 Star 2 Fork 0

Code Issues 14 Pull requests 2 Actions Projects 0 Wiki Security Insights Settings

Open a pull request

Create a new pull request by comparing changes across two branches. If you need to, you can also [compare across forks](#).

base: ArmWrist-PressPID compare: STAGING **Can't automatically merge.** Don't worry, you can still create the pull request.

Example

Write Preview AA B i “ <> @

Example

Attach files by dragging & dropping, selecting them.

Reviewers

Suggestions

- jxfireruby Request
- ptoil Request
- racereric Request

Assignees

No one—assign yourself

Labels

None yet

Projects

None yet

Create pull request



Merging and Merge Conflicts

- Merge conflicts can occur during
 - Pull Requests
 - Pulling code from the remote
- Must be manually fixed with
 - Git command line
 - VS Code or other IDE

```
21  * @param args the command line arguments
22  */
23  public static void main(String[] args) {
24      // TODO code application logic here
25      Scanner keyboard = new Scanner(System.in);
26      <<<<<<< HEAD
27      System.out.print("Enter an input: ");
28      int input = keyboard.nextInt();
29
30      System.out.println(input * 5);
31
32      =====
33      System.out.println("Enter user input: ");
34      int operand = keyboard.nextInt();
35      >>>>>>> origin/master
36      }
37  }
38  }
```



Issue Linking

- We can now link issues to specific branches
- Helps specify problems and makes code integration easier
- Project boards for issues make resolving much cleaner and nicer!

The screenshot displays a GitHub issue page. At the top, the title is "Link an existing branch or PR #1". Below the title, there is a green "Open" button and a status bar indicating "dipree opened this issue 11 minutes ago · 0 comments". A comment by "dipree" is visible, followed by a text input area for a new comment with a rich text editor toolbar. On the right side, there are several metadata sections: "Assignees" (No one—assign yourself), "Labels" (None yet), "Projects" (None yet), "Milestone" (No milestone), "Development" (Create a branch for this issue or link a pull request.), "Notifications" (Unsubscribe), "1 participant" (dipree), and "Lock conversation", "Pin issue", and "Transfer issue" options.



FRC and Git: Best Practices

- One repository for each season
- One user account for each programmer
- Master/Main branch holds current 'perfect' robot code
- Staging branch holds robot code being tested



FRC and Git: Best Practices

- Once a feature works on its own, Pull Request into staging
- Once staging is verified working, Pull Request again into master
- Create branches for each competition attended, to keep track of on-the-fly changes made in between matches
 - After competition, review changes, and Pull Request into staging/master accordingly

This creates a “chain of custody” that ensures only quality, working code makes its way into master, so there is no question of which code to be running at competition.



Github is on IOS and Android!

- Helps with Build Season notifications
- Creates habits with consistently reviewing and refactoring code that others have committed
- What's not to love about Github being on your phone?
<https://github.com/mobile>



Live Demo Time :)

Follow along if you want



Kahoot Time!

<https://create.kahoot.it/details/0c9aca68-6545-4379-bc48-710288eb93ff>



Resources

<https://web.archive.org/web/20180108144941/https://www.firstinspires.org/robotics/frc/kit-of-parts/#VirtualKit>

<https://help.github.com/en/github/teaching-and-learning-with-github-education/applying-for-an-educator-or-researcher-discount>

<https://github.com/mobile>

