[[](https://camo.githubusercontent.com/d9cf9ee31887a3901288b88af700393f523b2901/68747470733a2f2f73332e616d617a6f6e6177732e636f6d2f6465766d6f756e7461696e2f726561646d652d6c6f676f2e706e67)](https://camo.githubusercontent.com/d9cf9ee31887a3901288b88af700393f523b2901/68747470733a2f2f73332e616d617a6f6e6177732e636f6d2f6465766d6f756e7461696e2f726561646d652d6c6f676f2e706e67" \t "_blank)

**Project Summary**

Practice using git + Github

This project will consist of three separate mini-projects to get you comfortable with the kinds of activities you'll be using git for throughout the class.

In the first mini-project, you'll be mimicking the steps you'll take when you first start your personal project. Creating a repository, linking it to your computer, then pushing those changes up to your GitHub.

In the second mini-project, you'll be mimicking the steps you'll take with nearly every DevMountain project you do. You'll 'fork' the DevMountain repository, link your computer with your fork, then pushing those changes up to your GitHub.

Finally, in the last mini-project you'll be mimicking the steps you'll take during the group project portion. You'll fork your group's repo, link your computer with your fork, push changes to your GitHub, then make a 'Pull Request' into your group's repo.

**Mini-Project 1: Personal Project**

**Step 1**

**Summary**

In this step we will create a repository on GitHUB.

**Instructions**

* Go to [GitHub](https://github.com/).
* Sign in to GitHub.
* On the right side of the page, click on the green New repository button.
* Give your repository any name you like and make sure that the repository is public.
* Also make sure that the Initialize this repository with a README is **NOT** checked.

**Step 2**

**Summary**

In this step we will setup the origin for the repository. We'll do this by connecting code on our computer to the GitHub repository we just created.

**Instructions**

* Create a folder called myProject.
* Go into that folder.
* Create a file called myName.js and add your name to that file.
* Save the file and open a terminal window.
* In your terminal window, cd to your myProject folder. O
* Run git init.
  + What just happened?

You've just told your computer that you want git to watch the myProject folder and to keep track of any changes. This also allows us to run git commands inside of the folder. (Warning: Be very careful to make sure you're in the right directory when you run git init!)

* Run git remote add origin [Repository URL goes here]. You can get your URL from going to repository you made earlier in your browser and copying the address.
  + What just happened?

Basically, we tell our computer "Hey, I created this repo on GitHub, so when I push, I want my code to go to this GitHub repo." Now whenever you run git push origin master your computer knows that origin is pointing to your repo you made on GitHub and it pushes your changes there.

( If you accidentally DID initialize your repository with a README, you must do a git pull origin master first - to get the README file on your computer - before you'll be able to push. )

**Step 3: Push your code to GitHub**

**Summary**

In this step, we will push code to GitHub.

**Instructions**

* Open a terminal window and make sure it is in the directory of myProject.
* Run git status.
  + What does this do?
* Run git diff.
  + What does this do?

This will show the actual code that has been changed. Again, we want to make sure we don't push anything to GitHub that shouldn't be there.

* Run git add nameOfMyFile.fileExtension.
  + What does this do?
* Run git commit -m "The sentence I want associated with this commit message".
  + What does this do?

This tells your computer: 'Hey, the next time code is pushed to GitHub, take all of this code with it.' The message also specifies what GitHub will display in relation to this commit.

* Run git push origin master
  + What does this do?

Your code is now pushed to GitHub. Be sure to include origin master, as this tells GitHub which branch you want to push to, and creates the branch if it doesn't exist yet.

* Go to your repository on GitHub and see your updates.

**Mini-Project 2: DevMountain Project**

**Step 1**

**Summary**

In this step, we will fork this tutorial repository.

**Instructions**

* On this current GitHub repository, scroll to the top and look for a button that says fork.
* Click the fork button.
  + What does this do?

**Step 2**

**Summary**

In this step, we will take the forked repository and clone it down to our machine.

**Instructions**

* Go to your forked repository on GitHub. It should appear under Your repositories which is next to the New repositorybutton.
* Click on the green clone or download button and copy the URL.
* Open a terminal window and navigate to your Desktop.
* Run git clone [the url you copied].
  + What does this do?

**Step 3**

**Summary**

In this step, we will make changes to our clone and push them to GitHub.

**Instructions**

* Open the folder in your coding IDE.
* Make a change in a file.
* Run through the steps outlined in Step 3 of the first project ( status, diff, add, commit, push ).
  + Since you've cloned this repository, it is already pointing to your forked version. Therefore, you don't need to tell your computer where to push the code.

**Mini-Project 3: Group Project**

**Step 1**

**Summary**

To help this process stick in memory we are going to repeat the process of the second project. We'll delete our current fork on our machine and restart the process.

**Instructions**

* Delete the folder on your Desktop that is the forked repository.
* Re-clone the fork to your desktop.
* Make a change to any file.
* Run through the process of pushing to GitHub ( status, diff, add, commit, push ).

**Step 2**

**Summary**

Here is where things start to get different. Let's imagine we're working in groups. If we have everyone pushing to one repo without verifying the quality of the code, things can get messy pretty quick. GitHub fixed this solution with 'Pull Requests.' Basically, you fork a project, make changes to your fork, then you make a Pull Request (PR) back into the original project requesting that some piece of code be added to the original repo. This is how the vast majority of open source code projects work. In this step, we will make a pull-request.

**Instructions**

* Go to your forked repo on GitHub.
* Locate the button that says Pull Request and click it.
* Locate the green button that says New pull request and click it.
  + You should now see the file changes you've made and how they differ from the original repo.
* Click on the Create pull request button to submit your PR.
* Now if you navigate to the [original repository](https://github.com/DevMountain/learn-git/pulls) and take a look at the Pull Requests yours should be there.

**Contributions**

If you see a problem or a typo, please fork, make the necessary changes, and create a pull request so we can review your changes and merge them into the master repo and branch.

**Copyright**

© DevMountain LLC, 2017. Unauthorized use and/or duplication of this material without express and written permission from DevMountain, LLC is strictly prohibited. Excerpts and links may be used, provided that full and clear credit is given to DevMountain with appropriate and specific direction to the original content.

[](https://camo.githubusercontent.com/d9cf9ee31887a3901288b88af700393f523b2901/68747470733a2f2f73332e616d617a6f6e6177732e636f6d2f6465766d6f756e7461696e2f726561646d652d6c6f676f2e706e67)

* © 2019 GitHub, Inc.
* [Terms](https://github.com/site/terms)
* [Privacy](https://github.com/site/privacy)
* [Security](https://github.com/security)
* [Status](https://githubstatus.com/)
* [Help](https://help.github.com/)
* [Contact GitHub](https://github.com/contact)
* [Pricing](https://github.com/pricing)
* [API](https://developer.github.com/)
* [Training](https://training.github.com/)
* [Blog](https://blog.github.com/)
* [About](https://github.com/about)

Press h to open a hovercard with more details.