IGME-230 Website Design & Development

Git and Github Exercise

Pre-Class Preparation

In our week 1 exercise, I asked you to set up an account on GitHub (if you didn't already have one), and to apply for the Github Student Developer Pack so that you'll be able to have private repos. Today's exercise also assumes that you have completed either the GitHub for Web Designers or the Version Control for Everyone tutorial on Lynda.com.

Because the files we're using for this exercise are in a private repository in the LawleyClasses organization, you need to be part of the 230-spring2017-students team in that organization to access them. That's why I asked for your Github username on Tuesday. If you haven't provided that to me yet, you'll need to do it now before you can proceed. When I send the invitation, you'll get an email with a link to join the team.

Review: What is Git?

Git is a distributed version control software (DVCS). In traditional version control systems, there was a central repository that held all of the code.

A *repository* is a data structure used by version control software to hold all of your files and the history of each change. This allows you to rollback to previous versions of files and see who has done what.

In older version control systems, like SVN, files were stored in a central server-based repository, and developers "checked out" a file from the server when they were working on it. That meant if the server went down, or you lost your network connection, you could not sync your code. It also meant that if one developer had a file checked out, nobody else could work on that file.

In a distributed version control system like Git, every person's computer is a full-fledged repository. In the event of a server failure or a connection error, you can still sync your code to your local machine and to anyone else who needs it. It is a peer-to-peer system where one or more of the "peers" may be a server. This makes sharing your code with people much easier.

Review: What is Github?

Github is an git repository web service that also provides a web interface, a task tracker, wiki pages and more. It actually works a lot like a social network in that has feeds, followers, organizations, etc. At its core though, Github is primarily a server system for hosting git repositories.

IGME-230 Website Design & Development

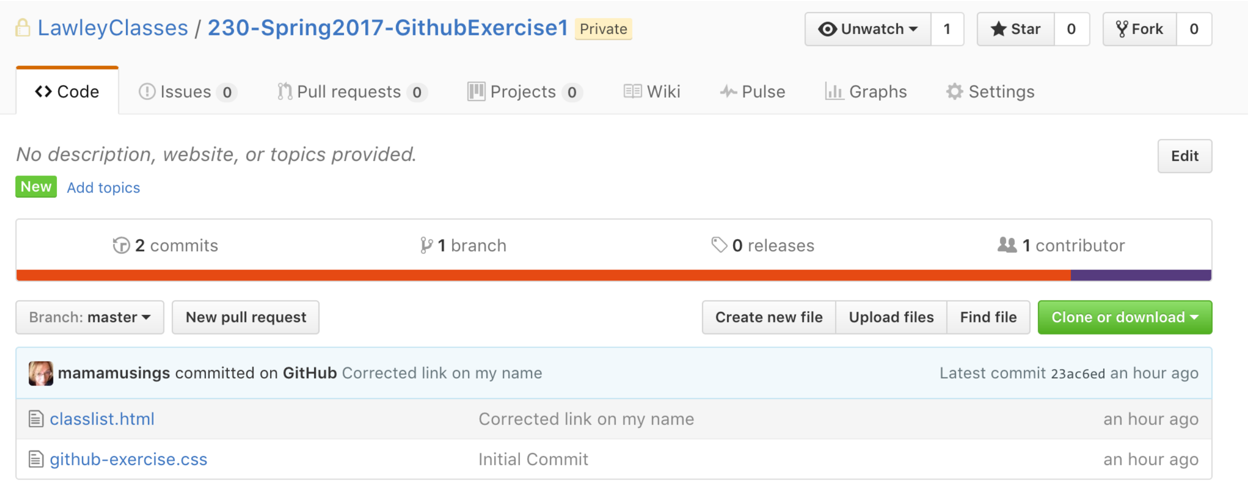
Git and Github Exercise

A Github repository is just one of the “peers” in the peer-to-peer organization of git. It may be one of the places you sync your code to. It makes it easy for other people to see your code and sync with it. In multi-developer situations, Github lets each developer sync their code back to Github so that the others can get it.

Since Github is just a web service that lets you host a copy of your repository online, you can use git without it. Github is also not the only service that does this. There are several other similar services available, including Bitbucket and Gitlab.

Using Git Part I

1. Go to the repository for exercise part 1: https://github.com/LawleyClasses/230-Github-Exercise1. You should see two files, an HTML file called classlist.html, and a CSS file called github-exercise.css



No description, website, or topics provided.

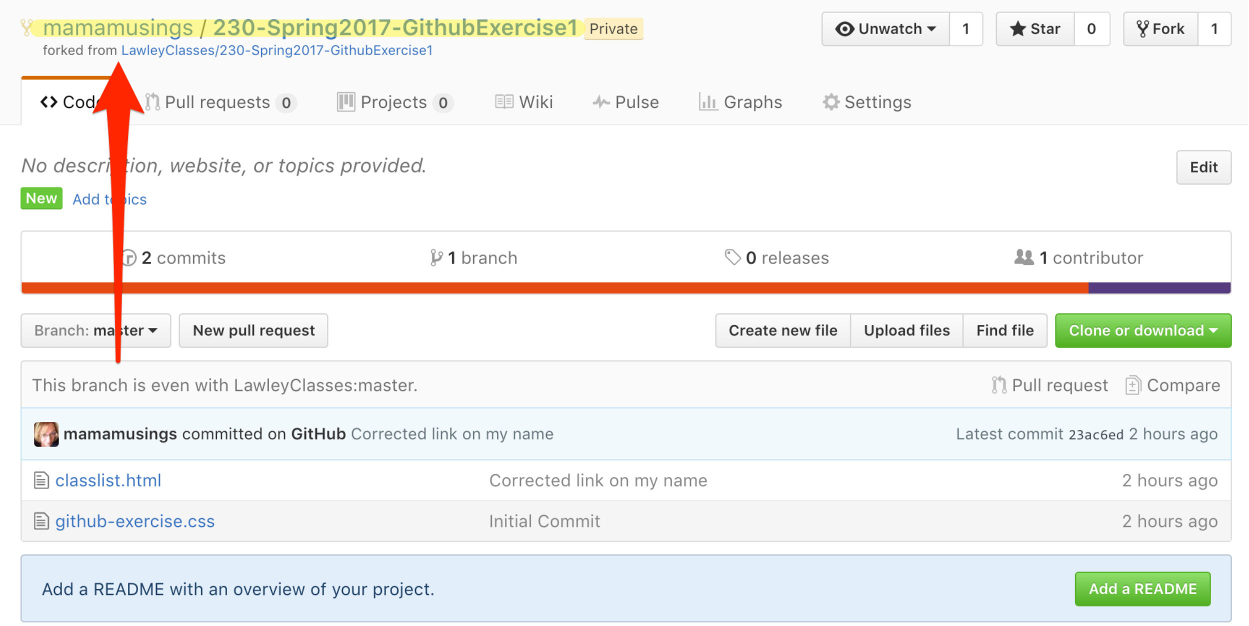
mamamusings committed on GitHub Corrected link on my name — Latest commit 23ac6ed an hour ago

classlist.html	Corrected link on my name	an hour ago
github-exercise.css	Initial Commit	an hour ago

2. You are going to make your own fork of the repository on Github. A fork is a new repository that is a copy of an existing repository. One reason to make a fork is to take a project in a new direction. For instance, a faculty member at another university might want to use my syllabus repository as a starting point for their own class. They could fork the class repository so that they have the basic structure and content, and then edit the files to reflect their own class details.

   A fork can also be used if someone who doesn’t have access to modify a repository wants to make changes—for instance, a programmer who wants to help fix a bug on a big open source project. In that case they would make a fork of the code, fix a bug or add a feature, and then send a request to the original owner to pull in their changes. This is referred to as a pull request.

In the top right corner of the page is a button labeled "Fork"—click that, and if prompted, select your personal account. This will create a copy of the repository owned by your account. (If you've set up your Github Education Pack, you should be able to create this as a private repository.) You should be taken to the new repository page. While the content is the same you can see that the path to the repository now has your user name rather than LawleyClasses.


mamamusings / 230-Spring2017-GithubExercise1 Private
forked from LawleyClasses/230-Spring2017-GithubExercise1

No description, website, or topics provided.

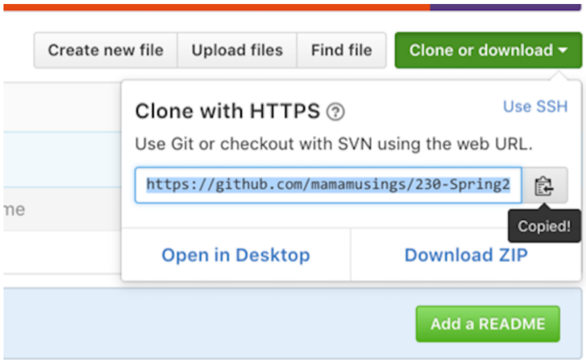
This branch is even with LawleyClasses:master.

mamamusings committed on GitHub Corrected link on my name — Latest commit 23ac6ed 2 hours ago

classlist.html	Corrected link on my name	2 hours ago
github-exercise.css	Initial Commit	2 hours ago

Add a README with an overview of your project.

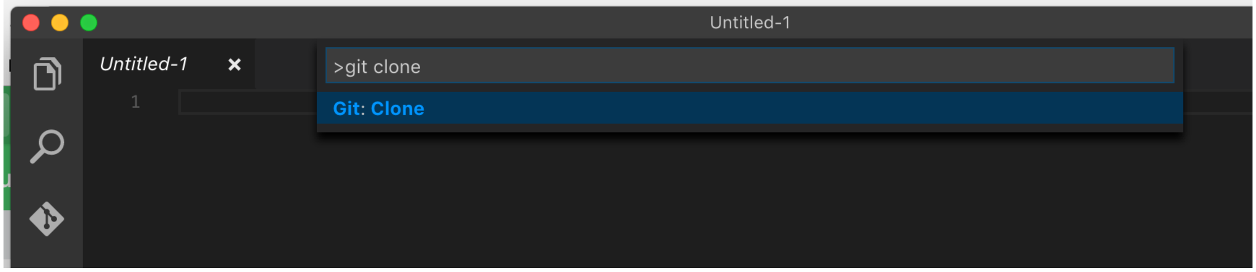
3. Now you need to clone your Github repository to your local machine, so you can edit the files locally. Copy the CLONE URL from **your** new repository (the one under your account, not the one in LawleyClasses), using the "Clone or Download" button. If you hit the little clipboard looking button next to the URL, it will copy the entire URL to your clipboard for pasting.


Clone with HTTPS

Use Git or checkout with SVN using the web URL.

```
https://github.com/mamamusings/230-Spring2
```

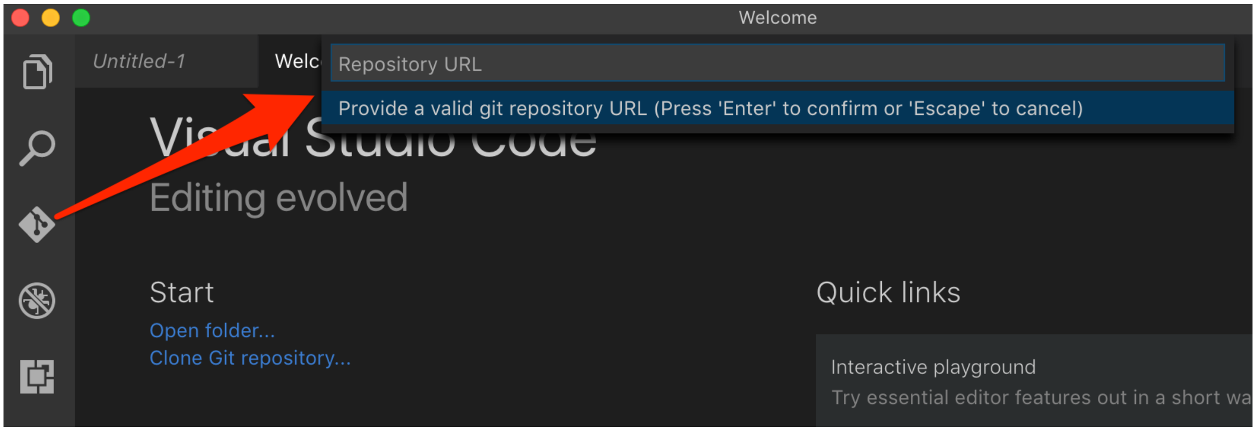
4. Launch Visual Studio Code (or, if it's already open, choose "New Window" from the File menu). Press F1 for the command menu, and type Git Clone. You should see the Git: Clone command. Press enter.



>git clone

Git: Clone

Visual Studio Code will now prompt you for a valid Git repository URL, which is what you just copied from Github. Paste the URL in here.



Repository URL

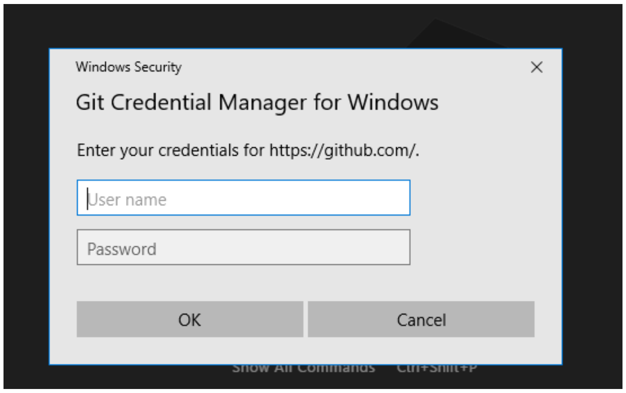
Provide a valid git repository URL (Press 'Enter' to confirm or 'Escape' to cancel)

Try essential editor features out in a short wa

Hit enter once you've pasted in the URL, and VS Code will prompt you for a destination directory. It will create a new folder with the repository name in whatever folder you designate, so you can just pick Desktop if you want.

VS Code will then prompt you for your GitHub username and password, so it can access your files. (If you've enabled 2-factor authentication on Github, this step won't work properly, and you'll have to authenticate using the command line. Ask me for help with this.)

If this all works properly, you'll now have an exact copy of your Github repo on your local computer, and displayed in VS Code.


Windows Security

Git Credential Manager for Windows

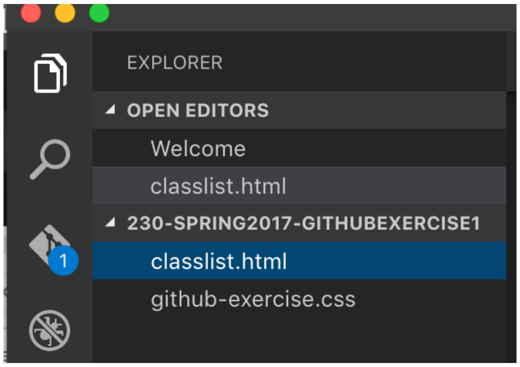
Enter your credentials for https://github.com/.

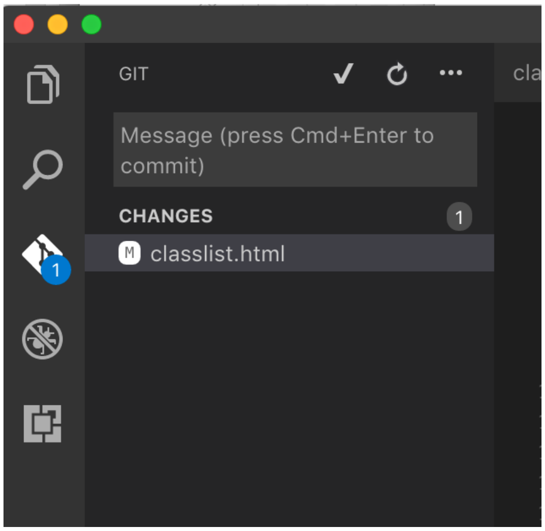
User name

Password

5. In VS Code, select the classlist.html file, and edit it to add a link from your name to your igme230 directory on people.rit.edu. Save the file.

6. You should now see a notification on the Git icon in the left sidebar. Click on that icon to see the current status of your repository.





You should see the classlist.html file listed under “Changes”, with an M (for “modified”) next to the file.

To see the difference between the original file and the modified version, click on the filename. VS Code will show you a side-by-side comparison noting any content that has been removed or added. The original file (noted with “HEAD”) is on the left, and your modified file is on the right.

The file has been changed on your computer, but they have not changed in the local Git repository. The local repository is the data structure on your machine tracking the history and changes of the code.

To update the repository, we need to *stage* this file, which you can do by clicking on the + sign to the right of the file, or, if you have multiple changed files, by clicking on the + sign to the right of “Changes”. When you do this, the file will move from “Changes” to a new category of “Staged Changes.” This adds the file to the **queue** of files to be added to the repository. It does not actually add it to the repository yet, it just stages it for review and prepares it to be added to the repository.

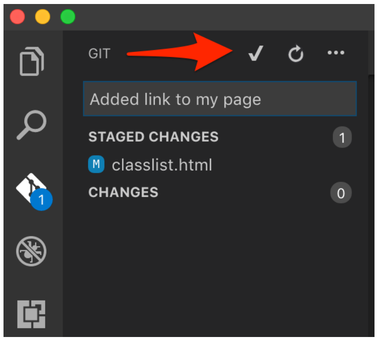
Why the extra step? When adding a new feature to your code on a collaborative project, it's possible that you will be altering multiple files. (For instance, you might update an HTML file to include an image link, which would require you to add the image file to the repository.) You want to make sure that all the related files are updated at once, in case someone else is also working on some of those files.

To reduce the chance of that problem, git waits until you have queued all of the files you want to add and then it adds them in all at once when you commit the files. When someone else syncs their code with the repository, they get all of the files from your update.

7. Now that you have staged your file, you need to "commit" it to the repository. Committing means that you are adding all of the changes in the queue to the repository, along with a message summarizing what you did.

   Type your commit message in the box above "Staged Changes." **Remember:** Make these messages useful. Explain briefly what you did. One line is fine, but it should be a clarification of what you changed, not just 'did stuff'.

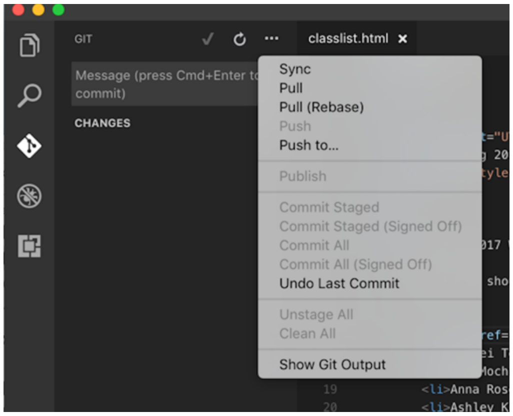
   Once you've added your commit message, click the check mark to commit the files to your local copy of the repository.



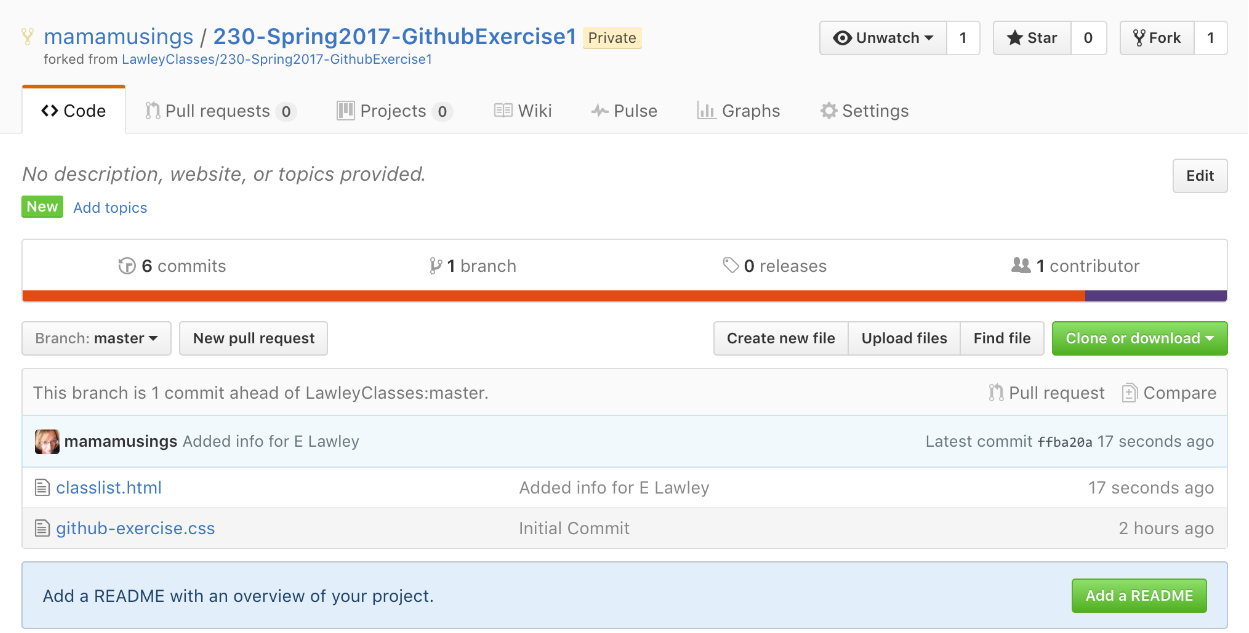
Added link to my page

8. Your changed files are now part of your local repository, but they are not yet synced to any other version of the repository. Now you're going to push your changes to the Github version of your repository.

   You can see the various options for interacting with the Github version of your repository by clicking on the … at the top of the Git status sidebar. VS Code lets you do this by either using "Push" (which pushes your changed files to the "origin" repository on Github), or "Sync" (which pushes your changes to the origin and **also** pulls any changes that might have been made by others). You can see the vSince you're the only one working on this repository, you can choose either one. In part 2, when you'll be collaborating on a file, you'll have to approach this process differently.



9. Now look at your forked repository on Github. At the top of the list of files you should see your most recent commit message, and that message will also show up next to the classlist.html file you modified. There will also be a message above the latest commit saying "This branch is 1 commit ahead of LawleyClasses:master", because Github keeps track of the differences between the original repository and your forked copy. That would be important if you wanted to contribute back to the original repository through a pull request, something we won't be doing today.



No description, website, or topics provided.

This branch is 1 commit ahead of LawleyClasses:master.

mamamusings Added info for E Lawley		Latest commit ffba20a 17 seconds ago
classlist.html	Added info for E Lawley	17 seconds ago
github-exercise.css	Initial Commit	2 hours ago

Add a README with an overview of your project.

10. Go back to the LawleyClasses version of the repository: (https://github.com/LawleyClasses/230-Spring2017-GithubExercise1) . Notice that it has not been changed. This is because your "fork" of the repository is entirely separate from the original. (You also don't have the right permissions to change this directory.)

11. Since your repository on Github has been updated, you can now delete that directory code on your local machine (if you want to). You will always be able to clone the latest set of files from your Github repository onto any machine with git installed.

Using Git Part II

1. This exercise uses the https://github.com/LawleyClasses/230-Spring2017-GithubExercise2 repository. It is very similar to the first repository—with the exception that the HTML files is called "sharedclasslist.html", and everyone in the the class has the ability to directly modify

the files. Instead of making a fork of the repository in your own Github account, you're going to clone the LawleyClasses version of the repository to your computer. Since everybody else in the class will be cloning the same repository, we're all going to be making changes to the same HTML file.

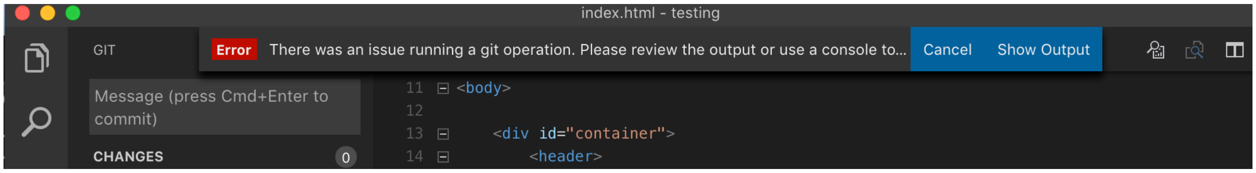
2. Using the process described in step 3 in the first part of the exercise, copy the clone URL for the shared repository, and clone it to your local computer in VS Code. By the time you do this, some people may have already started modifying the code on Github. That is okay!

3. Open the sharedclasslist.html file in VS Code, find your name, and add a link to your igm230 directory on people.rit.edu.

4. Once you have finished, stage and commit the changed file to your local repository. Make sure your commit message has your name (e.g. "Added link to Liz Lawley's page")

5. Now sync your changes back to the organization's Github repository using the Push command.

   If someone else has modified the repository since last time you pulled or cloned (which will be true for most of you), your changes will get rejected because of the conflict, and you'll see this message:



index.html - testing

Error There was an issue running a git operation. Please review the output or use a console to...

```
<body>

    <div id="container">
        <header>
```

   Clicking on "Show Output" will let you see why Github rejected your commit—it should say something like "the remote contains work that you do not have locally."

6. If do receive an error, that means you need to retrieve the updates to the file before pushing your changes. You do this by running the Pull command. This will pull down any updates to the file (specifically, URLs that other people have added) and merge them into your code. Assuming that nobody else changed the line of the file that you were editing, the pull should be successful. If you have an error at this point, let me know.

7. Once you have pulled the changes from Github to your local file, try running the Push command again. If you are lucky, no one else made another change during that time. If they did, however, you will need to repeat steps 5 and 6 until successful.

8. If, after pushing your changes, you don't get an error message, take a look at the repository on Github, and view the sharedclasslist.html file. You should see your change reflected in the file.

Submission

I will be checking the Github repositories to make sure that (a) you forked the first exercise repository, and (b) you added your URL to the sharedclasslist.html file in second repository.