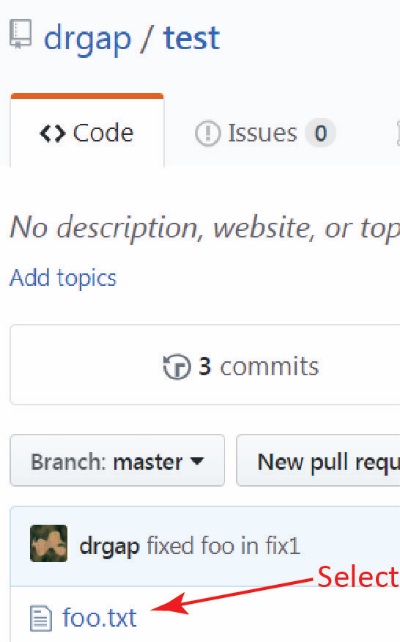
**Tutorial 8**

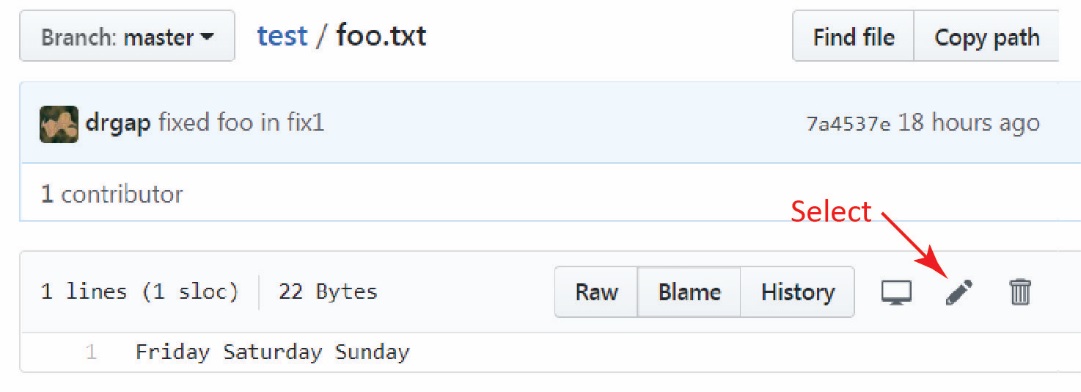
***Pulling Changes from GitHub***

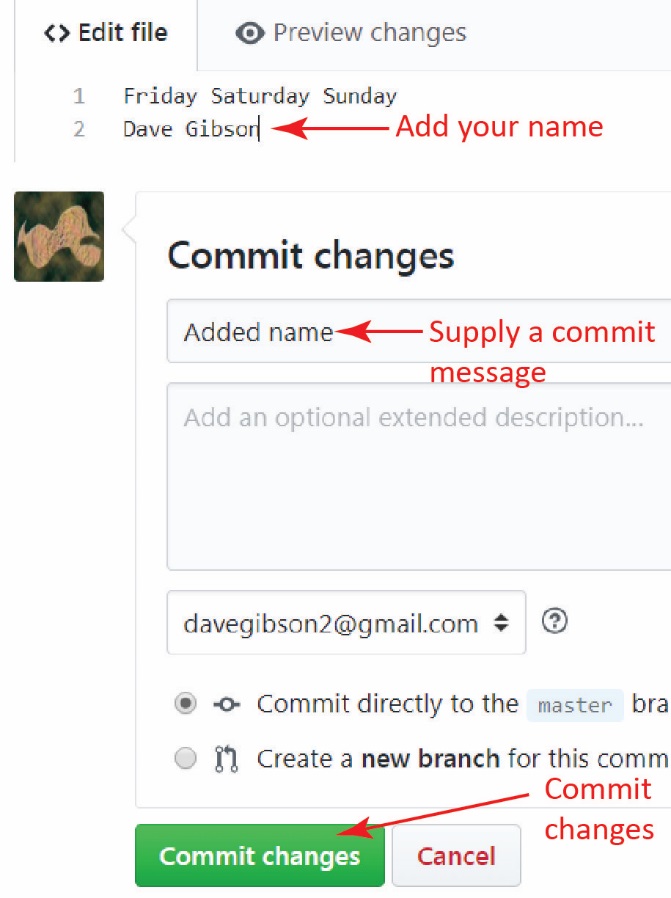
In this tutorial we will explore pulling changes that have been made on GitHub (*e.g.* changes a collaborator makes and pushes to GitHub) to your local computer.

This tutorial continues from Tutorial 7.

**Part 1 – Steps to Complete**

1. Login to GitHub and open *foo.txt* in your *test* repo, by selecting it in the list of files in the master branch of the repository as shown on the right.
2. Edit the file by selecting the pencil icon on the right at the top of the file as shown in the figure below.

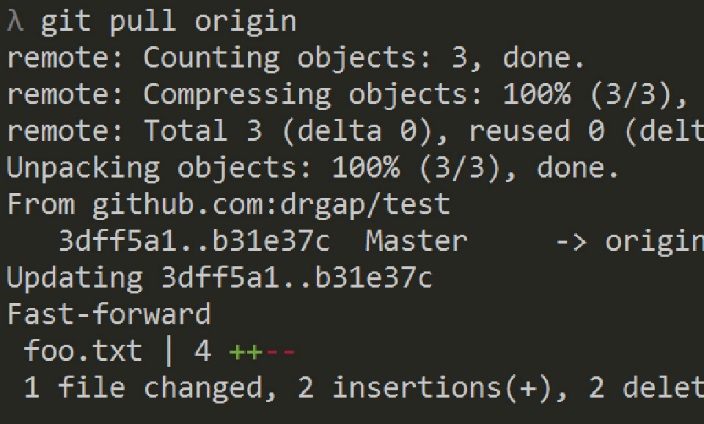




1. As shown on the right, do the following:
2. Add your name to the end of the file.
3. Supply a commit message.
4. Commit the changes (will probably need to scroll down)

Our objective here is to illustrate several things:

* You can make edits to files in the repository directly on GitHub. Normally, we would probably only do this for a simple change.
* We don’t “save” the file; we commit the changes, supplying a commit message, just as we did in our local repository (but we don’t have to *stage* the file as we did locally).
* We are “simulating” that another user has made a change from their local repository and pushed the changes back to GitHub.
* In the next steps you will *pull* these changes back down into your local repository.

1. Click the name of your repository (*i.e.* click the “test” and notice that now there is another commit (mine says “4 commits”). Select the commits link and verify that the commit you just made is listed.
2. Return to your local workspace and pull the changes from GitHub:

**λ**  git pull origin

The result is shown on the right. The message is a bit cryptic, but it did succeed. I believe if you had made changes on your local files that “conflicted” with the changes on GitHub, the *pull* would “fail” in a sense and Git would have indicated this to us (similar to an earlier tutorial when we had a conflict when trying to merge a branch). We will probably explore this in another tutorial.

1. (Optional, Read, no action required) *git pull* actually does two commands: *get fetch* and *get merge*. A detailed explanation can be found on [stackoverflow](https://stackoverflow.com/questions/292357/what-is-the-difference-between-git-pull-and-git-fetch).
2. Next, we will make a change locally, commit it, push the changes back to GitHub, and then verify that the changes are present. Do the following:
3. Open *foo.txt* and add a line at the end, “Fun about to begin.”

**λ**  notepad foo.txt

1. Commit the changes:

**λ**  git commit -a -m "More stuff added to foo"

1. Push the changes back to GitHub:

**λ**  git push -u origin master

1. Return to GitHub, refresh the page, open *foo.txt* and verify that the change is present.

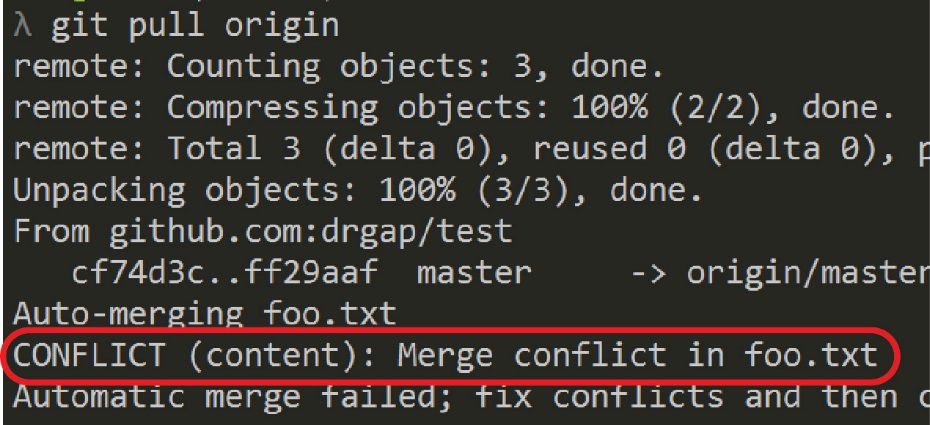
**Part 2 – Steps to Complete**

1. (Read, no action required) Next, we consider a conflict on a pull. We will do the following: (a) make a change on GitHub and commit, (b) make a change locally and commit, (c) pull from GitHub which results in conflict, (d) resolve the conflict, (e) push back to GitHub.
2. Do the following:
3. Open *foo.txt* on GitHub
4. Add a line at the end of the file: “Change from someone else”
5. Commit the change.
6. Do the following:
7. Open *foo.txt* on you local machine. Add a line at the end of the file: “Groundhog day”

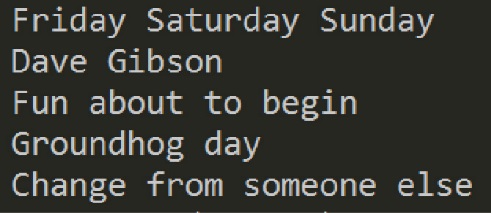
**λ**  notepad foo.txt

1. Commit the change.

**λ**  git commit -a -m "Foo continues to change"

1. Next, we will pull from GitHub and we expect a conflict between the two versions of *foo.txt.*

**λ**  git pull origin

1. Edit *foo.txt,* resolving the conflict so that it looks as shown on the right.

**λ**  notepad foo.txt

1. Stage and commit the changes locally

**λ**  git add foo.txt

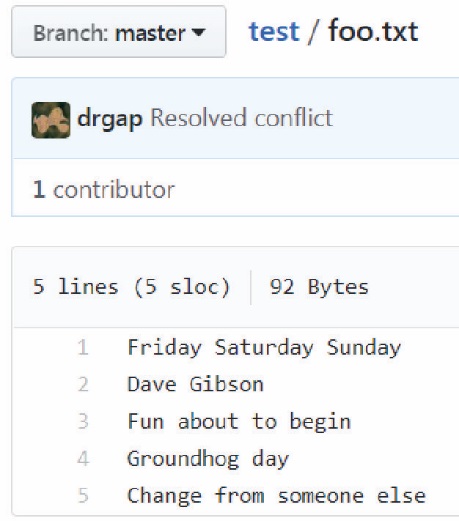
**λ**  git commit –m “Resolved conflict”

1. Check the status. Should say “Your branch is ahead of ‘origin/master’ by…”

**λ**  git status

1. Push changes back to GitHub

**λ**  git push -u origin master

1. Open *foo.txt* on GitHub and verify the changes as shown on the right.
2. **Do the following:**
3. Make a screen shot similar to the one in Step 16 above. Make sure it shows your User ID and your name in the text.
4. Place the image in the *HW VCS* document in the appropriate place.
5. The image should easily readable without zooming in or out.