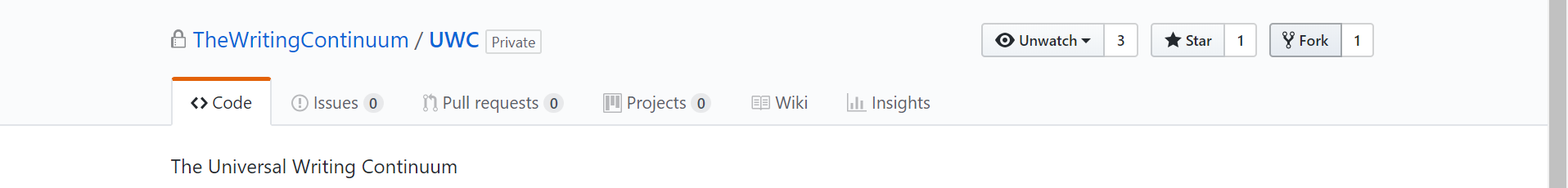
The Writing Continuum GitHub Guide

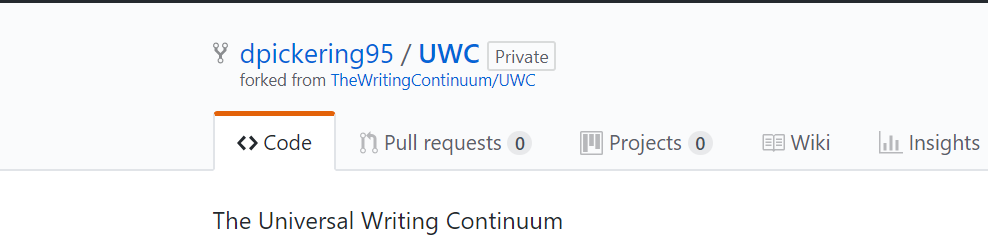
It is important that we ensure that we can make changes to our code without a) breaking anything or b) overwriting anyone’s changes. To do this, we are going to make use of the “GitHub Fork & Pull Request” model. This is a straightforward approach to making changes to the codebase that, when used correctly, will make for a seamless way to make changes to the codebase.

**Forking the repo**

For any new developer, the first thing that you want to do is make a fork of the main repository. This is done by heading over to the main repo (https://github.com/TheWritingContinuum/UWC) and clicking the fork button in the upper right corner. This fork will be your local copy of the main repo, it will be where you check all your changes into.



After forking the repo, you should see your local copy.



**Creating an upstream remote**

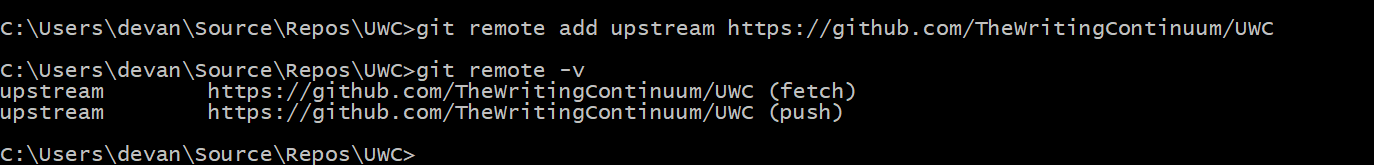
This is a onetime step that allows GitHub to associate our main repo with the name “upstream”

In the git command line execute the following:

git remote add upstream <https://github.com/TheWritingContinuum/UWC>

Verify that the remote is created by executing the following:

git remote -v

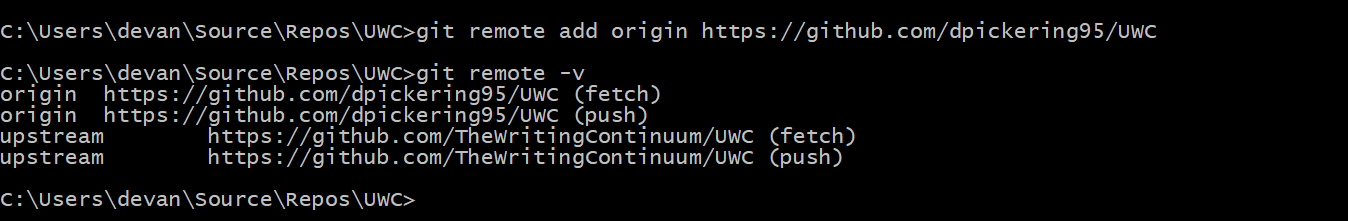


**Creating an origin remote**

This is a onetime step that allows GitHub to associate our local (forked) repo with the name “origin”

In the git command line execute the following:

git remote add origin [link to your local repo]



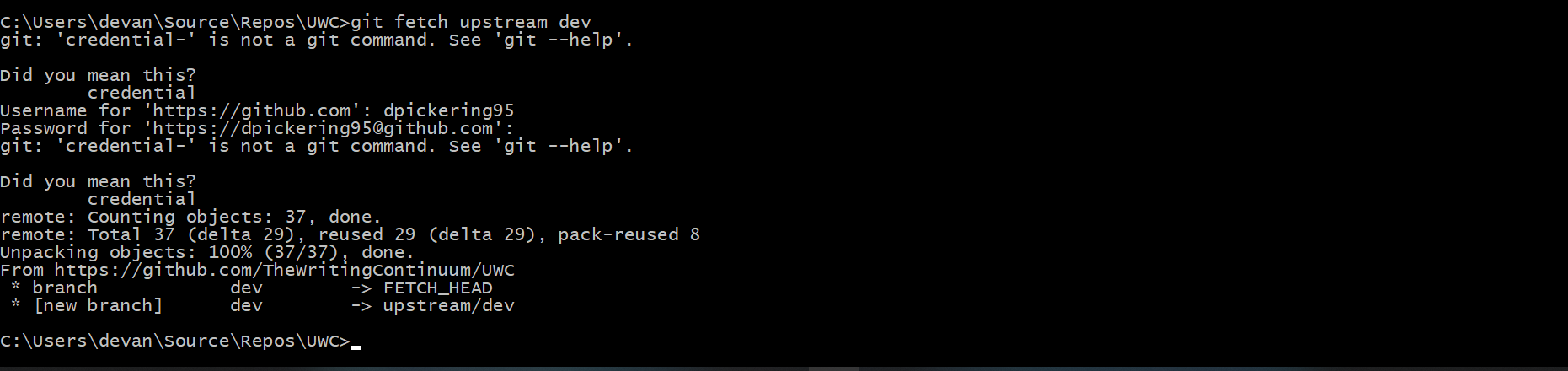
Now you should have a remote to the main repo (upstream) and your local copy of the repo (origin).

**Keeping your origin repo in sync with upstream**

Because there are going to be multiple developers making changes, we want to make sure that we have the most up to date snapshot of the code before making our changes. This is done by fetching all the changes from the dev branch in the main repo. Make sure that this step is executed every time you plan to make a change!

In the git command line execute the following:

git fetch upstream dev



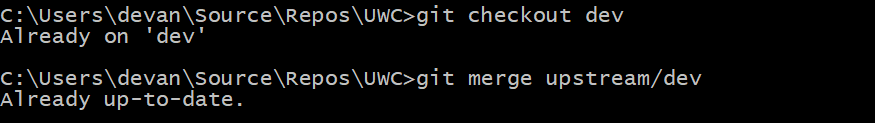
You’ll be prompted to enter your GitHub credentials. This step gathers all the changes that have been checked in to the dev branch in the upstream repo. You’ll only ever need to fetch changes from the dev branch.

Next, check out your local version of the dev branch by executing:

git checkout dev

Finally, merge the upstream changes into your local:

git merge upstream/dev



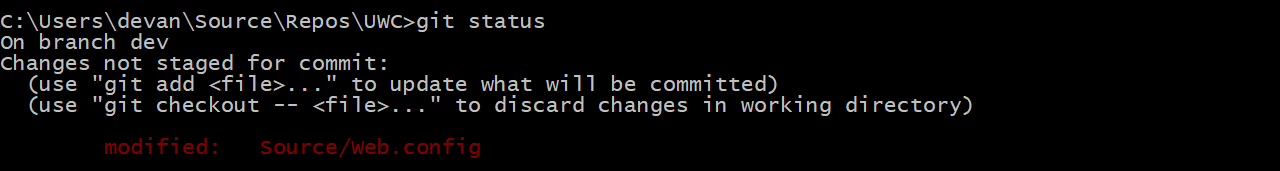
Additional info: https://help.github.com/articles/syncing-a-fork/

**Checking in New Changes**

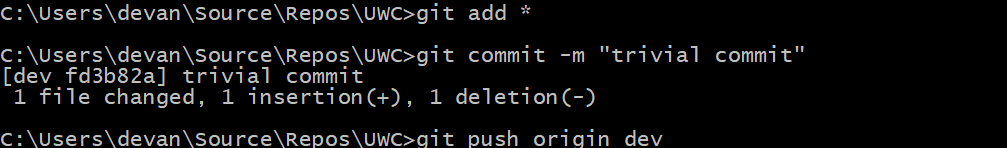
After you’ve implemented a new feature or fixed a defect, the next step is to check those changes into your dev branch, then raise a pull request to have them be merged into the main dev branch. The first part of this process is to commit your changes. Personally, I make use of the git command line, but Visual Studio can do this as well.

GIT CMD

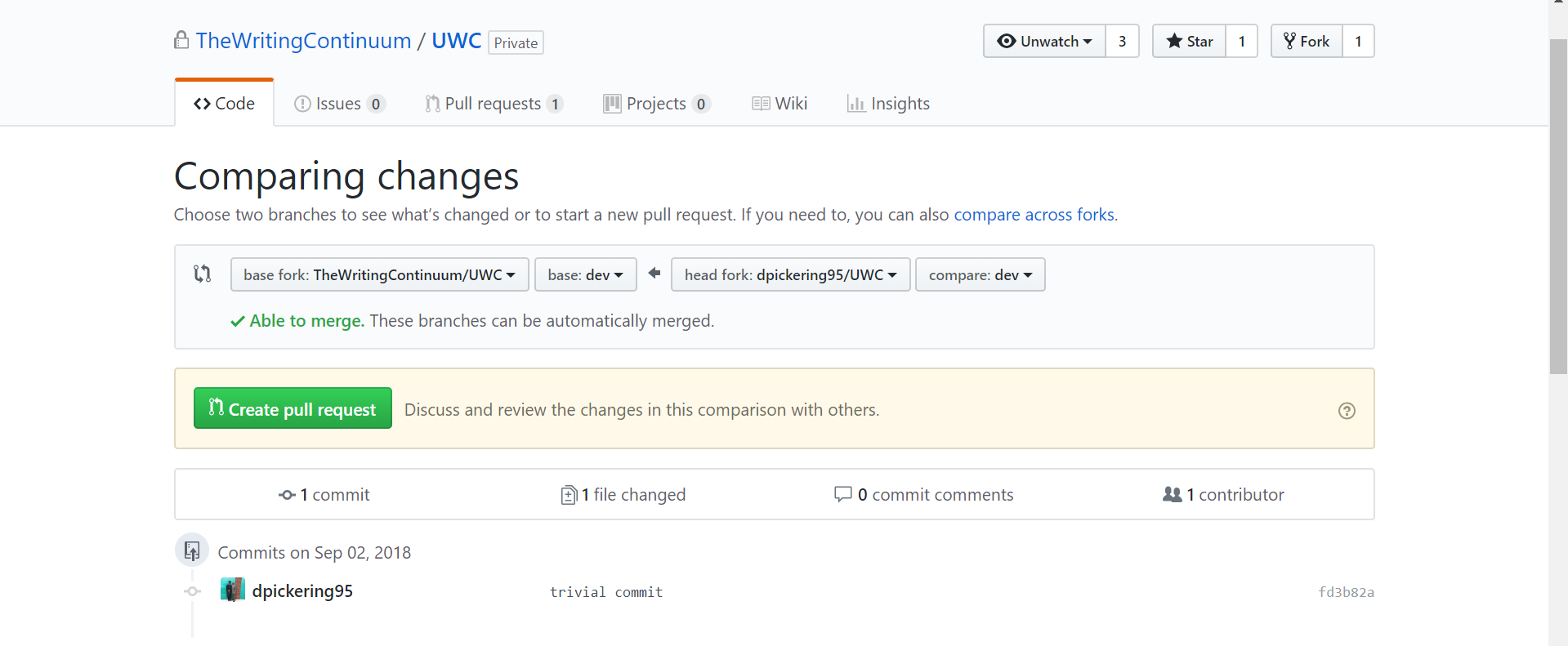
1. Run “git status” – this shows you all the changes you’ve made since your last commit

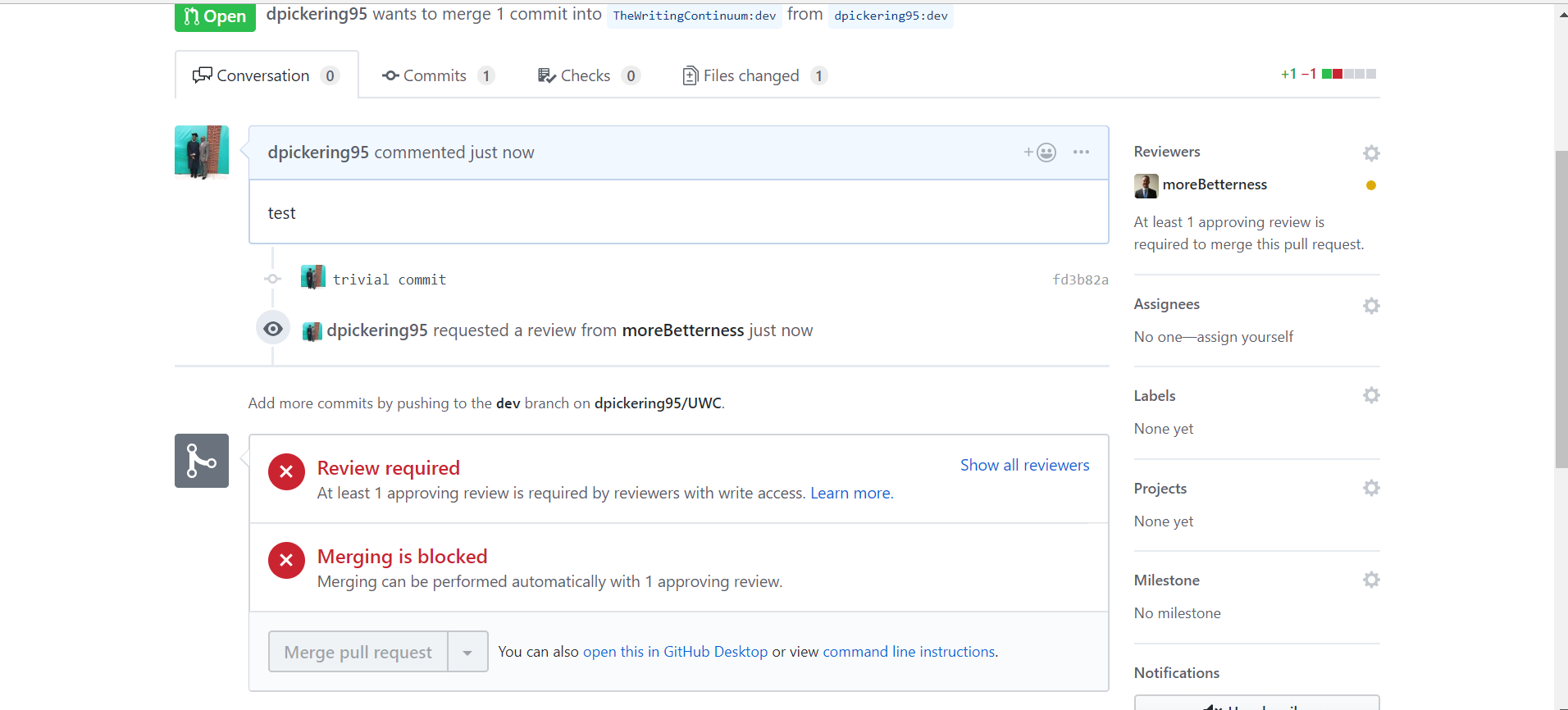


1. Run “git add [path to file]” add files to your commit one by one or run “git add \*” to add all files.
2. Run “git commit -m [commit message]” this commits your changes. Please add a brief message detailing the change that was made.
3. Run “git push origin dev” this pushes your commit



1. Go to your local fork on GitHub. It will recognize that a new commit has been added and prompt you to raise a PR. If you’re ready to check in the changes, raise the PR and request a code review. You’ll notice that merging is blocked until the code has been review.





**SQL Changes**

For any SQL changes (new tables, new triggers, new stored procedures, etc.) manually insert the scripts into the SQL folder in your forked repo and raise a PR to check them into the main repo. When we push changes from qa to prod, we’ll need to execute those scripts on the production db.