Using git for Homework

Terry Sergeant

1 Background

The program `git` is an example of distributed version control software. It is used by programmers for the purpose of tracking changes to a code base, especially for projects that involve multiple programmers.

Prospective employers have provided us with feedback multiple times that it would be nice for CSCI grads to have experience in the use of version control software. To that end we are introducing `git` in the Program Design and Development 2 course and will use it in various capacities in other programming courses.

We will teach command-line use of `git` because it is portable across multiple systems. Although you will be allowed to use a graphical tool when doing your own work we will not provide help with those tools. We recommend you exclusively use the command-line in whatever environment you choose for doing your work.

There are some significant benefits of using version control software for homework assignments:

- You can revert code to a previous state when you make a mess of things (as long as you commit your code strategically)
- Your code will be backed up on a server (assuming you push it regularly). Imagine having a hard drive crash (or you lose your USB drive) ... and not losing your CSCI homework!
- For group projects it becomes much easier to track and manage multiple threads of development.
- You will be able to put `git` as a skill on your resume.
- It will cause you to pay attention to code changes rather than mindlessly editing code.
- If you have multiple computers where you like to work you can share code seamlessly among them.
- It will allow an instructor to make modifications to your code and easily share them with you.
- It will provide you with a portfolio of programs you have written.

2 Setting Up Your Workspace

NOTE: These steps only need to be done once per semester. Most people have a particular computer they like to do their work on. If you have multiple computers you will need to take these steps for each computer:

1. Install `git` on your computer. See `https://git-scm.com/book/en/v2/Getting-Started-Installing-Git` Make sure that `git` will be available on the command-line.
2. On the command line (run Git Bash on Windows to open a Linux-friendly command-line interface) on your computer identify your name and HSU email address to `git` by entering these commands:

```
git config --global user.name "Your first and last name here"
git config --global user.email "your-email@hsutx.edu"
```

NOTE: Please use your HSU email address. Please do *not* use a screen name.

3. On the command line enter this command: `git clone URL` where `URL` is the location of the homework repository for the course in question. (See your homework assignment).

4. Make note of the name of the directory/folder created by the command. That directory/folder contains the homework files and you will do your work for the semester in it. NOTE: For Program Design and Development 2 course the directory will be named: `p2_homework`

5. NOTE: By cloning from the course repository, `git` will automatically designate that URL as a remote named `origin`. So, when working in a repository set up in this way, commands that use the name `origin` are referring to the course repository. NOTE: In git, a "remote" is just a shortcut name for a hosted repository. In this case there was a remote named "origin" created which means you can just use the short, easy-to-type name "origin" to refer to the long URL of the original repository.

To summarize, the steps above accomplish two things:

- create a versioned copy of the homework files (i.e., a repository) on your local computer

- set up a remote name of `origin` to refer to the course repository

3 Setting Up a Hosted Repository

NOTE: These steps only need to be done once per semester. Now you are ready to create a hosted repository where you will a copy of your work. The work you save at this hosted repository is the work that will be graded by the instructor. For Sergeant's courses you will be using `https://bitbucket.org` to host repositories. Follow these steps:

1. If you haven't done so already create an account at `https://bitbucket.org` using your `hsutx.edu` email address. If you have an account but it is associated with a different address you can either create a separate account or add your HSU address as an alias (see settings) to the existing account.

2. Create a repository by importing from the course repository:

   - Log in to your `bitbucket.org` account.
   - In the menu at the top choose: Repositories-¿Import repository.
   - Under "Old repository":
     - choose "git' from the drop-down
     - enter the URL for the course repository (same URL as in previous section)
   - Under "New repository":
     - keep yourself as the owner
     - click in the "Repository name" box ... the name should autofill ... do not change the name
     - make sure the repository is marked as private

- click “Import Repository”

NOTE: When the import operation completes you should be at the overview page of the newly created repository.

3. NOTE: The instructions in this step are subject to change from time-to-time. The goal of the step is to share your newly established repository with your instructor. You should give the instructor permissions to write to the repository. There are a couple of different interfaces BitBucket makes available and I’m not sure which one is the default. (Optional: You can change the interface settings in BitBucket by clicking on the account icon in the bottom left of the screen and then choosing “BitBucket Labs”. Then toggle the “new source browser experience” setting.) Share the newly imported repository with your instructor:
   - From the newly imported repository’s overview page click “Settings” in the menu to the left (you may have to scroll down in the menu).
   - Under “General” click on “User and group access”.
   - Under “Users” enter the email address specified by your instructor (`terrys@gmail.com` for Sergeant) and choose “Write” from the drop-down box. Then click “Add”.
4. Return to the overview page and highlight and copy (to the clipboard) the URL for your repository near the top-right of the screen. This is the URL you will use to refer to your hosted repository. (Depending on your interface you may have “Clone” button that when clicked shows the URL.)
5. Set up a new remote name to refer to your hosted repository on your local computer.
   - Return to the command-line on your local computer.
   - Navigate to the directory/folder and into the repository (i.e., `cd p2_homework` for the P2 course) you cloned earlier and issue this command: `git remote add mine URLofYourRepository` (where URLofYourRepository is the URL you just copied to the clipboard).
   - NOTE: In this command you are informing `git` of a remote and giving it the name `mine`. You can actually call it whatever want but the discussion below will make a lot more sense if you call it `mine`.
   - Issue the command: `git remote -v` to see what URLs `git` has assigned to what names. (`origin` should point to the original course-provided files and `mine` should refer to your hosted copy of those files).

4 Working on Your Homework

Some terms/assumptions in these work flows:

- **local repository**: the copy of your homework repository kept on the computer where you do your homework.
- **hosted repository**: the copy of your homework repository kept in your BitBucket account.
- **base repository**: the copy of the homework repository provided by the instructor and hosted at BitBucket.
- The setup procedures described above should be complete. If that is the case, in your local repository you will have designated the base repository as a remote named `origin` and your hosted repository as a remote named `mine`.

4.1 Work Flow for Typical Homework

In a typical homework assignment you will:

1. *Verify your local repository has been committed.* Use `git status` to see if anything has been changed since the last commit. If there are changes you want to hang onto you should go ahead and commit: `git commit -m "info about changes you are committing"`.

2. *Pull changes (if any) from the base repository.* Use `git pull origin master`.

3. *Work on your homework.* In a typical assignment you should have 5 or more commits. Students often wait until they think they have everything completely finished to commit. You should commit after each success. For example, if you are writing a program that will read words from a large file and count how many times each word appears you might follow this sequence:

   (a) Create a small words file for testing purposes. Add the file to repository using: `git add nameoffile`. Then commit with an appropriate message using
   `git commit -m "Created small words data file for testing purposes"`

   (b) Write code that opens the file, reads each word and displays each word to the screen. If code was in new source file, add file to repository. Commit with message "Opening data file and displaying each word to the screen."

   (c) Suppose this is all you have time for at this sitting. You will push your changes to your hosted repository using `git push mine`.

   (d) Write a method to search an array for a word. Document the method using JavaDoc conventions. Commit with message "Wrote method to search array for a given word."

   (e) Modify reading code to store words into array only if word not found in existing list. Commit with appropriate message.

   (f) Add code to count recurrences of words. Commit with appropriate message.

   (g) Suppose this is all you have time for at this sitting. You will push your changes to your hosted repository using `git push mine`.

   (h) Add code to track time elapsed in program and to display various stats/results. Commit with appropriate message.

   (i) Verify program is adhering to all documentation requirements and Java programming conventions. Switch to large data file and test. Commit with "Completed Homework to Count Word Frequency".

4. *Make sure* `git status` *is clean.* Use `git status` and verify that files you want to be staged for commit have been added with `git add`. If needed do a final commit. In the commit message you should indicate that the homework assignment is completed. Be sure to specify which assignment it is you completed. This will help the instructor navigate your repository when grading.

5. *Push all changes to your hosted repository.* Use `git push mine`. Verify that the repository you pushed matches what you thought you pushed by logging in to your BitBucket account and viewing the source of the most recent changes you made.

6. *Wait patiently for work to be graded.* The instructor will pull code from your hosted repository when grading. You can, of course, start on the next homework assignment before grading is complete.

4.2 Some Other Cases

The work flow suggested above will cover many cases. Here are some possible scenarios you may encounter and how to deal with them.

- *I committed with a message saying my homework was done and realized I forgot xyz.* You can get a "redo" on a previous commit by committing again with using `git commit --amend -m "your commit message"`. If you had already pushed the previous commit to you hosted repository you can force the modified commit using: `git push mine --force`

- *My instructor announced that the files in the base repository had an error that is now fixed . . . but I already started my homework!* Try this:

  1. Use `git status` and `git commit` to make a snapshot of your current working repository.
  2. Use `git pull origin` to import the updates.
  3. Continue working.

- *My @!#!$ instructor left a comment and posted it to my bitbucket account while I was working on the next homework assignment. When I tried to push my new homework to bitbucket I was told "Updates were rejected because the remote contains work that you do not have locally."* Do this:

  1. Use `git status` and, if necessary, `git commit` to make a snapshot of your current working repository. You've probably already done this if you are trying to turn in your homework.
  2. Use `git pull mine master` to pull/merge the comment your instructor left with your new homework.
  3. Use `git status` to verify there is nothing left to commit/merge.
  4. Use `git push mine` to turn in your new homework.