1 Submitting your assignment

The purpose of removing a friendly submit script from the CS24 assignments is to get you familiar with the git (and by extension version-control) development cycle. As discussed in class, we will be following the *Feature Development Workflow* [1].

This type of development keeps a project's history on its main, or `master` branch, while new features are added on separate branches. The main idea behind this design is to maintain the `master` with bugless code, and develop in isolation on feature branches.

Let's take this into practice. I am expecting you to find and execute the instructions to successfully clone the repos onto your (`local`) computers.

1.1 Feature branching

For either assignment, you will be creating a feature branch, copying and modifying the template assignment, then merging back to the master branch.

1. Navigate into the `time-capsule` directory
2. Create a branch
   (a) `git checkout -b myfeature master` (shorthand), or:
   (b) `git branch myfeature; git checkout myfeature`
3. Configure your assignment (§2)
4. Merge your branch
   (a) Either submit a *pull request* : Through the online bitbucket interface, submit a pull request. This will ask everyone on the team to review the changes you made to the code before it gets merged onto the master. This prevents you from accidentally adding bad code to the master, or
   (b) `git checkout master; git merge myfeature`
      i. Note that you can use `checkout` to switch between branches and spy on your classmates' branch.

Remember, to figure out where you are, you can always run `git status` (Figure 1).

[1] `https://www.atlassian.com/git/tutorials/comparing-workflows/feature-branch-workflow`

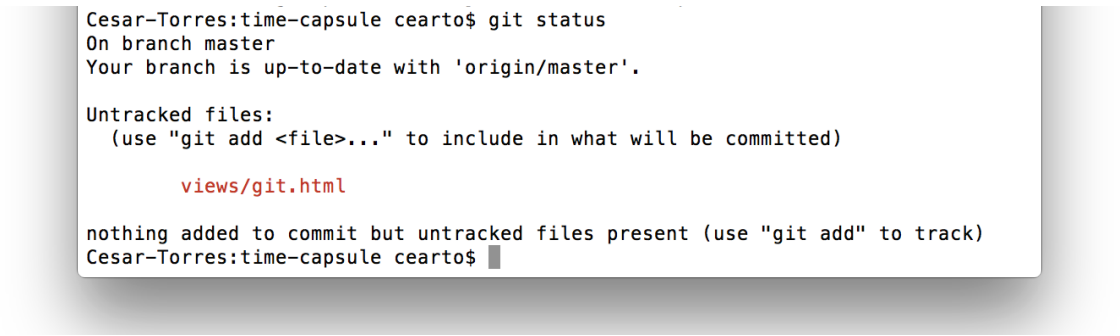

Figure 1: git status is your friend. Use it often. Note that the first line tells you the branch you are on, how old it is (e.g. 2 commits behind master [since someone else has merge their branch after your started yours), the things that are on your stage, and off your stage (untracked).

2 Configuring your recipe

Recipes are currently composed of three elements.

1. webpage in `\views\topic\recipe.html`
   (a) Create the directories using `mkdir` if they don't already exist
   (b) *To edit:* Open the html file using a text editor. Replace relevant text.
   (c) *To view:* Open your terminal, navigate to the root directory of the repo of interest. Run `python -m SimpleHTTPServer`. This starts a server on port 8000 of your computer. Point your browser to localhost:8000. Click on views, ..., down to your recipe's html page to view your changes. The server automatically loads the most recent code. Just refresh the browser any time you make an edit to the document.
2. .java files
   (a) Don't commit your .class files! Just your .java is
   (b) Lives in `\code\recipe`
   (c) In your html page, find the anchor tag `a` that links to the code. Update the `href` property with the new path: `\code\recipes`.
3. your folder
   (a) Lives in `\partners\name`
   (b) Symbolically link your html pages to this folder. This "copies" the html page to your personal folder, without actually copying the file. This is what happens under-the-hood when you create shortcuts on your desktop. If you make a change to your html page, the "copy" will also update.
   syntax: `ln -s` *pathtofiletosymlink dst*
   if you are in the partners directory, you can just run `ln -s` *pathtofilesymlink* . where the period (.) refers to the current directory.

3 Configuring your Q/A

This follows the same pattern except for some path names:

1. webpage in `\views\week\topic.html`
2. pdf files
   (a) Lives in `\views\week`
   (b) Generate the pdf of your questions using LaTeX, and place the file in the above location. Don't commit build files! (no .out, .syntex, etc.., only .pdf) Keep the answers to the questions in a separate file. You'll submit this in a couple of weeks.
   (c) In your html page, find the anchor tag `a` that links to the questions pdf. Update the `href` property with the new path: `\views\week\questions.pdf`.
3. your folder
   (a) Lives in `\groups\name`
   (b) Symbolically link your html pages to this folder.