15-150: Git Handout System

Last Revised: February 21, 2012

1 Introduction

We will handout homework assignments in this class through a read-only `git` repository. This will let us smoothly distribute starter code to you, including the evolutions of the libraries we'll be using for this class. Since `git` supports fully distributed version control, this will also let you take advantage of version control locally to manage the iterations of your solution.

`git` is available on the UNIX timeshares and for modern operating systems, so you can choose to work on your local machine or remotely on `unix.andrew.cmu.edu`. You should feel free to work on your assignments locally, but we strongly suggest you work remotely on `unix.andrew.cmu.edu`. We will grade your assignments with the version of SML installed there, and we know that the version of `git` installed there works.

`git` has a huge user base, so it's easy to find out information about it. We're only using a small subset of the features it offers, so a lot of that information will not be relevant. That said, the following are good places to learn more about both `git` and version control in general.

- `http://en.wikipedia.org/wiki/Version_control`
- `http://en.wikipedia.org/wiki/Git_(software)`
- `http://git-scm.com/`
- `http://git-scm.com/documentation`

2 Using Our `git` Repository

2.1 Initial Check Out onto AFS

To check out the repository initially, log in to AFS either through a cluster machine or by SSH to `unix.andrew.cmu.edu`. Change to the directory that you'd like to have your work in and run

```
git clone /afs/andrew.cmu.edu/course/15/150/handout 15150
```

Make sure that the directory you're working in is private to you. A good way to do this is to make it a subdirectory of the `private` directory in your AFS home directory.

This will create a subdirectory called `15150` and populate it with the files in the repository. You can replace `15150` with a directory name of your choosing in the command above.

2.2 Initial Check Out via SSH

It's also possible to check out the repository to a local machine—like your laptop—using SSH. Be careful, though: you very likely only want to have one copy of the repository, and we will grade your submissions against the version of SML installed on the UNIX timeshares. Setting up SML locally can be tedious, as well, so we strongly recommend that you work on the UNIX timeshares.

That said, the process to make a clone via SSH is basically the same, except you use the command

```
  git clone
   ssh://AID@unix.andrew.cmu.edu/afs/andrew.cmu.edu/course/15/150/handout
   15150
```

instead of the one above, where `AID` is your andrew ID.

2.3 Updates

To update your copy of the repository, change into the `15150` directory you created by running `git clone` and run

```
git pull
```

This command is how you will receive new assignments, solutions to previous assignments, and possible corrections to existing assignments.

2.4 Local Commits

You can make local commits by running[1]

```
git commit -a
```

The `commit` command preserves the state your working copy, so you could come back to it later if need arises. Therefore, you should make a commit whenever the collection of files you're working on reaches a state that you think you might want to get back to later. Since you'll be working in a strictly local repository, there are no other developers to complain if you commit code that doesn't compile, so: commit early, commit often.

`git` will ask you for a commit message each time you commit. It's tempting to leave this blank or trivialize it, but you should avoid the urge. Meaningful commit messages are invaluable when you're going through previous revisions looking for a working bit of code that you lost.

When you create new files, `git` doesn't know about them. If you want them under version control, you need to add them manually by running

```
git add file.sml
```

In general, the updates you pull for each new assignment will have empty files with appropriate names; these are already under version control and you won't need to add them again.

It is important to note that since this will be a strictly local repository, if you delete it or your computer crashes *you will lose all your work*. There is no server that you're pushing your commits to. This is not a substitute for conventional backup.

2.5 Assignment Submission

You will *not* be using `git` to submit your solutions. This is a read-only repository. Detailed instructions about submitting assignments will be provided with them..

[1] The option `-a` tells `git` to commit changes in all the files that have been modified or deleted. Without this option, `git` expects a list of files to commit.