Git Version Control for CS 5600

For homeworks in this class you will be using the Git distributed version control system. Git works with ***repositories*** and ***local copies***. A repository is a central copy of the files, while a local copy is just what it sounds like - a copy of the repository on your machine that you can edit and work with.

Theory of Operation

You ***clone*** a repository to make a local copy; you then synchronize the copy with the repository by ***pushing*** changes back, and ***pulling*** changes that have been made in other local copies (by yourself or your lab partner).

Your local copy has revision control features of its own - you ***add*** files to put them under Git control, and ***commit*** changes to files. Only committed changes can be pushed to the repository.

Specifics for CS 5600

Each student will be given repositories named ***name***-hw1, ***name***-hw2 etc. where ***name*** is your Blackboard / MyNEU user ID. Repositories will be initialized with the files you need to complete an assignment. You will have read/write permission to your own repository; when you request permission to work on a homework with another student, you will each be given permission to the other's repository for that homework. (E.g. if Fred Smith works with Joe on homework 1 and Bob on homework 2, then Joe will have access to smith.f-hw1, and Bob will have access to smith.f-hw2)

Authentication for the Git repository we are using relies on SSH cryptographic keys; if you go to "My Grades" on Blackboard there should be a 0-point grade; your SSH key will be contained in the comments associated with that grade.

To work with the Git server from a linux machine, you need to make a copy of this file named **~/.ssh/id_rsa**, and set its permissions correctly with the command **chmod 400 ~/.ssh/id_rsa**. Note that this is a text file, so you should be able to copy it from your browser and paste it into a terminal window:

```
$ cat > ~/.ssh/id_rsa
-----BEGIN RSA PRIVATE KEY-----
MIIEowIBAAKCAQEA1WHDHddFFn...          <-- pasted text
 ...
-----END RSA PRIVATE KEY-----
^D                                     <-- control-D to finish
$ chmod 400 ~/.ssh/id_rsa
```

If you use SSH keys already, there is a section at the end of this document showing how to add the new SSH key for Git while keeping your old key for everything else.

Assignments will be submitted by 'pushing' your final changes to the repository; the last revision submitted before the deadline will be retrieved for grading. Teams should pick one member's repository and use it; this repository should be identified by creating a file named “SUBMITTED_HERE” in that repository and pushing it to the server.[1]

[1] If you're a Git expert, it is possible to copy changes back and forth between multiple users' repositories. Please do not do this - it is easy to run into problems, and I will not provide help if you do so.

Git Commands

`git clone gitolite@xevious.ccs.neu.edu:smith.f-hw1`

- make a local copy of your homework 1 repository, assuming your Blackboard ID is smith.f

`git push`

- copy local commits to the repository

`git pull`

- get changes from the repository to your local copy

`git add` ***file***

- add file to the repository

`git commit -m '`*`message`*`' -a`

- commit changes in the local copy. Git forces you to add a commit message like this.

`git status`

- list locally changed files, etc.

If you've used subversion (SVN) before, this is mostly the same, except that there's an extra local step - rather than just committing changes to the repository, you commit them locally and then push the most recently committed change to the repository. It doesn't make much of a difference if you're working on your own, but if you're working on a team it means that you can commit frequently (so you can retrieve an earlier copy if you mess up) but only push changes to the repository when you're willing to let your teammate see it.

Multiple SSH keys

Advanced Unix/Linux users may use SSH and already have SSH keys on their machines that they don't want to replace. In this case you'll want to install the CS5600 key as a secondary key, only used for Git.

Given a key in a file named '`smith.f.key`', you'll need to copy it to `~/.ssh` and set its permissions to 400. Then add the following lines to `~/.ssh/config`:

```
host cs5600
     user gitolite
     hostname xevious.ccs.neu.edu
     identityfile ~/.ssh/smith.f.key
```

If you have to create ~`/.ssh/config`, you will need to set its permissions to 600. Now you can clone your repository like this:

```
git clone cs5600:smith.f-hw1
```

and you're done.

Git Example

Here is the sequence of operations if Alice and Bob work together, with a repository starting out with 2 files, “file1” and “file2”. Note that in real life things will probably be much more complex. For instance Alice may work on two different machines, and so she would clone the repository on each machine, and then push her changes back to the repository when she's done working on one machine, and pull changes when she sits down at the other machine.

Alice	Alice's local copy	Repository	Bob's local copy	Bob
Email instructor: “working with Bob”				Email instructor: “working with Alice”
		file1, file2		
`git clone ...:alice-hw1`		.		
	file1, file2	.		`git clone ...:alice-hw1`
	.	.	file1, file2	
<create file3>	.	.	.	
`git add file3`	.	.	.	
`git commit -m 'add file3' -a`	file1, file2, file3	. .	. .	
`git push`	. .	file1, file2, file3	. .	
	. .	. .	file1, file2, file3	`git pull`
	. .	. .	file1, file2', file3	Edit file2
	.	.	.	`git commit -m 'edit file2' -a`
	. .	file1, file2', file3	. .	`git push`
`git pull`	file1, file2', file3	. .	. .	

At this point if Alice and Bob are finished, they should each log into Blackboard, and under “Homework 1” enter a comment that their work is in repository 'alice-hw1', upload any additional materials, and submit. Now the instructor can do a 'git clone' of their repository, giving access to the latest copy as well as all the intermediate versions.