SSH Keys & Revision Control

This course relies heavily on the Git revision control system. There are many revision control software packages, but Git has some advantages in our online classroom environment.

Your first task in this course will be to access the Git repository for your term. To do that, you'll need to set up SSH keys that will grant you access. You may already have SSH keys set up on your computer. If you do, you may use the existing keys instead of generating new ones.

Objectives

In this assignment you will learn how to:

- Create SSH keys.
- Install and configure Git.
- Clone Git repositories.
- Manage content in a Git repository.

Reading and Resources

This document will walk you through the bare minimum to complete the assignment, but is not intended to teach you everything about SSH, public key encryption, or Git. The latter is the most important take-away, as you will be using Git to submit all of your source code for this course.

Scott Chacon, one of the founders of GitHub, has written and open-sourced the book *Pro Git*, which covers everything you could want to know about Git and more:

http://progit.org/book/

Instructions

In this document, when you see a green/yellow box, it will contain a command that you will need to issue in Terminal, like so:

```
ls -al
```

You will notice that console prompts, such as `$`, `>`, or `#`, have been left out.

Example output from commands will be in blue boxes:

```
drwx------   6 rosborne  staff    204 Mar 18 17:44 .
drwxr-xr-x+ 55 rosborne  staff   1870 Mar 18 19:01 ..
```

Finding Your SSH Keys

Git is a console application that runs from the command line, so you'll need to open Terminal. SSH keys are stored in a locked-down directory in your home folder. First, make sure you are in your home folder:

```
cd
```

Remember that the `cd` command by itself returns you to your home folder. Now look to see if you have a `.ssh` folder and keys inside of it (note that the folder name begins with a period):

```
ls -al .ssh
```

When I run this, I see:

```
drwx------   6 rosborne  staff    204 Mar 18 17:44 .
drwxr-xr-x+ 55 rosborne  staff   1870 Mar 18 19:01 ..
-rw-r--r--   1 rosborne  staff    403 Mar 18 17:44 authorized_keys
```

```
-rw-------    1 rosborne  staff   1675 Oct  6 02:27 id_rsa
-rw-r--r--    1 rosborne  staff    403 Oct  6 02:27 id_rsa.pub
```

If you already have SSH keys you will have both the `id_rsa` and `id_rsa.pub` files. You may skip past the next part if you do.

Generating SSH Keys

Make the `.ssh` directory:

```
mkdir -p .ssh
```

Change into the new directory:

```
cd .ssh
```

The `ssh-keygen` command generates SSH keys. It will prompt you for information if you don't give it any, but it goes faster if you specify the correct arguments (note the capitalization):

```
ssh-keygen -t rsa -C 'you@fullsail.edu' -N '' -v
```

With that command, you will generate RSA-style keys for your email address that are not protected by a password. You should now have your `.ssh` folder and the `id_rsa` files within it. Make sure:

```
ls -al
```

Compare the permissions on the `.ssh` directory and the `id_rsa` files to the permissions in the example in the previous section. Ensure the permissions are correct, or SSH clients will think you've been hacked and won't use the keys!

Install Git

Close and quite the Terminal application before you install Git. Git can be downloaded from:

http://www.git-scm.com/

Find the OSX installer. Either of the 64- or 32-bit versions will work for the small files and repositories we will use in this course. Git is distributed as a PKG file instead of as a standalone App, so ensure you run the installer—don't just drag it into your Applications folder.

If you have Xcode and GCC properly installed, you can also install Git with MacPorts, Brew, or Fink. Don't worry about it if you don't know what those are.

Configure Git

Once Git is installed, open a new Terminal window and ensure it's working correctly:

```
git --version
```

You will now need to set up Git with some basic information about yourself. This is easiest done through the command line:

```
git config --global user.name 'Firstname Lastname'
git config --global user.email 'you@fullsail.edu'
```

There are some additional settings that are optional but may help you:

```
git config --global core.compression 9
git config --global core.editor nano
git config --global color.ui auto
git config --global merge.conflictstyle diff3
```

When you have entered all of the configuration information, you should verify it by having Git list it out for you:

```
git config --global --list
```

You should see all of the information you just entered.

Deliverables: Completing the Assignment

To complete the assignment, email your public key (`id_rsa.pub`) to your instructor.

Don't worry! Even though it's a key, your public key is meant to be shared. You aren't compromising anything about your security or identity by giving it away. Your secret key—the one without the `.pub` extension, should never, ever be given away or transferred. You can and should make backup copies of both, but make sure your backups are secure.

The easiest way to do this is to copy the key to your Desktop first:

```
cp ~/.ssh/id_rsa.pub ~/Desktop
```

After the Assignment

Don't try this part until after you've heard back from the instructor about your SSH keys.

Once it has been confirmed that your key has been added to the Git repository you can clone it. Repositories are hosted on the `wddbs.com` domain and all follow the same format:

Gitosis Authentication	Course	Term
git@wddbs.com:	bdf	*yymm*

The term is the two-digit year followed by the two-digit month. For example, if you were in term `1105`, your repository would be:

```
        git@wddbs.com:bdfXXXX
```

Do not substitute your email address for the `git@wddbs.com` part. That's not really an email address, it's just how the authentication system for Git, called Gitosis, works.

Cloning the repository can be done from the command-line. You will want to clone into your MAMP web root:

```
cd /Applications/MAMP/htdocs
git clone git@wddbs.com:bdfXXXX
```

That should make a new folder in your web root named `bdfXXXX`. When the cloning is finished the folder will already contain a number of files. You should create a new folder inside that `bdf1105` folder with your name in `lastname_firstname` format and change into it:

```
cd bdfXXXX
mkdir lastname_firstname
cd lastname_firstname
```

All of your development should happen in this folder. You will be instructed in future assignments how to name the folders and files that will go in there. For now, to show that you have Git repository access, create another folder and file:

```
mkdir 01_revision_control
touch 01_revision_control/access_confirmed
```

You can then add, commit, pull, and push your new content:

```
git add 01_revision_control
git commit -m 'Confirming repository access'
git pull
git push
```

Until you are told otherwise, **do not** change any files in the repository outside of your directory. If you need to, copy files into your directory and make changes there. This will prevent problems until you know how to work in a repository with your teammates.