GitHub

Application Note for GitHub Revision Control System

Robert Miller
ECE 492 • University of Alberta • 19 March, 2013 • Rev 1

GitHub

Application Note for GitHub Revision Control System

HOSTING SERVICE FOR SOFTWARE DEVELOPMENT PROJECTS

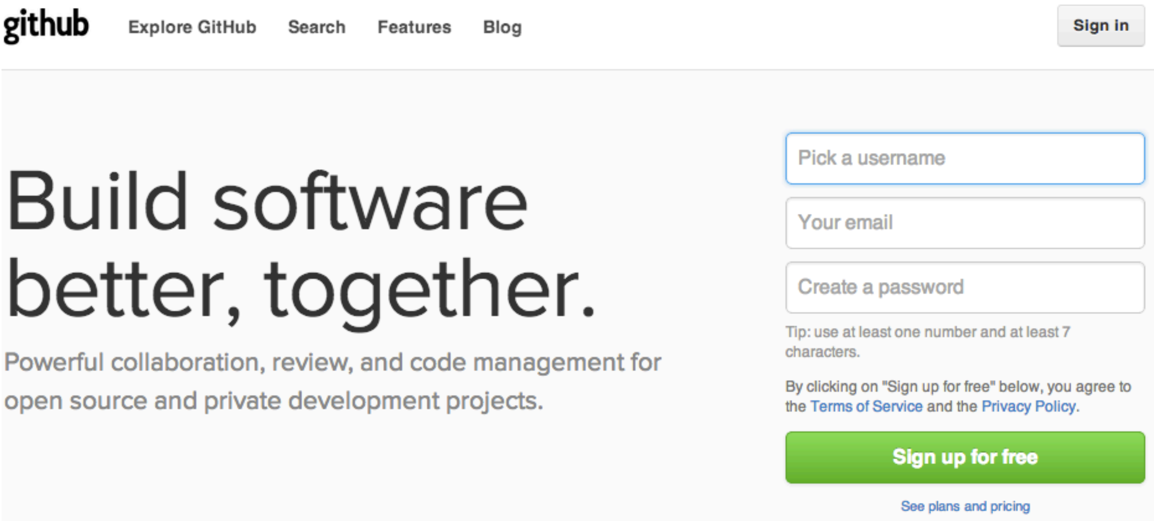
GitHub is one of the most popular open source code repository sites. It is compatible with Windows, Mac, and Linux. If you are a Terminal Junkie, you can use GitHub completely from the terminal. Otherwise you have the option of installing the native application with a full GUI. Both free and paid plans are available to host your online software repository. Version control is essential when working on software development projects, especially with multiple team members contributing to the source code. This document will walk you through setting up a GitHub account, creating a new repository, granting members access to your repository, and demonstrate how to use the basic Git commands.

"At the heart of GitHub is an open source version control system (VCS) called Git. Created by the same team that created Linux, Git is responsible for everything GitHub related that happens locally on your computer." - github.com

If you have never heard of Git, you may wish to review the basics before continuing with this guide > http://git-scm.com/book/en/Getting-Started-Git-Basics

1. Creating a GitHub Account

GitHub is cross platform and you can even choose to use terminal access or the standalone application with full GUI. Before we get into that, we will create a new account on the GitHub website. Go to https://github.com and sign up for a free account. A free account lets you create unlimited public repositories and lets you add unlimited public collaborators. Note that with a free account, anyone can see your code in your repositories, hence the open source. If you have a problem with this, you will need to pay a monthly fee for a private repository.



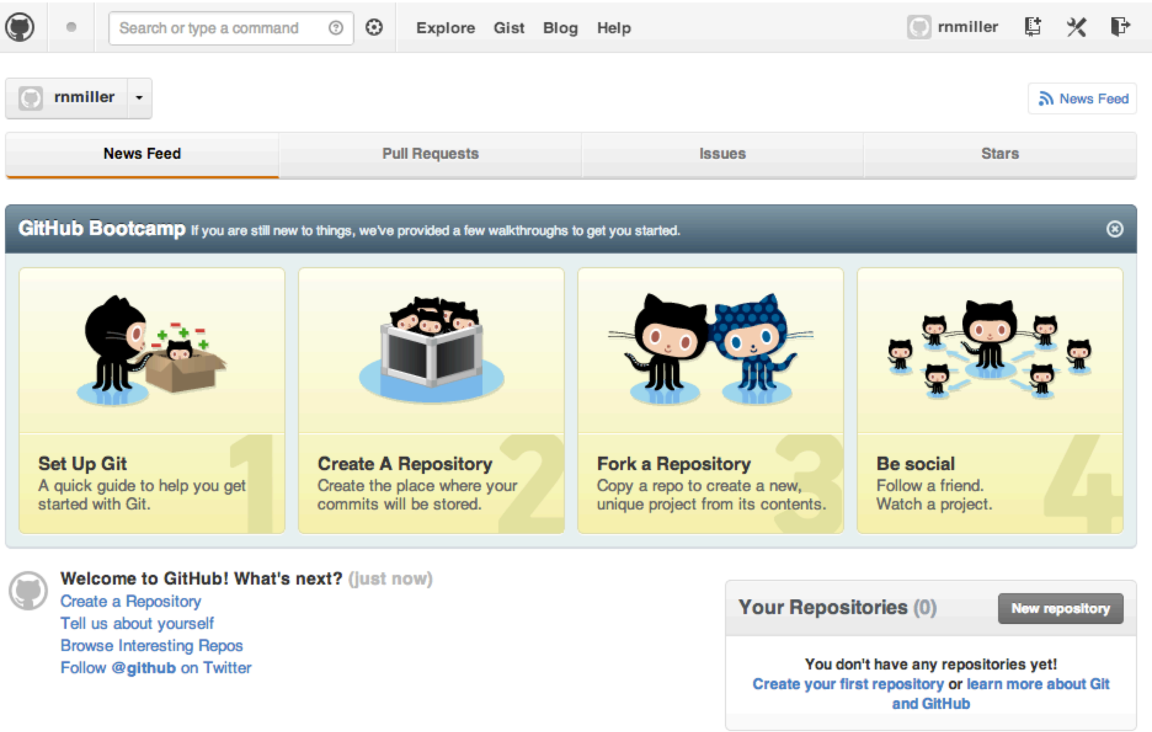
Build software better, together.

Powerful collaboration, review, and code management for open source and private development projects.

Tip: use at least one number and at least 7 characters.

By clicking on "Sign up for free" below, you agree to the Terms of Service and the Privacy Policy.

Once you fill in the three basic fields and click ‘Sign up for free’ you will be immediately brought to your GitHub home screen! Wasn’t that easy?



GitHub Bootcamp If you are still new to things, we've provided a few walkthroughs to get you started.

Set Up Git
A quick guide to help you get started with Git.

Create A Repository
Create the place where your commits will be stored.

Fork a Repository
Copy a repo to create a new, unique project from its contents.

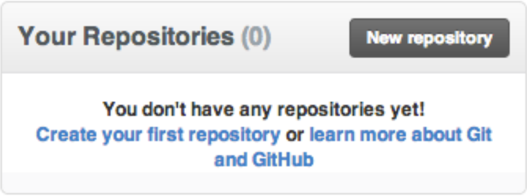
Be social
Follow a friend.
Watch a project.

Welcome to GitHub! What's next? (just now)
Create a Repository
Tell us about yourself
Browse Interesting Repos
Follow @github on Twitter

Your Repositories (0)

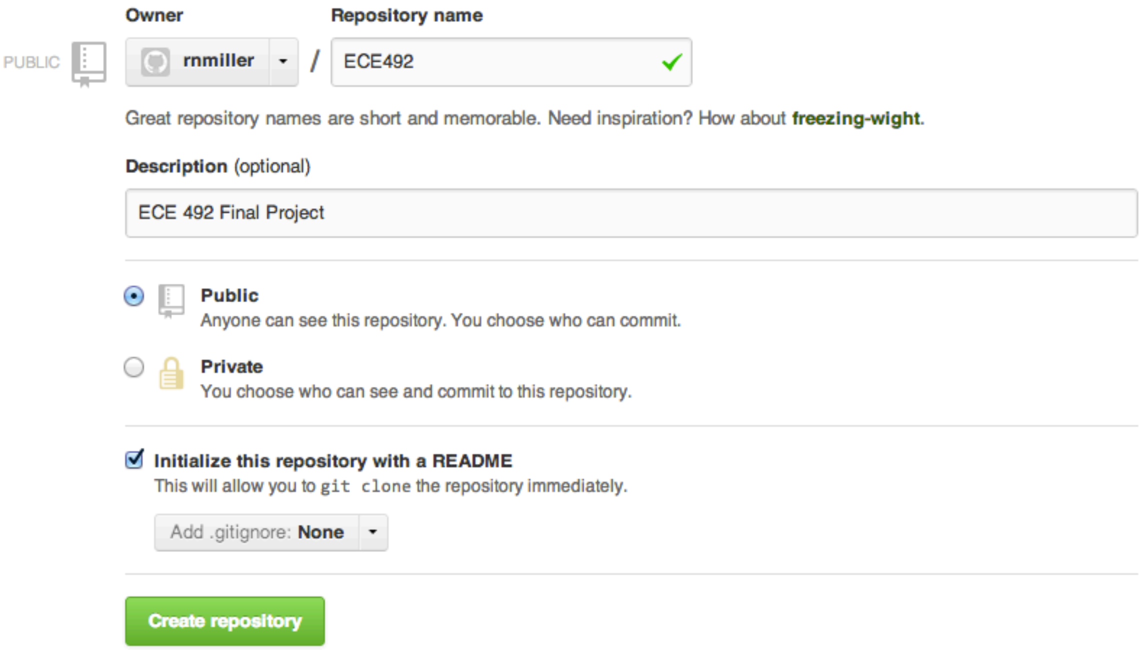
You don't have any repositories yet!
Create your first repository or learn more about Git and GitHub

Now it is time to create a new repository for your software development project. Click on the 'New Repository' button in your GitHub home page.



You don't have any repositories yet!
Create your first repository or learn more about Git and GitHub

Pick a repository name and give it a description. Make sure to check the box 'Initialize this repository with a README' as this will allow you clone the repository to your computer right away. Click on 'Create Repository' to finish.



Owner: rnmiller / Repository name: ECE492 ✓

Great repository names are short and memorable. Need inspiration? How about **freezing-wight**.

Description (optional): ECE 492 Final Project

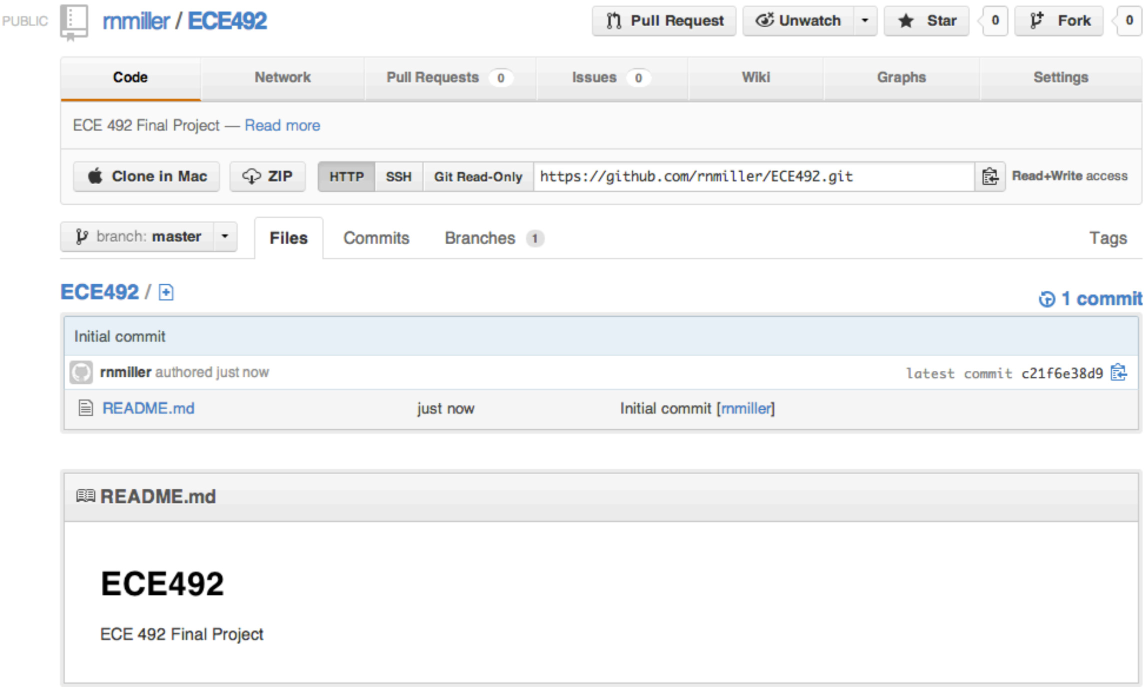
● **Public**
Anyone can see this repository. You choose who can commit.

○ **Private**
You choose who can see and commit to this repository.

☑ **Initialize this repository with a README**
This will allow you to `git clone` the repository immediately.

Add .gitignore: **None**

Congratulations! You have just setup your first software repository on GitHub. The next screen you are directed to is the home page for your repository. This page gives you access to a lot of information. You can track commits, browse the archived code, create a Wiki and track issues.



ECE 492 Final Project — Read more

https://github.com/rnmiller/ECE492.git

ECE492 /

Initial commit

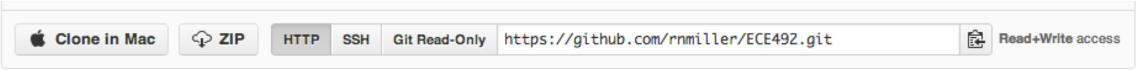
rnmiller authored just now latest commit c21f6e38d9

README.md just now Initial commit [rnmiller]

ECE492

ECE 492 Final Project

There are multiple ways to clone your repository to your computer. At the top of the repository home page you will see the following bar:

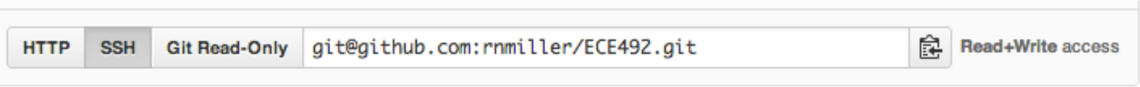


https://github.com/rnmiller/ECE492.git

If you are on a Macintosh or Windows PC, the first button 'Clone' will allow you to do a clone through the stand alone native application. If it is not already installed on your computer, it will prompt you to download it. If you are on a Linux computer (as in the lab) you will not see this option. The native application is user friendly and hence I will not discuss it further here.

The 'Zip' button will allow you to download the entire repository. We will not be using this feature at this time.

You have two options to access the repository from the terminal, HTTP or SSH. It has been suggested that HTTP is very slow and should only be used if the git port (9418) has been blocked. We will use the SSH method which works fine from the ECE 492 lab computers. If you click on the SSH button, you will see the git@github.com:*yourusername/repository.git* path which will be used later to clone the repository. First we will need to setup Git and your SSH keys.

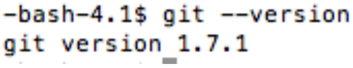


```
git@github.com:rnmiller/ECE492.git
```

2. Set Up Git

If you are installing Git (Terminal or native app) on one of your personal computers, you can follow the instructions on the following page > https://help.github.com/articles/set-up-git < which recommends using the HTTPS method for terminal users. I have had no issues using HTTPS when there is only one user. I have however had access issues when using multiple collaborators and therefore I recommend that you use the SSH method. If you prefer to use a GUI and are installing on your personal computer, the native app is self explanatory to install.

If you are going to use Git on the ECE 492 Linux stations, you are in luck! Git is already installed on these stations and can be confirmed by typing the following command at the terminal:



```
-bash-4.1$ git --version
git version 1.7.1
```

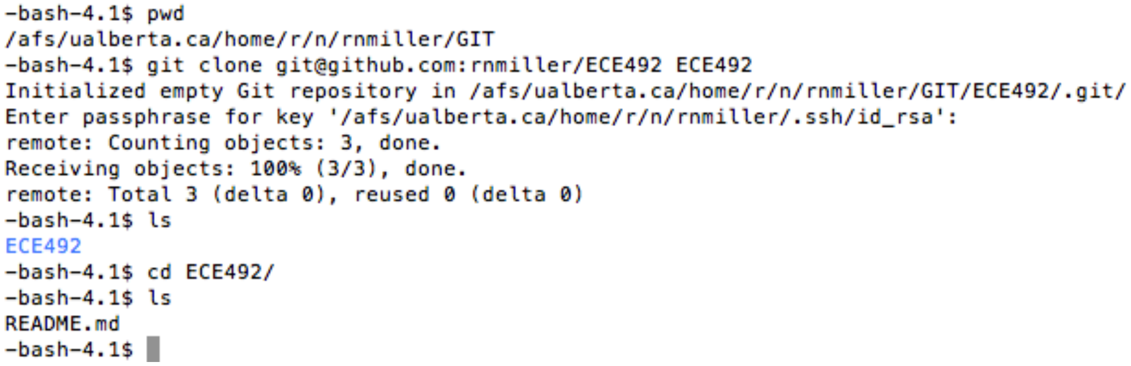
As of March 13, 2013 the latest version of Git is 1.8.2 however version 1.7.1 will work just fine with GitHub.

3. Generating SSH Keys

Since we are going to use the SSH method, you will have to generate an SSH key and add it to your Github account. You can consult the help page at https://help.github.com/articles/generating-ssh-keys#platform-all and then select your corresponding operating system.

4. Cloning a Repository

To clone a repository, you will use the following command (for SSH path):
"git clone git@github.com:*username/repository.git foldername*"
Here we are telling the program Git to clone the repository specified by the SSH path into the specified folder name. In the following example, I have created a folder called GIT in the root of my home directory. I then enter the command to clone the repository I created earlier, into a new subfolder called ECE492. I am then prompted to enter my password for my SSH key. Once accepted, Git clones the repository as specified. We can now see that the new folder was created and contains all the files in the repository (just the README.md at this time).



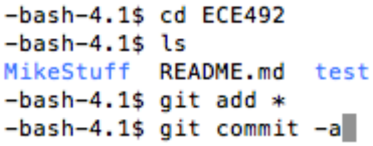
```
-bash-4.1$ pwd
/afs/ualberta.ca/home/r/n/rnmiller/GIT
-bash-4.1$ git clone git@github.com:rnmiller/ECE492 ECE492
Initialized empty Git repository in /afs/ualberta.ca/home/r/n/rnmiller/GIT/ECE492/.git/
Enter passphrase for key '/afs/ualberta.ca/home/r/n/rnmiller/.ssh/id_rsa':
remote: Counting objects: 3, done.
Receiving objects: 100% (3/3), done.
remote: Total 3 (delta 0), reused 0 (delta 0)
-bash-4.1$ ls
ECE492
-bash-4.1$ cd ECE492/
-bash-4.1$ ls
README.md
-bash-4.1$ 
```

4. Adding Collaborators to your Repository

Anyone can download and clone your repository, but only those who you grant access to can contribute or 'push' changes up to your repository. To add a contributor, navigate to your repositories GitHub home page > Settings > Collaborators. Here you enter a collaborator's GitHub user name and press add.

5. Saving your Progress

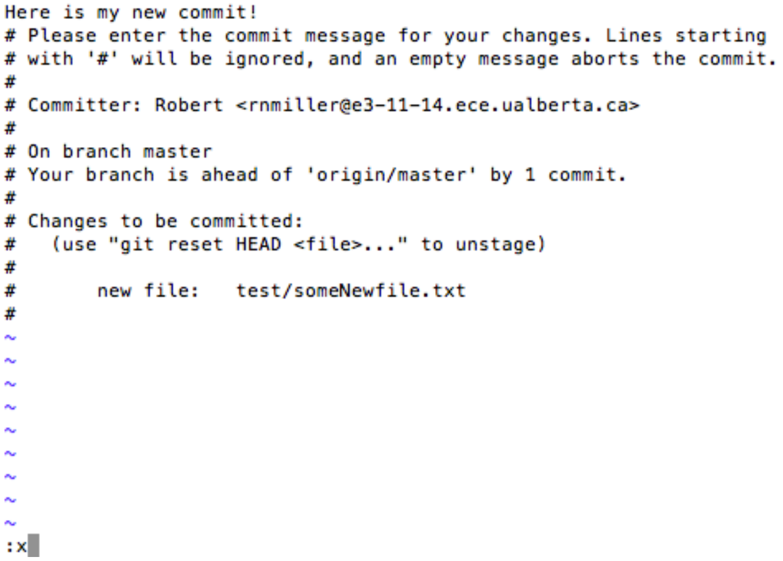
Once you have made some changes to your code base and want to save your progress, navigate in terminal to the top level folder of your local repository and use the command "git add *" this will add all files in this folder and it's subfolders to be 'tracked' by Git. Next use the command "git commit -a" which will commit all tracked changes to your local repository.



```
-bash-4.1$ cd ECE492
-bash-4.1$ ls
MikeStuff  README.md  test
-bash-4.1$ git add *
-bash-4.1$ git commit -a
```

After pressing enter, the Vi editor will open for you to make comments on this commit. You can search google for a list of Vi commands if you are unfamiliar with it. I will explain the minimum commands.

Press "i" to begin inserting text, and enter your comments. When you are done, press "esc" to exit insert mode. Then press ":" to enter command mode and finally type "x" and then enter to save and exit Vi.



```
Here is my new commit!
# Please enter the commit message for your changes. Lines starting
# with '#' will be ignored, and an empty message aborts the commit.
#
# Committer: Robert <rnmiller@e3-11-14.ece.ualberta.ca>
#
# On branch master
# Your branch is ahead of 'origin/master' by 1 commit.
#
# Changes to be committed:
#   (use "git reset HEAD <file>..." to unstage)
#
#       new file:   test/someNewfile.txt
#
~
~
~
~
~
~
~
~
~
:x
```

Once you exit Vi, a summary of your commit will be displayed detailing the files added, changes, and insertions etc. This commit is now saved to your local repository and you can push it up to the GitHub repository as you wish.

To push all of your recent commits up to the GitHub server, just make sure you are in the local repository's folder and use the command "git push". You will be prompted for your passphrase for your SSH key, and then once it has uploaded all files, a summary will be displayed.

6. Pulling Down Latest Version

To copy your collaborator's latest commits from the GitHub server use the command "git pull".

7. Undoing Changes

If you have made a mess of your local repository and would like to perform a hard reset back to the last local commit you can use the command "git reset" as documented here: http://git-scm.com/docs/git-reset