Git Demo Notes

Requirements

- Unix like environment
- Have `git` installed
- Have `man` installed
- Have `gcc` installed
- Unix, Linux BSD, etc
  - Use your package manager to install
  - Ex. Debian/Ubuntu derivatives: `sudo apt install <x>`
- MacOSX terminal
  - homebrew: https://brew.sh
  - Ex. `brew install <x>`
- Windows
  - Windows Subsystem for Linux (WSL) *Preferred*
    - Activate Windows features:
      - Windows Subsystem for Linux
      - Virtual Machine Platform
    - Distributions available in the Windows Store
    - Ubuntu (easiest, reqs pre-installed)
  - Cygwin
  - SCM-Git

Environment Sidebar

- `~` is shorthand for your home folder.

Start A Project

Hello World

- `mkdir hello`
- `vim hello.c`
- `touch branch1`

`vim` Sidebar

- Movement: h - Left, j - Down, k - Up, l - Right
- i - insert mode
- ESC - go back to movement mode
- To save and quit: Press ESC and then Shift-Colon and type `wq` (write and quit)

- :wq

Version 1

```
#include "stdio.h"

int main(int argc, char** argv){

   printf("hello world\n");
}
```

Version 2

```
#include "stdio.h"

int main(int argc, char** argv){

   if(argc > 1){
      printf("%s\n", argv[1]);
   } else {
      printf("No input\n");
   }
}
```

Setup Project with `git`

`git init .`

Add .gitignore for *.out

`git status`

`git add/remove`

- Concept of staging
- `git add -A`

`git commit`

- `git commit -a -m`
- `git commit -m`

Create a Remote Repository

`git init --bare ~/hello-repo.git`

`git remote add <remote name> <url>`

- `git remote add origin ~/hello-repo.git`

`git push --set-upstream <remote name> <branch>`

- First push: `git push --set-upstream origin master`
- `git push`

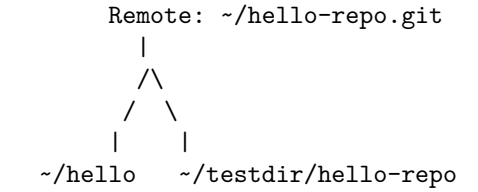
`git clone`

- `mkdir ~/testdir`
- `cd ~/testdir`
- `git clone ~/hello-repo.git`
- `ls`
  - You should see a a copy of your project in a new folder named: `hello-repo`

`git pull`

- Make changes in `~/hello` and then follow the add-commit-push flow.
- Come back to `~/testdir/hello-repo` and run
  - `git pull`
- You should see the changes applied to your *local* copy!
- If you had made local changes, you would have to tell `git` how to handle merge conflicts.

So, now we have 3 copies of the project:



Either one can push changes to the up-stream remote, if they have it set up.

SSH Integration

Git understands the SSH protocol. If you have your remote on a remote machine, you can set it up to use an off-site repository, just as easily.

`.git/config`

When you ran `git init .`, `git` added a hidden folder to your project: `.git`. This is where it stores the project configuration, and the compressed archive of previous versions. You can edit the configuration manually, which may actually be easier for some than using the "porcelain" commands.

If we did everything correctly, then our `.git/config` should look something like this:

```
[core]
    repositoryformatversion = 0
    filemode = true
    bare = false
    logallrefupdates = true
[remote "origin"]
    url = /home/james/hello-repo.git
    fetch = +refs/heads/*:refs/remotes/origin/*
[branch "master"]
    remote = origin
    merge = refs/heads/master
```

To setup a remote host, the easy way, is to add your username and access token to the url like so:

```
[remote "origin"]
    url = https://<username>:<access token>@<host domain>/PATH/TO/hello-repo.git
    fetch = +refs/heads/*:refs/remotes/origin/*
```

If you do not include an access token, it should prompt for a password, in the same manner as ssh.

Here is an example using a Raspberry Pi on my local network:

Branching

One of the purposes of branching, is to provide a mechanism where active development will not interfere with a known stable release. Another, could be to work on an experimental feature that may not make it's way into the final product.

Creating a branch is easy:

```
git branch <name>
```

```
git switch <branch name>
```

When you push from a branch it is mirrored in that branch on the remote host. And once you are ready to integrate your changes, to the main branch, you can just:

```
git switch master
```

```
git merge <branch>
```