Introduction to git

Brennon Brimhall

Why is git so cool?

- It's open source and widely used in the open source community.
- It's widely used in industry.
- Interfaces with other version control systems.
- Distributed, not centralized.
- Cheap branching.
- Automatic merging.
- No network access required.
- Language-agnostic.
- Cross-platform.
- Has integrity.
- Only adds; it doesn't remove things be default.
- Has helpful utilities to identify code changes that break things.
- Has helpful utilities to facilitate collaboration.

A brief history of git

- Not the same thing as GitHub.
- The brainchild of Linus Torvalds.
  - Same guy who originally wrote the Linux kernel.
- Originally written in 2005 in about two weeks because of a disagreement the Linux community had with BitKeeper.
  - Drew lots of inspiration from their tool.
  - Drew lots of anti-inspiration from other systems like CVS, SVN, etc.
- After about two months, Linus went back to working on the kernel and transferred responsibility for its maintenance to Junio Hamano of Google.
- Still is actively maintained; we’re now on version 2.16.
  - Released in January 2018

git init

Command to make a git repository.

```
cd path-to-my-cool-repo/ && git init
```

git clone

Command to copy a git repository from some online source.

```
git clone <url>
git clone https://github.com/torvalds/linux.git
git clone git@github.com:torvalds/linux.git
```

git status

Command to get an overview of what is going on in the repository.

```
git status
```

git diff

Command to get the difference between two commits, the previous commit and the current uncommitted state, etc.

```
git diff
git diff <commit-1>..<commit-2>
git diff <branch-1>..<branch-2>
```

git log

Get the history of the repository. Gives an overview of previous commits, etc.

```
git log
```

git add

Stage files for commit.

```
git add some-folder/some-file
git add some-folder/*
git add -u
```

git commit

Add staged items to the repository.

```
git commit
git commit -m “<my-commit-message>”
```

git branch

Create a branch off the current commit, or list all branches.

```
git branch <my-awesome-branch-name>
git branch
```

git checkout

Move between commits and branches.

```
git checkout <my-commit-sha>
```

```
git checkout <my-branch-name>
```

git merge

Merge a branch with the current branch.

```
git merge <some-other-branch>
```

git pull

Fetch and automatically merge changes from a remote repository. Usually used to update from a central, upstream repository in a work setting.

```
git pull
git pull <remote-name>
git pull <remote-name>/<branch-name>
```

git push

Publish local commits to a remote repository. Usually used to update a central, upstream repository in a work setting.

```
git push
git push <remote-name>
git push <remote-name>/<branch-name>
```

Where can I go to learn more about git?

- Manpages: man git <command-name>
- Linus Torvalds' presentation to Google about git: https://youtu.be/4XpnKHJAok8
- Scott Chacon's book, *Pro Git*: https://git-scm.com/book/en/v2/
- Interactive tutorial on GitHub: https://try.github.io/levels/1/challenges/1