An Overview of Git

• Built to manage the Linux kernel source code

• Developed by Linus Torvalds

• Currently the defacto standard for version control
Basic git commands

● Clone a repo (make a local working copy)
  ○ git clone <url>

● Commit changes
  ○ git add <filename>
  ○ git commit -m "Commit message"
  ○ git push

● Update working copy
  ○ git pull

● Other useful commands
  ○ git status
  ○ git init
Using git for Basic Tasks

- Normally clone an existing repo (can also create with init)
  
git clone https://github.com/KSU-SDML/srcML.git

- This creates a directory srcML with a .git folder inside it along with a working copy of the repo.

- Recording changes - files can be tracked or untracked
- Files can be modified or unmodified
- Changed files need to be staged before they are committed
Recording Changes

- Two files have been modified: foo.cpp, foo.hpp
  ```
  git add foo.hpp
  git add foo.cpp
  ```

- These files are now staged (ready to commit)
- Using git status will show what is staged, modified, and untracked in a directory (on the current branch - master)
  ```
  git commit -m "Updated foo class"
  ```

- Lastly push these commits to the remote repo
  ```
  git push
  ```
Update the Local Repo

- If changes have been made by others on the team you will need to get those updates
  
  git pull

- If working on the same branch as others use an svn like workflow before you commit - pull, add, commit, push
Workflow in git - Branching

- When using git, developers normally create a branch of the repository
- Work (new feature or bug fix) is done on the branch
- After the task is completed than the changes are merged back into the master

- Need to understand how git stores and manages branching
Managing and Storing Changes

● Git is an object database
  ○ Blobs
  ○ Trees
  ○ Commits

● Objects are stored in .git folder
Blobs

Filesystem:

Git Objects:

Blob: 1af3c5
Blob: 5fe43a
Blob: 5fe43a
Trees

Filesystem:

Git Objects:

Tree: 36b0fa
Blob: 1af3c5

Tree: 845acb
Blob: 5fe43a
Blob: 5fe43a
Commit

Filesystem:

Git Objects:

Commit: 8975f1

Tree: 36b0fa

Blob: 1af3c5

Blob: 5fe43a

Tree: 845acb

Blob: 5fe43a

Tree: 36b0fa

Blob: 1af3c5

Blob: 5fe43a

Tree: 845acb

Blob: 5fe43a
Role of Commits

- State of the repository
- Store:
  - Pointer to parent commit
  - Pointer to a tree
  - Other metadata
Branches / Reference

- A branch in git is a movable pointer to a commit object
- Default branch in git is called “master”
- Stored in the “refs” folder within the .git directory
- File stores the commit id

- Head points to the current branch
Commit change

```
- README
- Rakefile
- lib
  - simplegit.rb
```

Edit “simplegit.rb”

Relationships:
- Commit
- HEAD
- Master
- ./
- makefile
- README
- lib
- simplegit.rb
Commit another change

Edit "README"

HEAD

Master

Commit

Commit

Commit

/. 

makefile

/. 

lib

README

lib

README

lib

simplegit.rb

simplegit.rb

simplegit.rb

simplegit.rb

simplegit.rb
Creating and Using a Branch

- Create a repo
- Create a branch
- Merge change
Create a git repo

bash$: git init
Add files to Master branch

bash$: git add README
bash$: git commit -m "Add new file"
Commit changes to Master

bash$: git add *
bash$: git commit -m“Finished Parser”
Create a branch called Develop

bash$: git branch Develop
bash$: git checkout Develop

Short alternative: git checkout -b Develop
Commit to Develop

bash$: git commit -m "Add experimental feature"
Move back to Master

bash$: git checkout master
Commit to Master

bash$: git commit -m "Bug fix #1"
Move back to Develop, make changes to Develop

bash$: git checkout Develop
bash$: git commit -m “Add advanced look ahead”
Merge the changes from Master into Develop

bash$: git merge master
Resources

Pro Git
https://git-scm.com/docs

GitHub Guide
https://guides.github.com/

Git Ready
http://gitready.com/

Tutorials:
https://try.github.io/
https://learngitbranching.js.org/