Lecture 2
Making Simple Commits

Sign in on the attendance sheet!

AS A PROJECT DRAGS ON, MY GIT COMMIT MESSAGES GET LESS AND LESS INFORMATIVE.

credit: https://xkcd.com/1296/
Homework Reminders

• Great job getting the homework done this week!
• Future homework will be submitted through Autolab: https://autolab.andrew.cmu.edu/courses/98174-f17

Remember not to do this:
Andrewid.zip/
  question-2/
    left-pad/
      question-4.txt
Review of Last Lecture

• git init – creates a git repo in the current directory
• git clone <git url> – copies the remote git repo into the current directory
• git log [ --oneline ] – lists all commits in the git repo, starting with the most recent one
• git help <command>, git <command> --help, man git <command> – brings up the man help page for the git command
The .git folder

- Every git repository has a .git directory in the toplevel project directory
- This is where all git commit objects and metadata are stored
- **Don’t delete it!** Doing so deletes the repository
- Folders starting with a dot are hidden on UNIX
Today: The Git Commit Workflow

- Review: `git log`
- `git diff`
- `git status`
- `git add`
- `git commit`
- `git show`
From Last Time: git log

Also try git log --oneline:
What is 2eae45f?

• Commits are uniquely represented by SHA-1 hashes
• The first 6-7 characters of a hash are usually enough to identify it uniquely from all the other commits in the repository
• This is called the short hash
What is a commit?

1. A **snapshot** of all the files in a project at a particular time
2. A **checkpoint** in your project you can come back to or refer to
3. The **changes** a commit makes over the previous commit

Commits are identified by their SHA-1 **hash**
This is an example of how git diff works!

Here is a new line of text!
Git diff is my favorite command!

Git diff is my favorite command!
Here is a new line of text!
Git Diff

```
Aaron@HELIOS ~/Dropbox/Dropbox Documents/98174/testing (master)
$ git diff
diff --git a/demo.txt b/demo.txt
index 4fd054e..ff58225 100644
--- a/demo.txt
+++ b/demo.txt
@@ -1,3 +1,5 @@
  This is an example of how git diff works!
+Here is a new line of text!

-Git diff is my favorit command!
+Git diff is my favorite command!
+Here is another line of text!
```
Commits: Revisited

• Editing a file takes its state from 1 particular snapshot to the next
• When we edit a file, we can see it as a set of changes (a “diff”) from the snapshotted state of that file
• Commits bundle up sets of changes to a list of files

List of commits

bb2df1a
(HEAD)
file1.txt (v2)
file2.txt (v1)
file3.txt (v1)

782cb4f
file1.txt (v1)
file2.txt (v1)
file3.txt (v1)

ab628cc
file1.txt (v1)
file2.txt (v1)
git show <commit hash>
The Git Commit Workflow: Edit

Make changes to files
vim file1.txt file3.txt

Working Directory

file1.txt (v2)
file2.txt (v1)
file3.txt (v2)

List of Changes

- In file1.txt: add the line “here is a new line!” between lines 3 and 4
- In file3.txt: delete line 27
The Git Commit Workflow: Add

Working Directory

file1.txt (v2)
file2.txt (v1)
file3.txt (v2)

Add the current differences

git add file1.txt file3.txt

List of Changes

- In file1.txt: add the line “here is a new line!” between lines 3 and 4
- In file3.txt: delete line 27

Staging Area
The Git Commit Workflow: Commit

List of Changes

Staging Area

- In file1.txt: add the line “here is a new line!” between lines 3 and 4
- In file3.txt: delete line 27

Commit the currently staged differences

git commit -m "fixed bug in file1 and file3"

List of commits

- bb2df1a
  - file1.txt (v2)
  - file2.txt (v1)
  - file3.txt (v2)
- 782cb4f
  - file1.txt (v1)
  - file2.txt (v1)
  - file3.txt (v1)
- ab628cc
  - file1.txt (v1)
  - file2.txt (v1)
git add

Example use:
git add file1.txt file2.txt
(or)
git add . (adds changes to all files in directory)

- Creates a commit out of a snapshot of the staging area, and updates HEAD.
git commit

Example use:

git commit
(or)
git commit -m "commit message goes here"

• Creates a commit out of a snapshot of the staging area, and updates HEAD.
Aside: commit HEAD

• The “most recent commit” has a special name: HEAD
Good commit messages

• Good:
Build: Don't install jsdom3 on Node.js 0.10 & 0.12 by default

• Bad:
bugfix lol get rekt

http://whatthecommit.com
git status

Shows files differing between the staging area and the working directory (i.e. unstaged changes), the staging area and HEAD (i.e. changes ready to commit), and untracked files
git diff

Example use:
(show unstaged changes)
git diff
git diff

(show staged changes)
git diff --cached

• Shows unstaged changes or staged changes
git show

Example use:
git show [commit hash (default is HEAD)]

• Shows the changes in the specified commit
Activity: Practicing Making Commits

1. Make a new folder, and create a new git repository inside.
2. Create a file called “me.txt”. Inside, write your name and hometown.
3. Make a commit with this new file.
4. Make a new file called “neighbors.txt”.
5. Now, find 3 people sitting near you. For each person,
   1. Find out their name and hometown, and put it in neighbors.txt.
   2. Check the output of git status and git diff and verify it makes sense.
   3. git add neighbors.txt
   4. Check the output of git status and git diff and verify it makes sense.
   5. Commit the change.
   6. Check the output of git show and verify it makes sense.