CSE 403
Software Engineering
Spring 2020

Version control and Git

April 01, 2020
Logistics

Survey (timezone and connectivity)
● Please fill out this super short survey:
  https://catalyst.uw.edu/webq/survey/rjust/387671

Pre-lecture and happy-hacking playlist
● We are creating a spotify playlist
● Add your suggestions to Slack (# playlist)

Projects
● Overview, examples, and themes on Friday
● Project proposals and pitches next week
Today

- SE failure of the day
- Technical background
- Version control and Git
SE failure of the day

Installing 🔄 under OSX…

- (Ab)uses preinstall scripts
- Spoofs system dialog
- (Ab)uses root privileges
A little survey
Survey: Operating system

My operating system is

- Linux: 2
- Mac OS X: 17
- Windows: 26

Total Results: 45
Survey: Command line

How proficient are you with the Linux command line and shell?

- What is that? 1
- Beginner 22
- Advanced 21
- I can teach Linus Torvalds! 1

Total Results: 45
Survey: Git

How proficient are you with Git?

- What is that? 0
- Beginner 20
- Advanced 24
- Linus Torvalds asks me for help!

Total Results: 44
Survey: Build systems

How proficient are you with Java build systems (e.g., Ant or Maven)

- What is that? 11
- Beginner 31
- Advanced 2
- Expert

Total Results: 44
Survey: JDK installation

I know how to install a Java JDK and set the PATH and JAVA_HOME vars

- Nope: 3
- Somewhat: 18
- Mostly: 9
- No problem: 15

Total Results: 45
Version control
What is version control?

Version control records changes to a set of files over time. This makes it easy to review or obtain a specific version (later).

**Simple Example**

- René writes a paper, using version control: **v1.0**
- René corrects grammatical mistakes and typos: **v1.1**
- René discovers new findings and rewrites the paper: **v1.2**
- René realizes the new findings are wrong: restore **v1.1**

**Who uses version control?**

- Developers (obviously)
- Researchers
- Applications (e.g., (cloud-based) word processors)
Why use version control?
Why use version control?

Just kidding... this is far more realistic.
Centralized version control

- **One central repository.**
- All users commit their changes to the central repository.
- Each user has a working copy. As soon as they commit, the repository gets updated.
- Examples: SVN (Subversion), CVS.
Distributed version control

- **Multiple clones of a repository.**
- Each user **commits** to a local (private) repository.
- All committed changes remain local unless **pushed** to another repository.
- No external changes are visible unless **pulled** from another repository.
- Examples: Git, Hg (Mercurial).
Branches

- One main development **branch** *(master, trunk, etc.)*.
- To add a new feature, it’s useful to create a new **branch** -- an independent line of development.
- **Lightweight** branching *(branch)*.
- **Heavyweight** branching *(clone)*.
- **Forking** *(clone + metadata)*.

[master] 8c6faa5 My first commit - John Doe
[develop] 3e89ec8 Develop a feature - part 1 - John Doe
[develop] e188fa9 Develop a feature - part 2 - John Doe
[master] 665003d Fast bugfix - John Fixer
[myfeature] eaf518c New cool feature - John Feature
[master] 8f1e0e7 Merge branch `develop` into `master` - John Doe
[master] 6a3dacc Merge branch `myfeature` into `master` - John Doe
[master] abcdef0 Release of version 0.1 - John Releaser
Merge
Merge
Conflicts

- **Conflicts** arise when two users change the same line of a file.
- When a conflict arises, the last committer needs to resolve it.

Are conflicts more likely in (de)centralized version control? How to avoid merge conflicts?
Rebase
Rebase
Rebase
Rebase: changing the commit history

- Changes and sequentializes the commit history.
- Often used to squash commits or amend messages.
- A powerful tool, but … think about pros and cons.
Version control with Git

- “Because my hatred of CVS has meant that I see Subversion as being the most pointless project ever started, because the whole slogan for the Subversion for a while was 'CVS done right' or something like that. And if you start with that kind of slogan, there is nowhere you can go.”
- "‘what would CVS never ever do’-kind of approach"
Motivating Example: What is this Git command?

NAME
   git-______ - ______ file contents to the index
SYNOPSIS
DESCRIPTION
This command updates the index using the current content found in the working
tree, to prepare the content staged for the next commit. It typically ______s the
current content of existing paths as a whole, but with some options it can also
be used to ______ content with only part of the changes made to the working tree
files applied, or remove paths that do not exist in the working tree anymore.
Motivating Example: What is this Git command?

NAME
   git-add - Adds file contents to the index
SYNOPSIS
DESCRIPTION
This command updates the index using the current content found in the working tree, to prepare the content staged for the next commit. It typically adds the current content of existing paths as a whole, but with some options it can also be used to add content with only part of the changes made to the working tree files applied, or remove paths that do not exist in the working tree anymore.
Git: vocabulary

- **index**: staging area (located .git/index)
- **content**: git tracks what is in a file, not the file itself
- **tree**: git's representation of a file system
- **working tree**: tree representing what is currently checked out
- **staged**: ready to be committed
- **commit**: a set of database entries (snapshot of the working tree)
- **ref**: pointer to a commit object
- **branch**: just a (special) ref; semantically: represents a line of dev
- **HEAD**: a ref pointing to the working tree
Git: concepts and terminology
Git: concepts and terminology

SYNOPSIS

```
git-diff-index [-m] [--cached] [<common diff options>] <tree-ish> [<path>…]
```

DESCRIPTION

`git-diff-index` compares the content and mode of the blobs found in a tree object with the corresponding tracked files in the working tree, or with the corresponding paths in the index.
Git: concepts and terminology

SYNOPSIS

`git-diff-index` [-m] [--cached] [<common diff options>] <tree-ish> [<path>…]

DESCRIPTION

`git-diff-index` compares the content and mode of the blobs found in a tree object with the corresponding tracked files in the working tree, or with the corresponding paths in the index.

SYNOPSIS

`git-allocate-remote` [--derive-head | --massage-link-head | --abduct-commit]

DESCRIPTION

`git-allocate-remote` allocates various non-branched local remotes outside added logs, and the upstream to be packed can be supplied in several ways.
Git: concepts and terminology

SYNOPSIS

`git-diff-index` [-m] [--cached] [<common diff options>] <tree-ish> [<path>…]

DESCRIPTION

`git-diff-index` compares the content and mode of the blobs found in a tree object with the corresponding tracked files in the working tree, or with the corresponding paths in the index.

SYNOPSIS

`git-allocate-remote` [ --derive-head | --massage-link-head | --abduct-commit ]

DESCRIPTION

`git-allocate-remote` allocates various non-branched local remotes outside added logs, and the upstream to be packed can be supplied in several ways.

SYNOPSIS

`git-resign-index` [ --snap-file ] [ --direct-change ]

DESCRIPTION

`git-resign-index` resigns all non-stashed unstaged indices, and the --manipulate-submodule flag can be used to add a branch for the upstream that is counted by a temporary submodule.
Git: concepts and terminology

SYNOPSIS

git-diff-index [-m] [--cached] [<common diff options>] <tree-ish> [<path>…]

DESCRIPTION

git-diff-index compares the content and mode of the blobs found in a tree object with the corresponding tracked files in the working tree, or with the corresponding paths in the index.

SYNOPSIS

git-allocate-remote [ --derive-head | --massage-link-head | --abduct-commit ]

DESCRIPTION

git-allocate-remote allocates various non-branch local remotes outside added logs, and the upstream to be packed can be supplied in several ways.

SYNOPSIS

git-resign-index [ --snap-file ] [ --direct-change ]

DESCRIPTION

git-resign-index resigns all non-stashed unstaged indices, and the --manipulate-submodule flag can be used to add a branch for the upstream that is counted by a temporary submodule.
Git: concepts and terminology