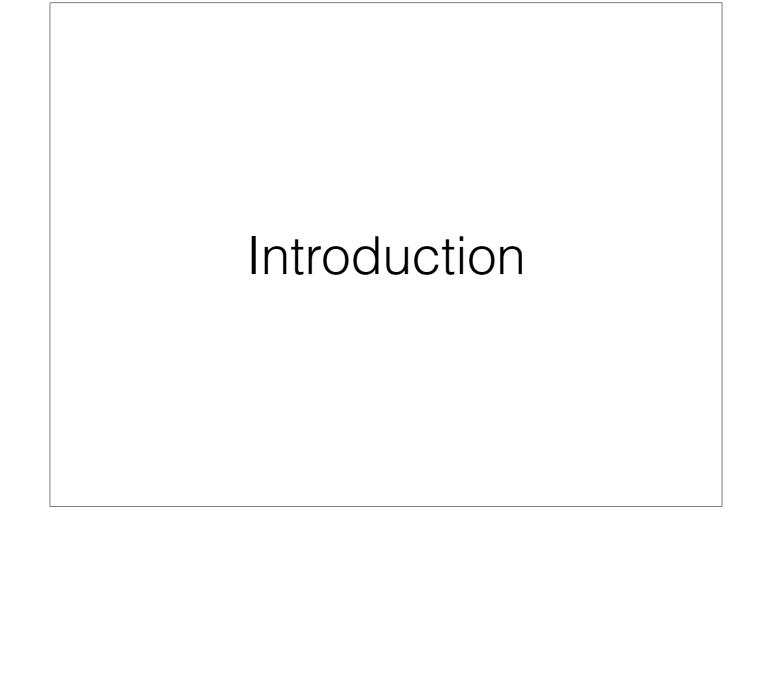
Intermediate Git and GitHub Workshop

JSConf Asia 2014 Donal Ellis

Topics

- Collaborating with Git and GitHub the GitHub workflow
- Forking, cloning and the GitHub Pull Request
- Git commands: branch, commit, diff, log, stash, alias
- Git reset and the reflog
- See https://github.com/donal/jsconf-asia-2014-git-and-github-workshop/blob/master/README.md



GitHub Stuff

- Signup to https://github.com
- Student pack at https://education.github.com/pack
- Pro Git v2, free at http://git-scm.com/book/en/v2
- https://training.github.com/

Command Line vs. GUI

- The workshop notes and exercises refer to the command line.
- If you're using a GUI for Git you'll have to execute these commands differently (but under the hood they will be the same commands as used on the line command).

The GitHub Flow

The GitHub Workflow

- 1. Create a feature branch
- 2. Do work on it (commit and push)
- 3. Create a pull request on GitHub
- 4. Discuss (on the pull request)!
- 5. Make more changes
- 6. Get the 👍 or 星
- 7. Merge the pull request
- 1. Person who opens the PR is responsible for closing/merging it
- 2. Good to practice this workflow even if you're the single developer on your own project (at minimum it means you have a deployable master branch)

Open Source Projects

- On GitHub you fork an existing project
- See https://help.github.com/articles/fork-a-repo/
- Follow the GitHub workflow

Add "student-questions" directory with README.md file.

- Your PR is between your forked repo and the project repo
- Checkout https://github.com/explore

Workshop Exercise: fork Donal's "jsconf-asia-2014-git-and-github-workshop" repo on GitHub

"Private" Git Workflows

- "Private" means invited collaborators only, so this often means within a company.
- Using the GitHub flow grants power and therefore promotes trust.
- But Git is very flexible setup whatever flow best suits your team.

Even within a company (who might have their own secured installation of Enterprise GitHub)

Git Branching

Workshop Exercise: https://github.com/donal/jsconf-asia-2014-git-and-github-workshop/blob/master/branch.md

Committing in Git

- A file in a git repository is either untracked or tracked
- Untracked files haven't been added to the repository
- Tracked files are either: unmodified (or committed), modified or staged.

Question: What is the state of files in a newly cloned repo?

Answer: they're all tracked and unmodified.

Git Diff

- When you've changed a tracked file (but you haven't staged it), you can use the git diff command to view the changes.
- When you've **staged** a file, you can use the git diff --staged command to view the changes.
- The output is a patch view of the changes.

Committing in Git

Workshop Exercise: https://github.com/donal/jsconf-asia-2014-git-and-github-workshop/blob/master/commit.md

Push (to your pull request) early and often is strongly encouraged as part of the GitHub flow.

Git Log

• Now that you've committed, it's worth viewing the Git log.

Workshop Exercise: https://github.com/donal/jsconf-asia-2014-git-and-github-workshop/blob/master/log.md

Pull Requests in GitHub

Workshop Exercise: https://github.com/donal/jsconf-asia-2014-git-and-github-workshop/blob/master/pull_request.md

- For Open Source projects you create a PR from your fork back to (usually) the master branch of the project repo.
- * For private projects you (usually) create a PR from your feature branch back to master branch.

Remote Changes

- So what happens when there are changes made on the repository that you forked from?
- You need to get these changes from the project repository and merge them into your project/branch.
- First you must add the project repository as a new remote.
- Then in two steps: git fetch to get the changes, and git merge to apply them.

Remote Changes

Will need to make changes to the project, e.g.. the README.md file, before this exercise can be done.

Workshop Exercise: https://github.com/donal/jsconf-asia-2014-git-and-github-workshop/blob/master/pull_request.md

Deleting Branches

- Once your pull request has been merged (into the master branch) you should pull down the changes (see above)
- Now you have a branch that points to the same commit as the master branch
- Which means you can delete it.

Deleting Branches

Workshop Exercise: https://github.com/donal/jsconf-asia-2014-git-and-github-workshop/blob/master/delete_branch.md

Merge Conflicts

- Handling conflicts in Git is relatively pain free
- When you attempt a git merge, if the same part of a file has been changed differently in each branch, there will be a merge conflict

Merge Conflict

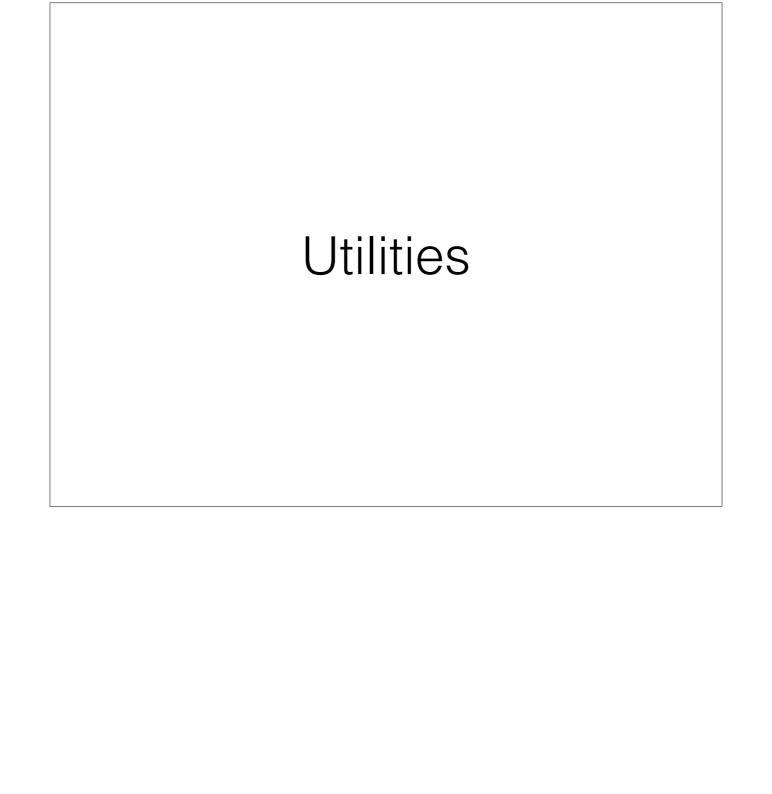
Need to edit the README.md file (probably the one in students/) then push it to the GitHub repo.

Workshop Exercise: https://github.com/donal/jsconf-asia-2014-git-and-github-workshop/blob/master/merge_conflict.md

Git Stash

- Sometimes you want to merge or change branches while you still have modified code
 - If the merge or branch change would result in a conflict, then Git won't allow you
- An easy way to handle this situation is to git stash the working code

Workshop exercise: https://github.com/donal/jsconf-asia-2014-git-and-github-workshop/blob/master/stash.md



Git Aliases

- There's (possibly) some handy Git aliases already in your Git config file
- The git config -1 lists the variables in the config file.
- For example: alias.lg=log --graph -- pretty=oneline --abbrev-commit --decorate
 - This is the log command earlier, which means you can run it with the simple command: git lg

Git Aliases

- You can obviously add your own aliases.
- For example, run: git config --global alias.rv "remote -v"

The Structure of Git

Git "Trees"

- Git has three areas (sometimes referred to as "trees"):
 - HEAD the current branch, the latest commit, the snapshot (stored in .git/)
 - Index the proposed next commit, the staging area (stored in .git/index)
 - Working directory a copy of HEAD, a sandbox (these are the files you can manipulate directly)

Git File Stages

- Tracked files (in the working directory) are in one of three states:
 - modified the file has been changed in the working directory only
 - 2. **staged** the file has been moved to the staging area, so it's the same in staging and in the working directory
 - 3. **unmodified**/committed the file has been committed, so it's the same in HEAD, staging and the working directory

Undoing Things

Git Amend

- If you've committed but you want to change something about the commit:
 - Add or remove a file
 - Modify the commit message
- Use git commit --amend (but only if you haven't pushed).

Git Reset

- git reset is the command used to undo changes
- You can move files from the git directory (HEAD) to staging
- You can move files from staging to the working directory
- You can reset modified files in the working directory

Git Reset Soft

- To move a file from the git directory (HEAD) to staging:
 - git reset --soft <filename>

Git Reset (Mixed)

- To move a file from staging to the working directory:
 - git reset [--mixed] <filename>
- "mixed" is the default option for git reset

Git Reset Hard

- To remove a file from the working directory:
 - git reset --hard <filename>
- **Danger!** This is the version of reset that can DELETE data.

Git Reflog

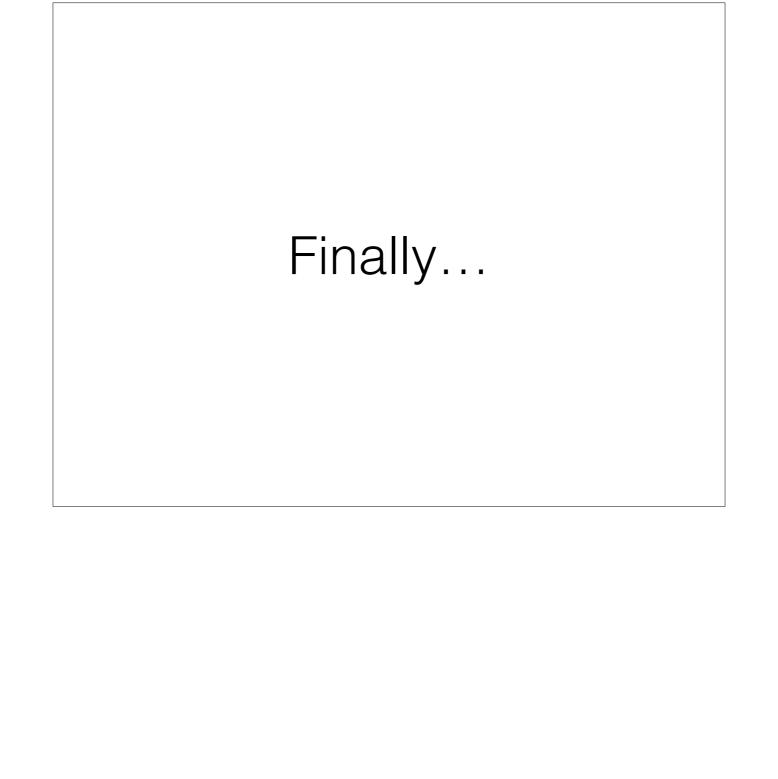
- git log shows the commit log
- git reflog shows the reference log
- What's a reference? It's a named "pointer" to a commit, in other words, a branch or tag.

Git Reflog

- Run git reflog to see a history of all commits against HEAD
- This is a way to find "lost" commits and retrieve them
- Use git cherry-pick to restore/apply a commit

Git Reset and Reflog

Workshop Exercise: https://github.com/donal/jsconf-asia-2014-git-and-github-workshop/blob/master/ reset_and_reflog.md



More Pull Requests

- As a final exercise:
 - 1. Create your own open source project repository on GitHub
 - 2. Share the repo URL with the other workshop participants and get their repo URL
 - 3. Fork their repo
 - 4. Create a branch and make some changes
 - 5. In GitHub create a PR to their repo
 - 6. Discus the PRs on GitHub
 - 7. The repo owner can then merge or reject the PR