Fundamentals of Git

By Chris McKnight
March 26th, 2015

Overview

- History of Git
- Project setup
- Staging Changes and Committing
- Branching, Merging, and Rebasing
- Working with remotes

History of Git

- Linux kernel developers started using BitKeeper[1]
- BitKeeper license revoked
- Linus Torvalds created Git in 2005

[1] http://git-scm.com/book/en/v2/Getting-Started-A-Short-History-of-Git

What is Git?

- Distributed revision control system
- Goals
  - Efficiency
  - Speed
  - Non-linear workflows

Configure Git

- Install Git[1]
- Configure user settings
  - git config --global user.name “John Doe”
  - git config --global user.email “jdoe@example.com”
- Configure line ending conversions (OS dependent)[2]

[1] https://help.github.com/articles/set-up-git
[2] https://help.github.com/articles/dealing-with-line-endings

Project Setup

- New Project
  - init
  - Create a .gitignore file
  - Create a .gitattributes file (optional)
  - stage changes
  - commit
- Existing project (not on computer)
  - clone from a remote

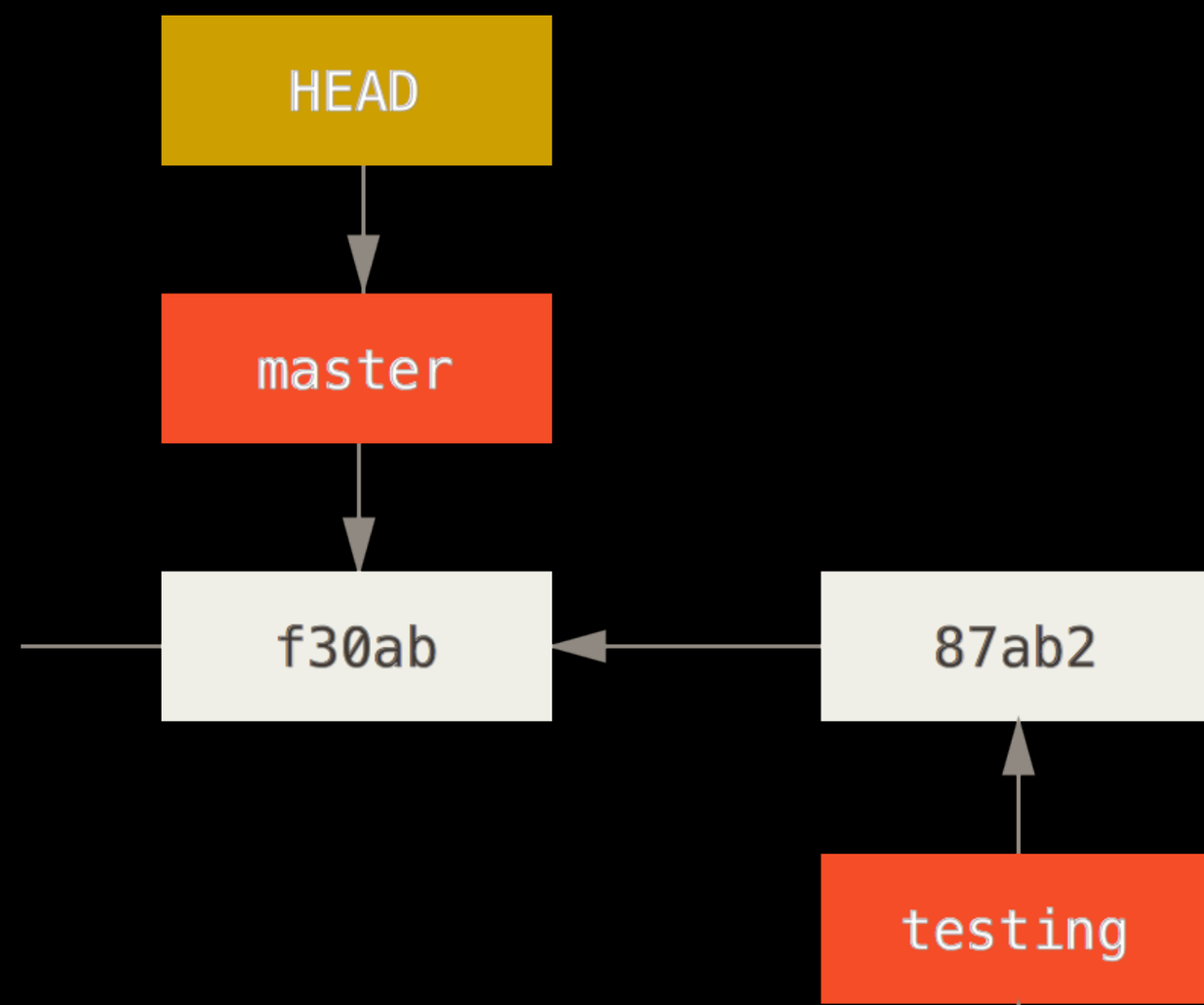
```
-> % git init
Initialized empty Git repository in /Users/Chris/presentations/git-project/.git/
Chris@McKnight [03:39:16] [~/presentations/git-project] [master *]
-> % touch .gitignore
Chris@McKnight [03:39:34] [~/presentations/git-project] [master *]
-> % git add .
Chris@McKnight [03:39:44] [~/presentations/git-project] [master *]
-> % git status
On branch master

Initial commit

Changes to be committed:
  (use "git rm --cached <file>..." to unstage)

        new file:   .gitignore
        new file:   README.md

Chris@McKnight [03:39:46] [~/presentations/git-project] [master *]
-> % git commit -m "Initial commit"
[master (root-commit) 6a271a8] Initial commit
 2 files changed, 1 insertion(+)
 create mode 100644 .gitignore
 create mode 100644 README.md
Chris@McKnight [03:39:55] [~/presentations/git-project] [master]
-> %
```

Staging Changes & Committing

- Staging changes
  - git add <path>
  - git rm <path>
  - git commit
- Unstage changes
  - git reset
- Can also move the file pointer of the current branch or discard all changes using git reset

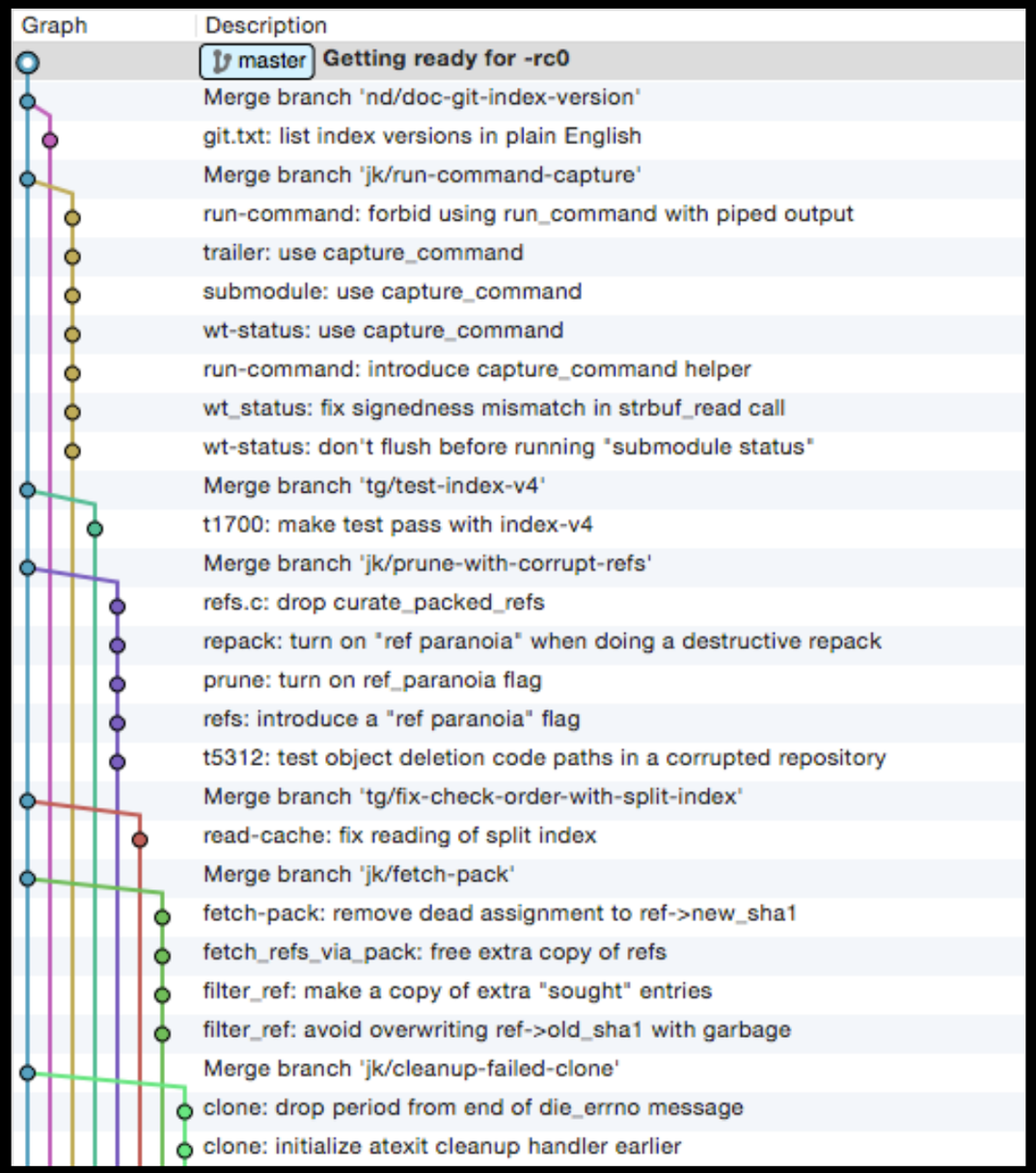
Branching

- Default branch is called master
- Lightweight pointer to a commit
- Useful for work in progress, features, and bug fixes



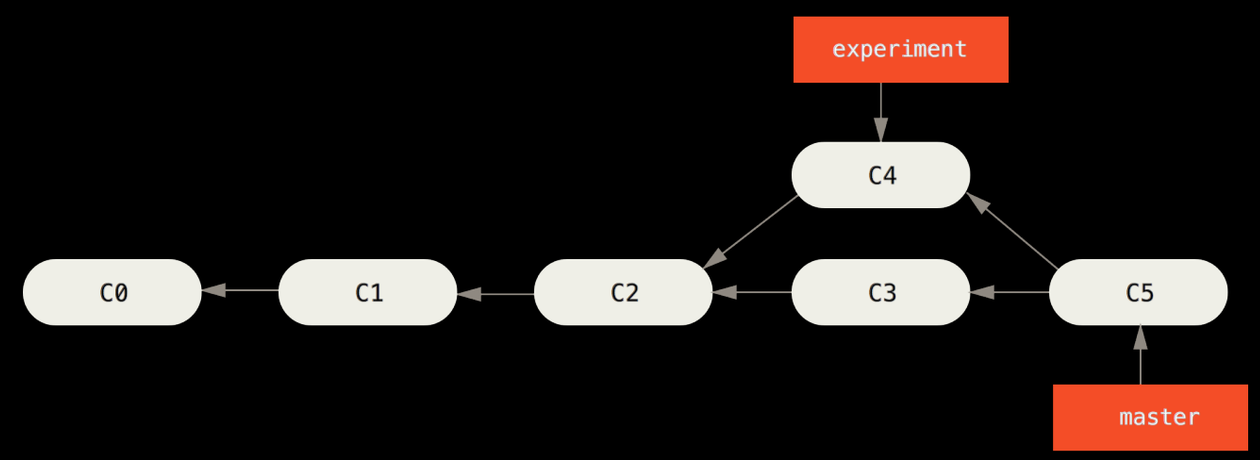
Merging

- Integrates changes from one branch to another
  - Example: merging a feature branch into master for deployment



Rebasing

- Another method of integrating changes of a branch into another branch
- Rewrites history
  - Moves to common ancestor of the two rebased branches
  - Figures out diffs
  - Moves commits from the current branch into a temporary area
  - Resets the current branch to the branch you are rebasing from
  - Applies commits from the temporary area onto the current branch
- Usage
  - Updating a feature branch with bugfix changes
  - Cleaning up a local feature branch before pushing to a remote



Remotes

- Typically the main remote is named origin
  - origin is not a special remote
  - can have multiple remotes
- Another Git repository used for collaboration
- Changes are pushed and pulled

A basic workflow

- Create a new project
- Stage files and commit
- Create a remote and push
- Iterative development by using branches, merging and pushing changes

GUI Tools

- Sourcetree GUI (http://www.sourcetreeapp.com/)
- GitHub app
  - https://windows.github.com
  - https://mac.github.com
- Others available at http://git-scm.com/download/gui/linux

Diff & Merge Tools

- Kaleidoscope for Mac (http://www.kaleidoscopeapp.com)
- WinMerge for Windows (http://winmerge.org)
- Meld for Windows (http://meldmerge.org)
- Araxis Merge (http://www.araxis.com/merge)

Resources

- Pro Git Book (http://git-scm.com/book/en/v2)
- GitHub (https://help.github.com)
- Try Git (https://www.codeschool.com/courses/try-git)
- Git Real (https://www.codeschool.com/courses/git-real and https://www.codeschool.com/courses/git-real-2)