# Intro to Git

and GitHub

#### **University** of Idaho Library

Evan Peter Williamson
<a href="mailto:ewilliamson@uidaho.edu">ewilliamson@uidaho.edu</a>
2016-09-28



#### "FINAL".doc



FINAL.doc!



FINAL\_rev.2.doc





FINAL\_rev.6.COMMENTS.doc



FINAL\_rev.8.comments5. CORRECTIONS.doc



JORGE CHAM @ 2012





FINAL\_rev.18.comments7. corrections9.MORE.30.doc

FINAL\_rev.22.comments49. corrections.10.#@\$%WHYDID ICOMETOGRADSCHOOL????.doc

Jorge Cham, PHD Comics, <a href="http://www.phdcomics.com/comics/archive.">http://www.phdcomics.com/comics/archive.</a> <a href="php?comicid=1531">php?comicid=1531</a>

# About Git

# Git distributed version control system





#### Example

Software Carpentry, "Version Control with Git"

Lesson content is created collaboratively in the repository, <a href="https://github.com/swcarpentry/git-novice">https://github.com/swcarpentry/git-novice</a>

Set in a collaboratively maintained template, <a href="https://github.com/swcarpentry/lesson-example">https://github.com/swcarpentry/lesson-example</a>

That generates a web site hosted on Github, <a href="http://swcarpentry.github.io/git-novice/">http://swcarpentry.github.io/git-novice/</a>

## Basic Workflow

```
git clone... / pull
Edit files
git add...
git commit...
git push...
```

1. Git in a web browser

## GitHub

# git repository hosting service

Use Git in a web browser!

#### Create a Repository:

- https://github.com/
- Log in
- Click + to create new repository
- Name it
- Click initialize with README
- Click "Create repository"

#### Create a commit:

- Click on a file
- Edit in browser
- Scroll down to "Commit changes"
- Fill in the information describing the change
- Click "Commit changes"

# 2. Git in a GUI application

## GitHub Desktop

Use Git in a GUI application! <a href="https://desktop.github.com/">https://desktop.github.com/</a>

#### Clone your repository:

- Click +, then clone
- Choose your repository from the list
- Choose directory to save in (leave default)

#### Create a commit:

- Open the folder
- Change the README file
- Check changes on Github app
- Create a Commit message
- Click "Commit to master"

#### Sync:

 Click Sync button (this pushes our local changes to the GitHub, plus pulls any updates from the web version)

3. Git in the shell

## Shell

terminal, command line, Bash, etc

An application that lets you communicate with your operating system.

- Command-line interface (CLI)
- read-evaluate-print loop (REPL)

On Windows use Git Bash for Unix-like commands so you can follow Mac or Linux tutorials.

## Git Config

one time initial setup

Set your name and email:

```
git config --global user.name "Evan Will"
git config --global user.email "myemail@gmail.com"
```

Set your text editor (Windows "notepad", Mac "edit -w", Linux "nano -w"):

```
git config --global core.editor "notepad"
```

Worried about privacy, see <a href="https://help.github.com/articles/keeping-your-email-address-private/">https://help.github.com/articles/keeping-your-email-address-private/</a>

## Basic Workflow

```
git clone... / pull
Edit files
git add...
git commit...
git push...
```

## Create repository locally

mkdir test

cd test

git init

Your new folder "test" will have a hidden folder ".git" which contains the full history. It is hidden for a reason--you don't need to know anything about it!

If you want to add this repository to GitHub, you have to "git remote add".

For most basic workflows it is easier to start a repository on GitHub, then "clone" it to your local machine.

### Clone repository from GitHub

Go to your GitHub repository, Click "clone or download" button, Copy the clone url.

git clone https://github.com/uidaholib/gitworkshop.git

Clone will download a full copy of the repository to your local machine and record its origin on GitHub. This ensures there is a connection so you can Push your changes to GitHub or Pull updates.

Your clone has the full history stored in the ".git" hidden folder.

### Track a file [ git add ]

```
git status
```

Git status is your friend. Type it often. Create a new file: echo "some interesting notes" > notes.txt

git status

Start tracking the file with "git add", adding the file to the "staging area":

git add notes.txt

git status

### Commit [ git commit ]

We are ready to store a set of changes, let's commit!

A message is required and will be recorded along with name and date. It is your note to posterity so you can remember what you did in the future.

```
git commit -m "add notes file"
git status
```

A commit is like a snap shot. We have stored away a new version of the documents in the repository and we will be able to navigate this history.

Make many smaller focused commits, rather than big ones, since small changes will be easier to undo, review, and merge.

## History [ git log ]

Check how far we have come! Take a look at the repository's history of commits.

```
git log
git log --oneline
```

## Check changes [ git diff ]

Now modify one of the file in your repository using a text editor or the CLI.

```
echo "more important notes!" >> notes.txt
git status
git diff
```

Diff allows us to see what changes were made to the currently unstaged files. Its best to check this before you commit to ensure you know what you are changing.

#### Repeat!

You will use "add", "commit", and "status" over and over!

Git add fills the staging area.

Git commit takes the snapshot of changes in the staging area.

git add notes.txt
git commit -m "more notes"
git status

#### Undo [ git checkout ]

Wait that last commit was no good! Let's get the earlier version back.

You can track individual commits using their id given by git log. Or we can refer to them sequentially. The most recent commit is called "HEAD". One back is called "HEAD~1" (head minus one). If you git checkout a specific file, it will reset it to that commit.

```
git checkout HEAD~1 notes.txt
git diff HEAD notes.txt
git checkout HEAD notes.txt
```

These two checkouts undid each other!

[ if you get 'detached HEAD' warning, type "git checkout master" ]

## Send changes to GitHub [ git push ]

#### git status

Notice git status now says "your branch is ahead of origin/master".

We finished our changes to the repository locally, now we have to add them to the version hosted on GitHub.

#### git push origin master

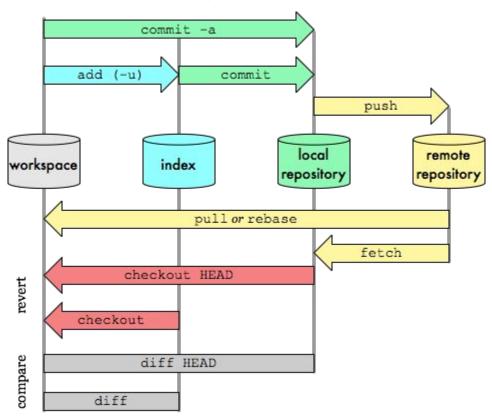
Origin/master is the standard way to name the main branch of the central repository.

Push sends only the commits, so it is efficient network use.

#### Review basic workflow

```
git status
git pull origin master
git add file.txt
git commit -m "message"
git push origin master
```

## Git Data Transport Commands



Oliver Steele, <a href="http://blog.osteele.com/posts/2008/05/my-git-workflow/">http://blog.osteele.com/posts/2008/05/my-git-workflow/</a>

#### Extra Credit [ fork ]

Go to the gitworkshop repository, <a href="https://github.com/uidaholib/gitworkshop">https://github.com/uidaholib/gitworkshop</a>

Click the "Fork" button in the upper right.

Edit the README adding your name and profile url.

Click "New pull request" to send me a message asking to add your edits to the main repository.

### Extra Credit [ gh-pages ]

GitHub pages is a quick and easy way to host static web pages. One option is to auto-generate a web page for your repository.

Click on "Setting"
Click on "Automatic page generator"
Follow the wizard steps
Publish

Your website will be [username].github.io/[repositoryname]

For example, <a href="https://uidaholib.github.io/gitworkshop/">https://uidaholib.github.io/gitworkshop/</a>

#### Interesting things not covered

Branch

Merge

Pull requests

Workflow strategies (Github flow, Gitflow, feature branch, etc)

Appendix

## Learning resources

Try Git (15min tutorial), <a href="https://try.github.io/">https://try.github.io/</a>

Software Carpentry:

Unix Shell, <a href="http://swcarpentry.github.io/shell-novice/01-intro/">http://swcarpentry.github.io/shell-novice/01-intro/</a> Git lesson, <a href="http://swcarpentry.github.io/git-novice/">http://swcarpentry.github.io/git-novice/</a>

GitHub Guides, <a href="https://guides.github.com/">https://guides.github.com/</a> (check out Hello World to learn GitHub features)

Git Tutorials (Bitbucket), <a href="https://www.atlassian.com/git/tutorials">https://www.atlassian.com/git/tutorials</a>

Git Book, <a href="https://git-scm.com/book/en/v2">https://git-scm.com/book/en/v2</a>

#### Recent articles

Art of the Commit, <a href="http://alistapart.com/article/the-art-of-the-commit">http://alistapart.com/article/the-art-of-the-commit</a>

Git for designers, <a href="https://medium.com/@dfosco/git-for-designers-856c434716e#.831v9cwbg">https://medium.com/@dfosco/git-for-designers-856c434716e#.831v9cwbg</a>

Git in the Classroom, <a href="https://opensource.com/education/16/1/git-education-classroom">https://opensource.com/education/16/1/git-education-classroom</a>

Git for artists, <a href="https://opensource.com/life/16/2/version-control-isnt-just-programmers">https://opensource.com/life/16/2/version-control-isnt-just-programmers</a>

Openstax textbooks, <a href="https://github.com/philschatz/algebra-trigonometry-book/">https://github.com/philschatz/algebra-trigonometry-book/</a>

Github-pages student portfolio lessons, <a href="https://dannguyen.github.io/github-for-portfolios/">https://dannguyen.github.io/github-for-portfolios/</a>

#### Repository hosting services

#### The Big ones:

- GitHub, <a href="https://github.com/">https://github.com/</a> (Free public repos, gh-pages web hosting, huge community, Github Desktop app)
- Bitbucket, <a href="https://bitbucket.org/">https://bitbucket.org/</a> (Free private repos (limited collaborators), Git or Mercurial, SourceTree app)
- Gitlab, <a href="https://about.gitlab.com/">https://about.gitlab.com/</a> (Free public and private repos, unlimited collaborators)

#### Others:

- Gitbook, <a href="https://www.gitbook.com/">https://www.gitbook.com/</a> (focus on creating and publishing books)
- Penflip, <a href="https://www.penflip.com/">https://www.penflip.com/</a> (focus on collaborative writing)

#### Other Version Control Software

Mercurial (Hg), <a href="https://www.mercurial-scm.org/">https://www.mercurial-scm.org/</a> (some say its easier than Git?)

Subversion, <a href="https://subversion.apache.org/">https://subversion.apache.org/</a> (old standard centralized system)

Fossil, <a href="http://www.fossil-scm.org/">http://www.fossil-scm.org/</a> (unique web interface based system)

#### Install Git on Windows

GitHub Desktop: <a href="https://desktop.github.com/">https://desktop.github.com/</a>

If you are interested in using a visual GUI application integrated with Github, install "Github Desktop" using the default options, . This will give you Git, Git Shell, and Github Desktop GUI. You can install Github Desktop even if you have another version of Git already installed.

Git for Windows: <a href="https://git-scm.com/downloads">https://git-scm.com/downloads</a>

Install using the default options which will give you Git, Git Bash, and Git GUI. Git Bash is a great terminal that lets you use UNIX style commands on Windows.

#### Install Git on Mac

GitHub Desktop: <a href="https://desktop.github.com/">https://desktop.github.com/</a>

If you are interested in using a visual GUI application integrated with Github, install using the default options. This will give you Git and Github Desktop GUI. You can install Github Desktop even if you have another version of Git already installed.

Your system might already have Git installed. Open terminal and type "git --version". If you do not have it, it will prompt you to install. Download the official Mac installer from, <a href="https://git-scm.com/downloads">https://git-scm.com/downloads</a>

#### Install Git on Linux

For Linux users, install Git using your package management (such as "sudo apt-get install git") or software center.

There are a few GUI apps available, check <a href="https://git-scm.com/download/gui/linux">https://git-scm.com/download/gui/linux</a>