Get Your Git On!

Intro to Git & GitHub

Jenn

starsplatter

Cornell University Library
Ithaca, NY
jrc88@cornell.edu
Joined on Aug 13, 2012

3 Followers **64** Starred **8** Following

Organizations

Contributions | Repositories | Public activity

Edit profile

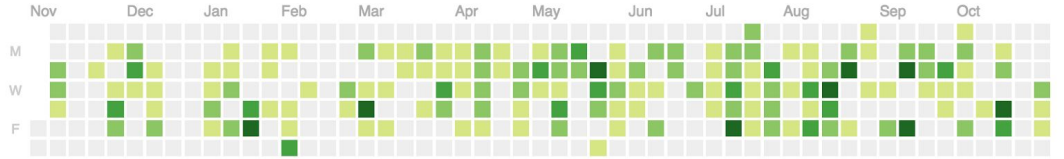
Popular repositories

- **Ubiqu-lty** — Tool for rapidly tagging texts in a corpus — 0 ★
- **arduino** — 0 ★
- **compass** — The Jekyll theme for your personal landing pa… — 0 ★
- **customizing-blacklight** — Jekyll site with content about customizing blac… — 0 ★
- **demo-book** — Demo book to see how github-bookeditor wor… — 0 ★

Repositories contributed to

- cul-it/**singlecore-library-cornell…** — 0 ★
- cul-it/**blacklight-cornell** — 0 ★
- cul-it/**collections-library-cornell…** — 0 ★
- cul-it/**digital-collections-solr** — 0 ★
- cul-it/**law-library-cornell-edu-th…** — 0 ★

Contributions



Summary of pull requests, issues opened, and commits. Learn how we count contributions.

Less More

Contributions in the last year	Longest streak	Current streak
1,051 total	6 days	3 days
Nov 7, 2014 – Nov 7, 2015	July 19 – July 24	November 4 – November 6

Why use git?

- ➔ Make collaboration easier
- ➔ Make fixing mistakes easier
- ➔ Make code experimentation easier

What does Git do?

- ➔ Takes a snapshot of the current state of your files assigns an identifier
- ➔ Works locally
- ➔ Makes it hard to lose data
- ➔ Has files in three states: modified, staged and committed

What about GitHub?

- ➔ Hosting for git repositories
- ➔ Repositories can be public or private
- ➔ Pull request management, issues, wikis, other features for collaboration

Preparation

- ➔ Install Git
- ➔ Join GitHub
- ➔ Set up ssh keys

Set up SSH keys

https://help.github.com/articles/generating-ssh-keys/

Set your name and email in Git

git config --global user.name “Jenn”

git config --global user.email “jrc88@cornell.edu”

git config --list

Create a local repository

cd ~/

mkdir my-repo

cd my-repo

git init

git status

Add a file

touch helloworld.txt

git status

git add helloworld.txt

git status

Edit your text file

Add some text to helloworld.txt and save it

Back at the command line do: git status

git add helloworld.txt

git commit -m “first commit”

What did we do?!

We...

Used ‘add’ so Git would track our file

Used ‘add’ again so Git would track the current state of that file

Used ‘commit’ so Git would save the current snapshot of that file in its repository and assign it a hash

Check the log: git log

Fixing mistakes you haven’t staged

Change some text in helloworld.txt and save it

Go back to the command line and do: git checkout filename

Look at helloworld.txt again and your changes will be gone

Fixing mistakes you have staged

Change the text in helloworld.txt again

Go back to the command line and do: git add helloworld.txt

Check git status

unstage but leave your changes: git reset helloworld.txt

Check git status and then: git add helloworld.txt

git reset HEAD helloworld.txt

git checkout helloworld.txt

Fixing mistakes you have committed

Change helloworld.txt one more time and save

At the command line: git add helloworld.txt

git commit -m “add text to helloworld”

git log --pretty=oneline

git revert [HASH]

Delete a file

touch howdyworld.txt

git status

git add howdyworld.txt

git status

rm howdyworld.txt

git status

git rm howdyworld.txt

Branching

Create a new branch

git checkout -b newbranch

edit helloworld.txt

git add helloworld.txt

git commit -m “experimental changes”

Now what do we have?

to list branches do: git branch

to switch back to the master branch: git checkout master

look at the helloworld.txt file

back on the command line: git checkout newbranch

look at helloworld.txt again

Merging branches

Change to your target branch (the one you want to merge INTO)

git checkout master

and then merge in the branch that has the changes

git merge newbranch

What if there is a conflict?

Edit the first line of helloworld.txt on the master branch

git add helloworld.txt

git commit -m “change first line”

git checkout newbranch

Edit the first line of helloworld.txt

Add and commit your change on newbranch

git merge master

fix the conflicts in your text editor, then add and commit your changes

github.com

Create a repository on github.com

Go to github.com and click the ‘+’ then create new repository

Have it add a readme file

On your local machine do: cd ~/

And then clone the repository from github to your machine:

git clone https-address-from-github

cd repo-name

git status

Commit back to github

Create a file in the repo you cloned from github

Add and commit the new file to your local repo

Then push the new changes back up to the github repo:

git push origin master

Placing an existing directory in git control

Easy way:

Create new repo on github.com

Do a git init on your existing directory

Add all the files to git

Follow set of directions for importing existing repo

Fun with collaboration!

One model of collaboration work flow

Master branch - Everything at master should be ‘right’

Dev branch - Where new features get merged together

Feature branches - Where new work is being done. Don’t commit it to dev unless you think it will work

Collaborate - Forks

Fork a repository on github, mine: https://github.com/starsplatter/icgit or someone else's

Check out your fork to your own machine

Make a change, add and commit to your local repo then push it up to your fork

Go to your fork on the github.com site and create a pull request

The owner of the original repo will then need to merge the pull request

The original repo now has your change

Collaborate: Branches and collaborators

Add each other as collaborators on your project:

Go to your repo on github.com then select settings, then add collaborators

After you've been added as a collaborator on a project you can create your own branches and commits on a project

Other things to try on github

Github pages for yourself/projects: https://pages.github.com/

Repository wikis

Repository issue managers

Following projects/people

Creating groups/organizations