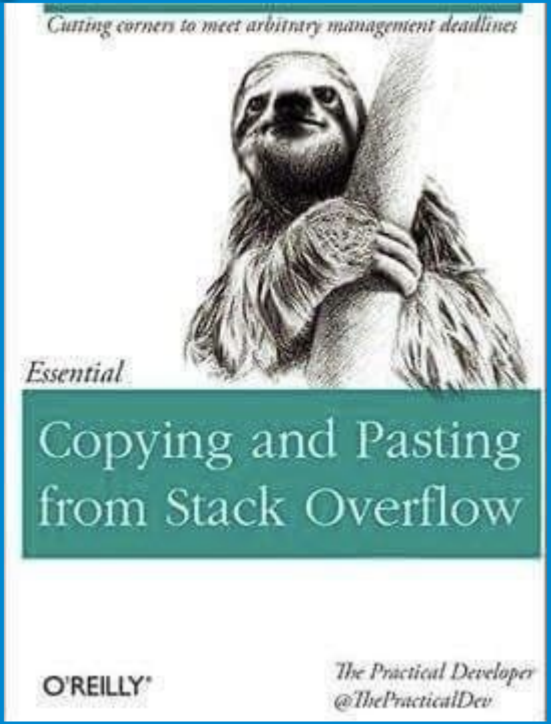
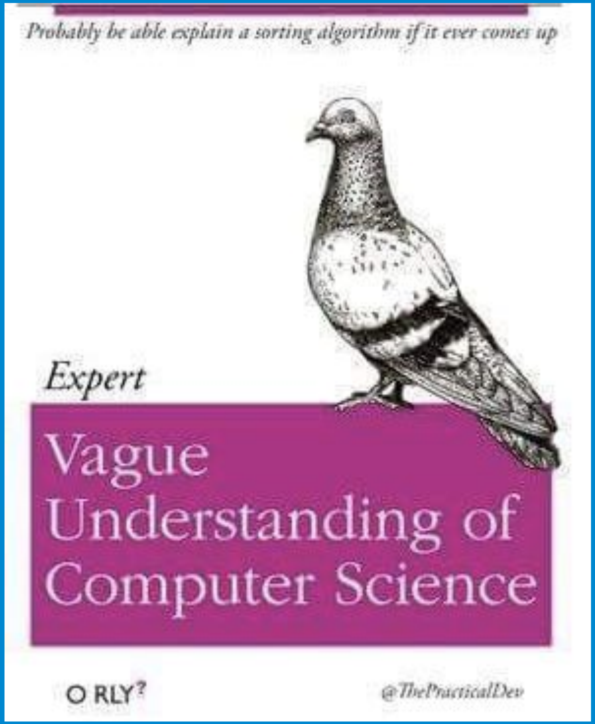
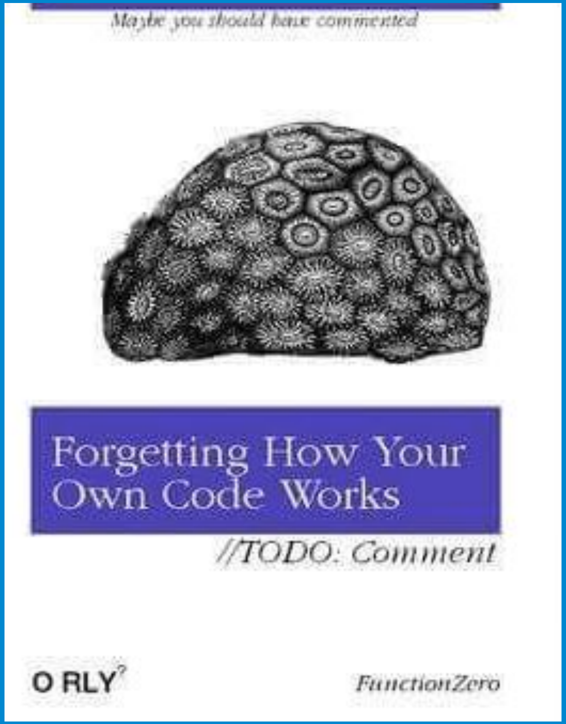
Today's Themes


Cutting corners to meet arbitrary management deadlines
Essential
Copying and Pasting from Stack Overflow
O'REILLY
The Practical Developer
@ThePracticalDev


Probably be able explain a sorting algorithm if it ever comes up
Expert
Vague Understanding of Computer Science
O RLY?
@ThePracticalDev

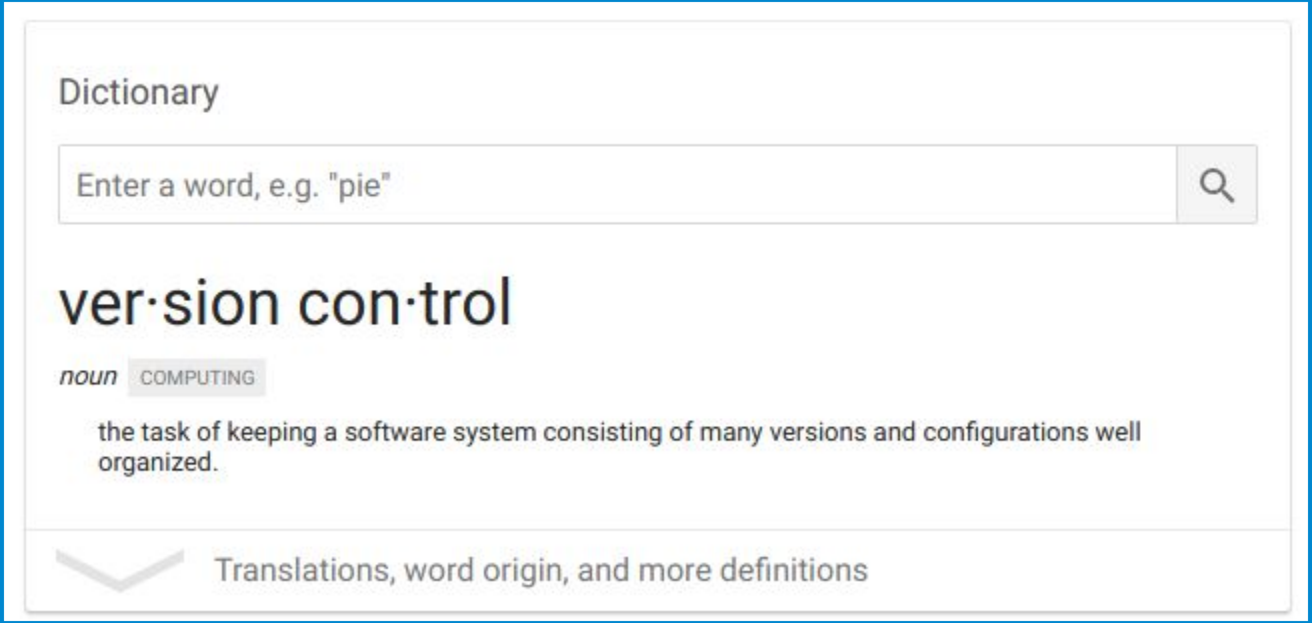

Maybe you should have commented
Forgetting How Your Own Code Works
//TODO: Comment
O RLY?
FunctionZero

Git & GitHub

Control your Code Versions!

Ryan T. Hamilton, Instrument Scientist
Lowell Observatory, September 2018

...Version Control?



Dictionary

Enter a word, e.g. "pie"

ver·sion con·trol

noun COMPUTING

the task of keeping a software system consisting of many versions and configurations well organized.

Translations, word origin, and more definitions

Go Read: https://en.wikipedia.org/wiki/Version_control

Why Should I Use Version Control?

- **When done correctly, it helps you organize and plan**
- Reproducibility is a growing theme in astronomy
  - No one cares about your result using a fancy analysis if they can’t repeat it themselves
- Sharing is Caring
  - How many person-hours have been wasted starting from scratch that could have been instead started at an intermediate point? Don’t always need to rewrite a parser.
- Hard drives are so stupid cheap that you can save almost everything
  - >= $500,000/GiB 1981; <= $0.03/GiB now
  - You can snapshot your entire development process from start to finish

Why Should I Avoid Version Control?

-

Why Should I Avoid Version Control?

Bad Answers to that Last Slide

- "This is a one-time thing"
  - Future you will hate current you
- "It's just me working on this"
  - Superman had visitors to his fortress of solitude. Same with Batman and the BatCave
- "It'll take too long to figure out how to do it"
  - Skill development is a real thing and isn't just a waste of time
    - ***Barriers can be self-imposed or related to norms of the work culture and astronomy is largely terrible at dealing with these***
- "I don't want to share my code"
  - I'm sorry you feel that way, hopefully you change your mind one day
- "I don't like 'cloud' apps or others hosting my data/work"
  - GitHub is an option, not a requirement. Can self-host with GitLab or work completely locally and backup in your preferred way

Ways of Controlling Versions

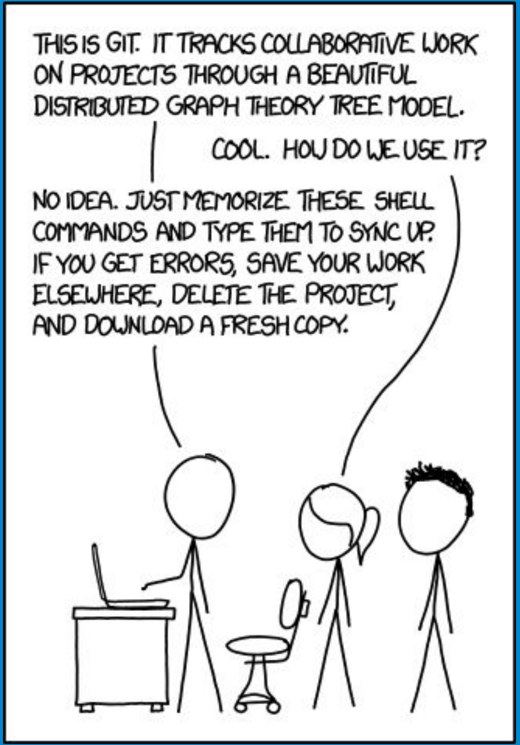
- Common tools:
  - Manual (Important_done.cpp, Important_working.cpp, Important_v5.cpp)
    - You almost always end up having to guess based on file modification dates
  - Git
    - Gold standard for a while now; created for Linux development
    - Integrated with code sharing website GitHub
  - Mercurial (Hg)
    - Solid choice, free online GitHub competitor called BitBucket
  - Concurrent Versions System (CVS)
    - *Please don't use this*
  - Subversion (SVN)
    - Evolution of CVS; it's fine, I just don't like it.
    - SourceForge was the O.G. and boosted SVN, but SourceForge went sketchy in 2015

Git

https://github.com/git/git/tree/e83c5163316f89bfbde7d9ab23ca2e25604af290

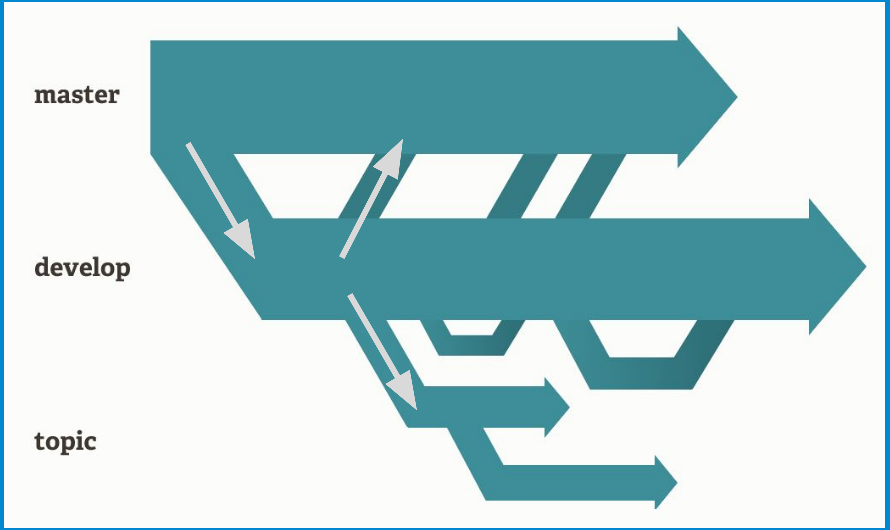
There's Always a Relevant XKCD

If that doesn't fix it, git.txt contains the phone number of a friend of mine who understands git. Just wait through a few minutes of 'It's really pretty simple, just think of branches as...' and eventually you'll learn the commands that will fix everything.

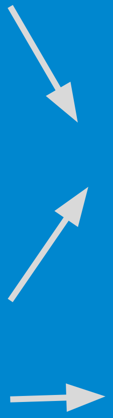

THIS IS GIT. IT TRACKS COLLABORATIVE WORK ON PROJECTS THROUGH A BEAUTIFUL DISTRIBUTED GRAPH THEORY TREE MODEL.
COOL. HOW DO WE USE IT?
NO IDEA. JUST MEMORIZE THESE SHELL COMMANDS AND TYPE THEM TO SYNC UP. IF YOU GET ERRORS, SAVE YOUR WORK ELSEWHERE, DELETE THE PROJECT, AND DOWNLOAD A FRESH COPY.

Theory and Lingo

- Development Model: trees and branches!
  - Can move vertically (up *and* down) in the diagram below using git commands



Time/Commits



The (slight) Catch

- “Commit” == checkpoint

- git only works if you “commit” your changes as you go
    - Each commit is actually stored as just a set of diffs from the previous commit
    - You get to choose when and what files to commit

- Committing binary files is possible, but can get out of hand
    - For that, try Git Large File Storage (Git LFS)

“But I’m not Good with Commitment”

- **Don’t let “future cleanup” stop you from performing commits**
  - Show how the sausage is made! Software doesn’t spontaneously get perfect
- Find what works for you
  - You don’t have to commit every character, but don’t only commit once a month
- *Regular commits allow you to go back and branch off easier*
  - *Also allows you to easier merge in other’s work*
- Get to the point but don’t be vague

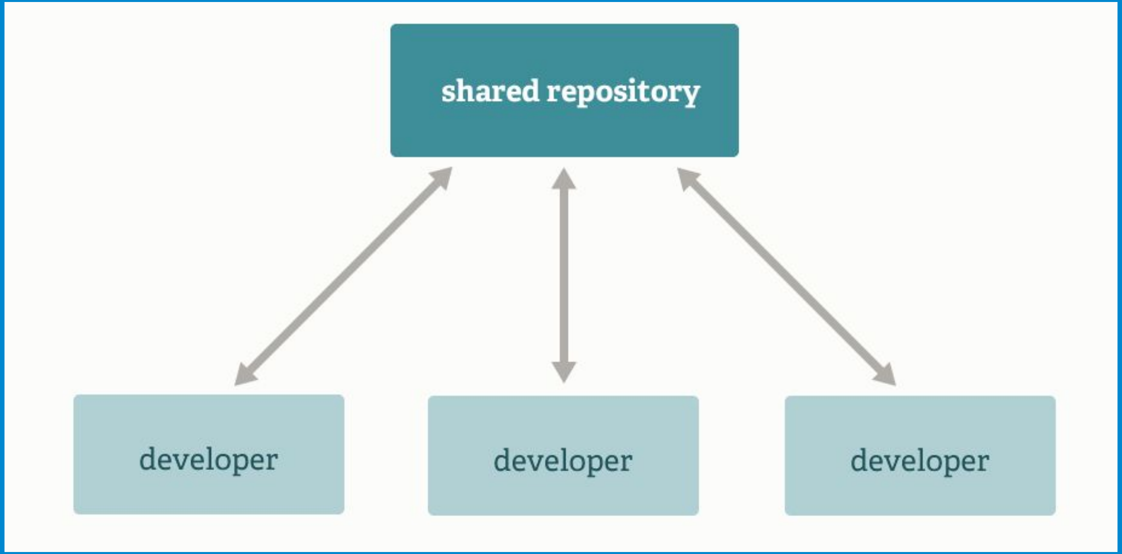
- **Remember:**

  You’re the most likely future user

COMMENT
DATE
CREATED MAIN LOOP & TIMING CONTROL
14 HOURS AGO
ENABLED CONFIG FILE PARSING
9 HOURS AGO
MISC BUGFIXES
5 HOURS AGO
CODE ADDITIONS/EDITS
4 HOURS AGO
MORE CODE
4 HOURS AGO
HERE HAVE CODE
4 HOURS AGO
AAAAAAAA
3 HOURS AGO
ADKFJSLKDFJSDKLFJ
3 HOURS AGO
MY HANDS ARE TYPING WORDS
2 HOURS AGO
HAAAAAAAANDS
2 HOURS AGO
AS A PROJECT DRAGS ON, MY GIT COMMIT MESSAGES GET LESS AND LESS INFORMATIVE.

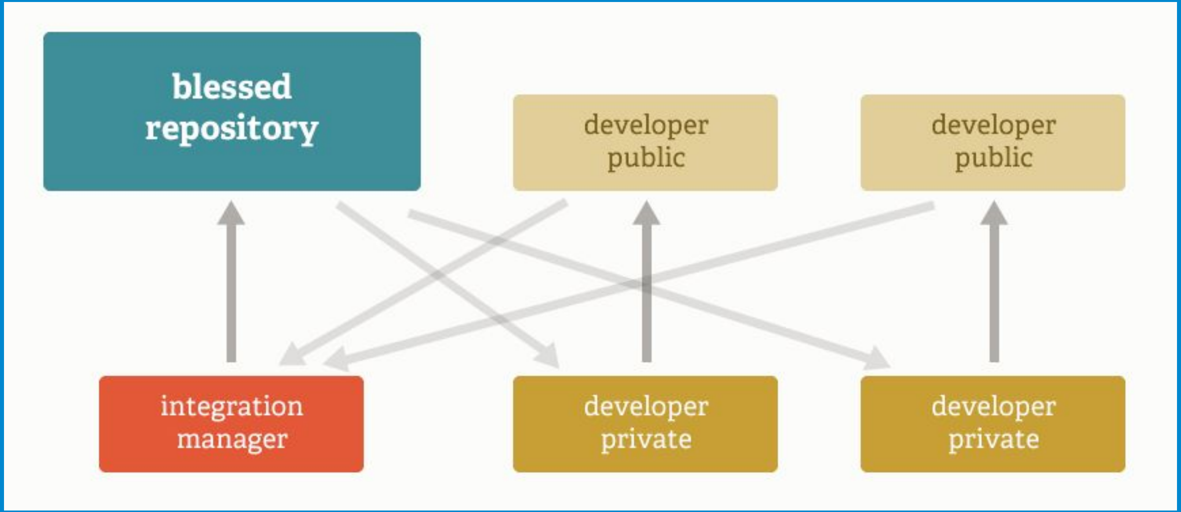
What Git Does

- Contribution types: shared (hippie)



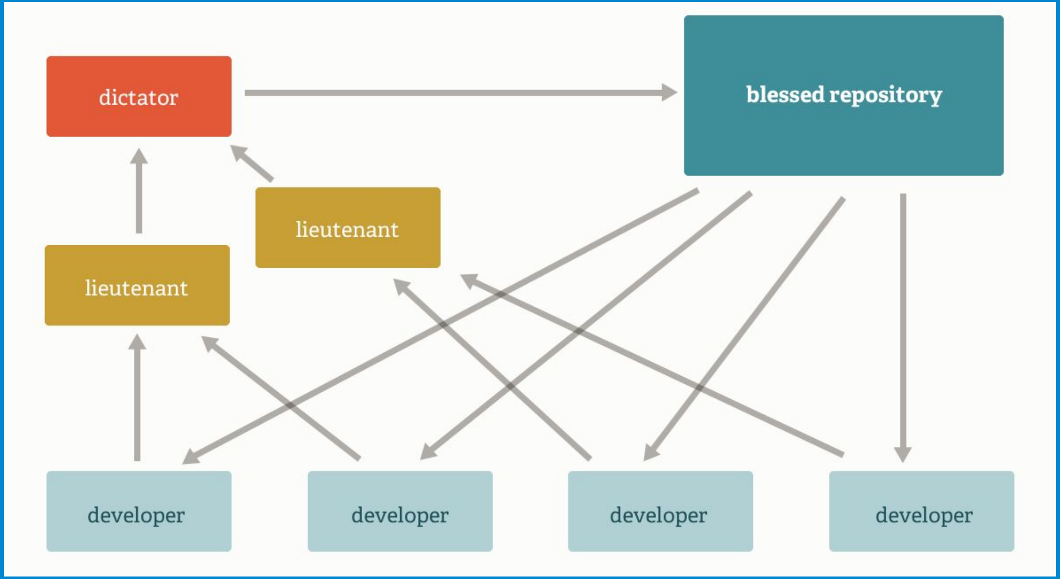
What Git Does

- Contribution types: managed (May The Fork Be With You)



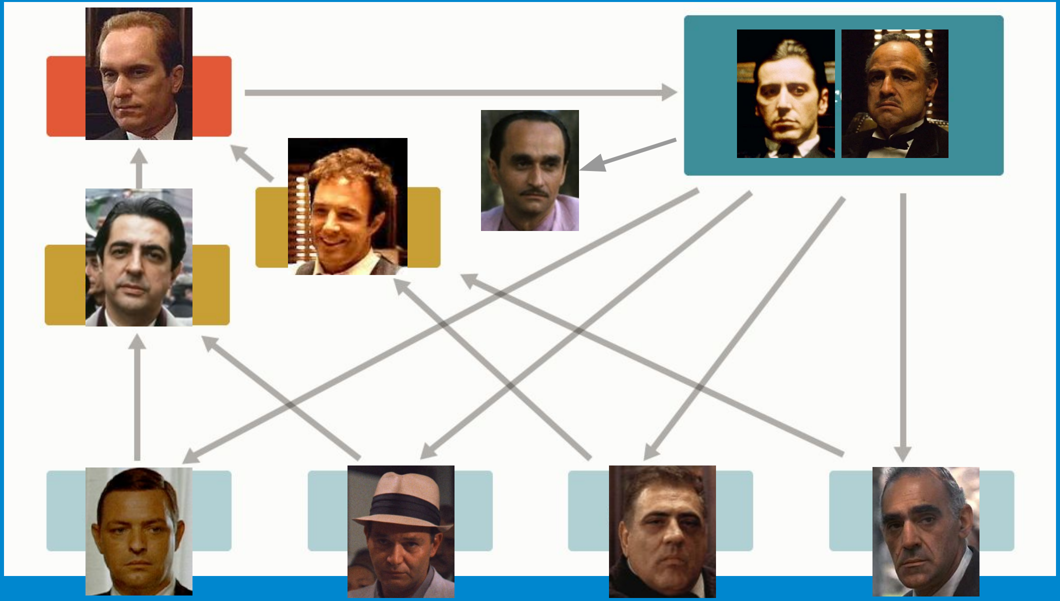
What Git Does

- Contribution types: mafioso



What Git Does

- Contribution types: mafioso



Git Quickstart

- On Macs, if you’ve installed Xcode, you’ve already got git
  - If you haven’t installed Xcode, you can use either Homebrew or MacPorts to get it easily. The Xcode git version tends to be old but either of those will give you a current one
- On Linux, it’s always just an apt-get (or yum or whatever) away
- On Windows, there’s an installer that git distributes
- On most platforms, there are also GUI options that sometimes give you a complete working bundle independent of the methods above
  - Atlassian’s SourceTree (Windows, Mac) is by far my favorite but they don’t have a Linux version. There, Git Kraken is king.
  - Git without a command line! Visual diffs, syntax colors, click to change branches. Awesome.
- ***Your IDE probably has a git plugin too! It’s almost essential.***

Before You Do Anything Else

- git config --global user.name "Your Name"
    - Tell git who you are, will be used for all commit messages
- git config --global user.email your.email@example.com
    - (same as above, really)

- **If you use GitHub (or any codesharing site, really) these could be public!**

- If you want to edit/find them in the future, look in ~/.gitconfig

Git Quickstart

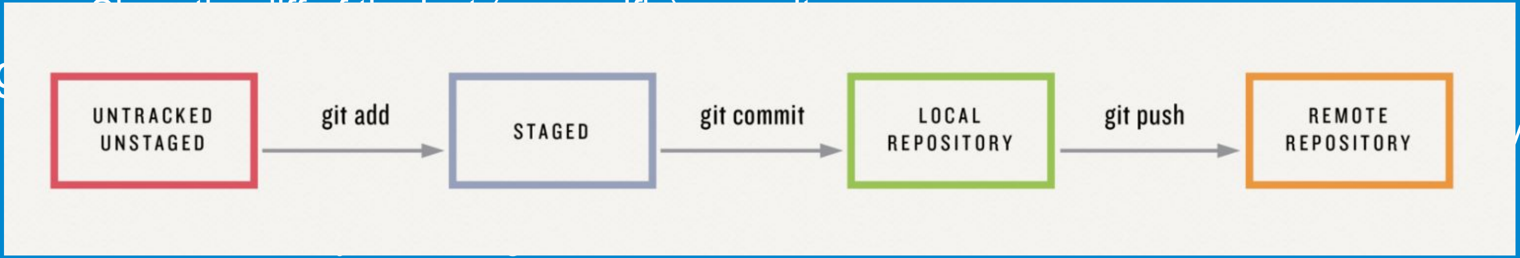
- Create a directory and go into it
  - git init
    - Make some initial files, or start from a completely blank slate
  - git add --all
  - git commit
    - Add in a message describing your changes and then save and quit
- Done! You’ve created your first git repository.
- <do some work>
- “Commit” your new work
  - git status
    - Will show list of modified files
    - git add <modified files>
    - git commit

Additional git Commands

- git log
  - Show your commit history
- git status
  - Show a list of modified files, including any that are new/untracked
- git show
  - Show the diff of the last (or specific) commit
- git [mv, rm]
  - Act on a tracked file, moving (or removing) it and updating it appropriately in the history
  - **‘git rm’ is *not* the inverse of ‘git add’ if you accidentally stage something**
    - For that, you want ‘git rm --cached <file>’
- git [fetch, push, pull]
  - Interact with a remote repository (like Github)
- .gitignore

Additional git Commands

- git log
  - Show your commit history
- git status
  - Show a list of modified files, including any that are new/untracked
- git show



- git [fetch, push, pull]
  - Interact with a remote repository (like Github)
- .gitignore

GitHub

Github Quickstart

- Follow this: https://guides.github.com/activities/hello-world/
- It lets you:
  - Create a repository
  - Copy (clone) another one and work on it
    - Request that the one you cloned include some of your changes ("pull request")
  - Create an "issue" describing a problem in a repository and talk about it
    - See also: Jira
  - Comment on a given commit, or even a line in a specific commit
  - Explore branches, the commit history, etc.
  - Create teams and groups, assigning permissions accordingly
  - Create releases, tagged versions, etc.
  - Use continuous integration systems to build and record each commit
  - **Host static webpages even with HTTPS and custom domain support!**

Demos

- Simple repository: https://github.com/LowellObservatory/DeadParrots
- Simple issue: https://github.com/LowellObservatory/DataServants/issues/8
- Simple project: https://github.com/LowellObservatory/DataServants/projects/1
- Complex everything: https://github.com/astropy/astropy
- Static HTTP: https://github.com/astrobokonon/astrobokonon.github.io
  - https://pages.github.com/
- Quick demo of SourceTree

Github Things of Note

- You should figure out what license you want (or are required) to post your code with. Unlicensed/ambiguous licensed code is dead code.
  - https://help.github.com/articles/licensing-a-repository/
  - https://choosealicense.com/
- Now owned by Microsoft so it could decay when they change CEOs
  - See also: Nokia, Skype, their enterprise/paywall shenanigans
- To have private repositories with collaborators, you're going to have to pay
- It's possible to host a git server which allows pushing from other machines but doesn't have any bells and whistles that a web interface that Github has
  - We have one on jumar already, for example