



(FOR HAMS)

TRISTAN HONSCHIED, NM0TH










FEB 2024

<

>

project

Documents

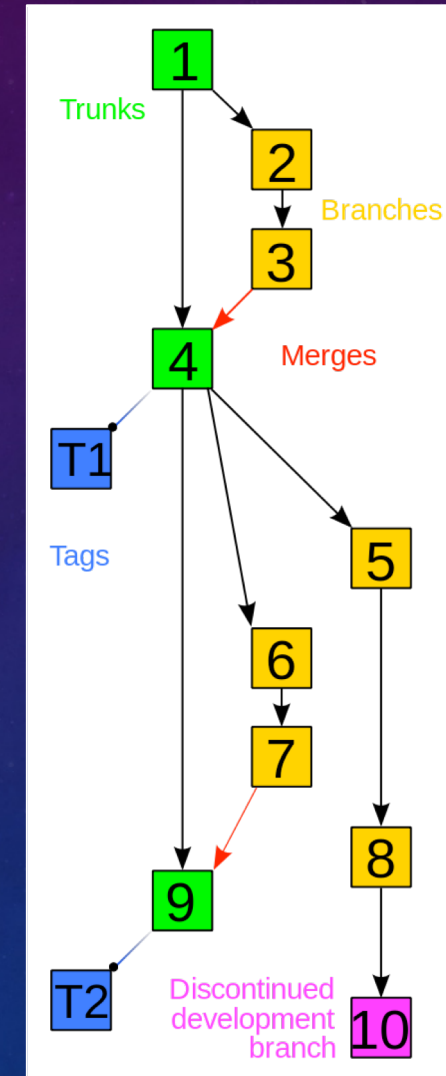
Name	Kind
 ~script (1).py	Document
 dont use this one FINAL.py	Python script
 original script from tyler wednesday.py	Python script
 script (1).py	Python script
 script (2).part.py	Python script
 script final v2.5.py	Python script
 script ready for prod FINAL.py	Python script
 script ready for prod.py	Python script
 script.py	Python script



WHAT IS GIT?



- Version control system
 - Designed for software development
 - But can be used and abused for other things
- Powerful tool for organizing, sharing, and tracking your work



Source:
https://commons.wikimedia.org/wiki/File:Revision_controlled_project_visualization-2010-24-02.svg



OTHER VERSION CONTROL SYSTEMS

- Subversion (SVN)
- Mercurial (Hg)
- Revision Control System (RCS)
- Commercial offerings...



PERFORCE



MISCONCEPTIONS

- “But I’m not a software developer!”
 - Software playing a larger role in Ham Radio than ever before
 - Also useful for storing configs, documentation, KiCad projects, codeplugs, etc
- “It’s too much overhead for what I need”
 - Simple workflows exist
 - Lots of benefits
- “Won’t everything I write be public on Github.com?”
 - No -- Totally different things



USING GIT

```
C:\Windows\System32\cmd.exe

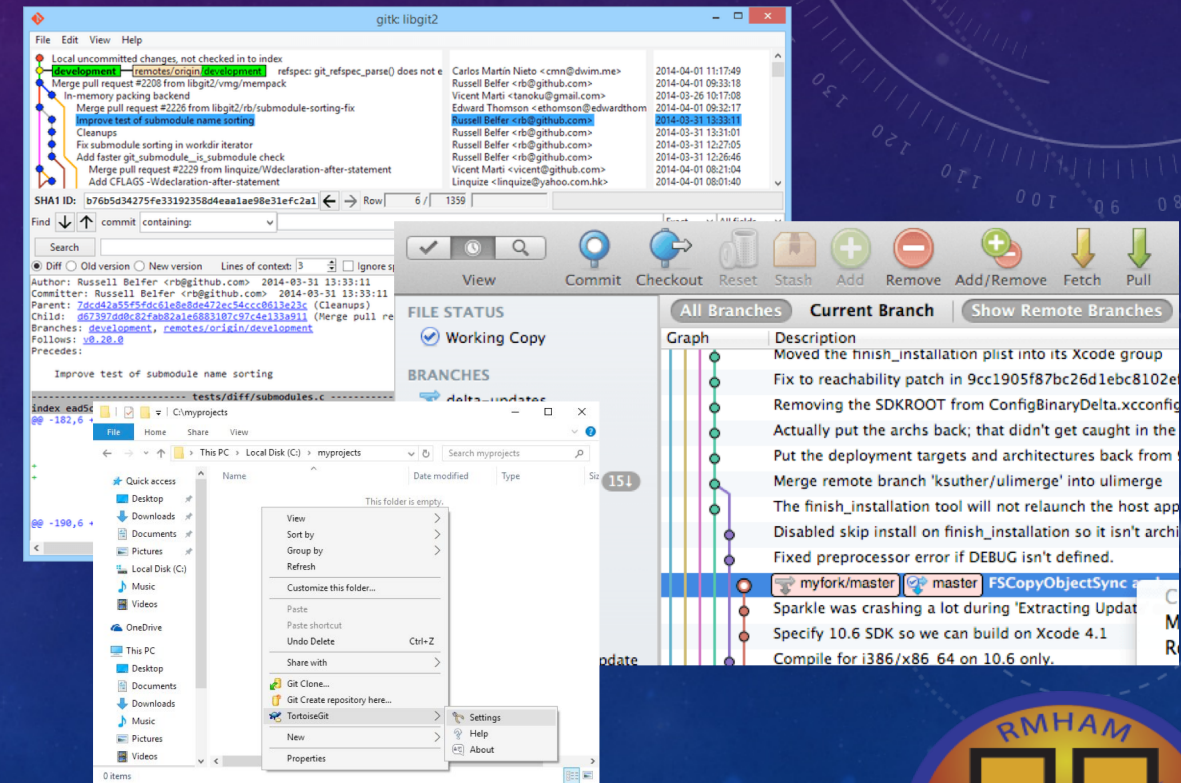
D:\GFG>git status
On branch master

Initial commit

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

    new file:   Basic Code/C++ Project.cbp
    new file:   Basic Code/README.md
    new file:   Basic Code/bin/Debug/C++ Project.exe
    new file:   Basic Code/data.in
    new file:   Basic Code/data.out
    new file:   Basic Code/main.cpp
    new file:   Basic Code/obj/Debug/main.o
    new file:   DS/stackPar.cpp
    new file:   DS/stackPar.exe
    new file:   DS/stackPar.o
    new file:   DS/stlSort.cpp
    new file:   DS/structcar.cpp
    new file:   DS/triplesum.cpp
    new file:   Debugging/Debugging.cbp
    new file:   Debugging/Debugging.depend
    new file:   Debugging/Debugging.layout
    new file:   Debugging/bin/Debug/Debugging.exe
    new file:   Debugging/bin/Release/Debugging.exe
    new file:   Debugging/main.cpp
    new file:   Debugging/obj/Debug/main.o
    new file:   Debugging/obj/Release/main.o
```

Command line



GUI clients



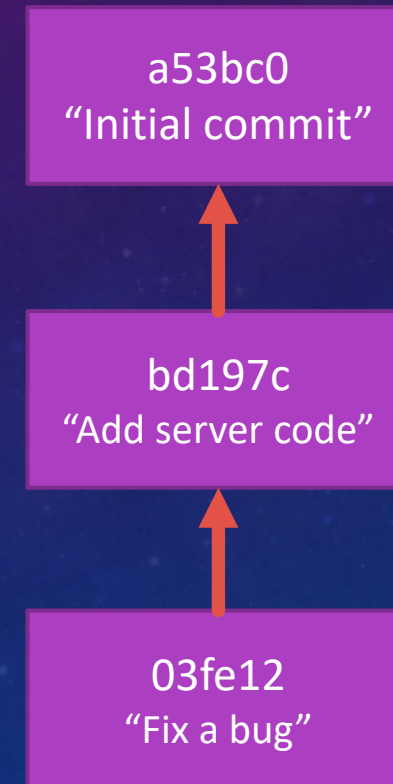
STARTING OUT

- **Init**: Start a brand new repository locally (`git init`)
 - Run inside your project directory
 - Creates a `.git` directory to store the Git database
 - Add files now or later
- **Clone**: Copy a repository from somewhere else (`git clone`)
 - Creates a populated working tree plus `.git` directory



COMMITS

- Each commit:
 - Stores a snapshot of your working tree
 - Has an author, date, and description (commit message)
 - Points to one or more parent commits
 - Is identified by a hexadecimal hash
- `git show <hash>`
- `git log`



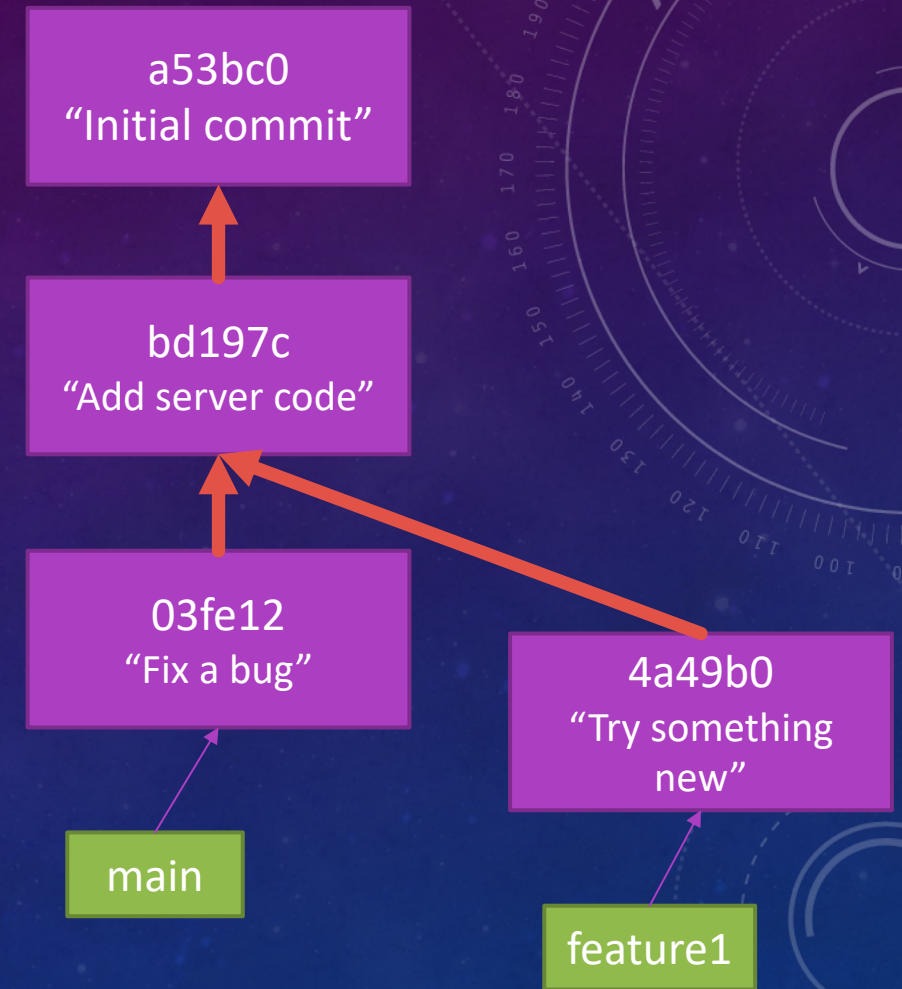
CREATING COMMITS

- Tracked vs untracked files
 - By default, everything is *untracked*.
 - Good to track: source code, documentation, etc...
 - Leave untracked: build outputs, temp/autosave files, editor or file explorer litter (thumbs.db, .DS_Store, etc)
 - .gitignore
- Staging
 - Select changes to go into a commit
 - Stage changes by **add**-ing them (`git add <file>`)
- Commit the staged changes (`git commit`)



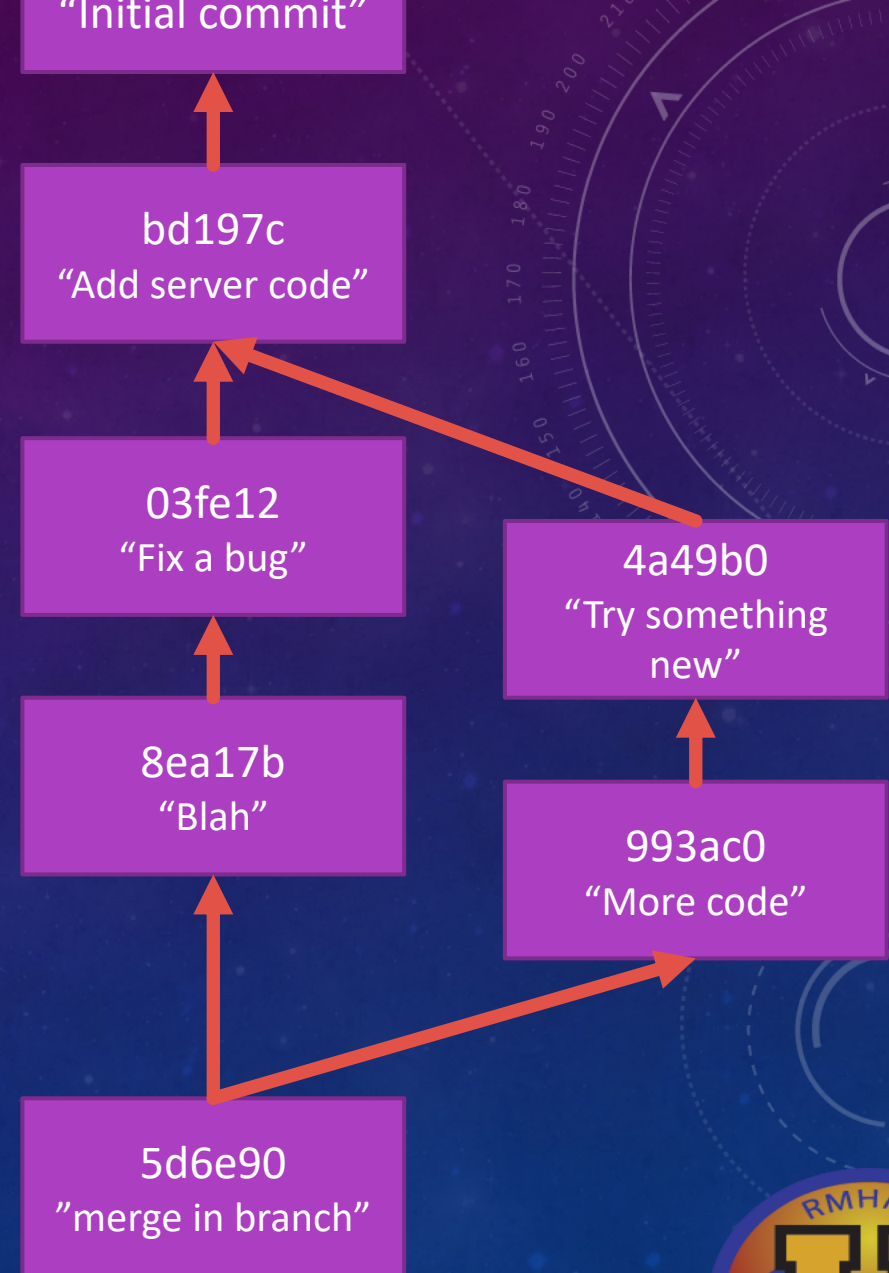
BRANCHING

- You've been on a default branch called `main`.
- Branches are just pointers
- Let you diverge and work on concurrent features
- `git branch` and `git checkout`



MERGING

- Combine lineages back together (`git merge`)
 - Creates a merge commit with multiple parents
 - Potential for **merge conflicts**
 - Need to be manually rectified
- Alternative method: rebasing
 - Replay commits over another branch



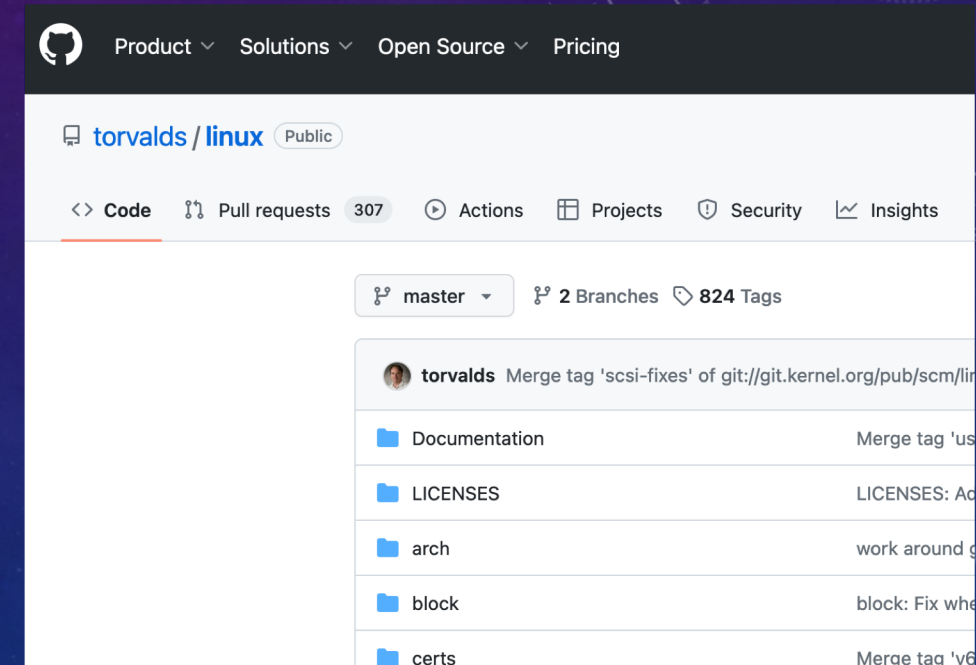
REMOTES

- Git can send and receive commits to/from other instances of the repository
 - Collaborate among multiple users
 - Can be on same filesystem but usually a remote system accessed via HTTP or SSH
- Frequently, the remote is a repository hosting service like Github, Gitlab, Bitbucket, etc...
- `git push`, `git pull`, `git clone`, `git fetch`

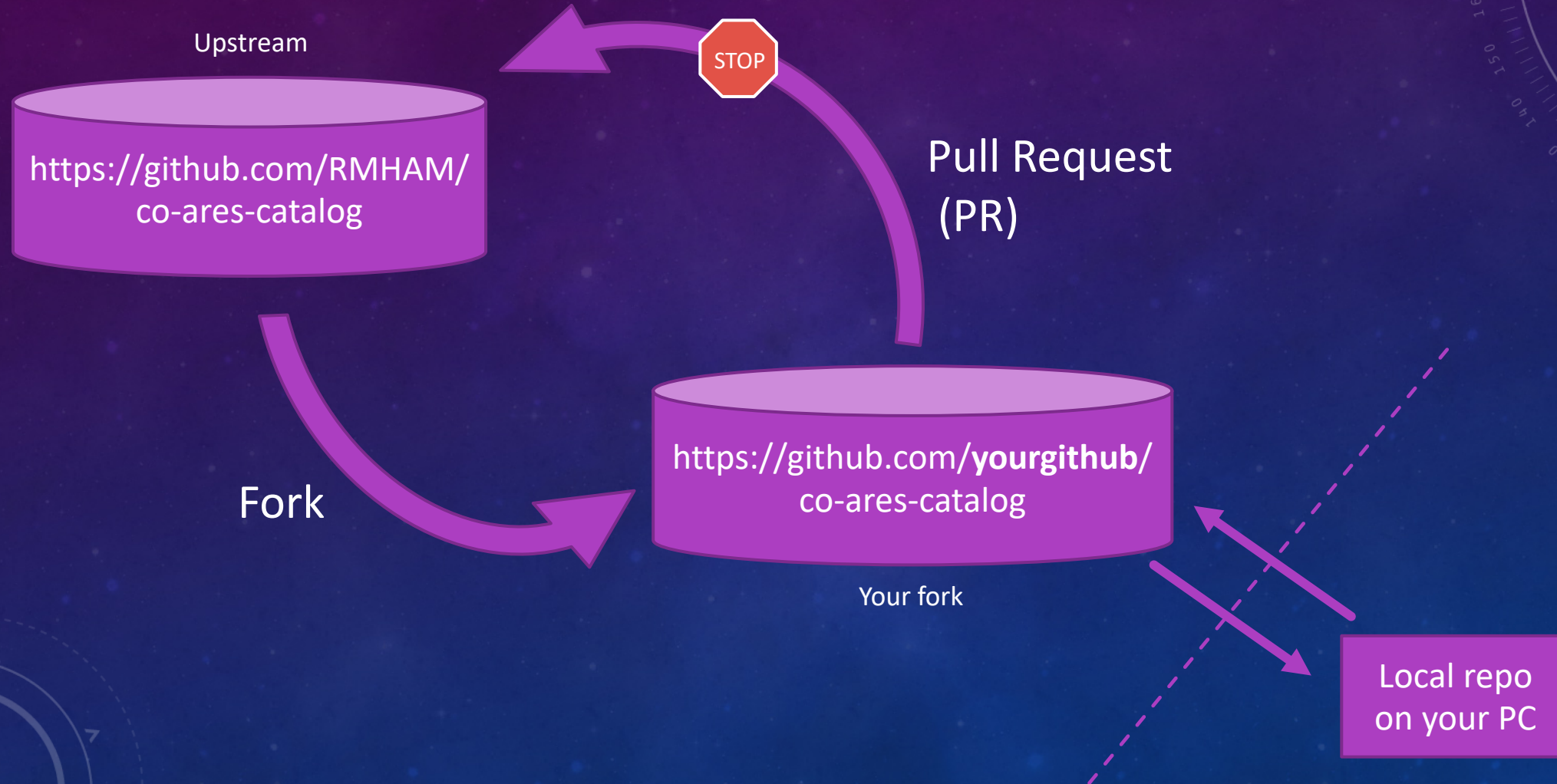


GIT HOSTING PROVIDERS (GITHUB, ETC)

- Provide a "hub" for multiple users to work off of.
- Safe storage of the repository
- Web front-end for viewing code
- Collaboration tools (issue tracker, wikis, etc)
- CI/CD systems
- Access control
- Fork somebody else's repo so you can work on it
- Pull request / merge request –procedure for approving proposed changes before merging



TYPICAL OPEN-SOURCE CONTRIBUTION WORKFLOW



FURTHER READING

- Official reference and tutorial: <https://git-scm.com/doc>
- Lots of other guides and tutorials available online

