Git Workshop

Download git

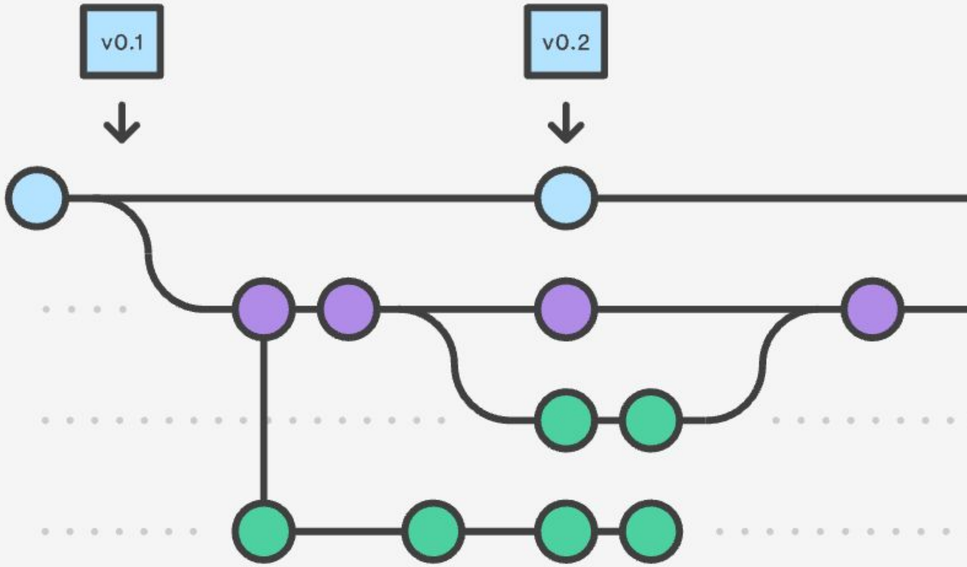
git-scm.com/downloads

Brief History

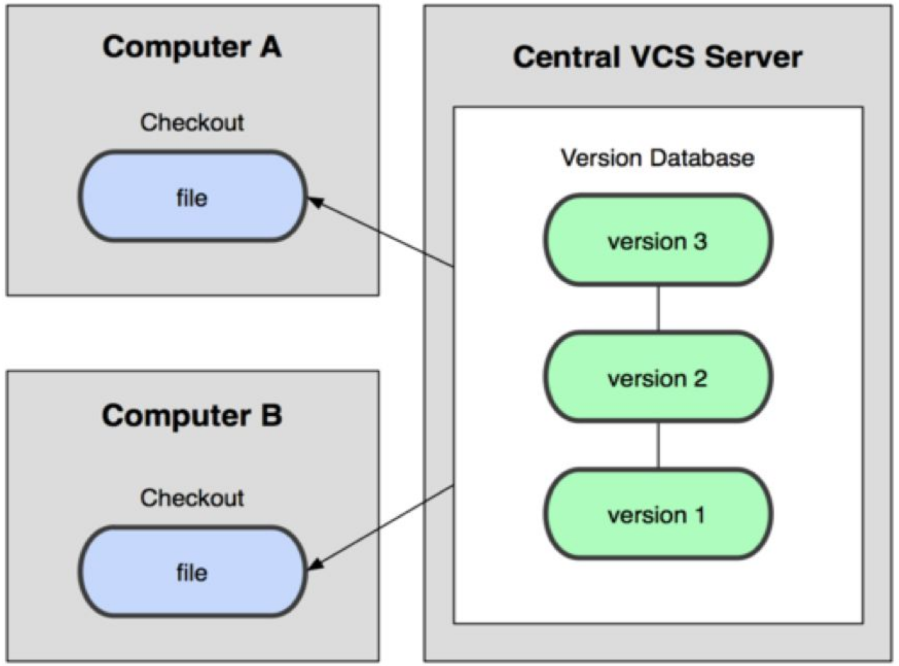
- Large projects require complex coordination - want to avoid “duplicated work”
- Used to be word-of-mouth (I am working on file A, you do file B) Only one person could be working on a file at a time
  - Examples: RCS, SCCS. Very simple (also old)
- Line-based coordinations, merge before commit, people could modify lines at a time, changing the same file as another person
  - Examples: CVS, Subversion
- Changesets, commit before merge
  - Bazaar, Git, Mercurial

What’s VCS? (Version Control System)

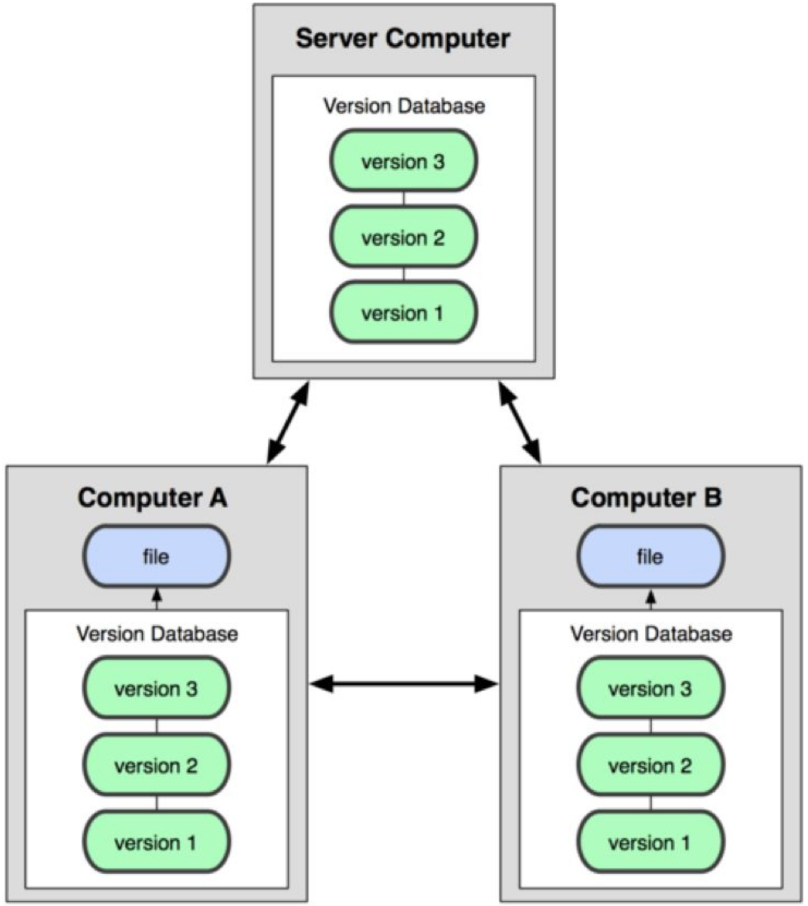
- Allow easy organization of large projects
- Classify changes in “commits” which you broadcast to other contributors.
- Tracks and provides control over changes to source code



Centralized



Distributed



So how about Git?

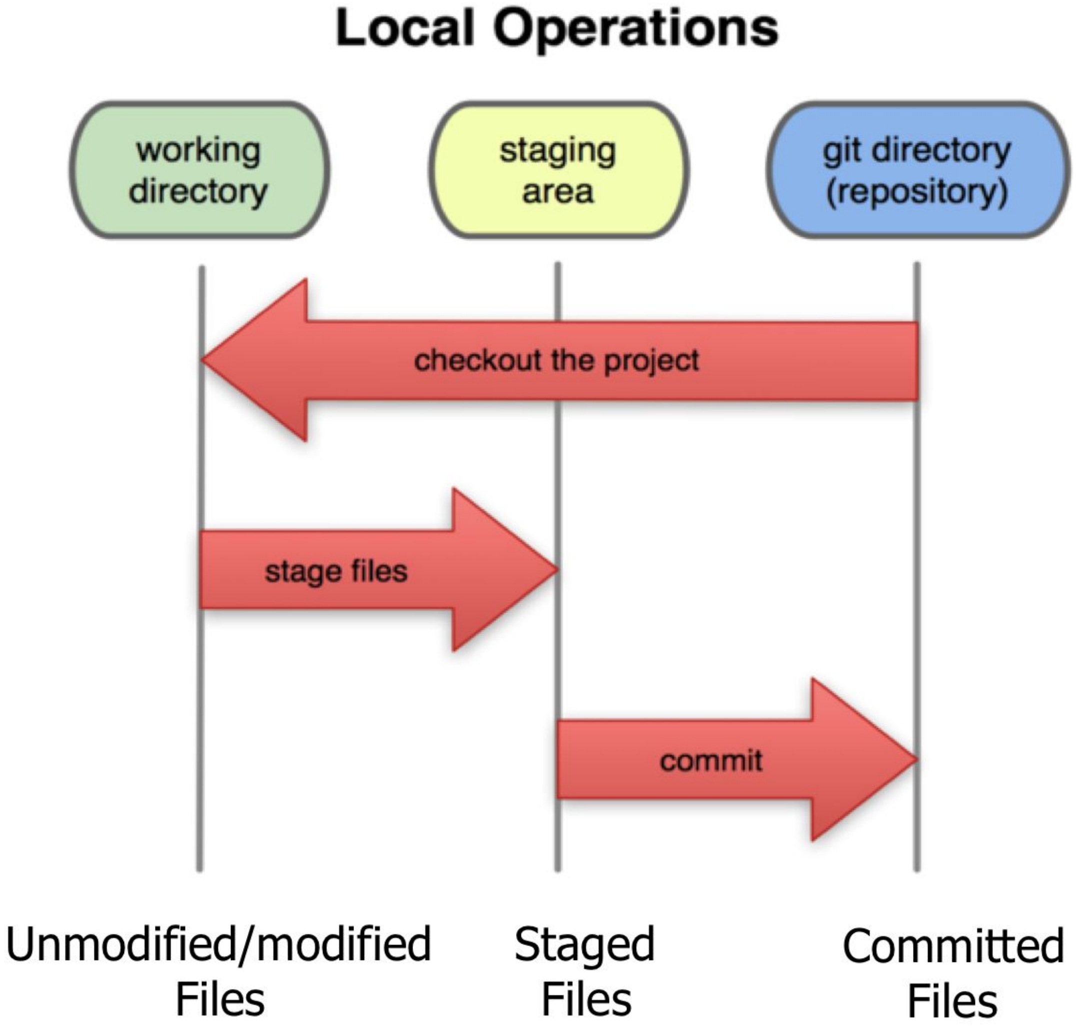
- Created by Linus Torvalds - The guy who wrote linux
  - The linux kernel is vast, very complex, and has many contributors
  - He was annoyed with existing solutions just not quite cutting it
- Some goals:
  - Speed
  - Support for non-linear development (thousands of branches)
  - Fully distributed (no-one owns the project, technically)
  - Able to handle large projects efficiently

How’s it work?

- When you want to contribute to a project, you clone it
- This gives you a complete copy of the project - called “local”
- Your local copy has a reference to where you cloned from, called “remote” (typically github or bitbucket)
- Make changes to local (commits, merges, etc…) and push, or publish, to the remote.

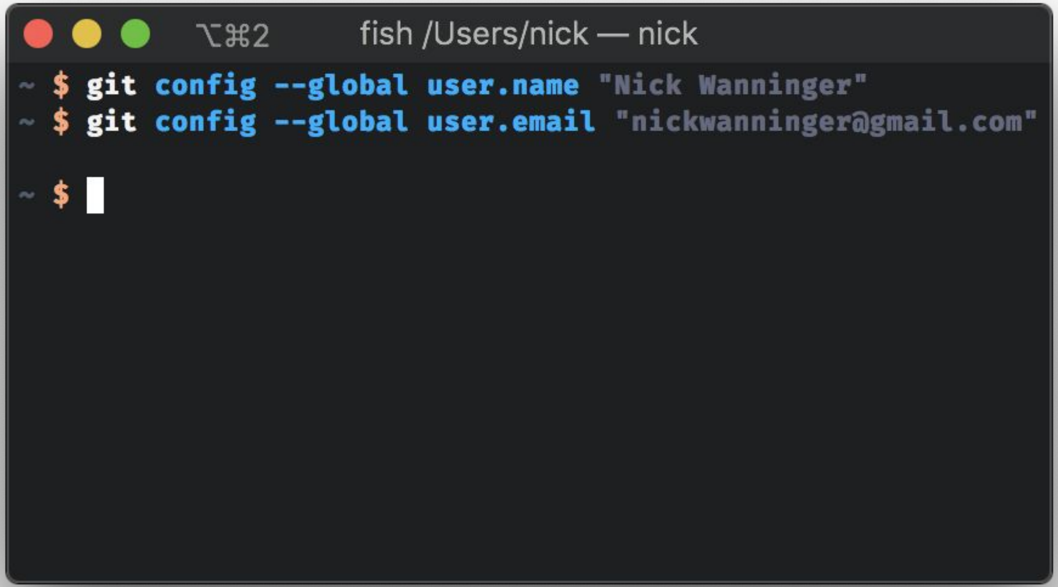
Making Changes

- Just make changes, and when your feature or bugfix works, commit it!
- Stage your changes,
- Committing makes a new “node” on the tree, at the end of your branch

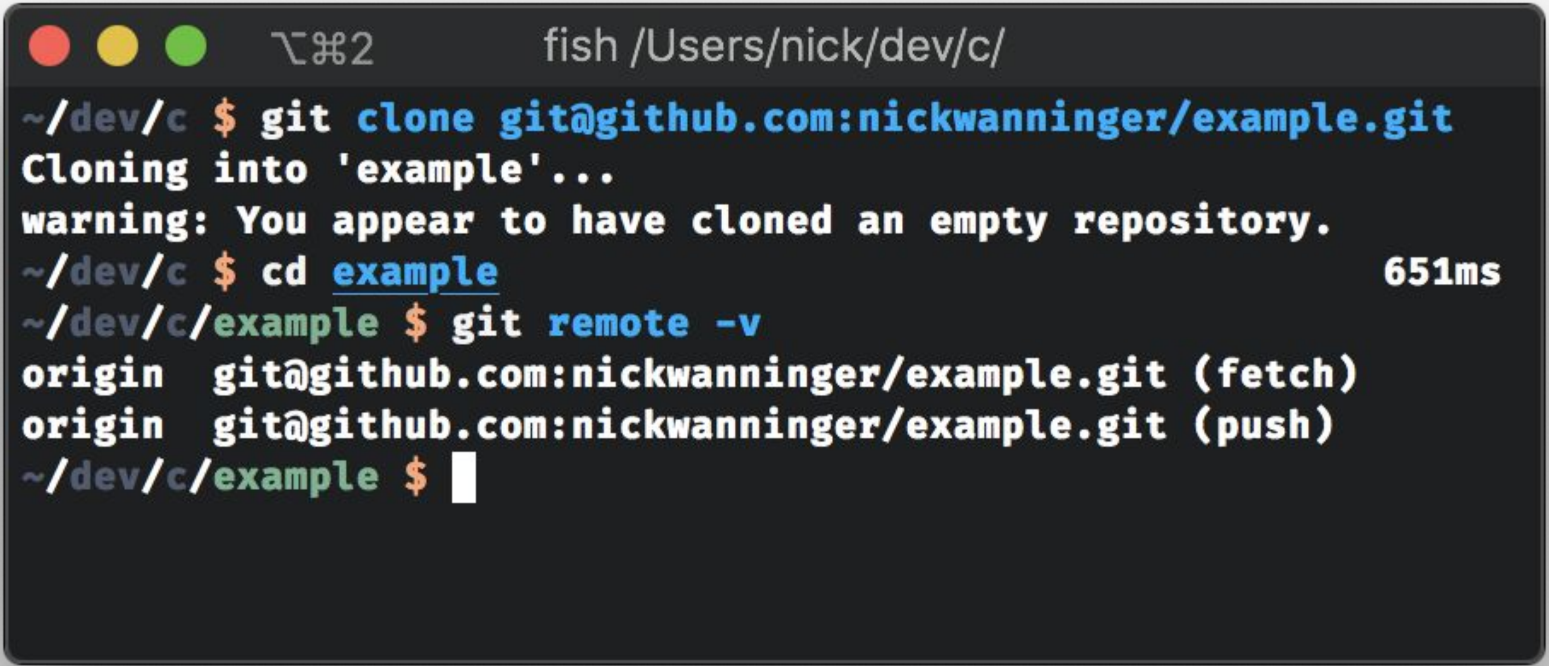


Configuring

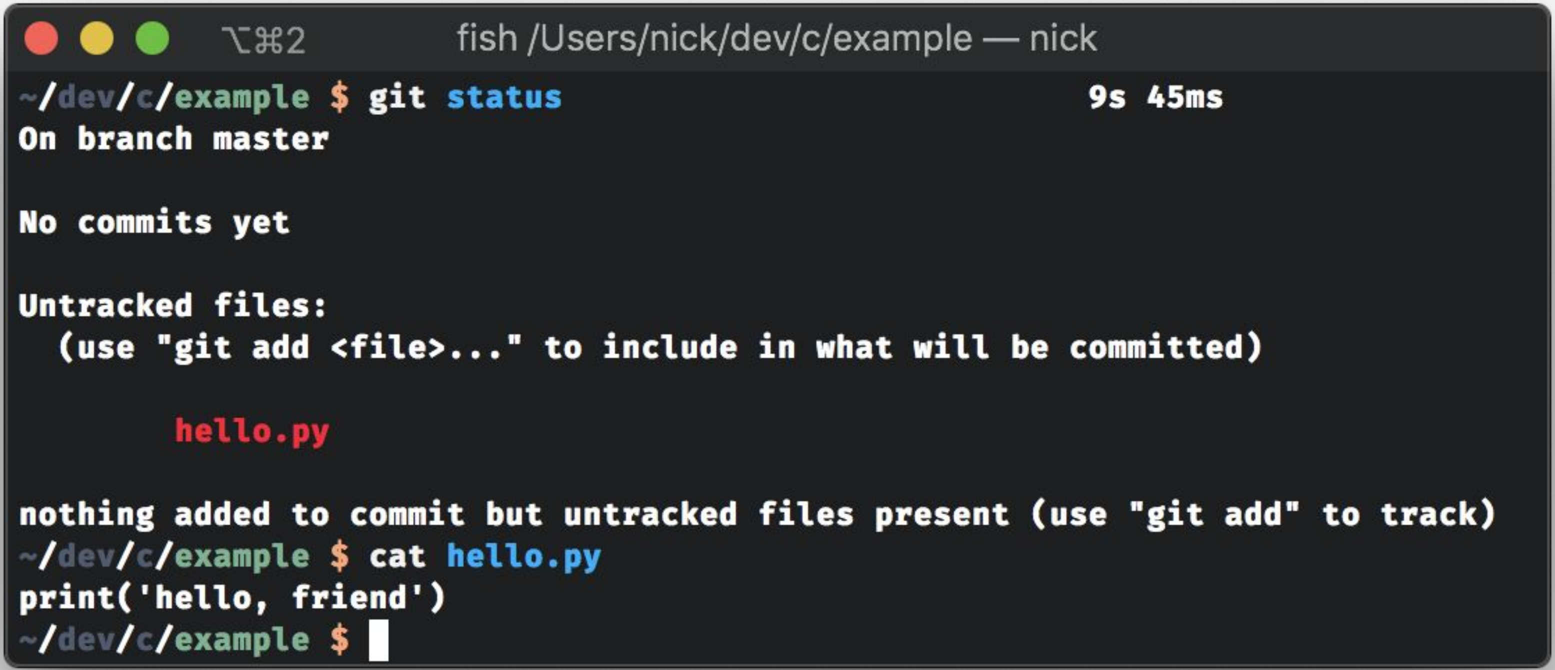
git config allows you to adjust settings, such as global user and email. These are important since they are used to identify you when you make commits.



```
~ $ git config --global user.name "Nick Wanninger"
~ $ git config --global user.email "nickwanninger@gmail.com"

~ $
```

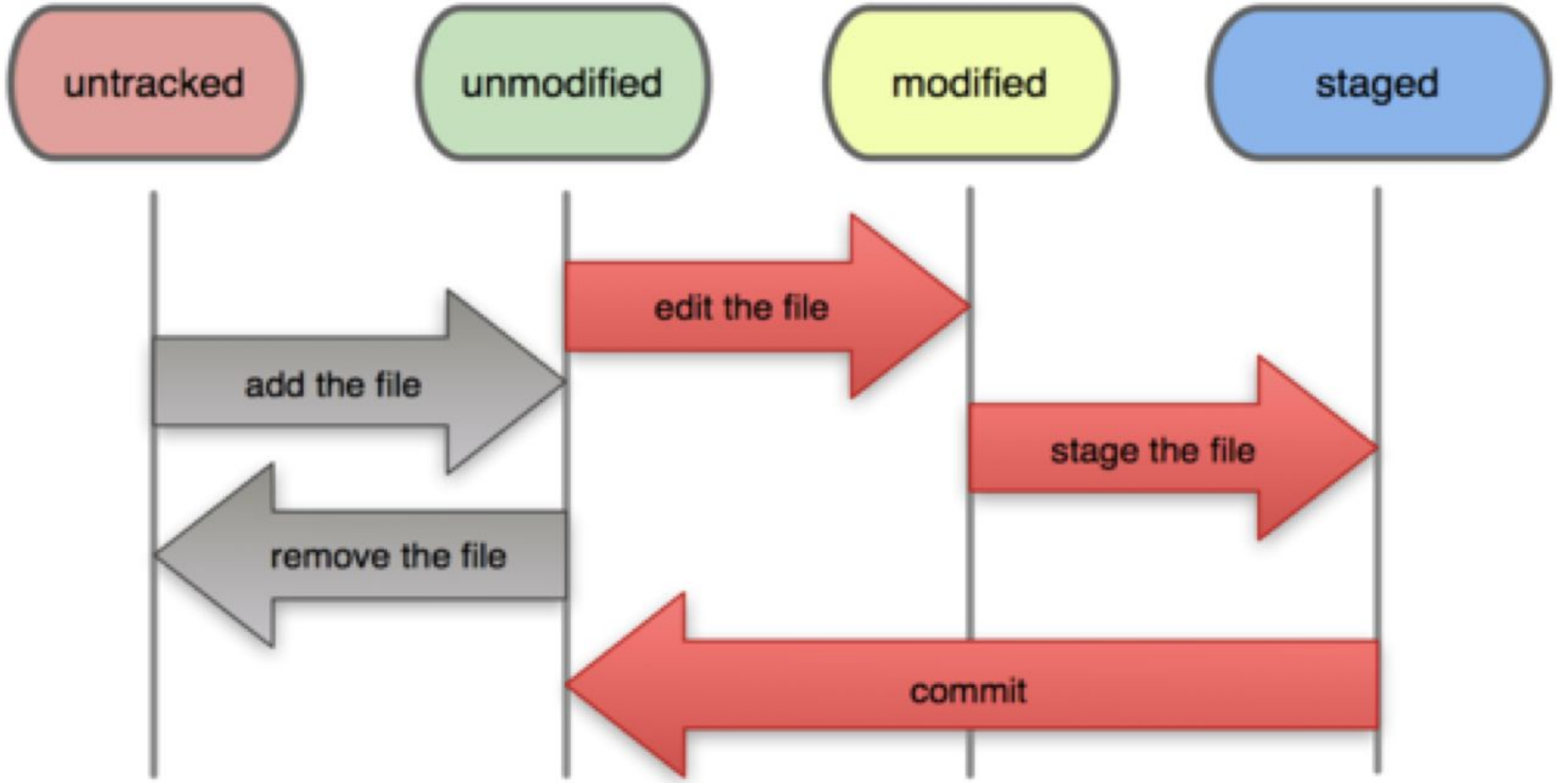


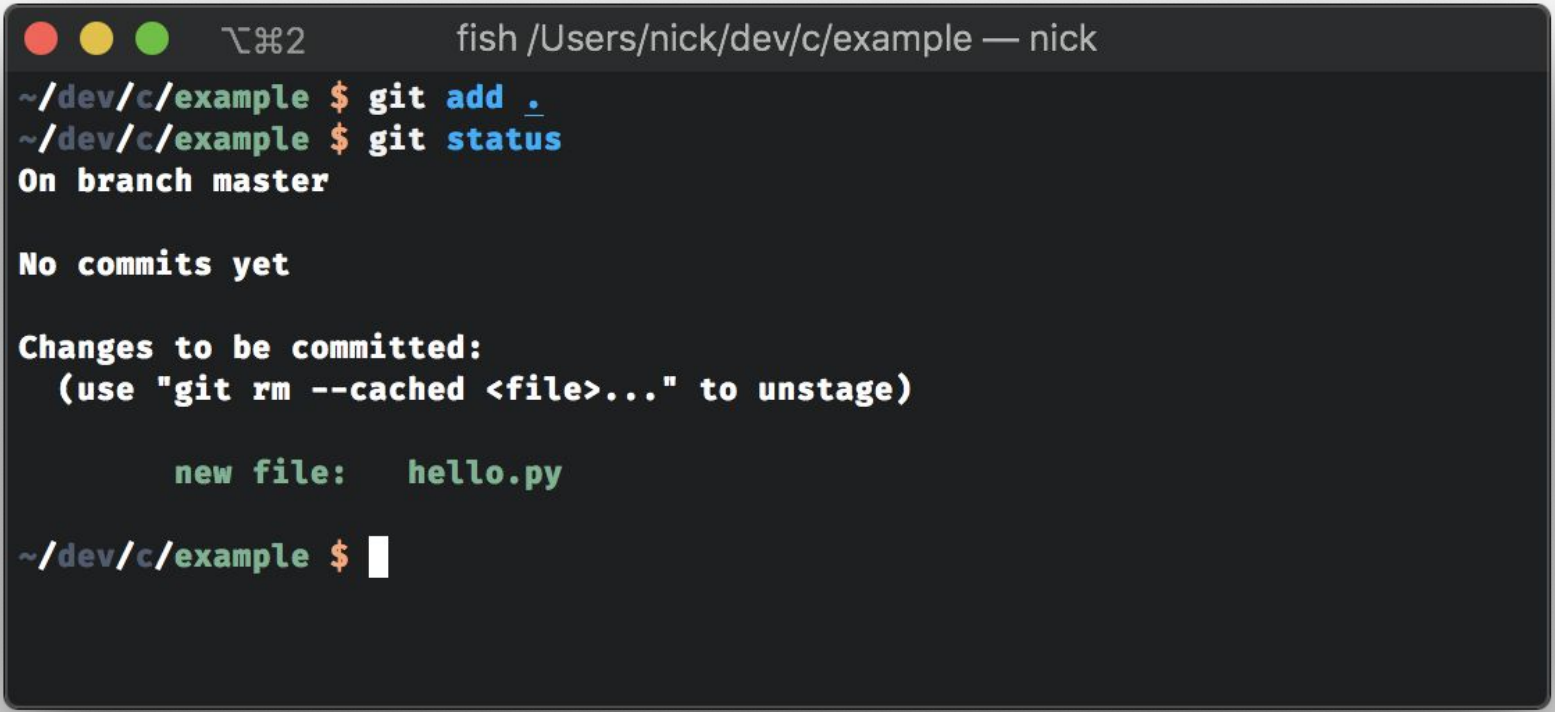
```
~/dev/c $ git clone git@github.com:nickwanninger/example.git
Cloning into 'example'...
warning: You appear to have cloned an empty repository.
~/dev/c $ cd example                                              651ms
~/dev/c/example $ git remote -v
origin  git@github.com:nickwanninger/example.git (fetch)
origin  git@github.com:nickwanninger/example.git (push)
~/dev/c/example $
```



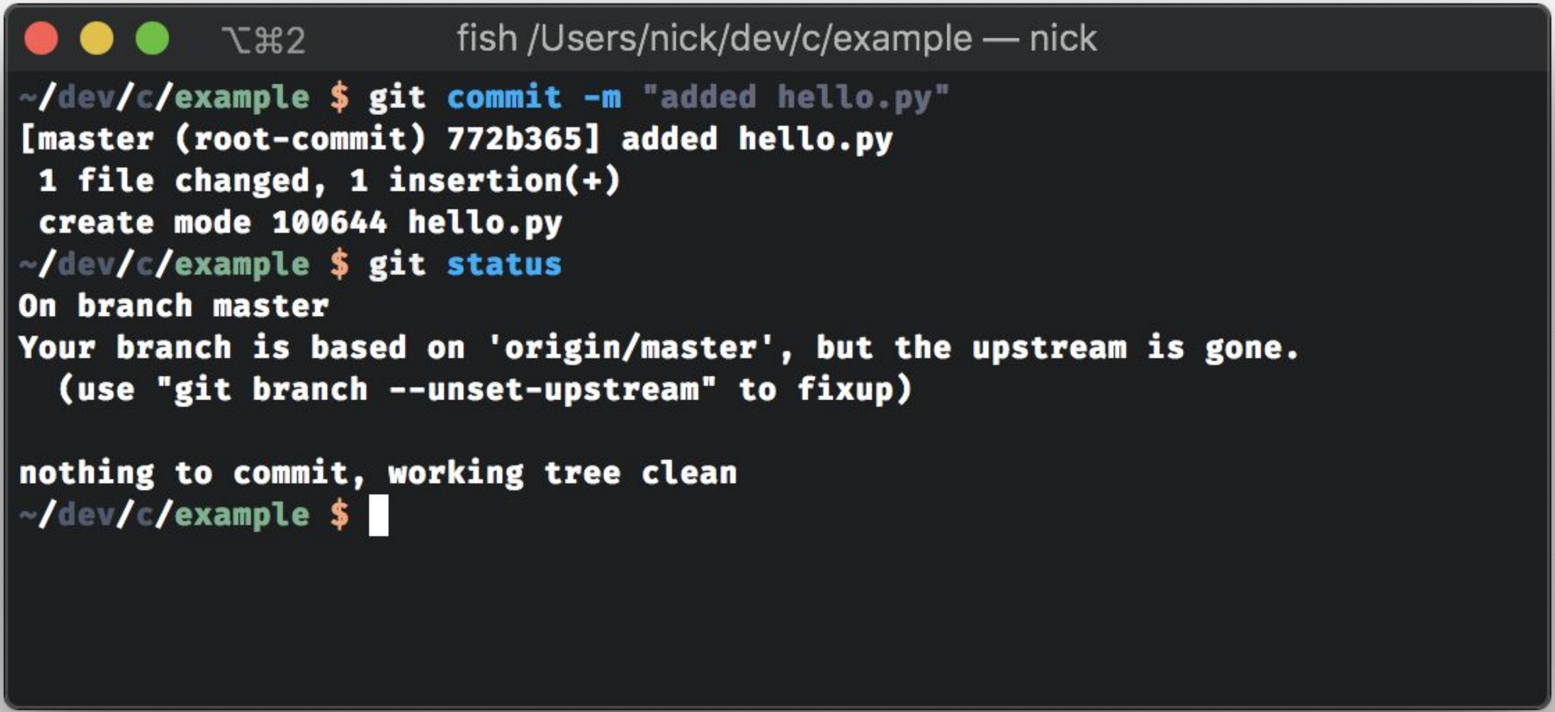
```
⌥⌘2          fish /Users/nick/dev/c/example — nick
~/dev/c/example $ git status                                  9s 45ms
On branch master

No commits yet

Untracked files:
  (use "git add <file>..." to include in what will be committed)

        hello.py

nothing added to commit but untracked files present (use "git add" to track)
~/dev/c/example $ cat hello.py
print('hello, friend')
~/dev/c/example $ 
```

File Status Lifecycle





```
~/dev/c/example $ git add .
~/dev/c/example $ git status
On branch master

No commits yet

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

        new file:   hello.py

~/dev/c/example $
```



```
~/dev/c/example $ git commit -m "added hello.py"
[master (root-commit) 772b365] added hello.py
 1 file changed, 1 insertion(+)
 create mode 100644 hello.py
~/dev/c/example $ git status
On branch master
Your branch is based on 'origin/master', but the upstream is gone.
  (use "git branch --unset-upstream" to fixup)

nothing to commit, working tree clean
~/dev/c/example $ 
```

Common commands

- **git status** // view changes, staging area, added files, etc…
- **git diff** // see what’s been changed
- **git diff --cached** // view staged changes
- **git log** // see a log of changes in your local repo
- **git add <file>** // stage a file
- **git commit** // commit all staged files
- **git push <remote> <branch>** // tell remote about changes
- **git fetch <remote> <branch>** // ask remote about changes
- **git merge** // merge remote changes into your local copy
- **git pull** // git fetch && git merge, basically (not really)
- **git help** // the most important one!

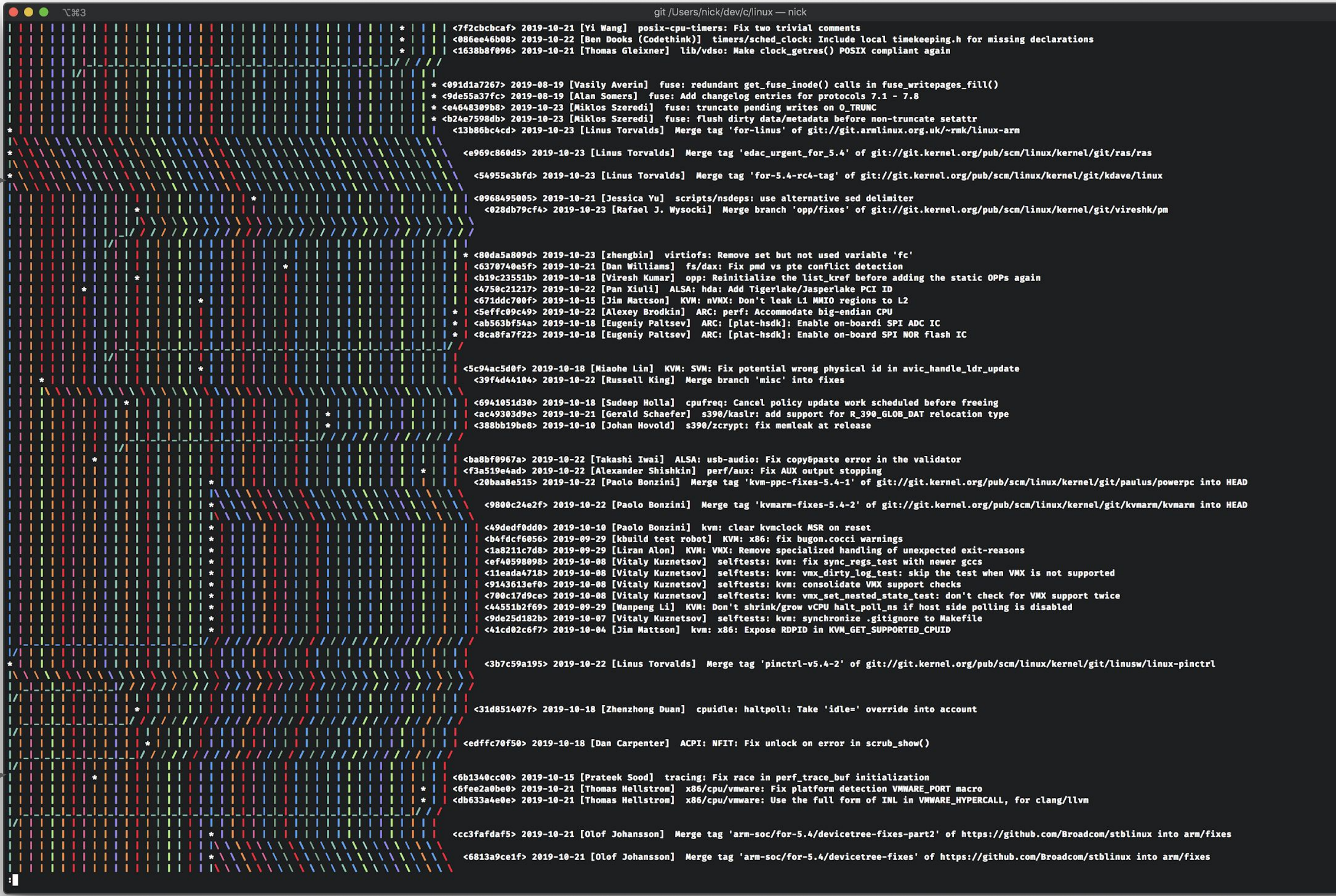
Branches

- Branches let you make changes independent of other people, then merge in later
- For example, you are working on a big feature - put it in a branch
  - Why? So you can freely change files you want, then resolve conflicts later on without muddying up someone else’s work while they are doing their own thing
- This can get very complicated, though

Linux Last Night

Merges
(pull request)

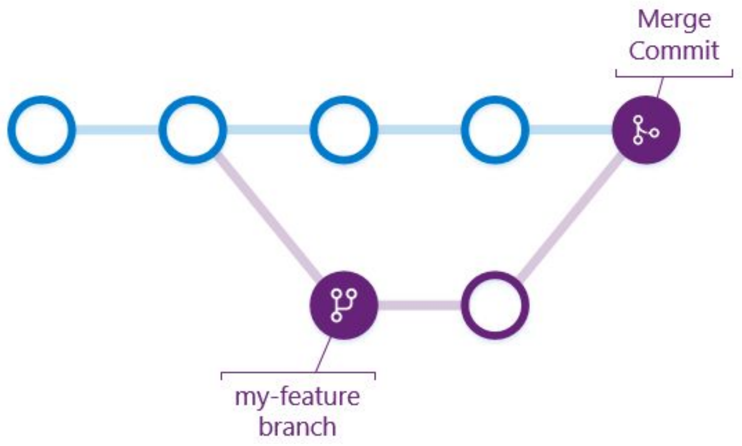
Branch
`git branch -b "..."`


git /Users/nick/dev/c/linux — nick

```
<7f2cbcbcaf> 2019-10-21 [Yi Wang]  posix-cpu-timers: Fix two trivial comments
<086ee46b08> 2019-10-22 [Ben Dooks (Codethink)]  timers/sched_clock: Include local timekeeping.h for missing declarations
<1638b8f096> 2019-10-21 [Thomas Gleixner]  lib/vdso: Make clock_getres() POSIX compliant again

<091d1a7267> 2019-08-19 [Vasily Averin]  fuse: redundant get_fuse_inode() calls in fuse_writepages_fill()
<9de55a37fc> 2019-08-19 [Alan Somers]  fuse: Add changelog entries for protocols 7.1 - 7.8
<e4648309b8> 2019-10-23 [Miklos Szeredi]  fuse: truncate pending writes on O_TRUNC
<b24e7598db> 2019-10-23 [Miklos Szeredi]  fuse: flush dirty data/metadata before non-truncate setattr
<13b86bc4cd> 2019-10-23 [Linus Torvalds]  Merge tag 'for-linus' of git://git.armlinux.org.uk/~rmk/linux-arm

<e969c860d5> 2019-10-23 [Linus Torvalds]  Merge tag 'edac_urgent_for_5.4' of git://git.kernel.org/pub/scm/linux/kernel/git/ras/ras

<54955e3bfd> 2019-10-23 [Linus Torvalds]  Merge tag 'for-5.4-rc4-tag' of git://git.kernel.org/pub/scm/linux/kernel/git/kdave/linux

<0968495005> 2019-10-21 [Jessica Yu]  scripts/nsdeps: use alternative sed delimiter
<028db79cf4> 2019-10-23 [Rafael J. Wysocki]  Merge branch 'opp/fixes' of git://git.kernel.org/pub/scm/linux/kernel/git/vireshk/pm

<80da5a809d> 2019-10-23 [zhengbin]  virtiofs: Remove set but not used variable 'fc'
<6370740e5f> 2019-10-21 [Dan Williams]  fs/dax: Fix pmd vs pte conflict detection
<b19c23551b> 2019-10-18 [Viresh Kumar]  opp: Reinitialize the list_kref before adding the static OPPs again
<4750c21217> 2019-10-22 [Pan Xiuli]  ALSA: hda: Add Tigerlake/Jasperlake PCI ID
<671ddc700f> 2019-10-15 [Jim Mattson]  KVM: nVMX: Don't leak L1 MMIO regions to L2
<5effc09c49> 2019-10-22 [Alexey Brodkin]  ARC: perf: Accommodate big-endian CPU
<ab563bf54a> 2019-10-18 [Eugeniy Paltsev]  ARC: [plat-hsdk]: Enable on-boardi SPI ADC IC
<8ca8fa7f22> 2019-10-18 [Eugeniy Paltsev]  ARC: [plat-hsdk]: Enable on-board SPI NOR flash IC

<5c94ac5d0f> 2019-10-18 [Miaohe Lin]  KVM: SVM: Fix potential wrong physical id in avic_handle_ldr_update
<39f4d44104> 2019-10-22 [Russell King]  Merge branch 'misc' into fixes

<6941051d30> 2019-10-18 [Sudeep Holla]  cpufreq: Cancel policy update work scheduled before freeing
<ac49303d9e> 2019-10-21 [Gerald Schaefer]  s390/kaslr: add support for R_390_GLOB_DAT relocation type
<388bb19be8> 2019-10-10 [Johan Hovold]  s390/zcrypt: fix memleak at release

<ba8bf0967a> 2019-10-22 [Takashi Iwai]  ALSA: usb-audio: Fix copy&paste error in the validator
<f3a519e4ad> 2019-10-22 [Alexander Shishkin]  perf/aux: Fix AUX output stopping
<20baa8e515> 2019-10-22 [Paolo Bonzini]  Merge tag 'kvm-ppc-fixes-5.4-1' of git://git.kernel.org/pub/scm/linux/kernel/git/paulus/powerpc into HEAD

<9800c24e2f> 2019-10-22 [Paolo Bonzini]  Merge tag 'kvmarm-fixes-5.4-2' of git://git.kernel.org/pub/scm/linux/kernel/git/kvmarm/kvmarm into HEAD

<49dedf0dd0> 2019-10-10 [Paolo Bonzini]  kvm: clear kvmclock MSR on reset
<b4fdcf6056> 2019-09-29 [kbuild test robot]  KVM: x86: fix bugon.cocci warnings
<1a8211c7d8> 2019-09-29 [Liran Alon]  KVM: VMX: Remove specialized handling of unexpected exit-reasons
<ef40598098> 2019-10-08 [Vitaly Kuznetsov]  selftests: kvm: fix sync_regs_test with newer gccs
<11eada4718> 2019-10-08 [Vitaly Kuznetsov]  selftests: kvm: vmx_dirty_log_test: skip the test when VMX is not supported
<9143613ef0> 2019-10-08 [Vitaly Kuznetsov]  selftests: kvm: consolidate VMX support checks
<700c17d9ce> 2019-10-08 [Vitaly Kuznetsov]  selftests: kvm: vmx_set_nested_state_test: don't check for VMX support twice
<44551b2f69> 2019-09-29 [Wanpeng Li]  KVM: Don't shrink/grow vCPU halt_poll_ns if host side polling is disabled
<9de25d182b> 2019-10-07 [Vitaly Kuznetsov]  selftests: kvm: synchronize .gitignore to Makefile
<41cd02c6f7> 2019-10-04 [Jim Mattson]  kvm: x86: Expose RDPID in KVM_GET_SUPPORTED_CPUID

<3b7c59a195> 2019-10-22 [Linus Torvalds]  Merge tag 'pinctrl-v5.4-2' of git://git.kernel.org/pub/scm/linux/kernel/git/linusw/linux-pinctrl

<31d851407f> 2019-10-18 [Zhenzhong Duan]  cpuidle: haltpoll: Take 'idle=' override into account

<edffc70f50> 2019-10-18 [Dan Carpenter]  ACPI: NFIT: Fix unlock on error in scrub_show()

<6b1340cc00> 2019-10-15 [Prateek Sood]  tracing: Fix race in perf_trace_buf initialization
<6fee2a0be0> 2019-10-21 [Thomas Hellstrom]  x86/cpu/vmware: Fix platform detection VMWARE_PORT macro
<db633a4e0e> 2019-10-21 [Thomas Hellstrom]  x86/cpu/vmware: Use the full form of INL in VMWARE_HYPERCALL, for clang/llvm

<cc3fafdaf5> 2019-10-21 [Olof Johansson]  Merge tag 'arm-soc/for-5.4/devicetree-fixes-part2' of https://github.com/Broadcom/stblinux into arm/fixes

<6813a9ce1f> 2019-10-21 [Olof Johansson]  Merge tag 'arm-soc/for-5.4/devicetree-fixes' of https://github.com/Broadcom/stblinux into arm/fixes
:
```

Pull Requests

Once you have a branch and want to commit the changes to the Master branch, submit a pull request to merge them.



Using .gitignore

There may be files you don’t wish to track. Adding these to the .gitignore file will stop Git from pestering you about untracked changes regarding these files.

For example, you might not want to commit your .class or .o files

Good Git Practices

Initialize repos with a README describing its purpose

Make meaningful commits (it’s not a save button)

For every commit, there is a descriptive message

Use git diff to check changes and avoid merge conflicts

Activity 1

1. Create your own repo
2. Create a README file describing your experience with Git
3. Commit the README file
4. Modify it to include an interesting fact
5. Commit again
6. Push to Master Branch

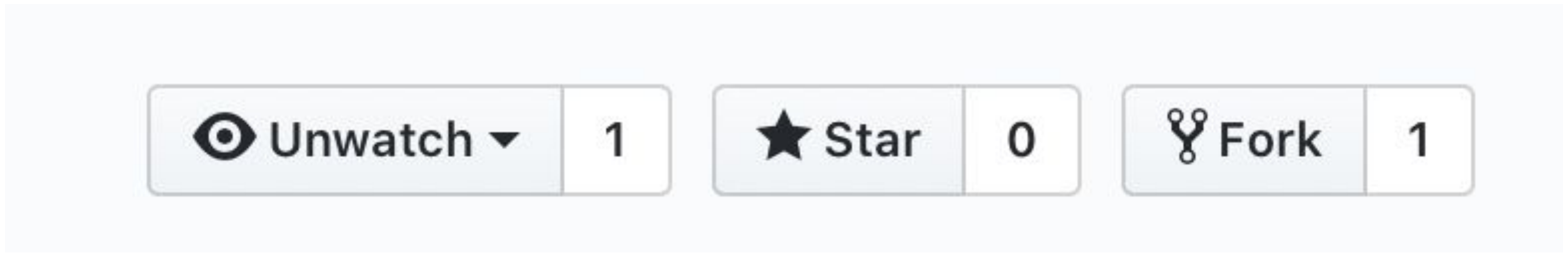
Instructions at https://github.com/nickwanninger/gitpres

Making Contributions

- Most of the times, with git, people work in large groups to get a project done.
- Communication happens in “issues”
- People often report bugs, ask questions, etc..
- If you see a bug that you could probably fix, you can!
- Just clone the repo, make a branch, make your changes, then push
- On github, you can merge the branches with a pull request (PR)
  - Let’s go do that now

Activity 2

1. Look at the issues on the repo
2. **http://bit.ly/git_workshop_repo**
3. Fork the repo, clone it, and make a feature branch
4. Make changes to an existing file
5. Fix the Issue
6. Commit these changes to your branch & push to the remote
7. Submit a pull request to merge with Master branch



Additional Resources

Pro Git (free ebook)

Github Guides

Google