STEM
CLUB OF AMERICA

git

Git basics

Git what

- Git is a:

  free

  open source

  distributed

  version control system

- We’ll cover just enough Git to make you dangerous

Free

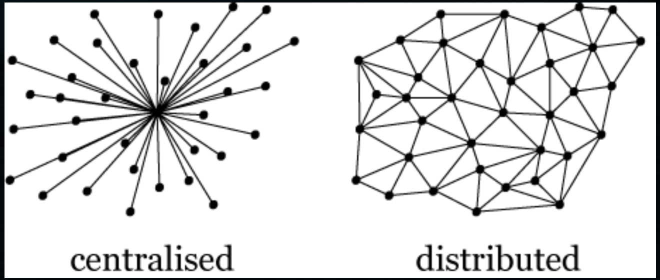
- What do you mean free?
  - 0 Dollars
  - 0 Yen
  - 0 Rupees
  - 0 Bitcoins

Open Source

- You can look at the code
  - https://github.com/git/git
- Linux Torvalds gave it to us
  - He also gave us Linux

Distributed

- I get a copy
  - and you get a copy
    - and you get a copy
      - and you get a copy
        - and you get a copy
          - and you get a copy



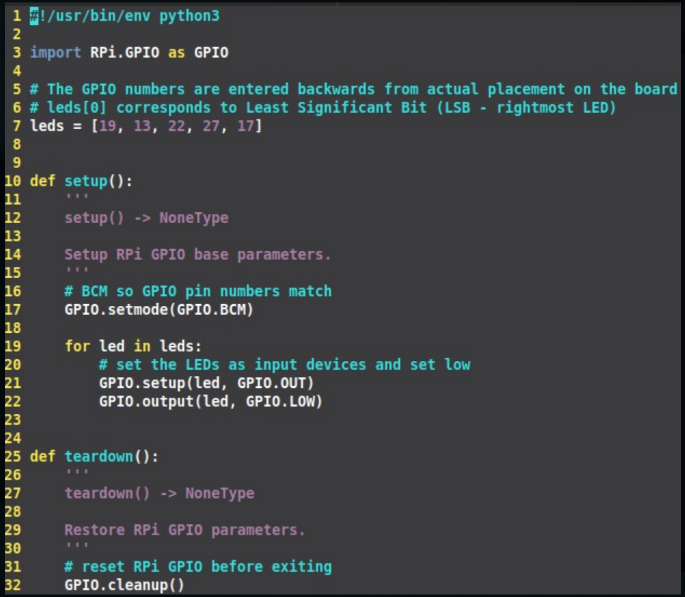
- If we all have a copy, who can make changes?

Version Control

- How do we keep track of changes?
  - Git keeps track of them for us
  - Changes are <u>*COMMIT*</u>ed
  - We <u>*PUSH*</u> our changes *to* others
  - We <u>*PULL*</u> in changes *from* others

What are we editing

- Usually things with text
  - Code
  - Notes
  - Grocery Lists
  - Presentations
  - Plans to take over the world



```
#!/usr/bin/env python3

import RPi.GPIO as GPIO

# The GPIO numbers are entered backwards from actual placement on the board
# leds[0] corresponds to Least Significant Bit (LSB - rightmost LED)
leds = [19, 13, 22, 27, 17]


def setup():
    '''
    setup() -> NoneType

    Setup RPi GPIO base parameters.
    '''
    # BCM so GPIO pin numbers match
    GPIO.setmode(GPIO.BCM)

    for led in leds:
        # set the LEDs as input devices and set low
        GPIO.setup(led, GPIO.OUT)
        GPIO.output(led, GPIO.LOW)


def teardown():
    '''
    teardown() -> NoneType

    Restore RPi GPIO parameters.
    '''
    # reset RPi GPIO before exiting
    GPIO.cleanup()
```

Collaboration

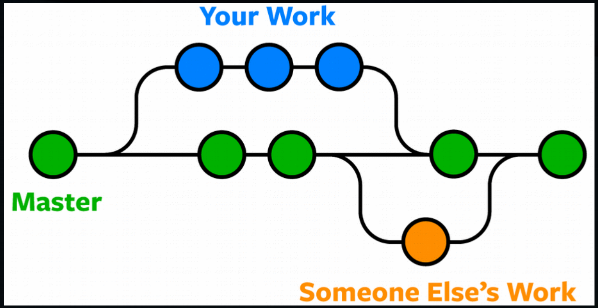
- Do you have to share your code?
  - Nope, but Santa knows your name…
  - Santa likes it when you share...

How does it work?

- Git tracks files in a repository
- You *ADD* files and the Git ninjas secretly begin tracking them for changes
- Every time you *COMMIT*, Git takes a snapshot of the staged (*ADD*ed) file
- Comments can be added to every *COMMIT*
  - Git, please remind me why I made these changes (*LOG*)
- Your changes only affect your local repository until you're ready to *PUSH*
- Changes can be *REVERT*ed
  - Woopsies

Life as a tree

- *BRANCH*es let you test out changes without affecting other contributors
- When you are done, you *MERGE* it back into the *MASTER* branch

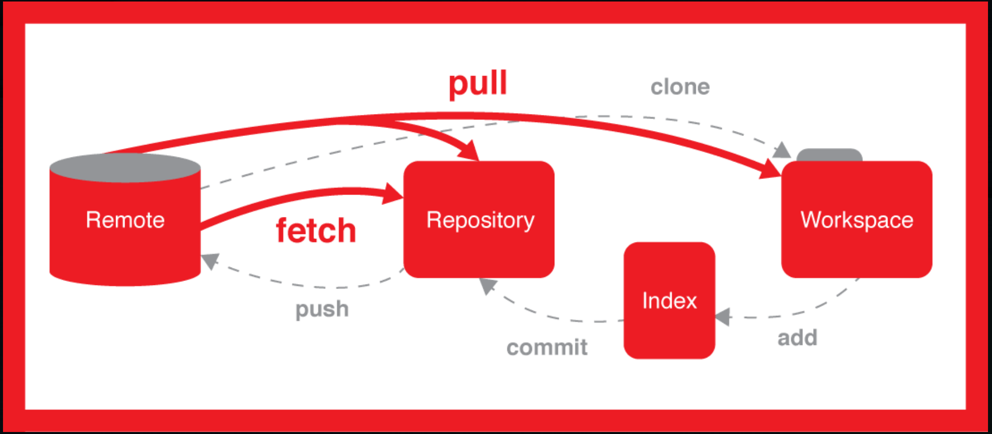


Sharing is caring

- You could share changes between your local repository and another contributor’s local repository
- This is not scalable
  - 10 person team?
  - 100 person team?
  - Connectivity between each contributor?

Remote Server

- Still distributed because we all have our own local repositories
- *PUSH* to and *PULL* from a remote server so that everyone's local repositories stay up-to-date



Git services

- www.github.com
- www.gitlab.com
- bitbucket.org

- Launch your own or subscribe

How do you begin

- *INIT* to initialize local repository

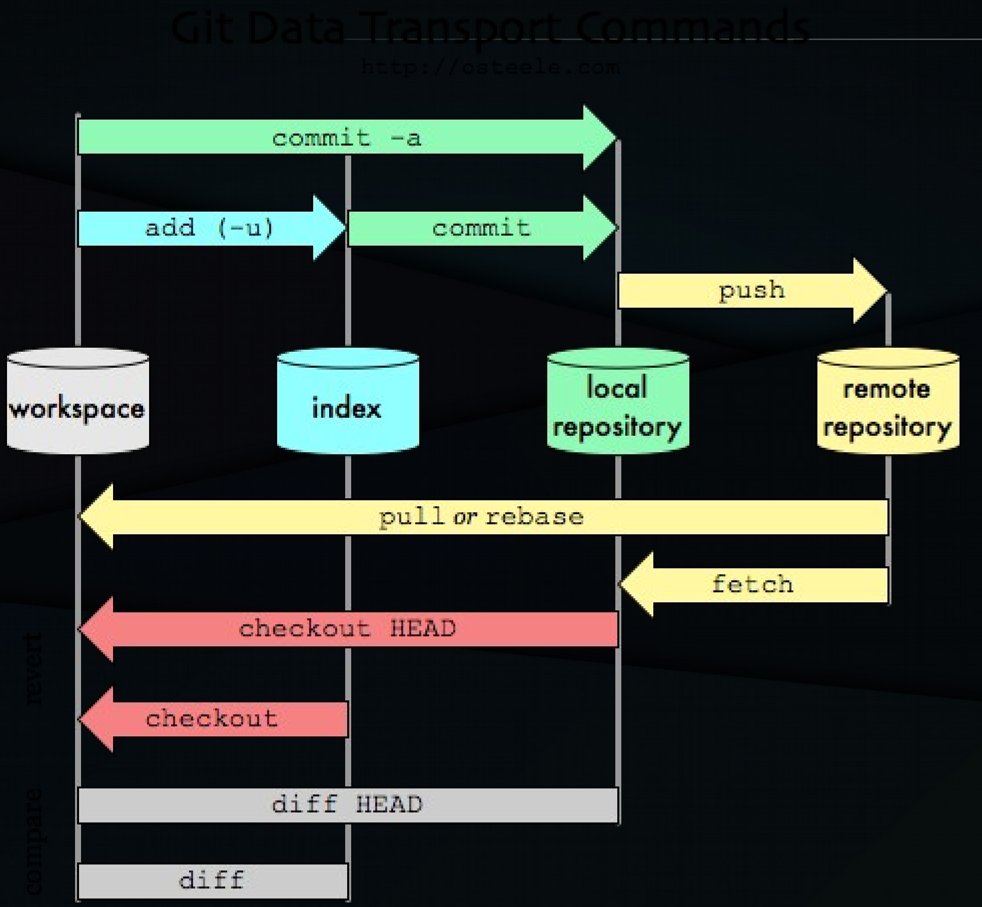
  or

- *CLONE* from remote repository (most common)

Typical flow

- Create a new feature *BRANCH* to work in and *CHECKOUT*
- Edit / create files
- *ADD* files to staging area (index)
- *COMMIT* your changes (local repository)
- *CHECKOUT* *MASTER* *BRANCH*
- *PULL* any changes that were made while you were editing
- *MERGE* your feature *BRANCH* back to *MASTER*
- *PUSH* changes back to remote (*ORIGIN*)

Typical flow



Check on things

- *STATUS* – whats the current state of the repository and staging area
- *LOG* – look back at the history
- *DIFF* – what changes are pending
- *BLAME* – who made a change
- *REVERT* – undo a commit (rewind)

Lab

Now that you are an expert, Git on that lab!