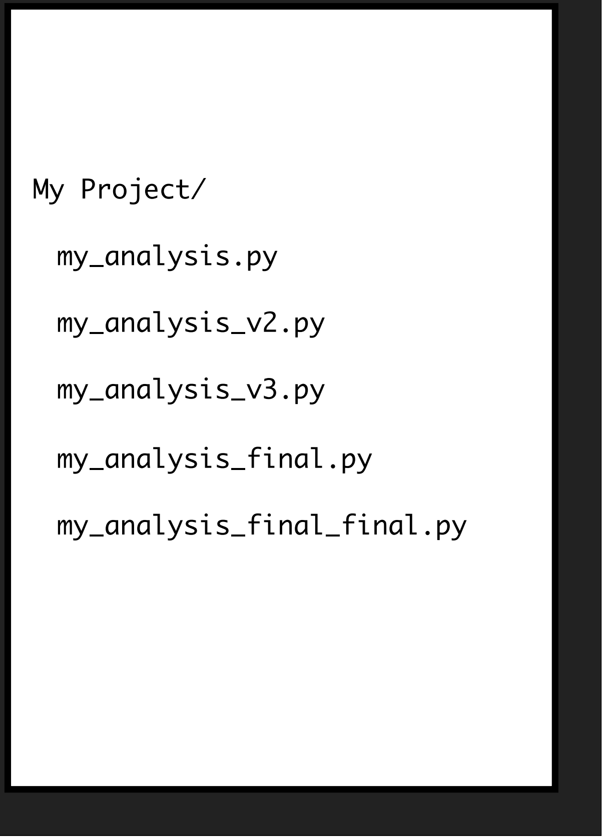
ESHIN JOLLY

Slack: @ejolly
Email: eshin.jolly@gmail.com

INTRO TO GIT/GITHUB

WHAT IS THE POINT OF THIS

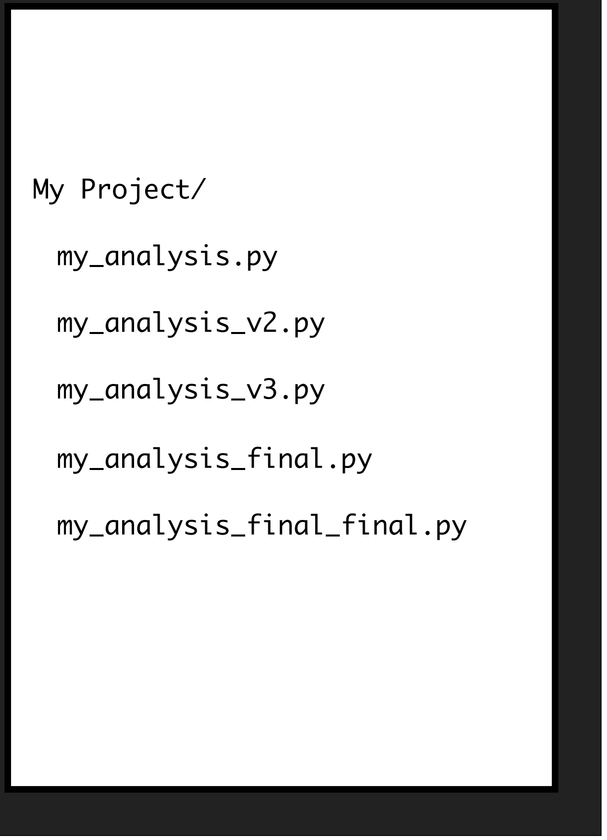
- We want to avoid this –> Why?
- What are the differences between different versions?
- Who else worked on these files?
- When/why were different version made?
- Arghh I can’t reproduce things and I don’t know what to trust!



```
My Project/
  my_analysis.py
  my_analysis_v2.py
  my_analysis_v3.py
  my_analysis_final.py
  my_analysis_final_final.py
```

WHAT IS THE POINT OF THIS

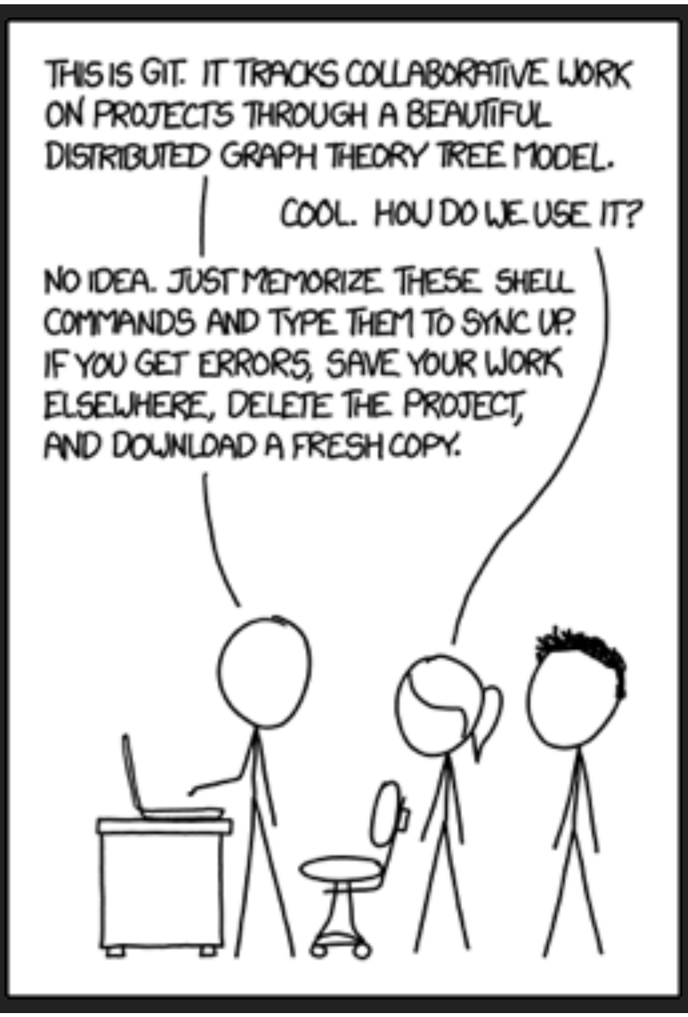
- We want to avoid this –> Why?
- We want to be able *easily* keep track of our work history
- We want to be able to *collaborate* with others
- We want to be able to try new things *without losing old work*
- We research to be *reproducible*



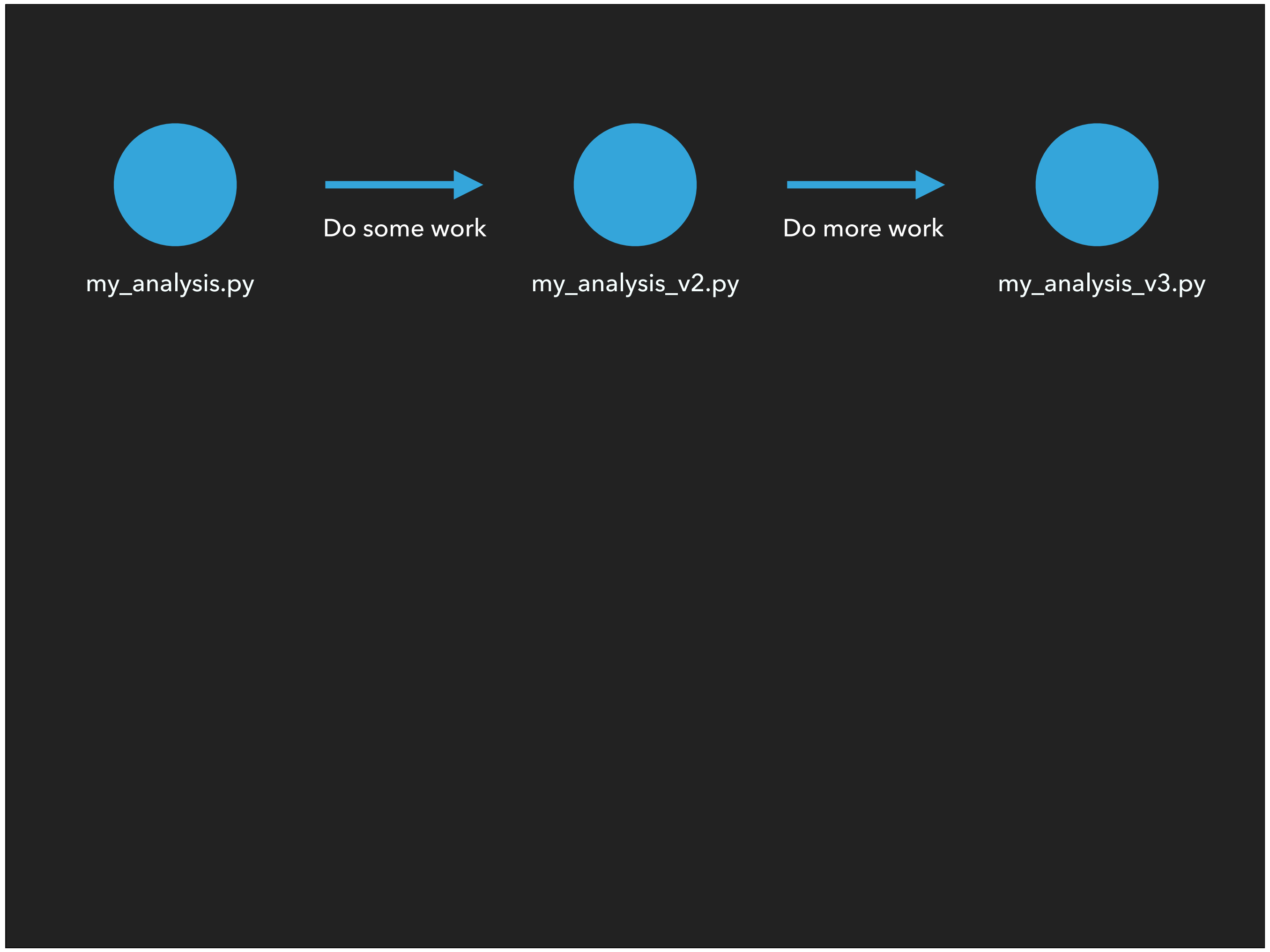
```
My Project/
  my_analysis.py
  my_analysis_v2.py
  my_analysis_v3.py
  my_analysis_final.py
  my_analysis_final_final.py
```

WE NEED A TIME-MACHINE

(With room for other people)


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.

VERSION CONTROL



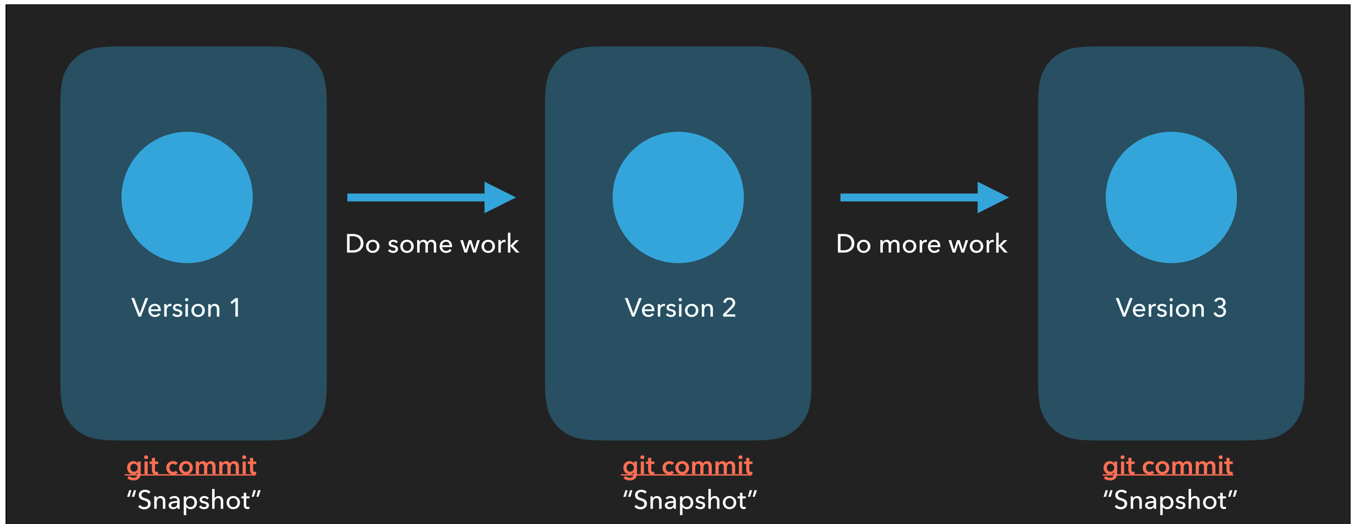
my_analysis.py

Do some work

my_analysis_v2.py

Do more work

my_analysis_v3.py



Version 1

Do some work

Version 2

Do more work

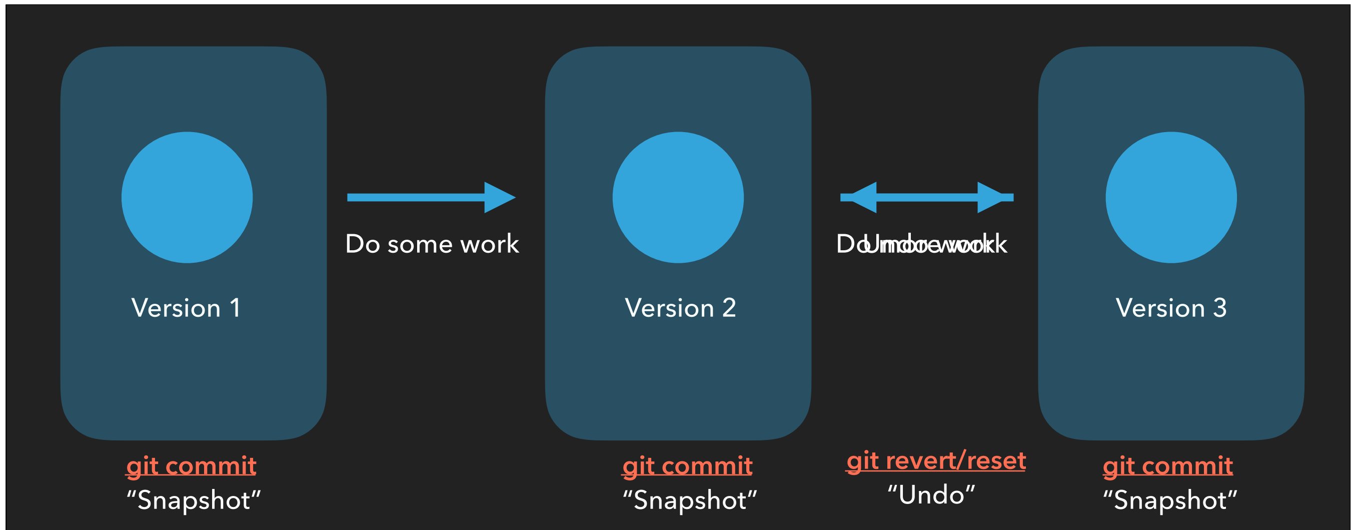
Version 3

git commit
“Snapshot”

git commit
“Snapshot”

git commit
“Snapshot”

GO BACK IN TIME



Version 1

Do some work

Version 2

Do more work

Undo work

Version 3

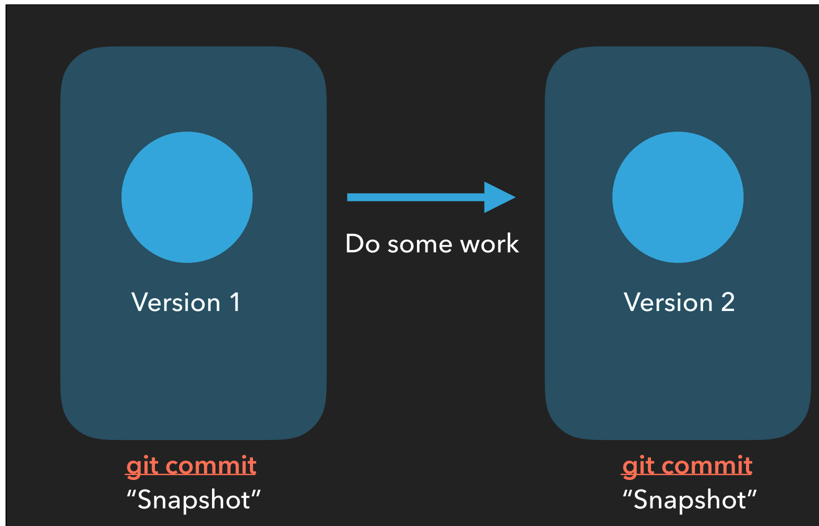
git commit
“Snapshot”

git commit
“Snapshot”

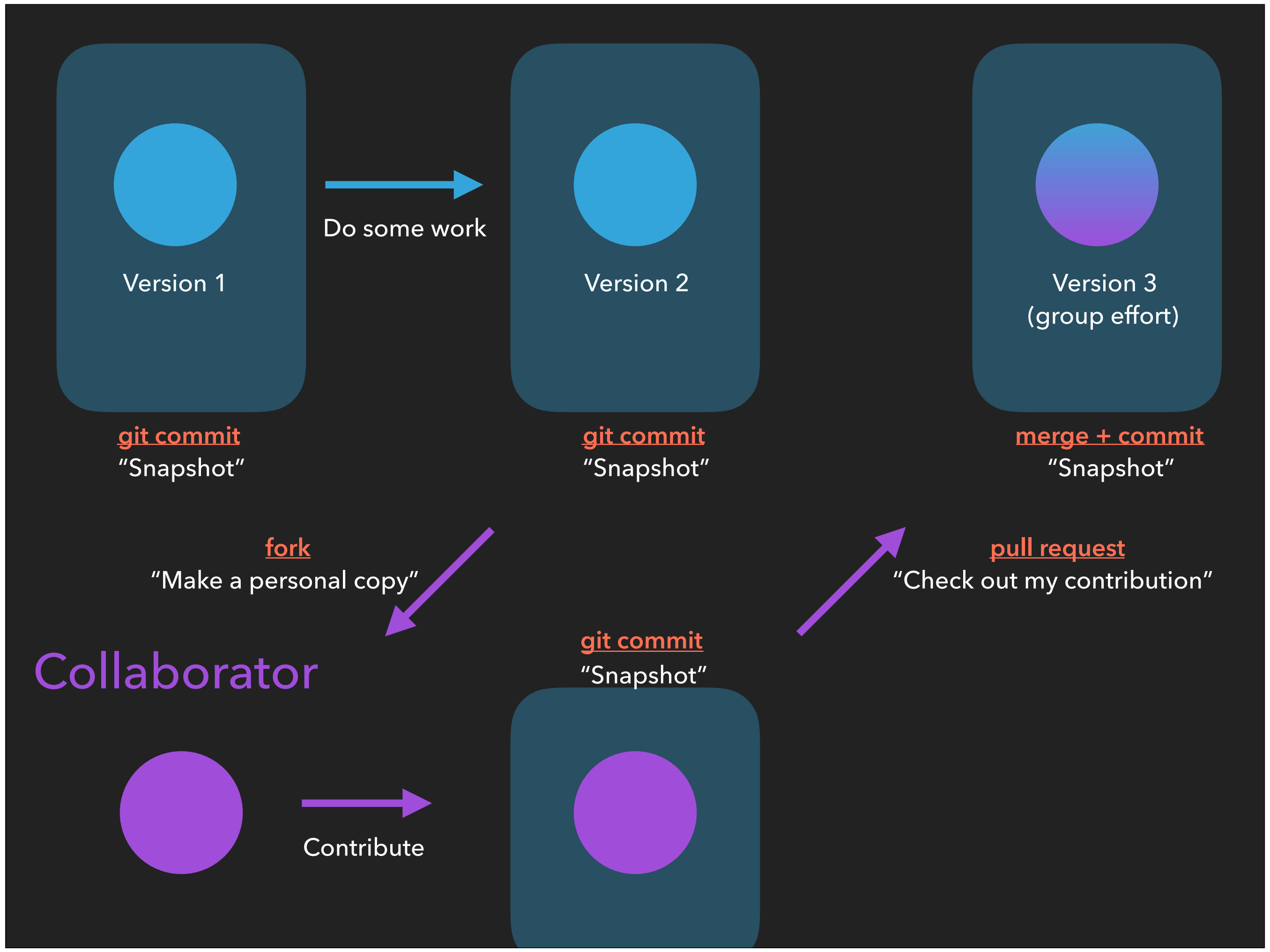
git revert/reset
“Undo”

git commit
“Snapshot”

COLLABORATION



Version 1
Do some work
Version 2
git commit
“Snapshot”
git commit
“Snapshot”



Version 1

git commit
“Snapshot”

Do some work

Version 2

git commit
“Snapshot”

Version 3
(group effort)

merge + commit
“Snapshot”

fork
“Make a personal copy”

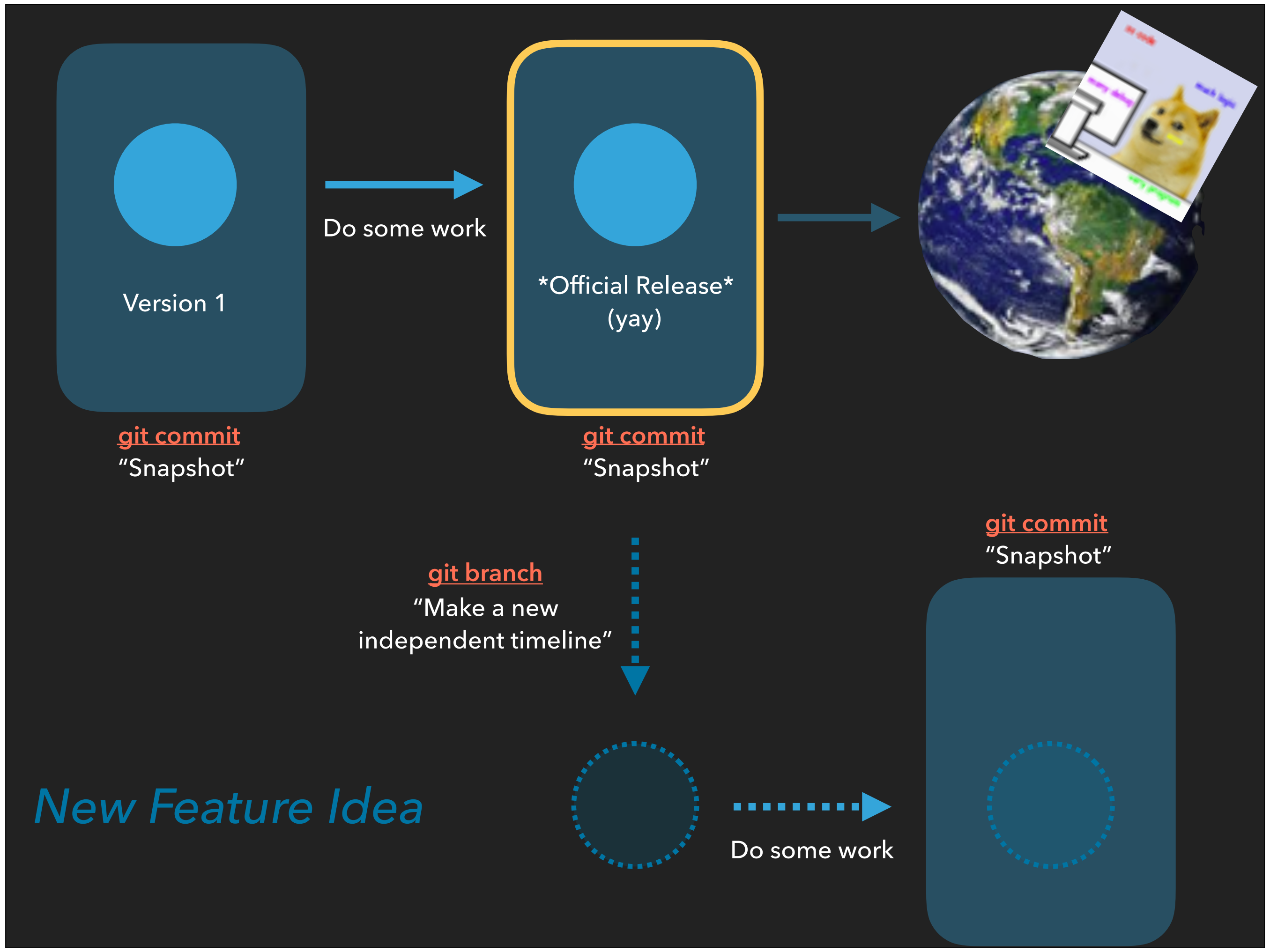
pull request
“Check out my contribution”

Collaborator

Contribute

git commit
“Snapshot”

PARALLEL DEVELOPMENT



Version 1

git commit
"Snapshot"

Do some work

Official Release
(yay)

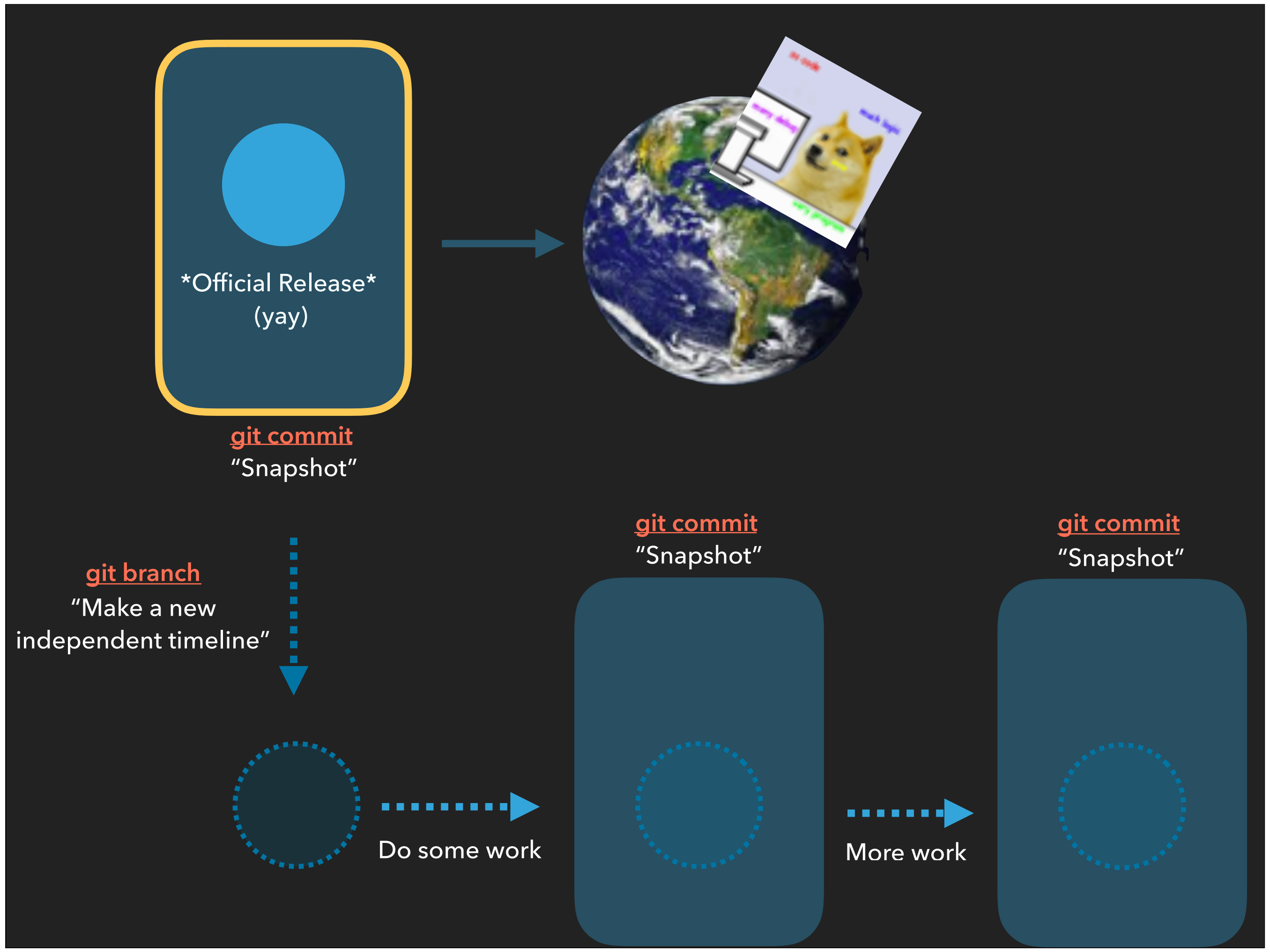
git commit
"Snapshot"

git branch
"Make a new
independent timeline"

New Feature Idea

Do some work

git commit
"Snapshot"



Official Release
(yay)

git commit
“Snapshot”

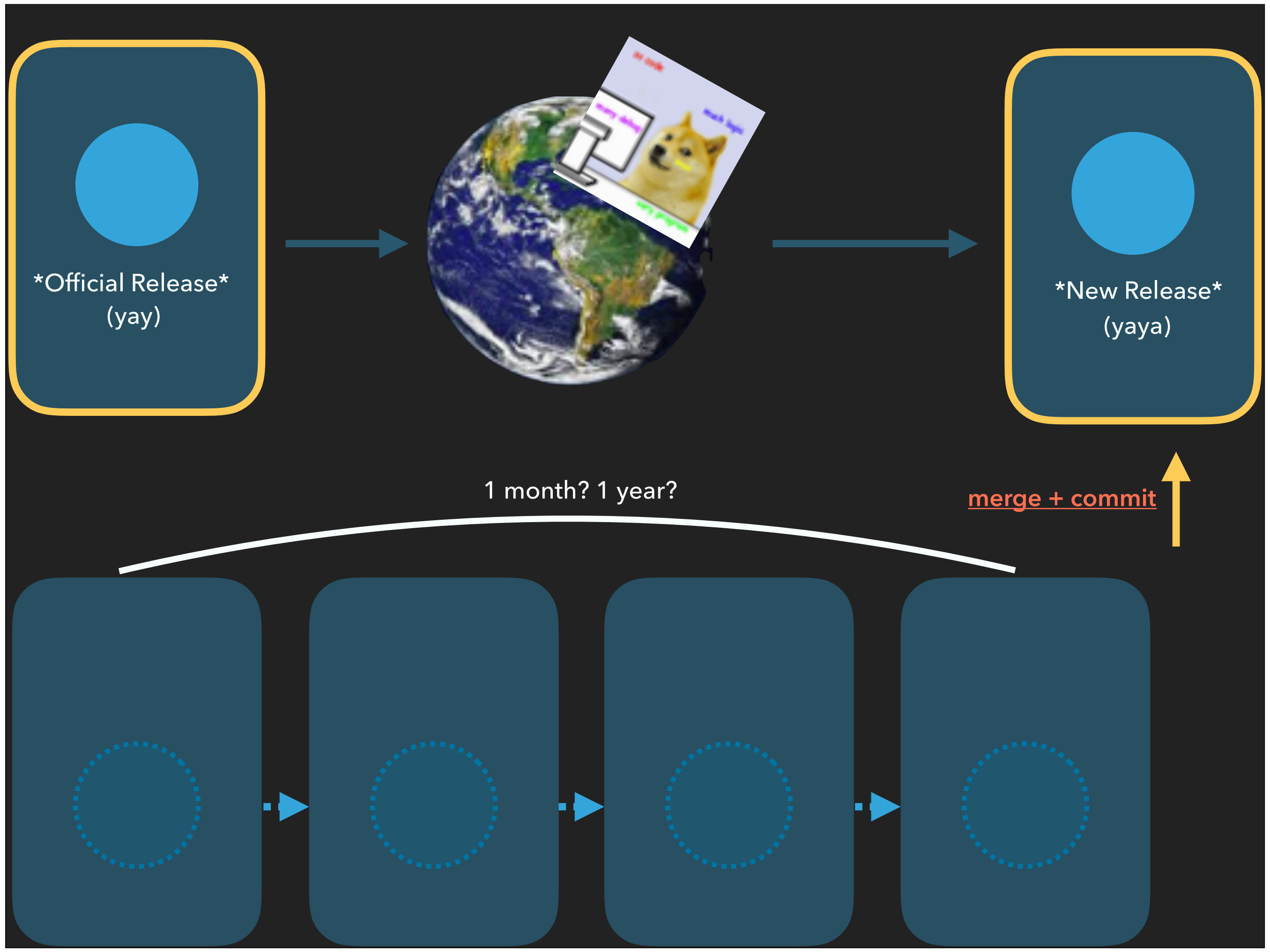
git branch
“Make a new
independent timeline”

Do some work

git commit
“Snapshot”

More work

git commit
“Snapshot”

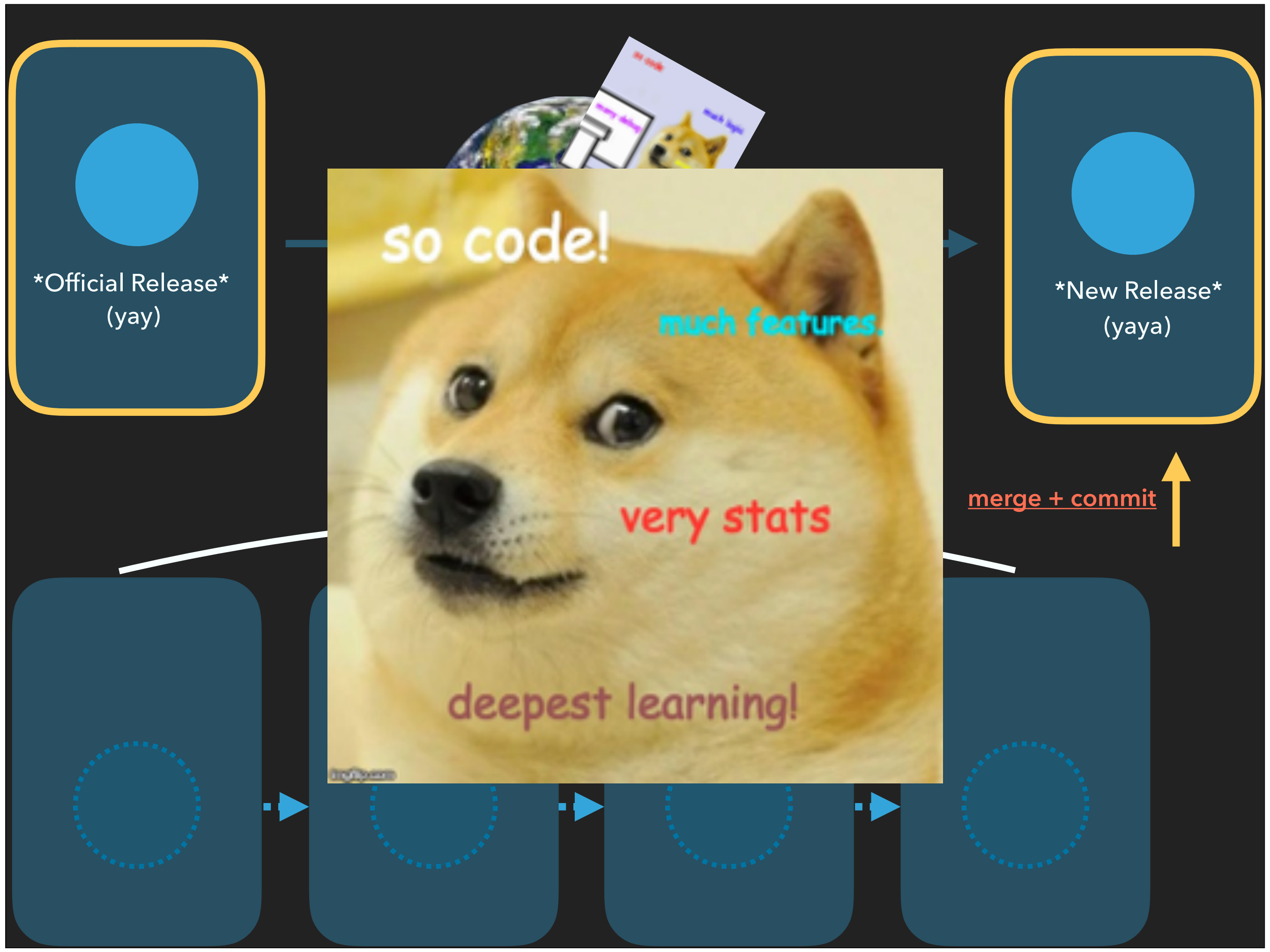


Official Release
(yay)

New Release
(yaya)

1 month? 1 year?

merge + commit



Official Release
(yay)

New Release
(yaya)

merge + commit

SUMMARY

GIT IS LIKE A FLEXIBLE TIME MACHINE FOR FILE MANAGEMENT

- Git is a local version-control-system
- Git is a command-line program (there are GUI apps too!)
- It can do almost everything a real time machine can do
  - Move backwards in time (retrieve old versions)
  - Move forwards in time (incorporate new changes)
  - Move sideways in time (parallel development)
  - Alter and merge branching timelines

SUMMARY

GIT VS GITHUB

- Github is a free cloud storage site for git repositories
- Github allows you to collaborate and share projects with others
- It keeps track of your complete git history (including parallel branches)
- It helps keeps track of who did what when
- It provides resources for tracking issues, enables communication, and hosts releases

ADDITIONAL RESOURCES

- GUI-based git apps
  - GitKraken
  - Atom Github Integration
- Atlassian Git Tutorials
- Github Hello World
- Blischak, Davenport & Wilson (2016) - Quick Intro to VCS

DEMO