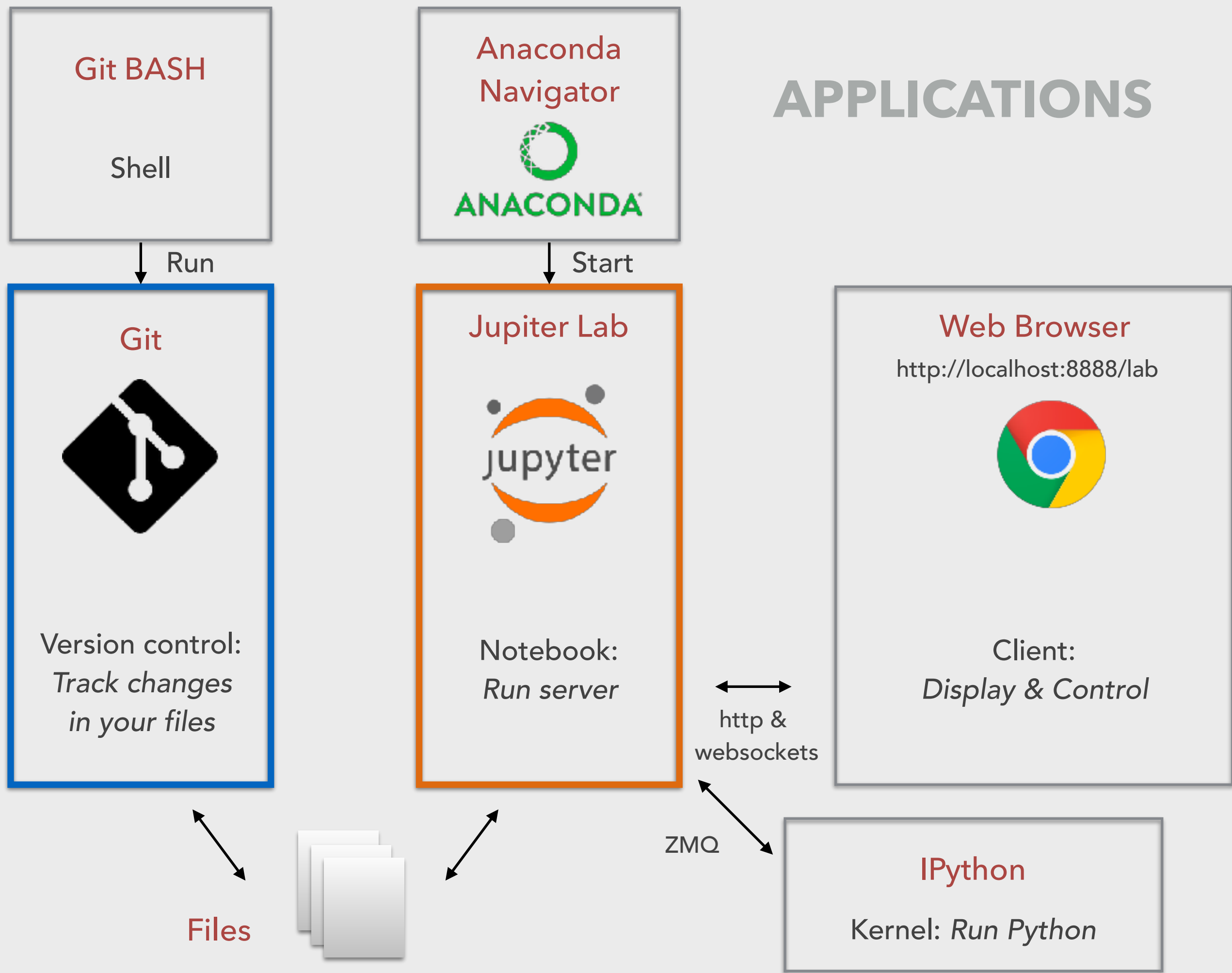


Recap, Git, & Good Practice

2018-10-17

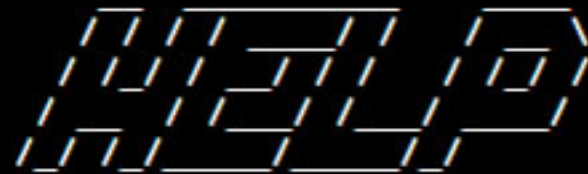
Application Overview

APPLICATIONS



Git

BASH BASICS:

The word "HELP" is displayed in a stylized, digital font. The letters are composed of multiple parallel lines, giving them a three-dimensional, wireframe appearance. The color of the lines is a light orange or yellow, which stands out against the solid black background.

- **cd**: Change current working directory
- **ls**: List files in current working directory
- **pwd**: Print current working directory to command line
- **git**: Operate git for repository of current working directory
- **cp**: Copy a file to a given destination
- **mv**: Move or rename a file or a directory
- **man**: Look up documentation for a command

GIT BASICS:

Remote repository



Github server



Pull



Push



Local repository



Local computer



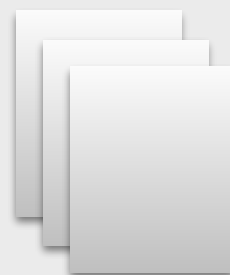
Checkout
(...)



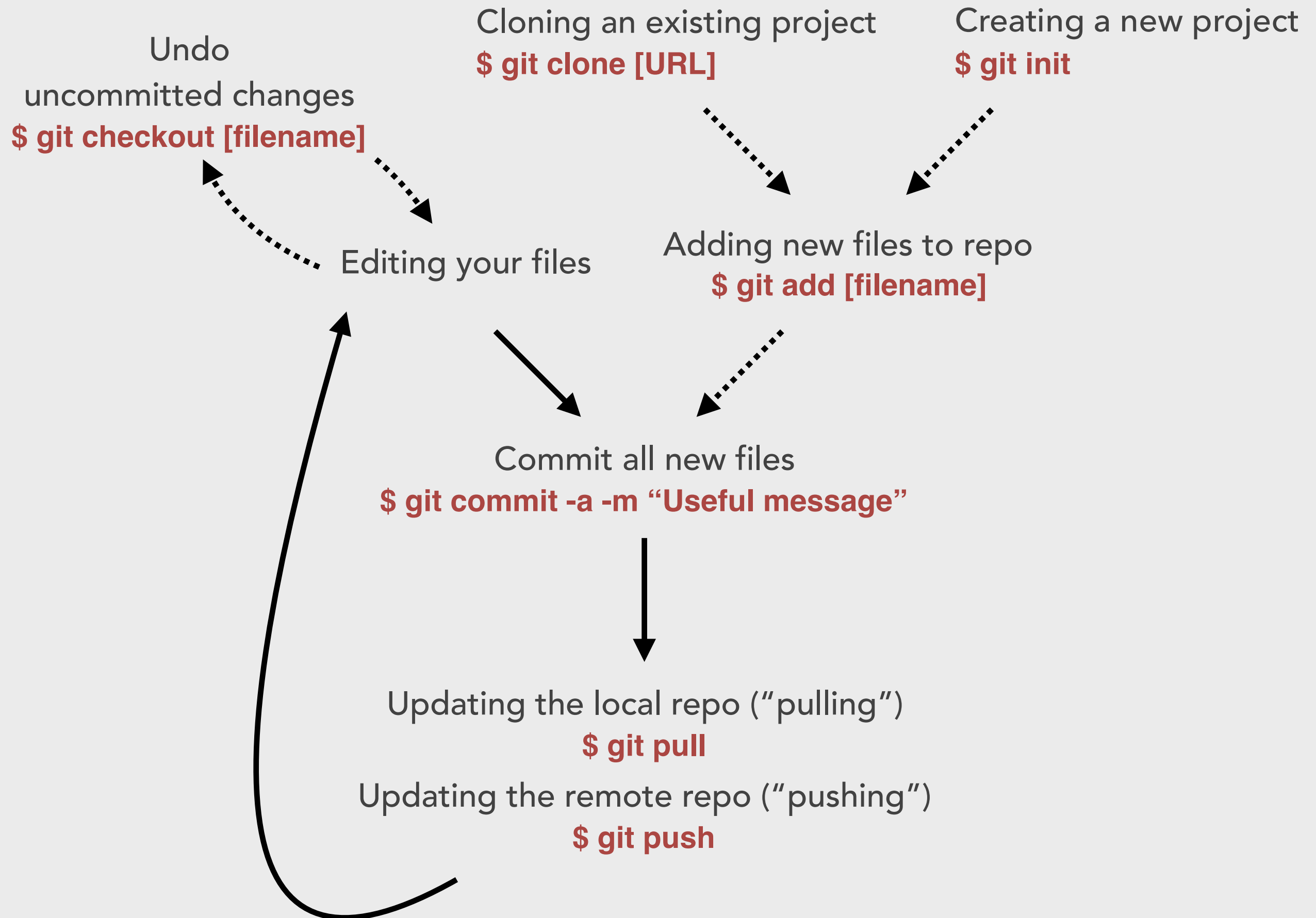
Commit
(...)



Files

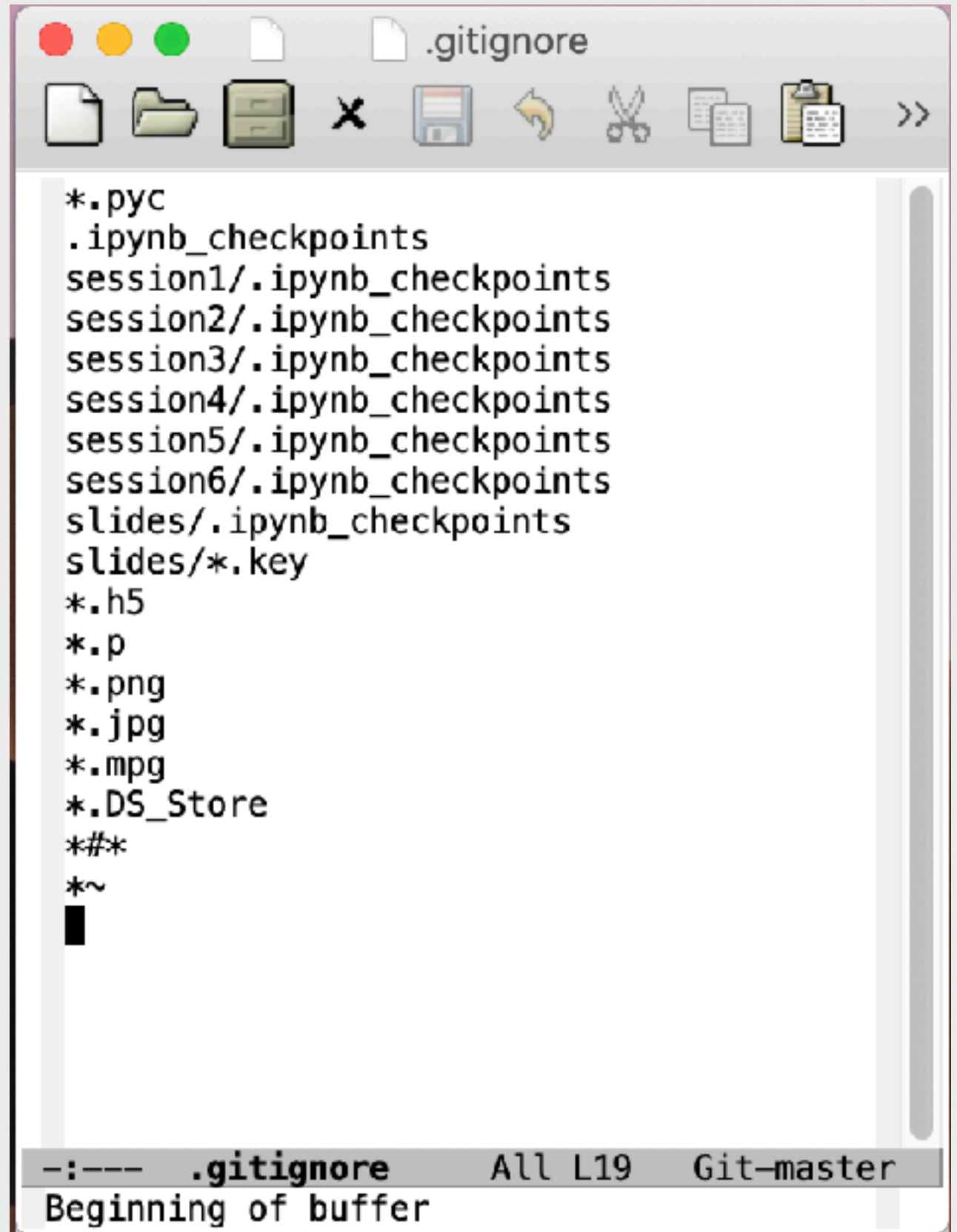


GIT BASICS:



TELL GIT WHICH FILES TO IGNORE

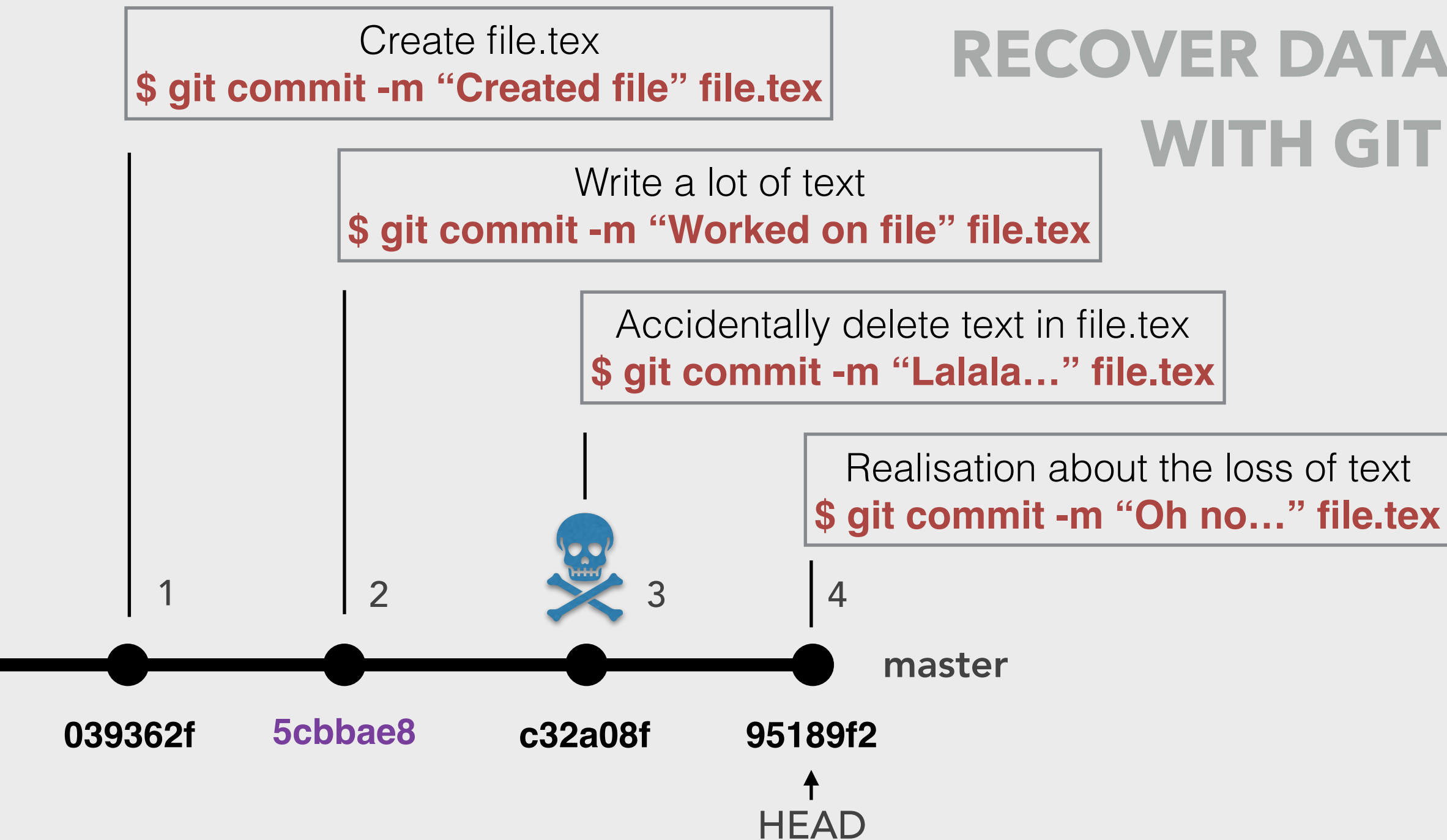
- The **.gitignore** file specifies intentionally untracked files that Git should ignore
- The file should be located on root level of a given repository
- Files already tracked by Git are not affected

A screenshot of a text editor window titled ".gitignore". The window has a standard macOS-style title bar with red, yellow, and green buttons. Below the title bar is a toolbar with icons for file operations like opening, saving, and deleting. The main area of the window contains the content of the .gitignore file, which lists various file patterns to be ignored. At the bottom of the window, there is a status bar showing the current file is ".gitignore", the line count is "All L19", and the branch is "Git-master".

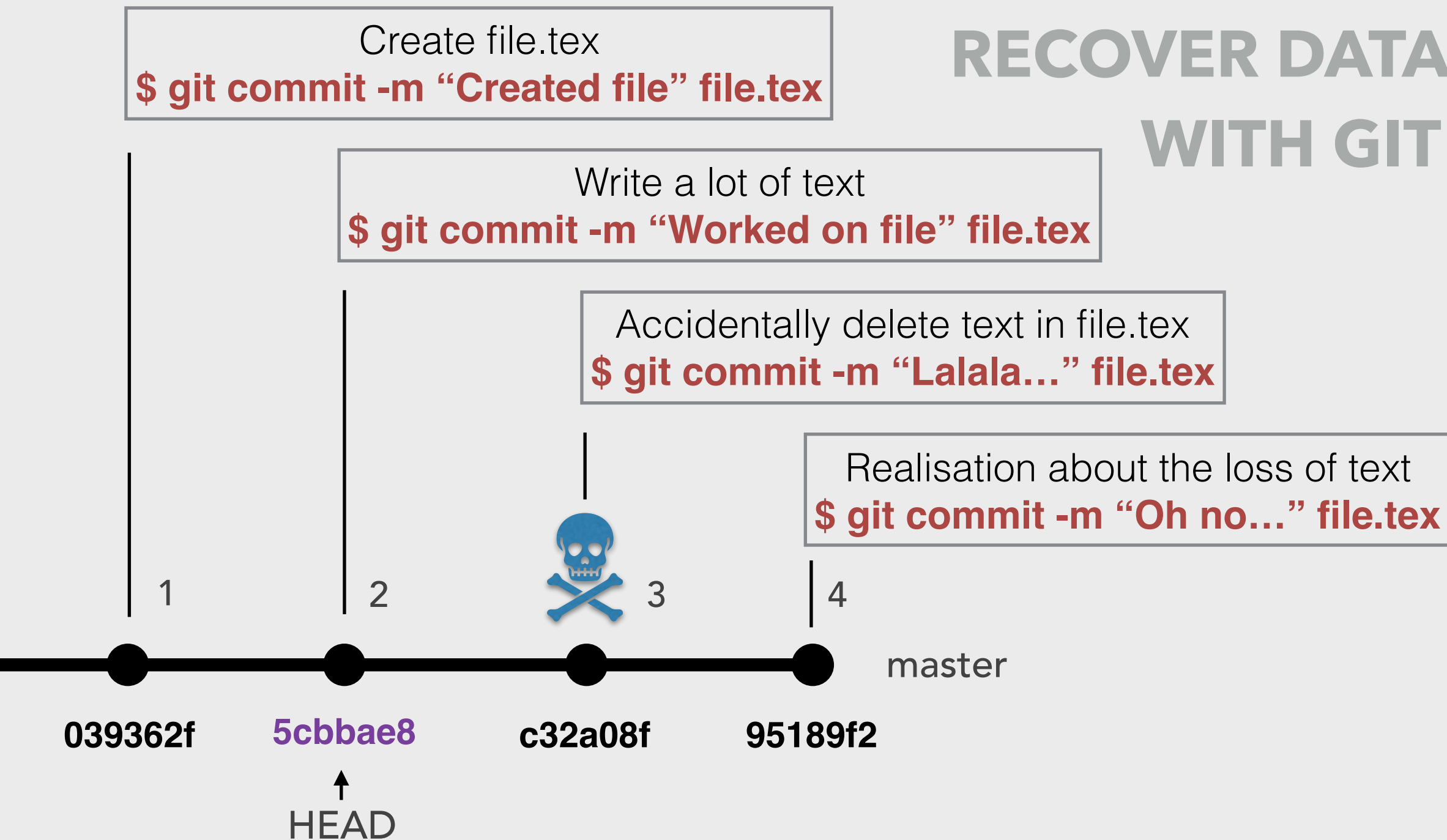
```
*.pyc
.ipynb_checkpoints
session1/.ipynb_checkpoints
session2/.ipynb_checkpoints
session3/.ipynb_checkpoints
session4/.ipynb_checkpoints
session5/.ipynb_checkpoints
session6/.ipynb_checkpoints
slides/.ipynb_checkpoints
slides/*.key
*.h5
*.p
*.png
*.jpg
*.mpg
*.DS_Store
*#*
*~
█
```

-:--- .gitignore All L19 Git-master
Beginning of buffer

RECOVER DATA LOSS WITH GIT



RECOVER DATA LOSS WITH GIT



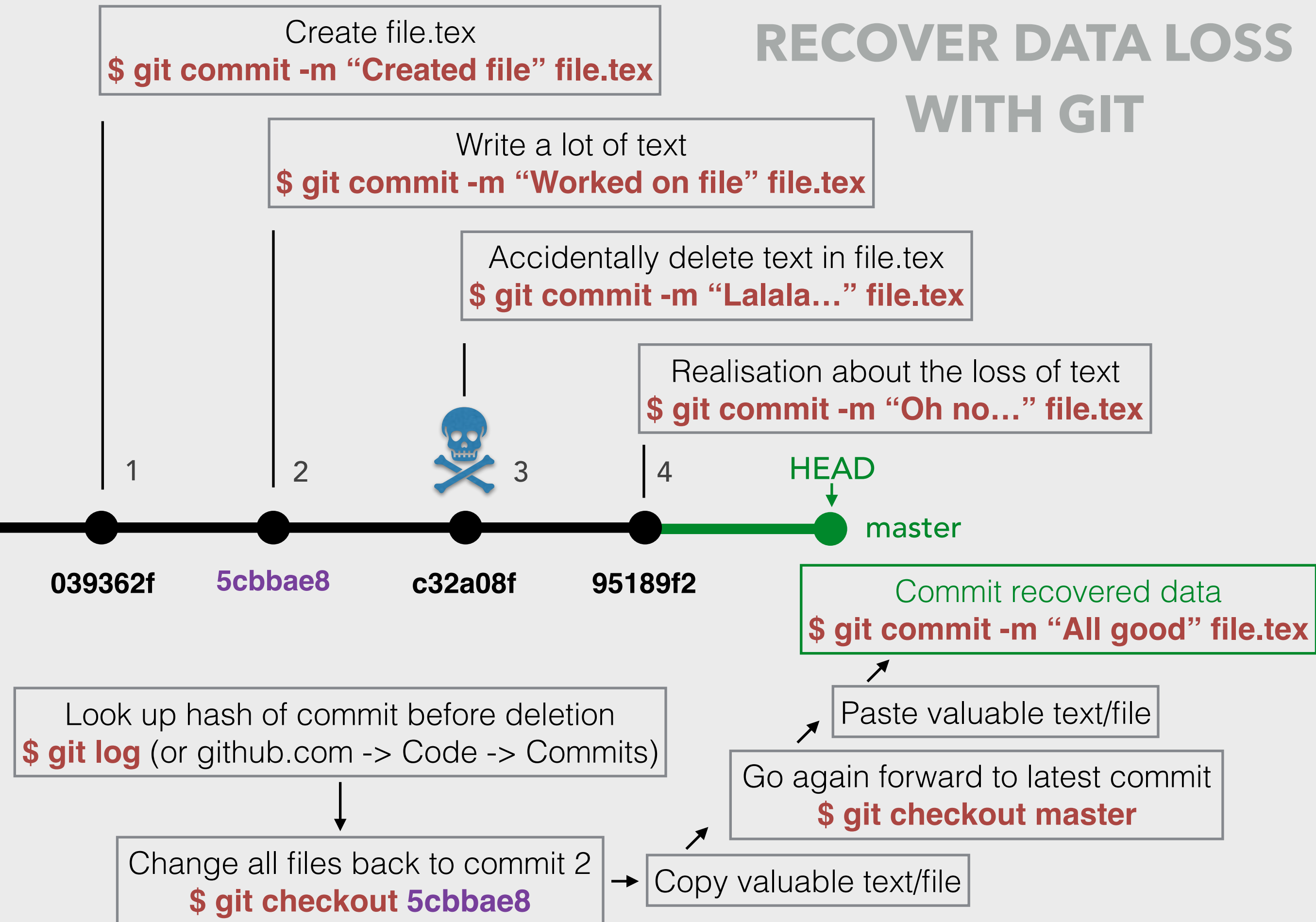
Look up hash of commit before disaster
\$ git log (or github.com -> Code -> Commits)

Change all files back to commit 2
\$ git checkout 5cbbae8

Go again forward to latest commit
\$ git checkout master

Copy valuable text/file

RECOVER DATA LOSS WITH GIT



Good Practice In Python

The Zen Of Python (PEP 20)

In [1]: **import this**

The Zen of Python, by Tim Peters

- Beautiful is better than ugly.
- Explicit is better than implicit.
- Simple is better than complex.
- Complex is better than complicated.
- Flat is better than nested.
- Sparse is better than dense.
- Readability counts.
- Special cases aren't special enough to break the rules.
- Although practicality beats purity.

The Zen Of Python

- Errors should never pass silently.
- Unless explicitly silenced.
- In the face of ambiguity, refuse the temptation to guess.
- There should be one-- and preferably only one --obvious way to do it.
- Although that way may not be obvious at first unless you're Dutch.
- Now is better than never.
- Although never is often better than *right* now.
- If the implementation is hard to explain, it's a bad idea.
- If the implementation is easy to explain, it may be a good idea.
- Namespaces are one honking great idea -- let's do more of those!