



Your attention Please



GIT

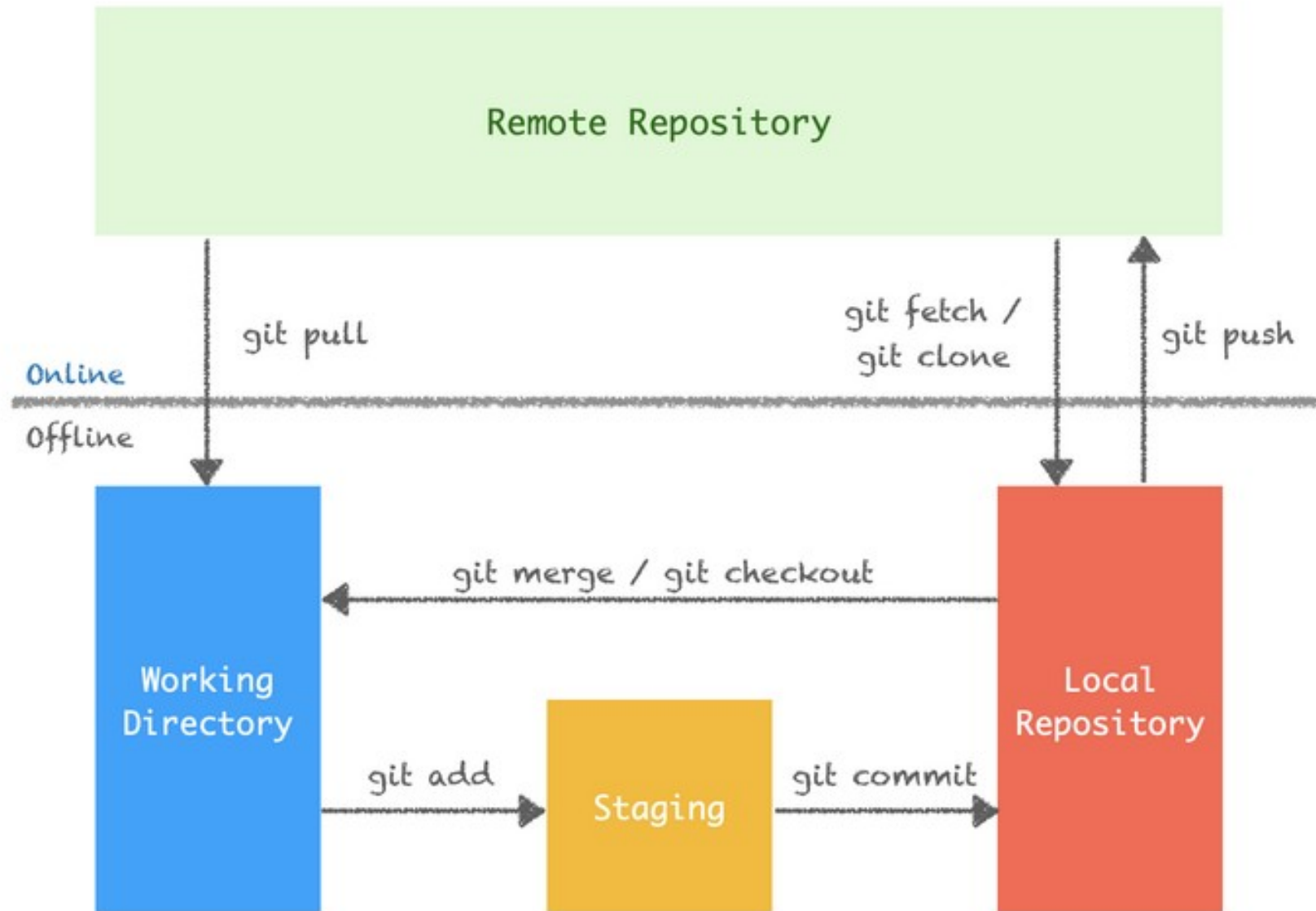
- **What is Git?**
- **Why Git?**
- **How ?**

What is Git?

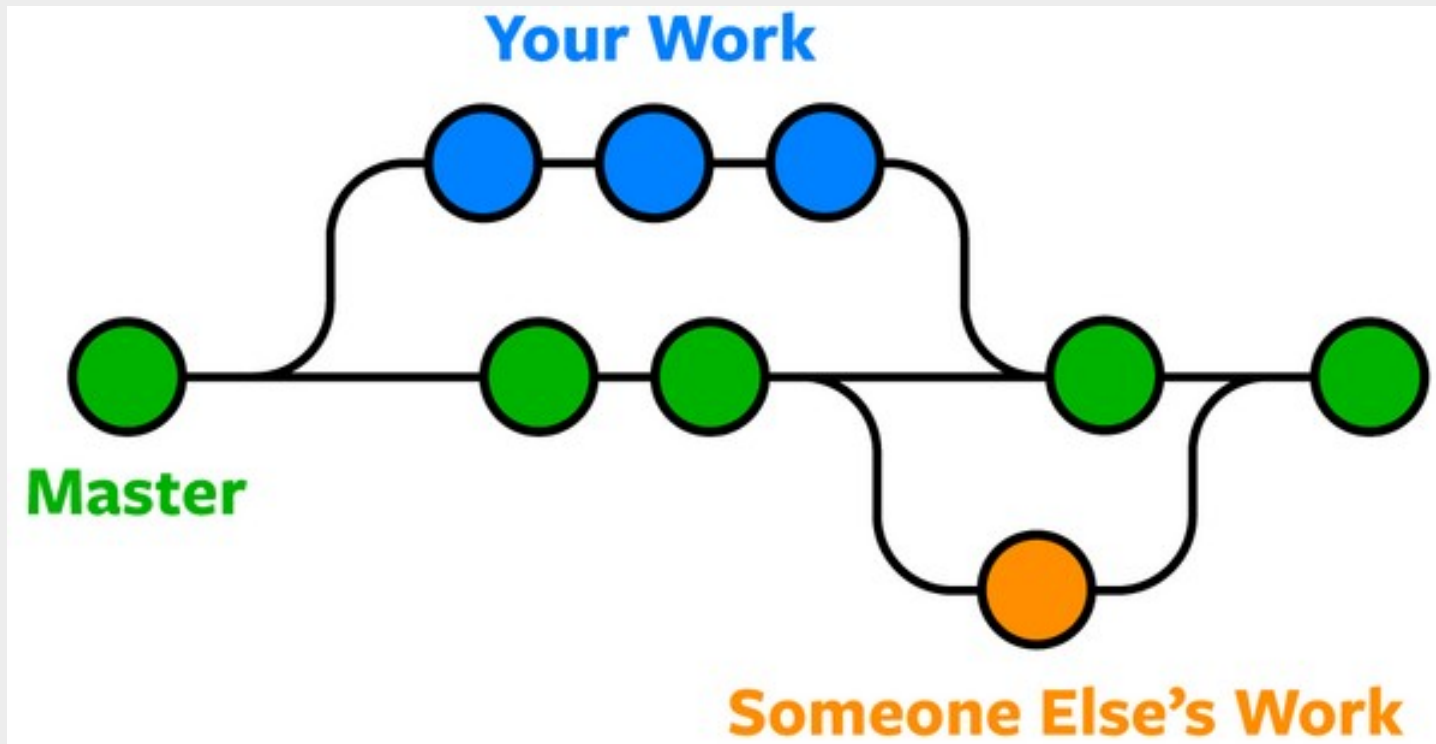
- **Git is a version-control system for tracking changes in source code during development. It is designed for coordinating work among programmers, but it can be used to track changes in any set of files.**



What is Git?



What is Git?



EXTRA CRANKY SLIDE

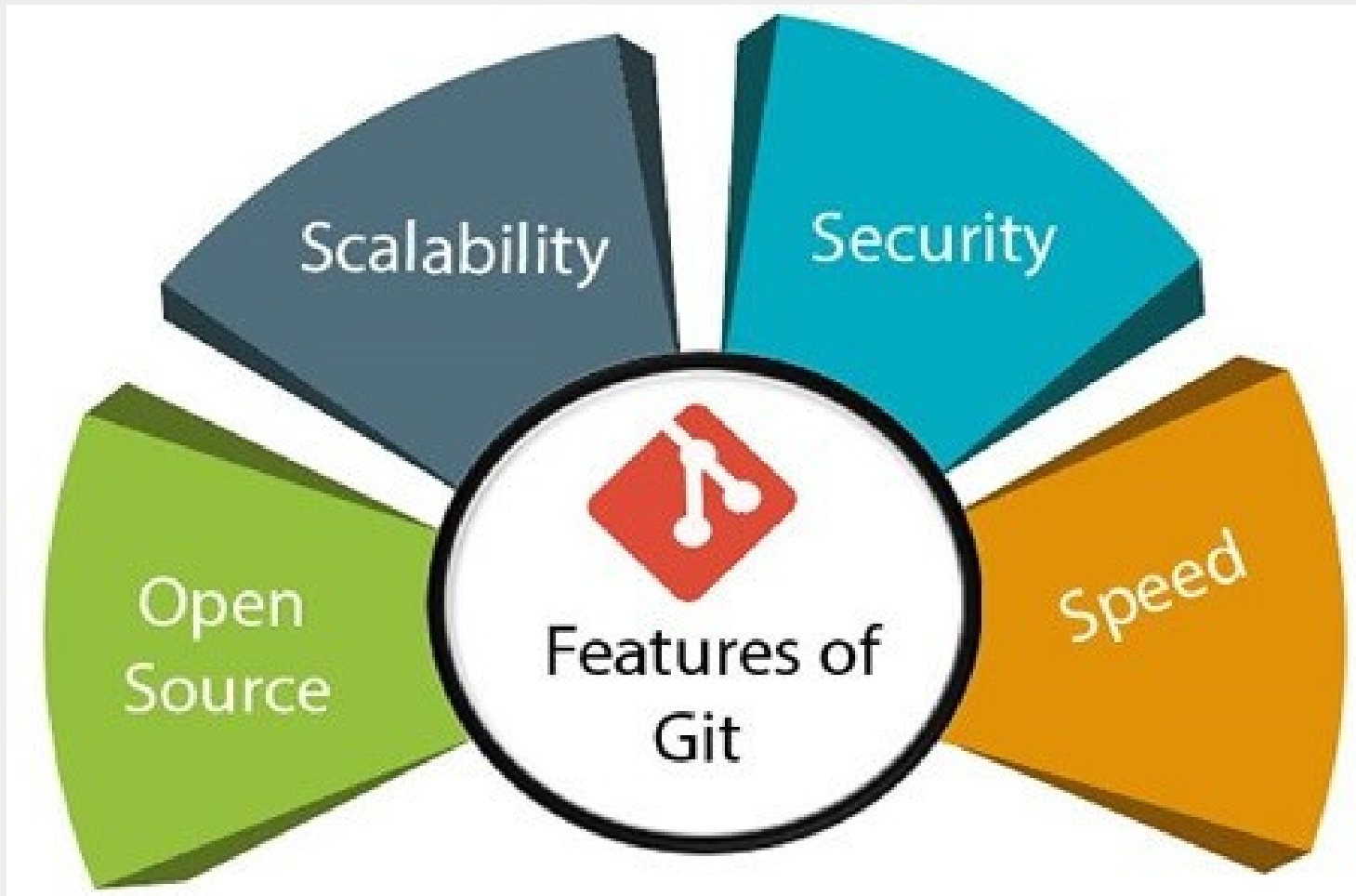
- Created by Linus Torvalds, creator of Linux, in 2005
 - Came out of Linux development community
 - Designed to do version control on Linux kernel



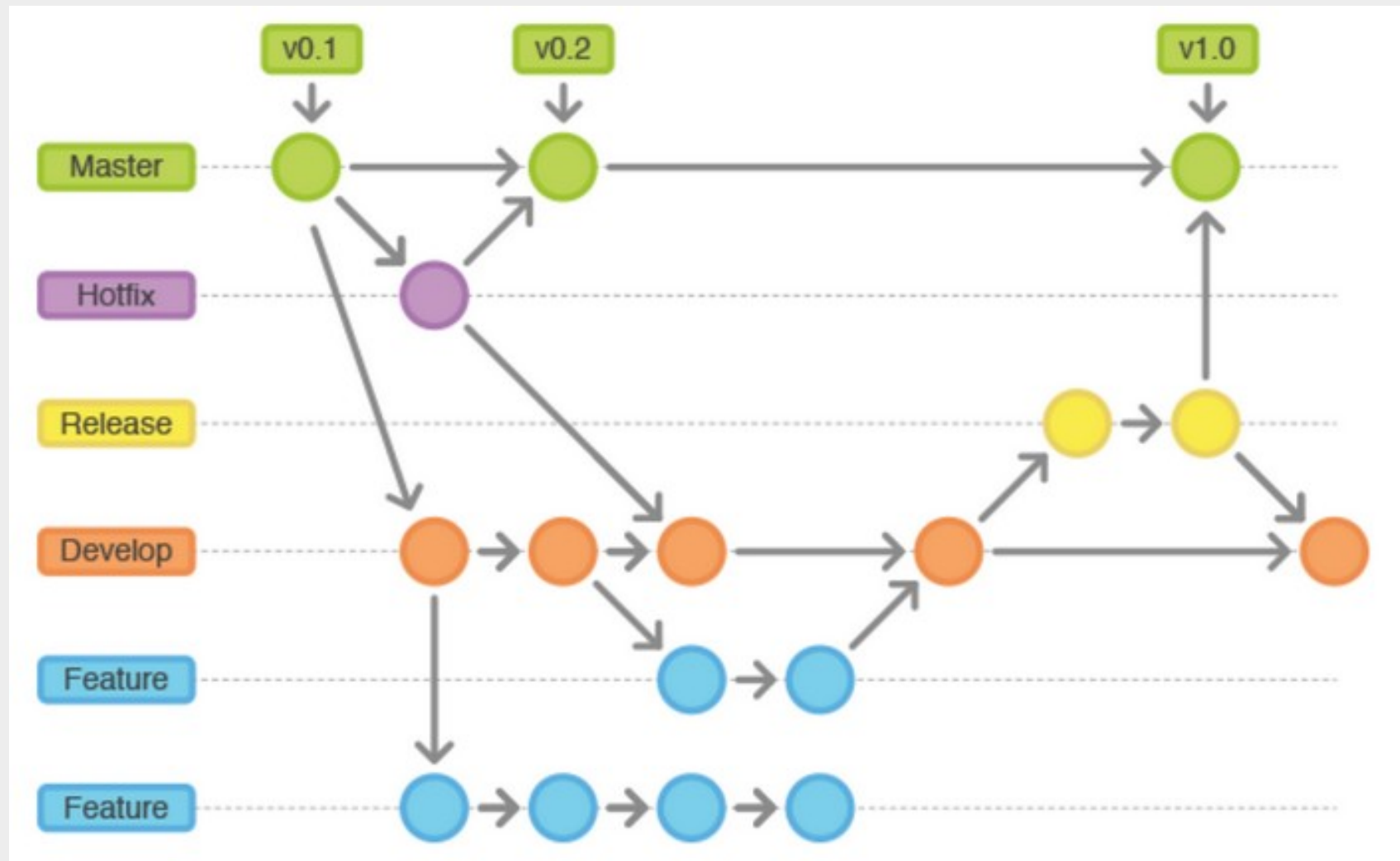
- Goals of Git:
 - Speed
 - Support for non-linear development (thousands of parallel branches)
 - Fully distributed
 - Able to handle large projects efficiently
- *(A "git" is a cranky old man. Linus meant himself.)*



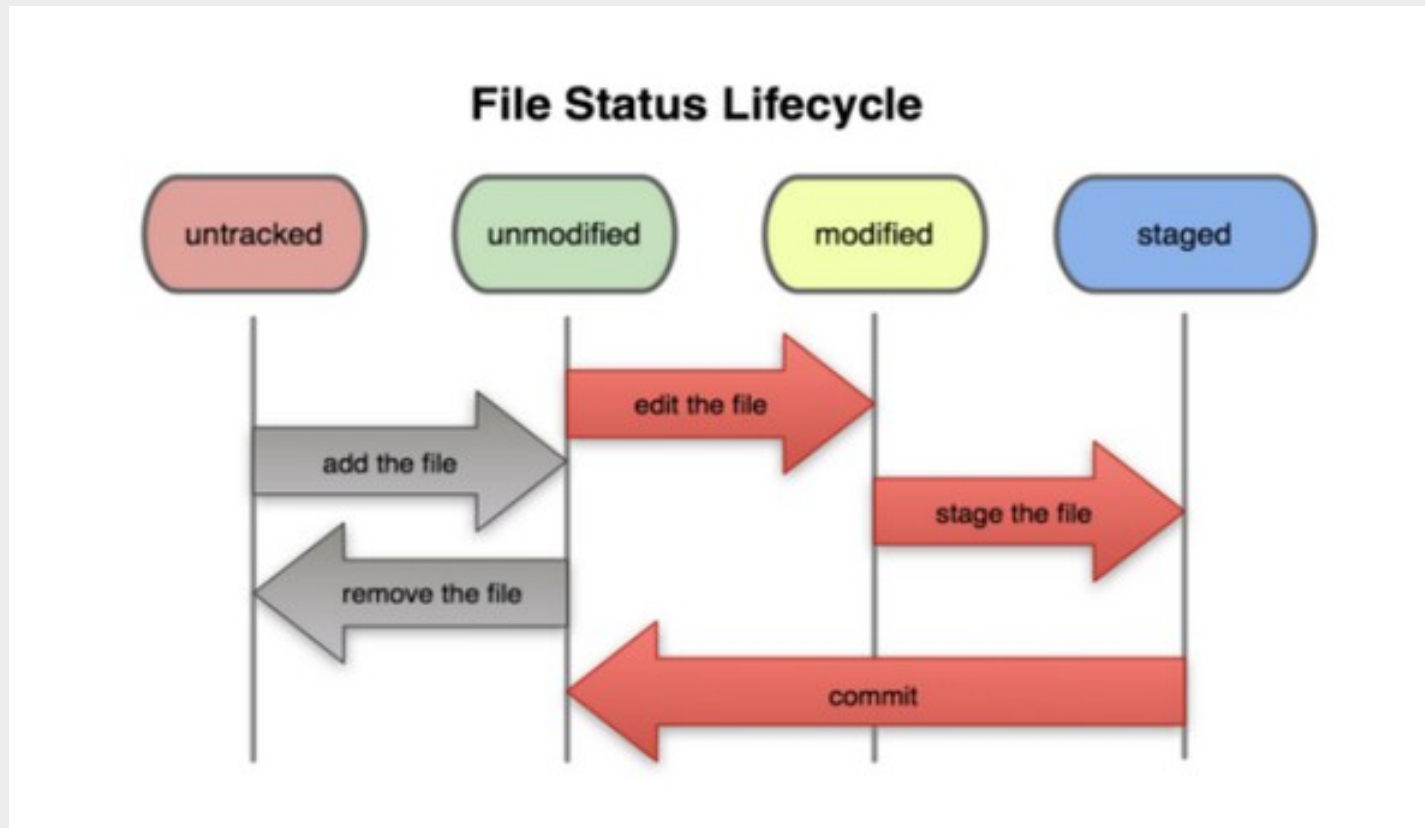
Why Git? (Features)



Why Git? (Features)



HOW? (VERY BASIC)



- Modify files in your working directory.
- Stage files, adding snapshots of them to your staging area.
- Commit, which takes the files in the staging area and stores that snapshot permanently to your Git directory.

HOW? (in terminal)



```
fullarostaky@malaka:~/Documents/ati_new/cursos/english/
[fullarostaky@malaka presentation]$ git init
Initialized empty Git repository in /home/fullarostaky/Documents/ati_new/cursos/english/presentation/.git/
[fullarostaky@malaka presentation]$ touch presentation.txt
[fullarostaky@malaka presentation]$ git status
On branch master

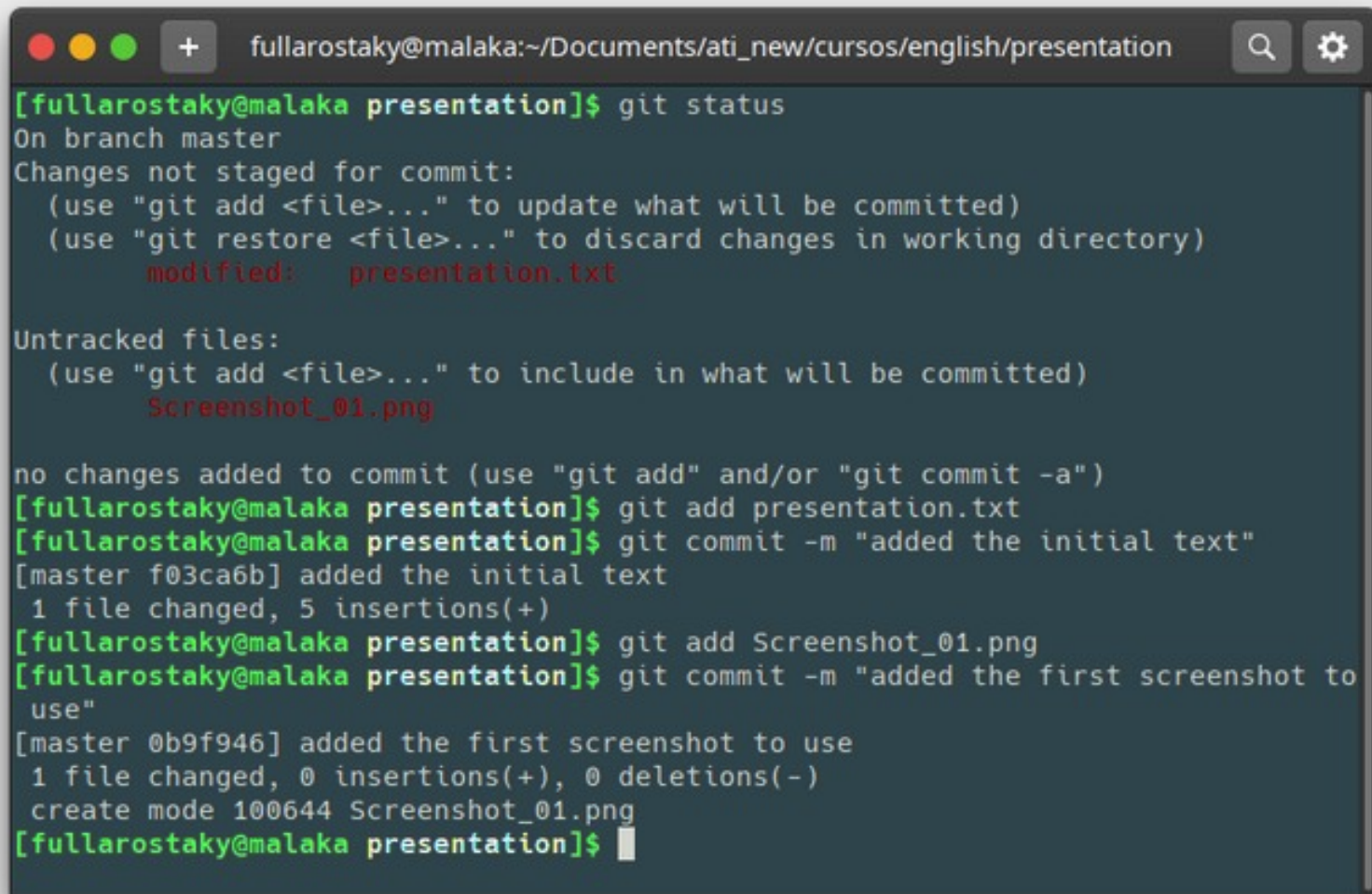
No commits yet

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

nothing added to commit but untracked files present (use "git add" to track)
[fullarostaky@malaka presentation]$ git add presentation.txt
[fullarostaky@malaka presentation]$ git commit -m "initial commit of my presentation"
[master (root-commit) d8203eb] initial commit of my presentation
 1 file changed, 0 insertions(+), 0 deletions(-)
 create mode 100644 presentation.txt
[fullarostaky@malaka presentation]$
```

- First we “init” a repo

HOW?



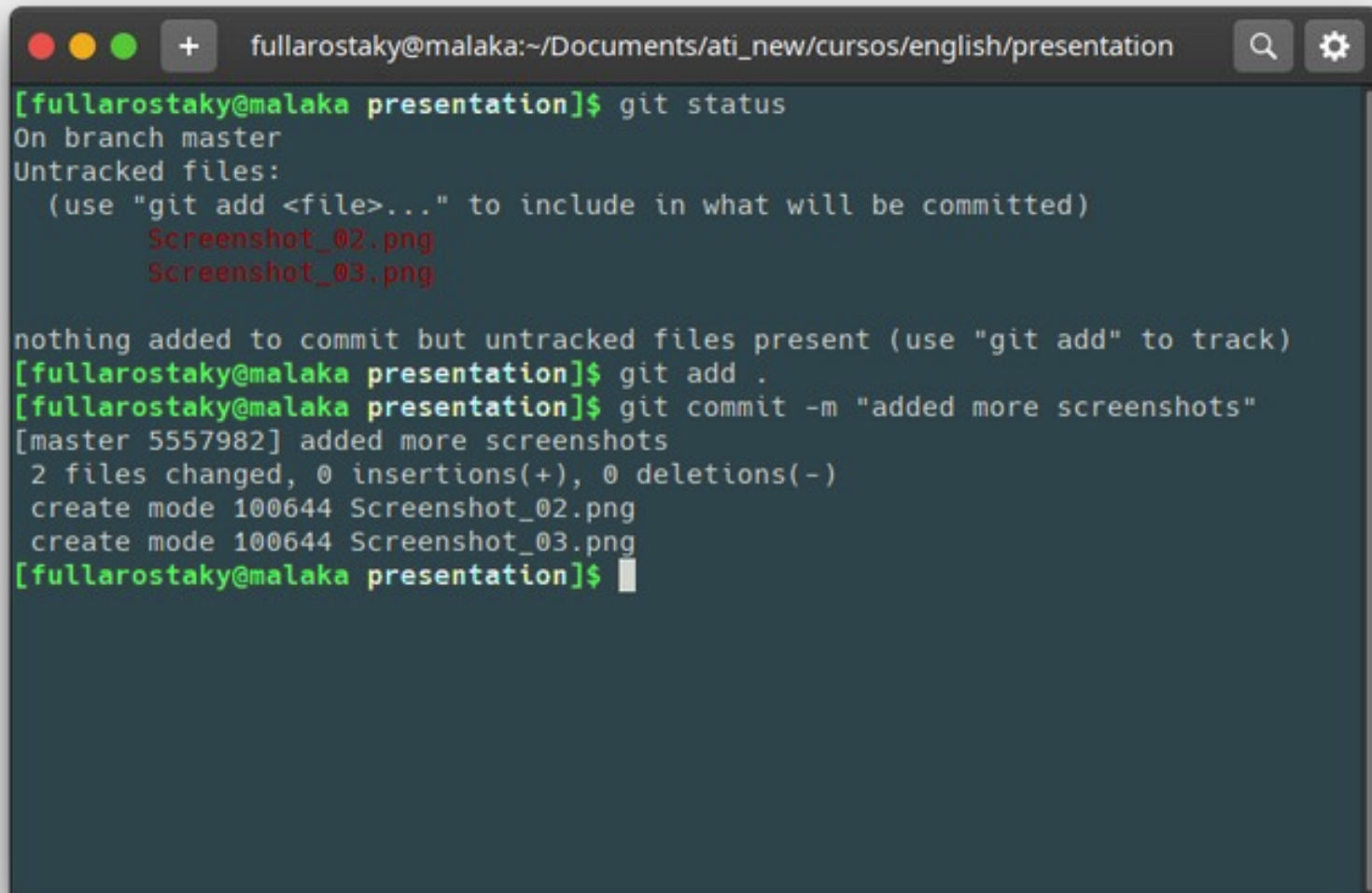
```
fullarostaky@malaka:~/Documents/ati_new/cursos/english/presentation
[fullarostaky@malaka presentation]$ git status
On branch master
Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git restore <file>..." to discard changes in working directory)
        modified:   presentation.txt

Untracked files:
  (use "git add <file>..." to include in what will be committed)
        Screenshot_01.png

no changes added to commit (use "git add" and/or "git commit -a")
[fullarostaky@malaka presentation]$ git add presentation.txt
[fullarostaky@malaka presentation]$ git commit -m "added the initial text"
[master f03ca6b] added the initial text
 1 file changed, 5 insertions(+)
[fullarostaky@malaka presentation]$ git add Screenshot_01.png
[fullarostaky@malaka presentation]$ git commit -m "added the first screenshot to use"
[master 0b9f946] added the first screenshot to use
 1 file changed, 0 insertions(+), 0 deletions(-)
 create mode 100644 Screenshot_01.png
[fullarostaky@malaka presentation]$
```

- Then we add files
- Then we commit our changes

HOW?

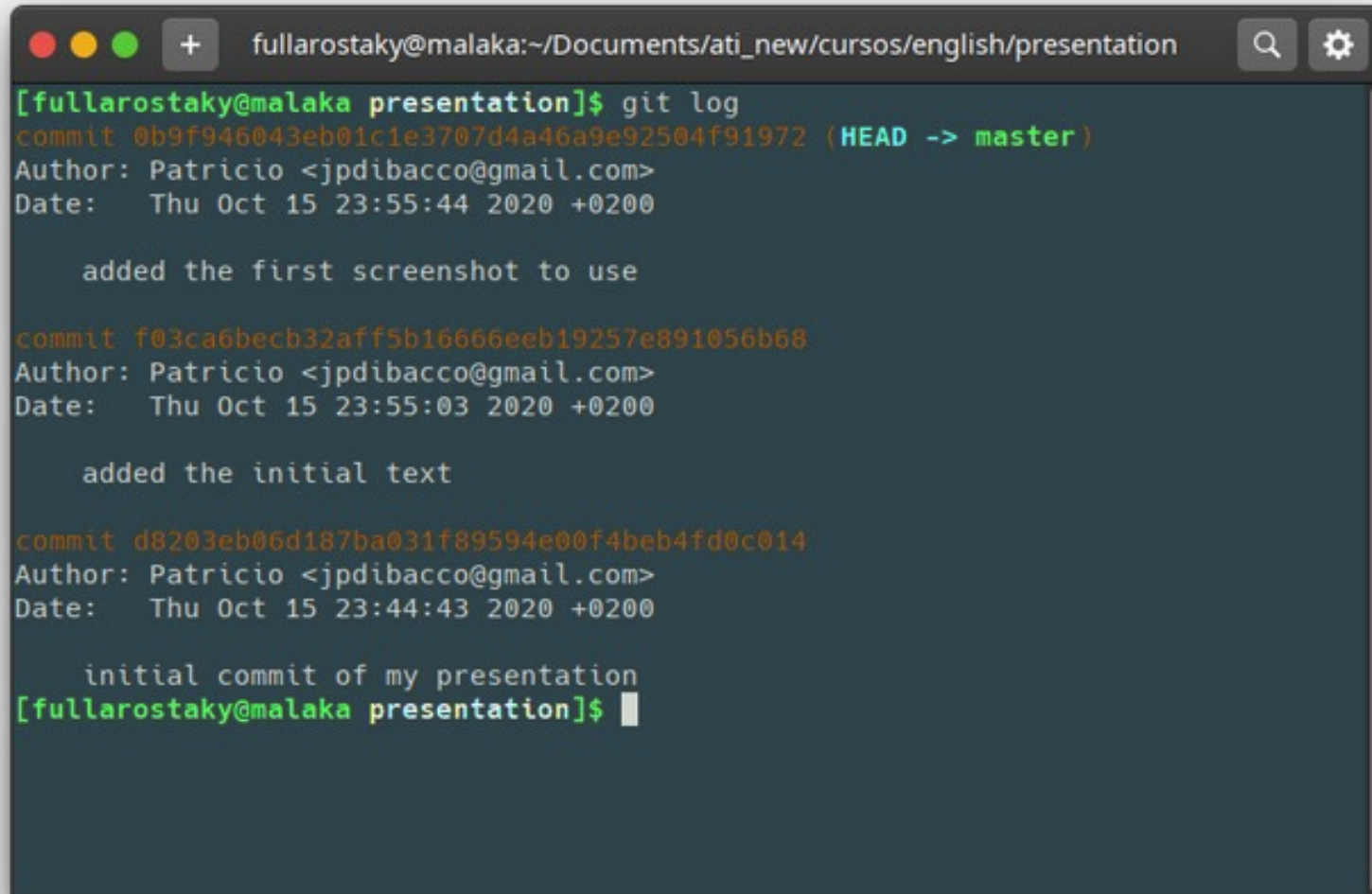
A terminal window with a dark background and light text. The window title bar shows the user 'fullarostaky@malaka' and the path '~/Documents/ati_new/cursos/english/presentation'. The terminal output shows the command 'git status' being executed, which reports that the user is on the 'master' branch and lists two untracked files: 'Screenshot_02.png' and 'Screenshot_03.png'. It then prompts the user to use 'git add' to track these files. The user then runs 'git add .' to stage the files. Finally, the user runs 'git commit -m "added more screenshots"', which creates a new commit on the 'master' branch with the message 'added more screenshots'. The commit details show that two files were changed, with no insertions or deletions, and both files were created with mode '100644'.

```
fullarostaky@malaka:~/Documents/ati_new/cursos/english/presentation
[fullarostaky@malaka presentation]$ git status
On branch master
Untracked files:
  (use "git add <file>..." to include in what will be committed)
        Screenshot_02.png
        Screenshot_03.png

nothing added to commit but untracked files present (use "git add" to track)
[fullarostaky@malaka presentation]$ git add .
[fullarostaky@malaka presentation]$ git commit -m "added more screenshots"
[master 5557982] added more screenshots
 2 files changed, 0 insertions(+), 0 deletions(-)
 create mode 100644 Screenshot_02.png
 create mode 100644 Screenshot_03.png
[fullarostaky@malaka presentation]$
```

- You can see changes in the working repository by using “git status”

HOW?

A terminal window with a dark background and light text. The window title bar shows the user 'fullarostaky@malaka' and the path '~/Documents/ati_new/cursos/english/presentation'. The terminal output shows the result of the 'git log' command, listing three commits in reverse chronological order. Each commit entry includes the commit hash, author name and email, date, and a description of the changes. The first commit is the most recent, followed by the second, and then the initial commit at the bottom.

```
fullarostaky@malaka:~/Documents/ati_new/cursos/english/presentation
[fullarostaky@malaka presentation]$ git log
commit 0b9f946043eb01c1e3707d4a46a9e92504f91972 (HEAD -> master)
Author: Patricio <jpdibacco@gmail.com>
Date: Thu Oct 15 23:55:44 2020 +0200

    added the first screenshot to use

commit f03ca6becb32aff5b16666eeb19257e891056b68
Author: Patricio <jpdibacco@gmail.com>
Date: Thu Oct 15 23:55:03 2020 +0200

    added the initial text

commit d8203eb06d187ba031f89594e00f4beb4fd0c014
Author: Patricio <jpdibacco@gmail.com>
Date: Thu Oct 15 23:44:43 2020 +0200

    initial commit of my presentation
[fullarostaky@malaka presentation]$
```

- You can see your “commits” in the repo with the command “git log”

CONCLUSION

- In conclusion, Git provides a way of keeping track of past versions of software and papers,
- making collaboration between various authors easy, and provides backup for your software.
- It has proven very useful to the open-source community and in academia as well.



Vocabulary

- Git = in English: “unpleasant person” , wiki: Git is a term of insult with origins in English denoting an unpleasant, silly, incompetent, annoying, senile, elderly or childish person. As a mild oath it is roughly on a par with prat and marginally less pejorative than berk.
- Torvalds sarcastically quipped about the name git (which means "unpleasant person" in British English slang): "I'm an egotistical bastard, and I name all my projects after myself. First 'Linux', now 'git'."The **man page** describes Git as "the stupid content tracker".
- Cranky = bad-tempered or irritable
- Repo (short for repository)
- “commit” in Git: A commit, or "revision", is an individual change to a file (or set of files). It's like when you save a file, except with Git, every time you save it creates a unique ID (a.k.a. the "SHA" or "hash") that allows you to keep record of what changes were made when and by who.
- “commit” in English: perpetrate or carry out (a mistake, crime or an immoral act
- a.k.a = also known as
- “Kernel” = in English: the central or most important part of something. The Linux kernel is the main component of a Linux operating system and is the core interface between a computer's hardware and its processes.



Sauce:

- <https://product.hubspot.com/blog/git-and-github-tutorial-for-beginners>
- <https://www.youtube.com/watch?v=USjZcfj8yxE>
(15 mins video, super nice)
- <https://courses.cs.washington.edu/courses/cse403/13au/lectures/git.ppt.pdf>
- <https://iamchuka.com/gitflow-workflow-continuous-integration-continu/>
I really recommend the last website, it explains everything in a very simple way.

Thank you!!

