git

By: Yazdan Jahedi

Shahid Beheshti University
Fall 2022

What is VCS?

- Version Control System

- Purpose?
- Pros?
- Types?

VCS types

- Distributed
- Centralized

Read more here

What is git?

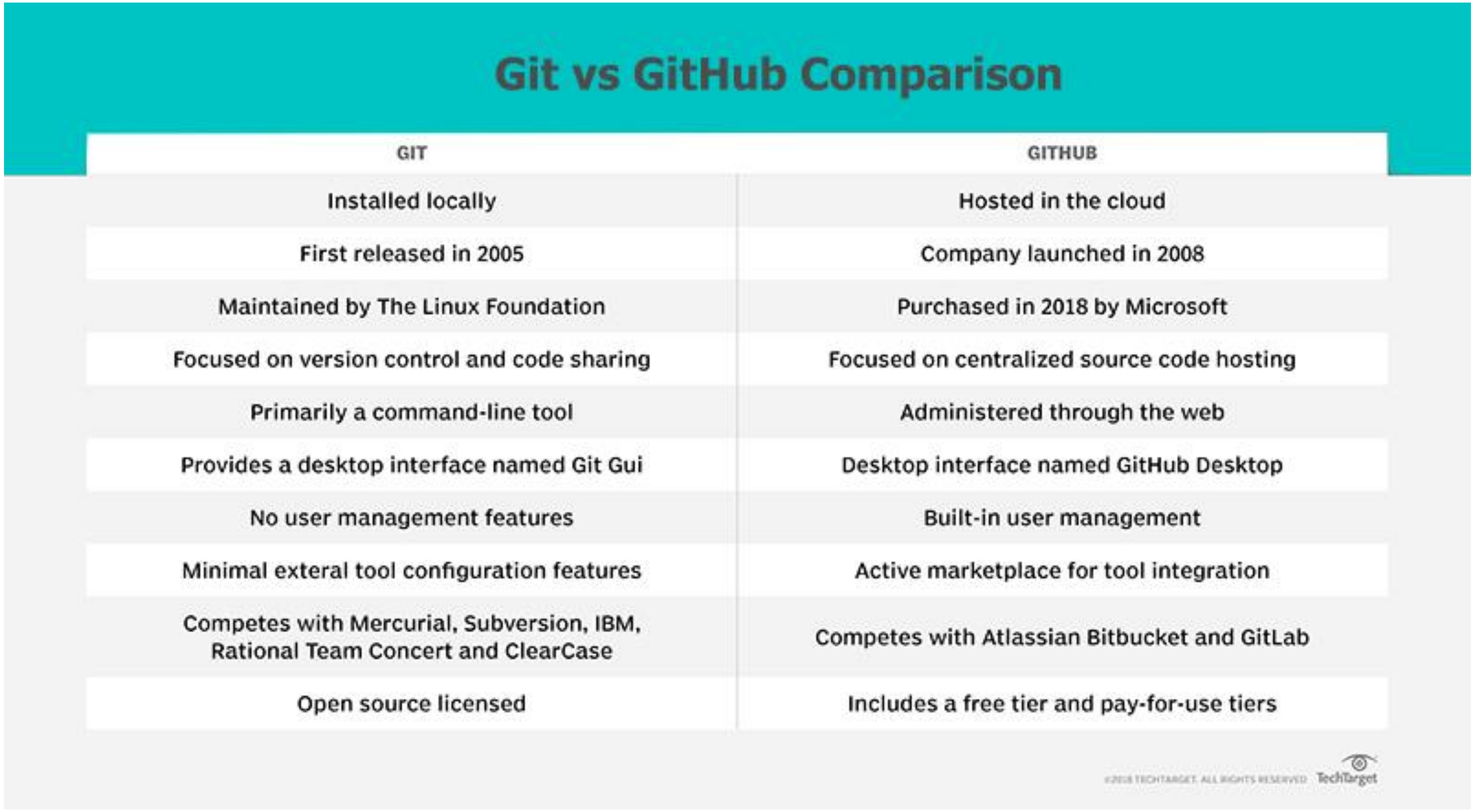
- distributed version control system

- Linus Torvalds

What else?

- Check here

git vs. GitHub



Git vs GitHub Comparison

GIT	GITHUB
Installed locally	Hosted in the cloud
First released in 2005	Company launched in 2008
Maintained by The Linux Foundation	Purchased in 2018 by Microsoft
Focused on version control and code sharing	Focused on centralized source code hosting
Primarily a command-line tool	Administered through the web
Provides a desktop interface named Git Gui	Desktop interface named GitHub Desktop
No user management features	Built-in user management
Minimal exteral tool configuration features	Active marketplace for tool integration
Competes with Mercurial, Subversion, IBM, Rational Team Concert and ClearCase	Competes with Atlassian Bitbucket and GitLab
Open source licensed	Includes a free tier and pay-for-use tiers

Install git

- Install from here

How to use git?

- For Windows   : git bash, power shell, CMD
- For Linux/Mac : terminal

After installation

- git --version

- Config:
  - git config [--global] user.name "myname"
  - git config [--global] user.email "myemail@gmail.com"

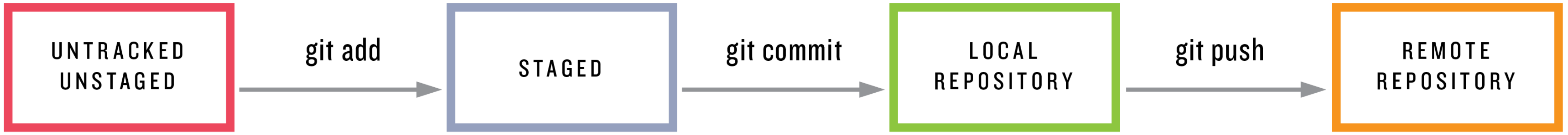
Note: You will probably also want to use your name and email when registering to GitHub later on
Note: Use global to set the username and e-mail for every repository on your computer

How to start?

- git init

Note: git now knows that it should watch the folder you initiated it on
Note: git creates a hidden folder to keep track of changes (.git)

Basics


UNTRACKED UNSTAGED
git add
STAGED
git commit
LOCAL REPOSITORY
git push
REMOTE REPOSITORY

git commit
AIRBUS A350-1000
1000
A350
git push
AIRBUS A350-1000
1000
A350
git add .

Basic commands

1. git status
   - Note: --short option

?? → Untracked files
A → Files added to stage
M → Modified files
D → Deleted files

Note: Files in your Git repository folder can be in one of 2 states:
1) Tracked: files that Git knows about and are added to the repository
2) Untracked: files that are in your working directory, but not added to the repository

2. git add <file or directory name>

- Note: Using "--all" or "-A" or "." will stage all changes(new, modified, and deleted) files

3. git commit –m ‘commit message’

- Commits must have a commit message
- Save point

- -a option : to commit without staging → NOT RECOMENDED
  - Example: git commit –a –m ‘commit message’

- A useful tip: changing the last commit message
  - git commit --amend
  - git commit --amend -m "my new message"

4. git log

- --oneline option:
  1) The first seven characters of the commit hash
  2) the commit message

5. git diff

- Delete, Rename, change path?
  - git mv
    - Example: git mv name1 name2
    - Example: git mv path1 path2
  - git rm
  - …

How to remember all these commands?

- No need to remember all!
- Just remember the basic ones

- Solution:
  - Searching
  - git [command] -help   : See all the available options for the specific command
  - git [command] --help : to open the relevant Git manual page
  - git help --all   : See all possible commands (NOT RECOMMEDED!)

git Revert & Reset

- revert vs. reset
  - Revert: take a previous commit and add it as a new commit, keeping the log intact.
  - Reset: move back to a previous commit, discarding any changes made after that commit.
  - Read more here

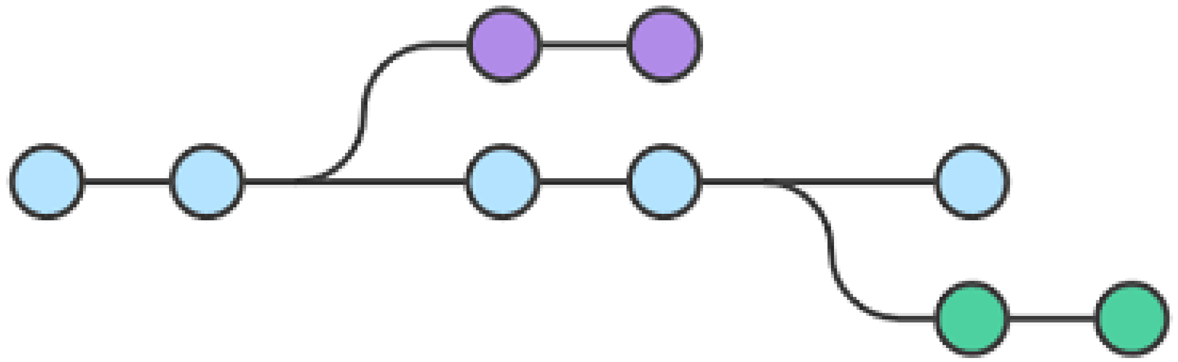
- git revert [commit_id or hash]

- git reset --hard [commit_hash]
  - we can find commit hash from : git log --oneline

Note: (undo reset!) Even though the commits are no longer showing up in the log, it is not removed from git.
→ If you know the commit hash you can reset to it

Branch

- Concept:



Branches allow you to work on different parts of a project "without impacting" the main branch.
When the work is complete, a branch can be merged with the main project.

Branch

- git branch new_branch
  - to create a new branch
- git branch
  - to see branches
  - current branch is shown with * sign
- git checkout branch_name
  - Move to another branch
  - -b option: create branch and move to it
- git branch -d branch_name
  - to delete a branch

- git branch -a
  - -a option: to see all local and remote branches

- git branch -r
  - -r option: to see remote branches only.

Merge branches

First, we need to change to the master branch:

- git merge branch_name

- **Merge Conflict:**
  - When we changed a file in master branch and also the files is changed in branch_nam, when we want to merge them, git can't merge them correctly.

.gitignore

- Blank lines are ignored
- Lines starting with # are ignored (comments)

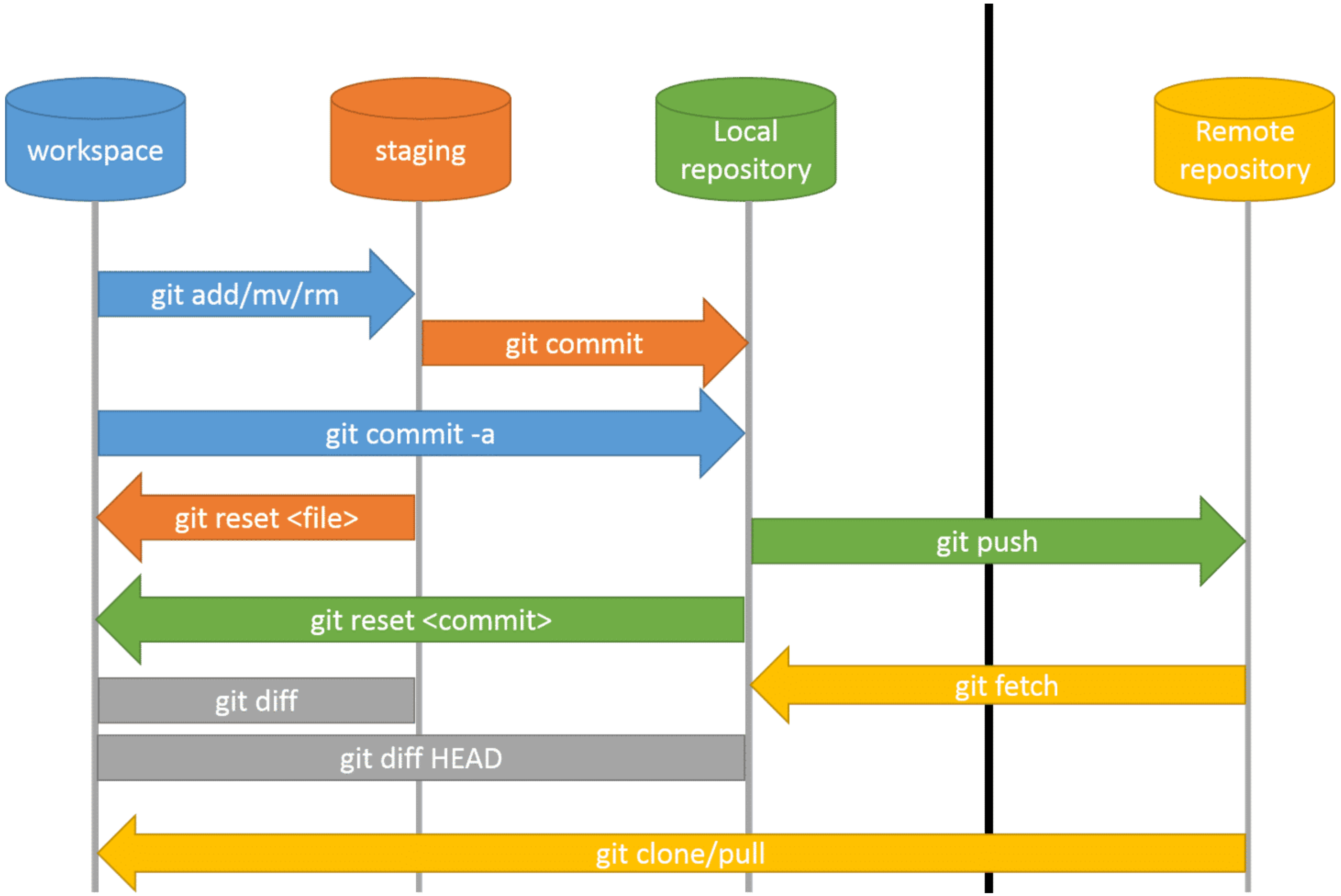
- Wildcards are used!

- *.file → All files withe .file extention
- !name.file → ! specifies a negation or exception. All files withe .file extention, except name.file

GitHub

GitHub

- git remote add origin <link of repo>
  - Find reop link on GitHub page
- git push origin master
  - Master → any other branch
- git pull origin master
  - Fetch and merge!
- git clone <repo link>


workspace
staging
Local repository
Remote repository
git add/mv/rm
git commit
git commit -a
git reset <file>
git push
git reset <commit>
git fetch
git diff
git diff HEAD
git clone/pull

Thank you!