

Introduction to GIT

Mohamad Mahdi Mahmoodian

Mahmoodian.m1999@gmail.com

Based on <https://www.atlassian.com/git/tutorials>

Table of content

- A historical problem in software development
- Version control, What? Why? How?
- A brief history of version control in software development
- What is GIT?
- Basic GIT concepts
- GIT commands
- GIT conflicts
- Q&A

A Historical Problem: *Final-Final-Final.zip*

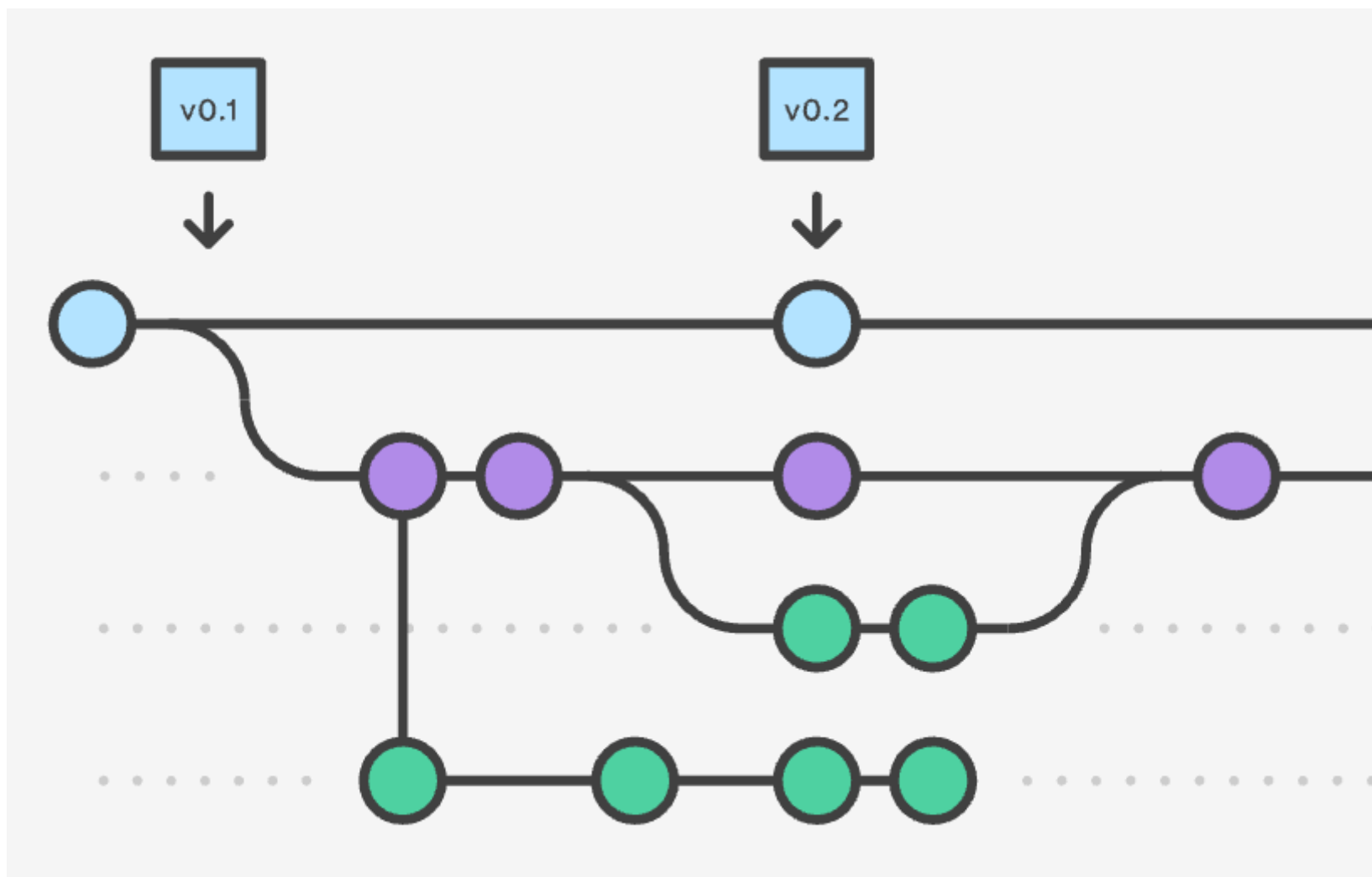
- Continues development
- Keep track of changes
- Team work



- Making new releases for project(also naming them)

Version control, What? Why? How?

- A version control system keep track of changes made to project/files
- A solution to continuously deliver/receive new versions
- Simply make new releases
- No need for naming conventions :D



What about team work?

A brief history of version control

- The Linux Kernel was developed using the commercial BitKeeper VCS
- Free software version control systems such as RCS and CVS
- additional restrictions on free version of Bitkeeper
- Searching for alternatives(reliable, efficient, and powerful)
- Birth of GIT!

What is GIT?



- “Git is a free and open source distributed version control system designed to handle everything from small to very large projects with speed and efficiency.” – [GIT website](#)
- Git track every changes in files, usually used for coordinating a group of programmers collaboratively developing a project. Its goals include speed, data integrity, and support for distributed, non-linear workflows

MASTER

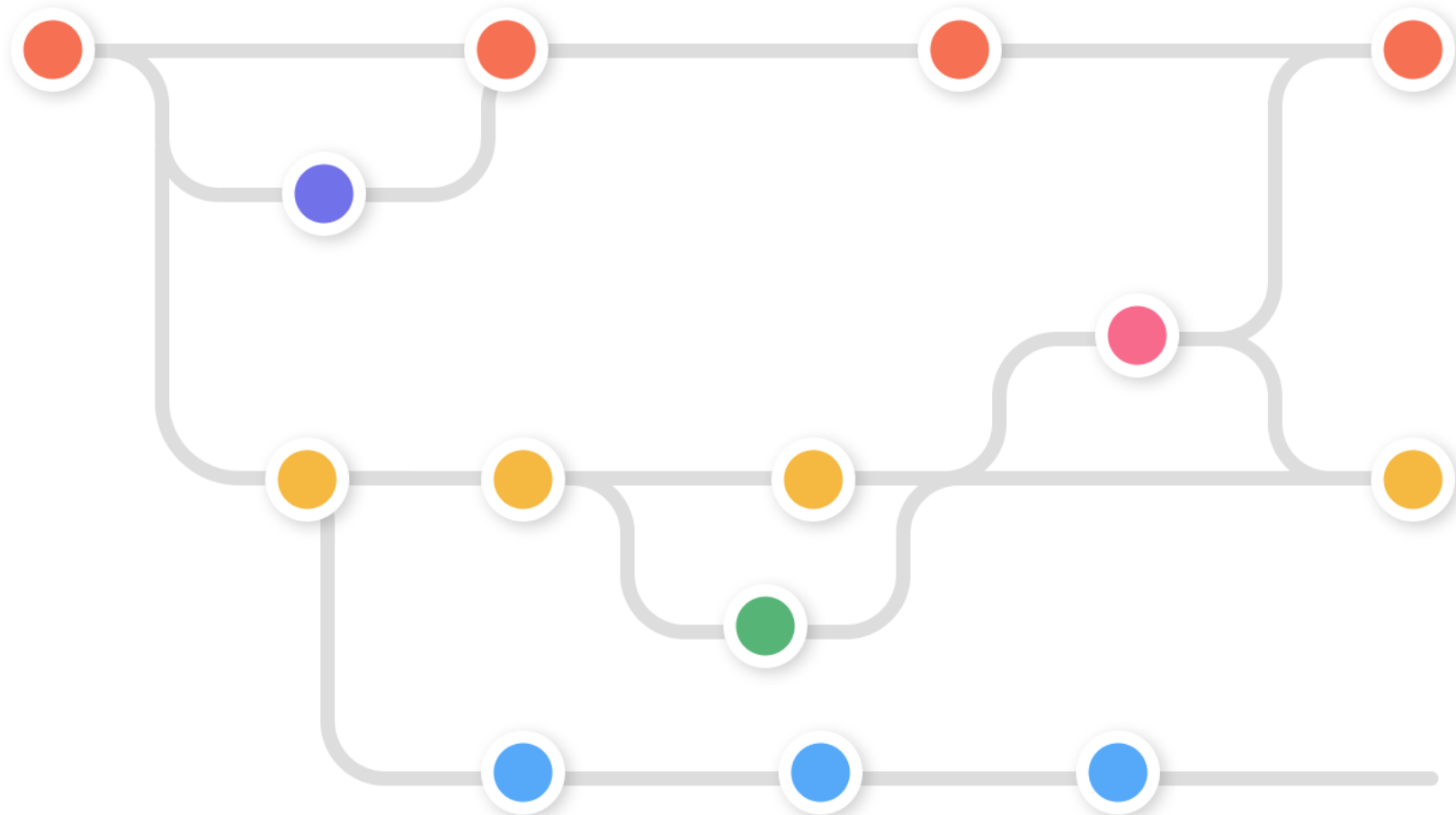
HOTFIX

RELEASE

DEVELOP

FEATURE

FEATURE



Basic GIT concepts – local

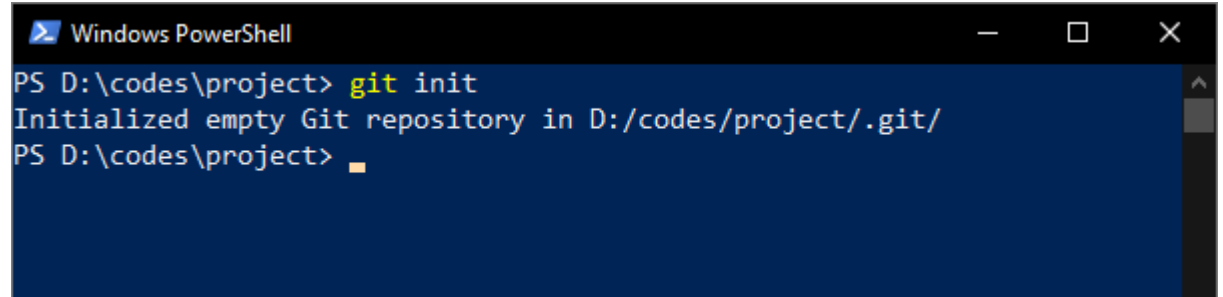
- Repository/Repo
 - Virtual storage of project
 - Contains all the changes, versions, etc.
 - Move forward/backward in changes or versions
- GIT add
 - Applying new changes/files to project
- GIT commit
 - Capturing a new snapshot of project

Basic GIT concepts – remote

- Remote repo
 - Virtual storage of project somewhere else(other than your machine)
 - Usually shared repositories for syncing changes with others
- GIT push
 - Send local changes to remote repo
- GIT pull
 - Update local repo from remote repo

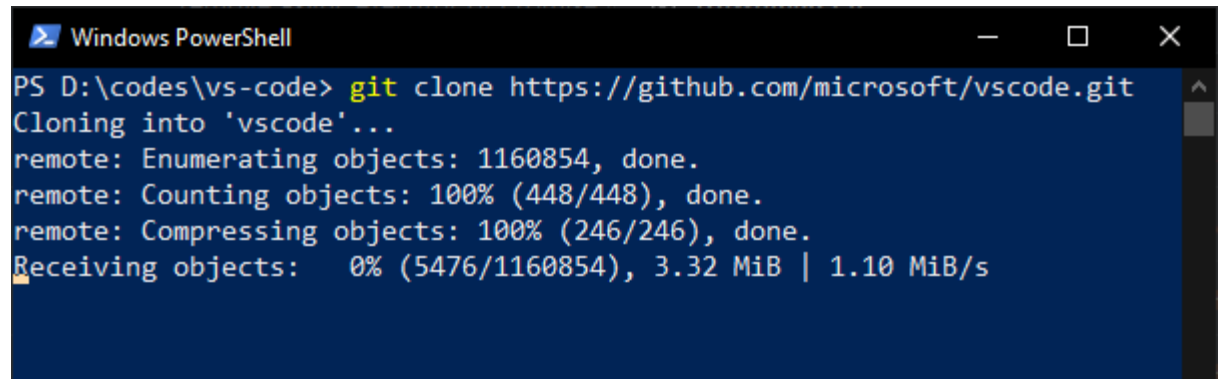
GIT commands - initializing

- Create new project
 - `git init`



```
Windows PowerShell
PS D:\codes\project> git init
Initialized empty Git repository in D:/codes/project/.git/
PS D:\codes\project> █
```

- Get an old project
 - `git clone remote_url`



```
Windows PowerShell
PS D:\codes\vs-code> git clone https://github.com/microsoft/vscode.git
Cloning into 'vscode'...
remote: Enumerating objects: 1160854, done.
remote: Counting objects: 100% (448/448), done.
remote: Compressing objects: 100% (246/246), done.
Receiving objects: 0% (5476/1160854), 3.32 MiB | 1.10 MiB/s
█
```

GIT commands – checking changes

- Checking changes/ updated files/ untracked files
 - Show all not applied/submitted changes in project
 - Command:
 - `git status`
- Checking differences(current changes with last snapshot)
 - `git diff`
 - `git diff file_name`

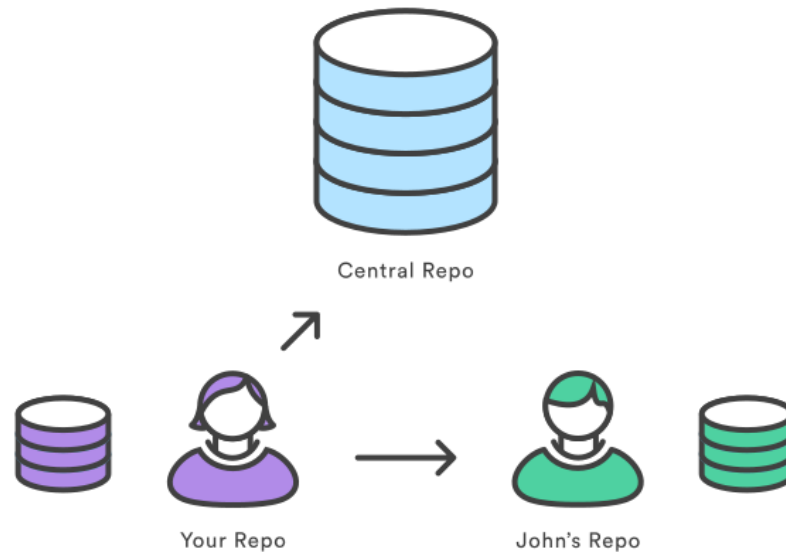
GIT commands – submitting changes

- Applying new changes/ adding new files to project
 - GIT only tracks added files
 - We need to add all files to repository
 - Commands:
 - `git add file` (single file)
 - `git add .` (all files)
- Capturing snapshots
 - Flagging a version for further use
 - `git commit`
 - `git commit -m "commit message"`

Subject	{	feat(2FA): Add two factor authentication
Body	{	Add 2FA [1] option in the user profile. This is an additional security option to verify user authenticity with SMS verification code. If user has a verified phone number he/she can use this feature and we send a verification code to his/her phone number. Otherwise upon enabling 2FA a prompt will appear, asking user to add/verify his/her phone number.
Footer	{	[1]: https://en.wikipedia.org/wiki/Multi-factor_authentication Closes #20

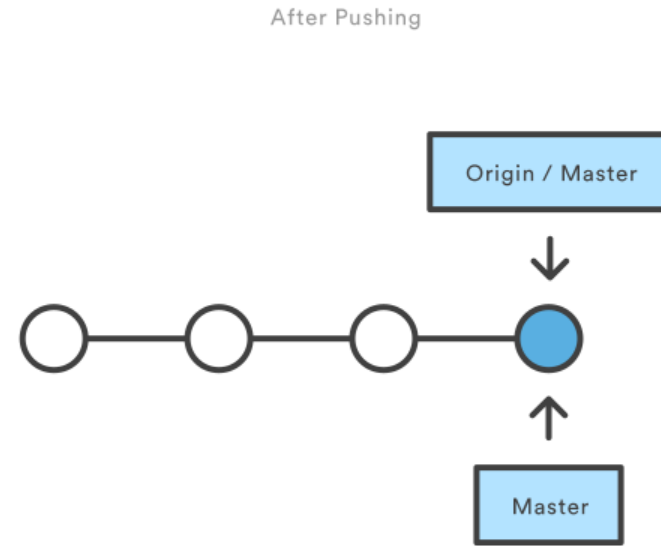
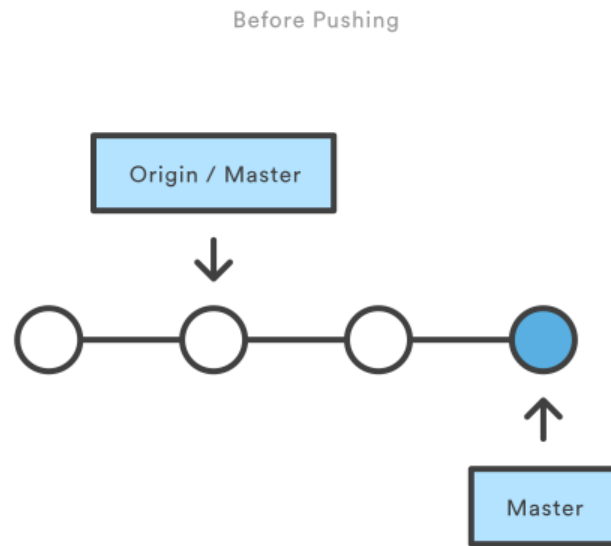
Git commands – remote

- Adding a new remote repo
 - `git remote add <name> <url>`
 - `git remote add origin https://github.com/microsoft/vscode.git`



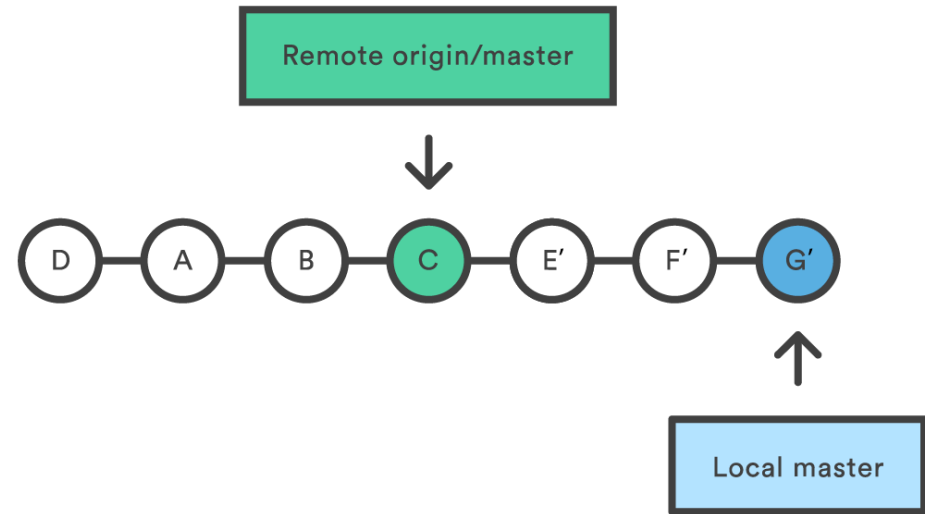
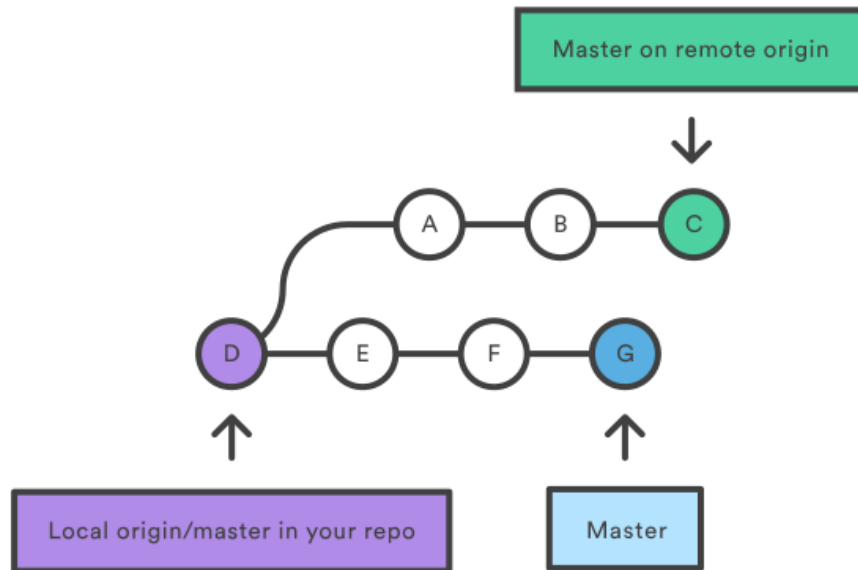
Git commands – push

- Send local committed changes to remote repo
 - `git push <remote_name> <branch>`



Git commands – pull

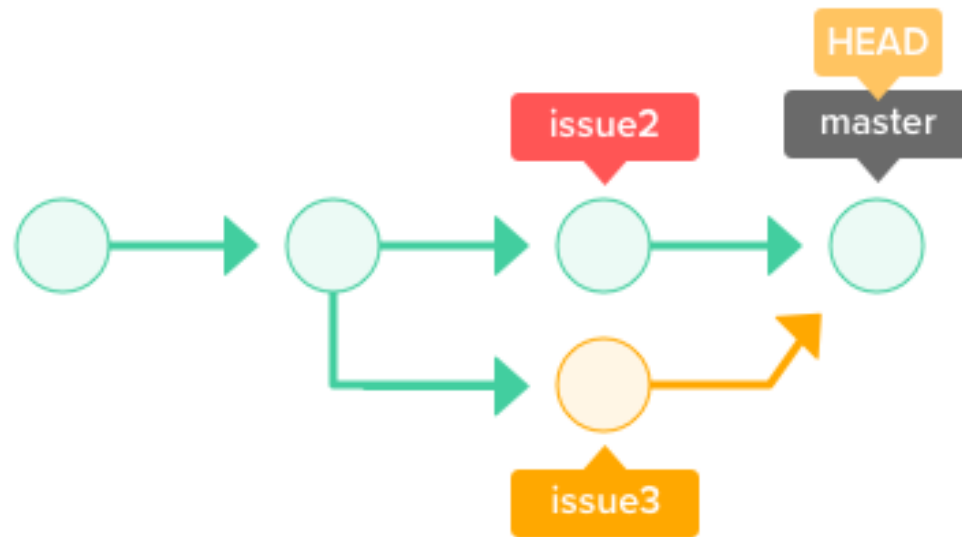
- update local repository to remote repo version
 - `git pull <remote_name> <branch>`



Conflicts!



GIT conflicts



Q&A