Git Hands-On

2023.12.29. 금 | 서폿

Git?

이전 시스템과 깃의 장점

Git

git --everything-is-local

Search entire site...

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 is easy to learn and has a tiny footprint with lightning fast performance. It outclasses SCM tools like Subversion, CVS, Perforce, and ClearCase with features like cheap local branching, convenient staging areas, and multiple workflows.

Version Control System (VCS)

회사원 오리씨 16화 https://hello-ori-c.tistory.com/4

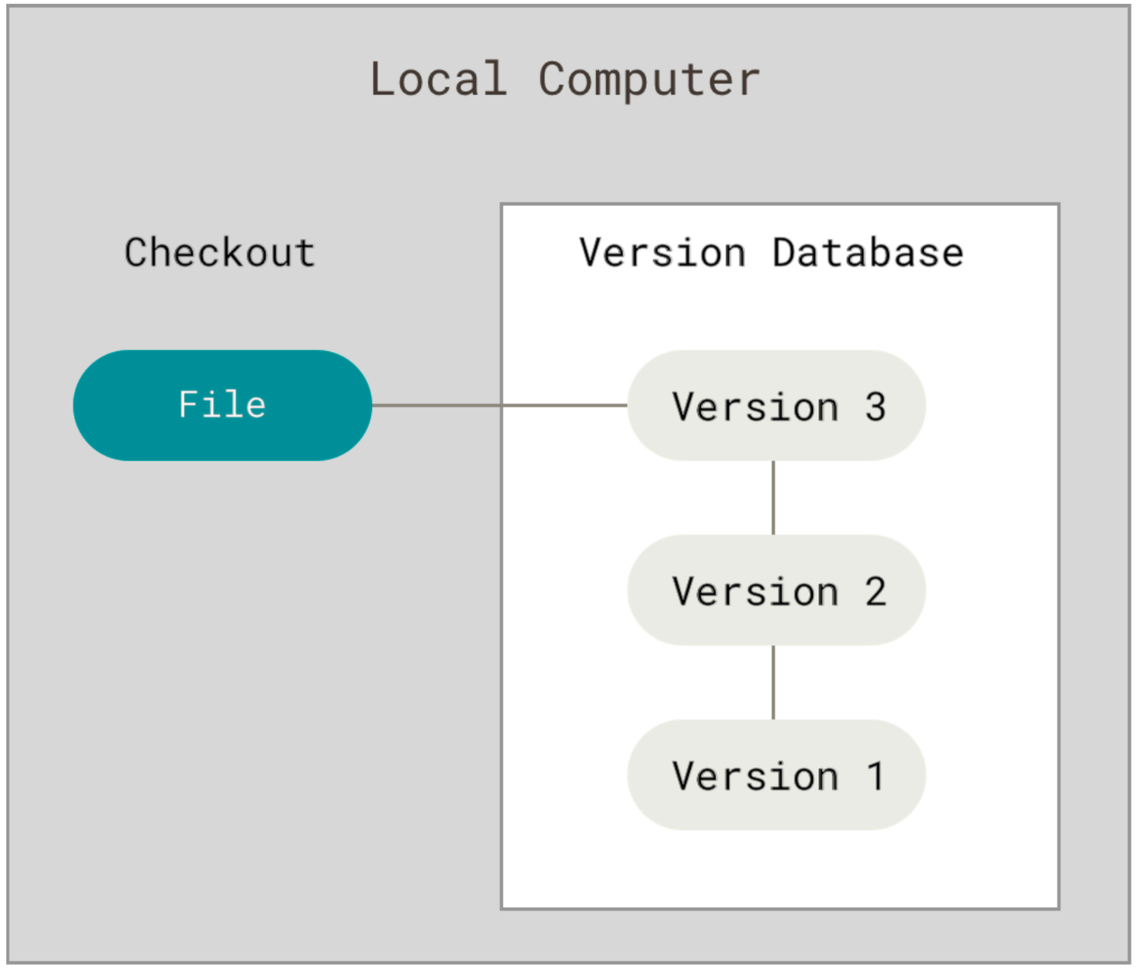
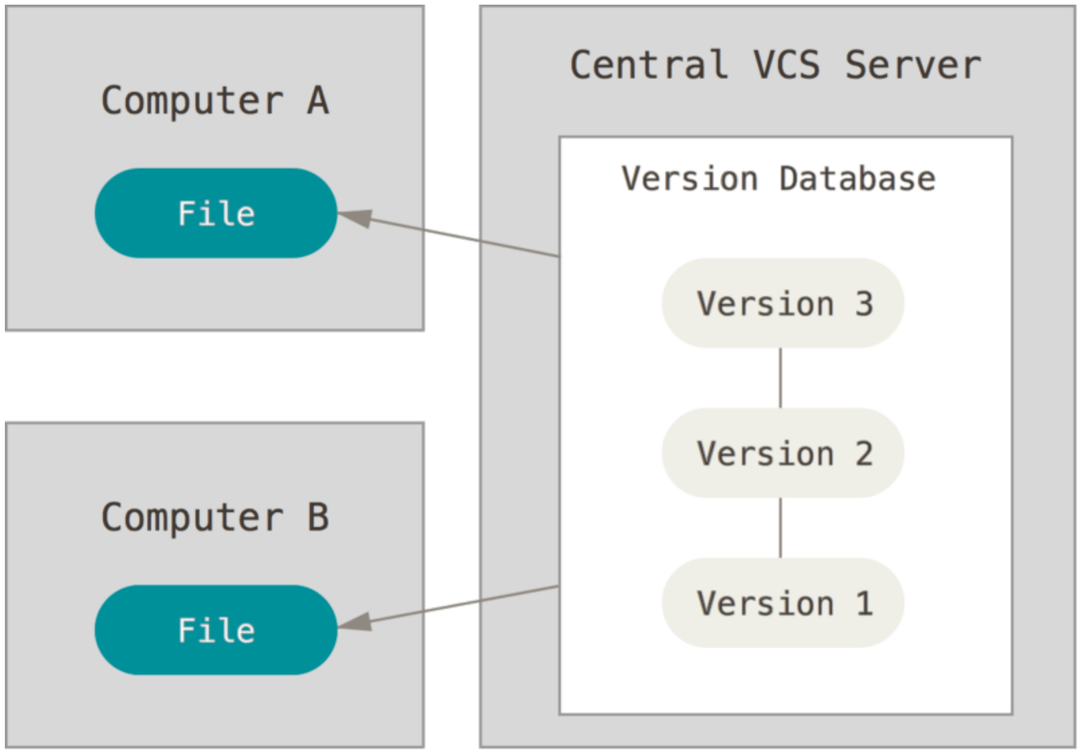


Fig 1. Local version Control Diagram, Chacon, S., & Straub, B. (2014).

Version Control System (VCS)



Centralized version Control Diagram, Chacon, S., & Straub, B. (2014).

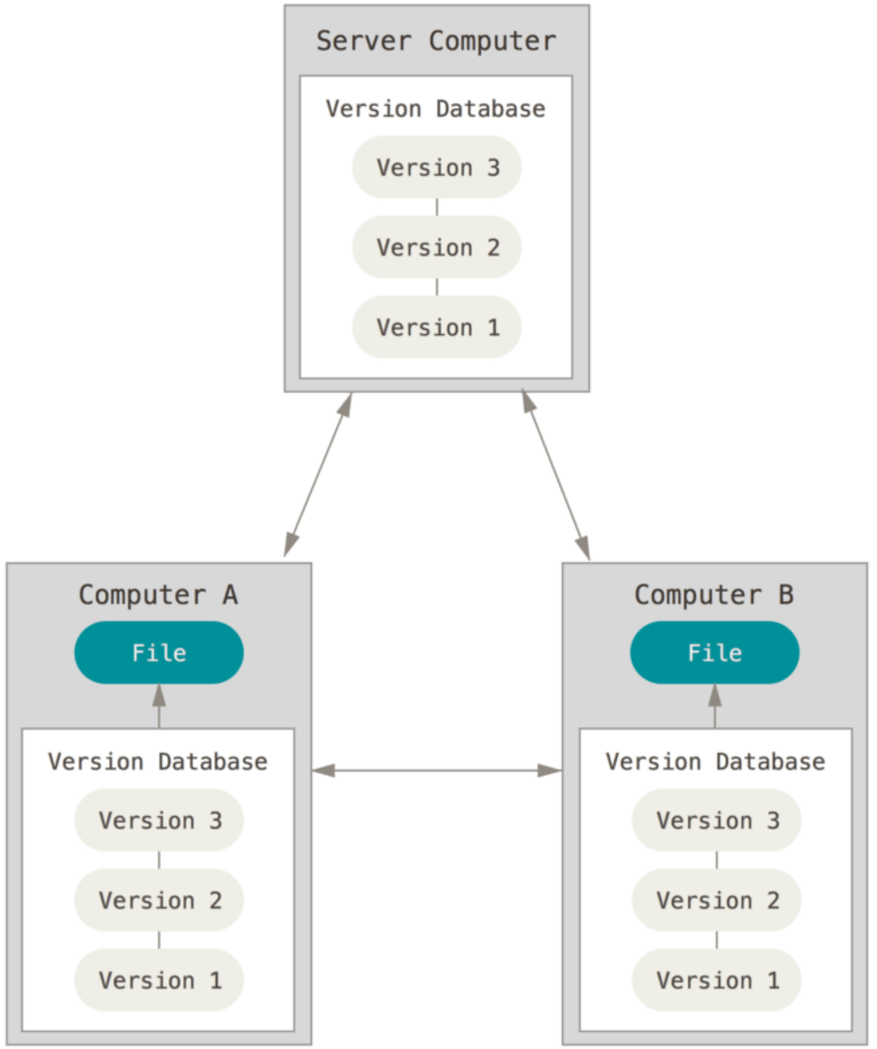


Fig 3. Distributed Version Control Diagram,
Chacon, S., & Straub, B. (2014).

Centralized VCS vs Distributed VCS

Version Control System	CVCS	DVCS
Repository	There is only one central repository which is the server.	Every user has a complete repository which is called local repository on their local computer.
Repository Access	Every user who needs to access the repository must be connected via network.	DVCS allows every user to work completely offline. But user need a network to share their repositories with other users.
Example of VCS Tools	Subversion, Perforce Revision Control System	Git, Mercurial, Bazaar, BitKeeper
Software Characteristics that suitable	i. Projects that allow only several users to contribute to the software development. ii. Team located in a single site.	i. DVCS is suitable for a single or more developers because the project repository is distributed to all the developers and this ability offer a great improvement for the projects. ii. It also can be applied for a small or big software projects because it makes less difficult for normal users to contribute to the development. iii. Team located in multiple site or different countries and different timezones.

Table 1. Comparison Between CVCS and DVCS , N., Ngah, A., & Deraman, A. (2018)

Git의 장점

git --everything-is-local

Search entire site...

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.

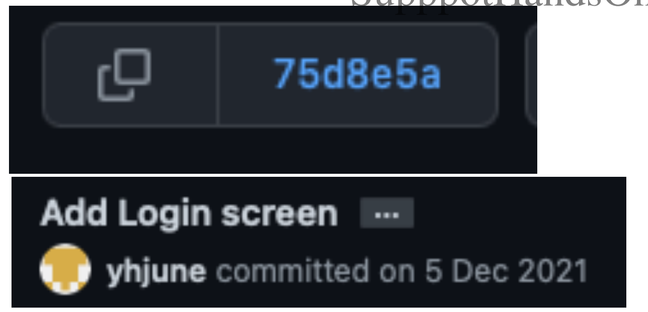
Git is easy to learn and has a tiny footprint with lightning fast performance. It outclasses SCM tools like Subversion, CVS, Perforce, and ClearCase with features like cheap local branching, convenient staging areas, and multiple workflows.

GitHub

Git Basic +

Nutshell and Workflows

Snapshot, Local, Integrity and Adds



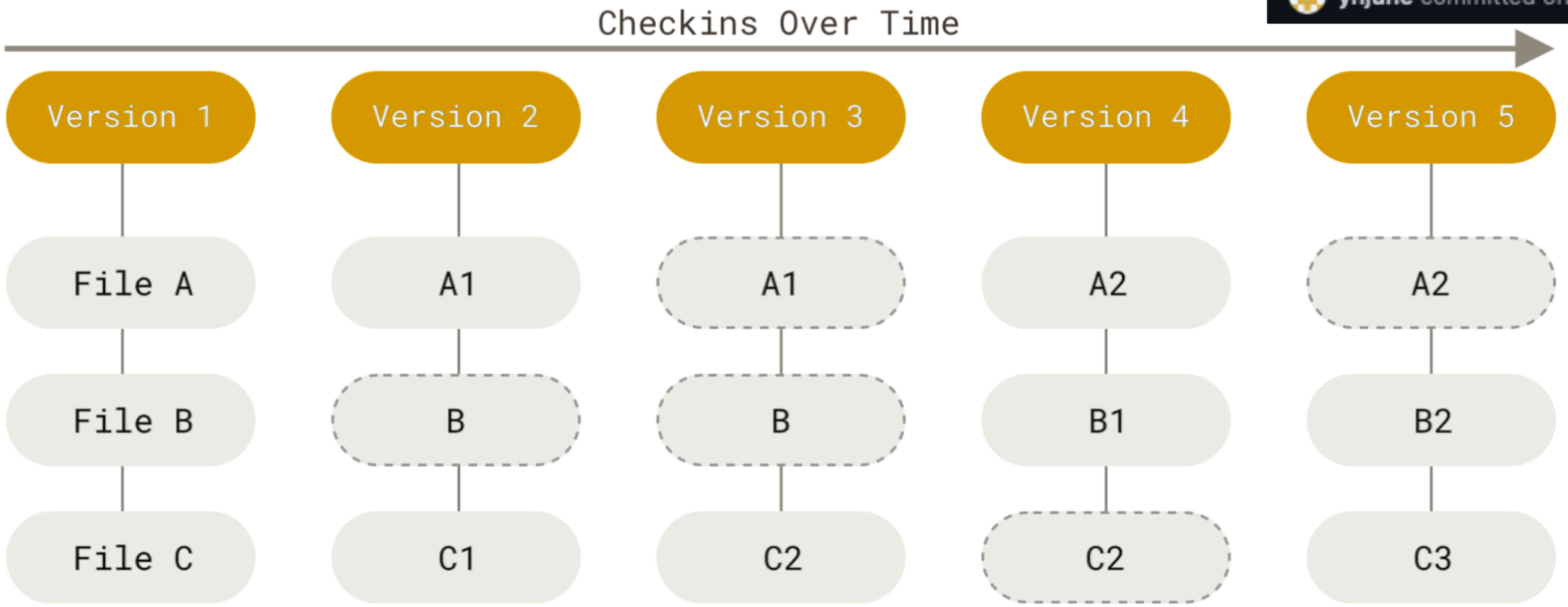


Fig 5. Storing data as snapshot of the project over time, Chacon, S., & Straub, B. (2014).

States

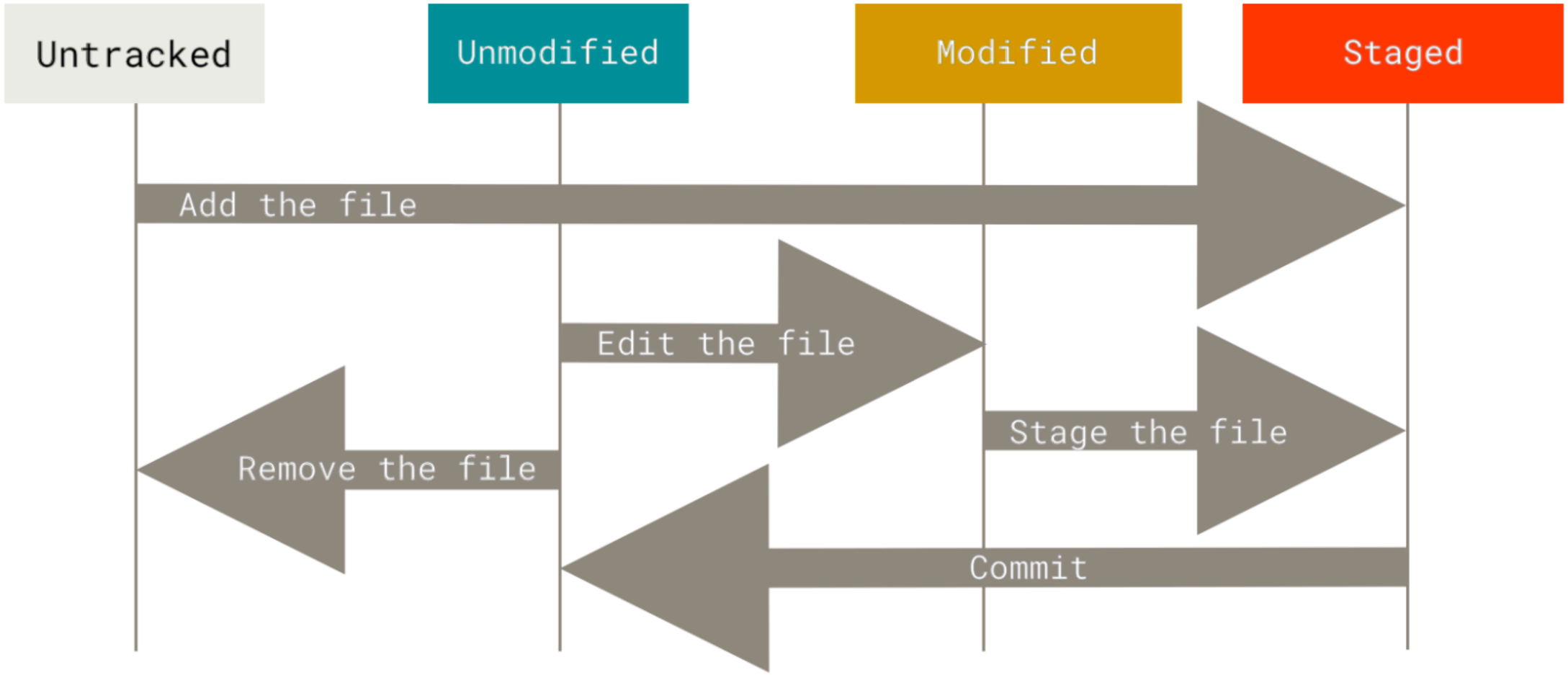
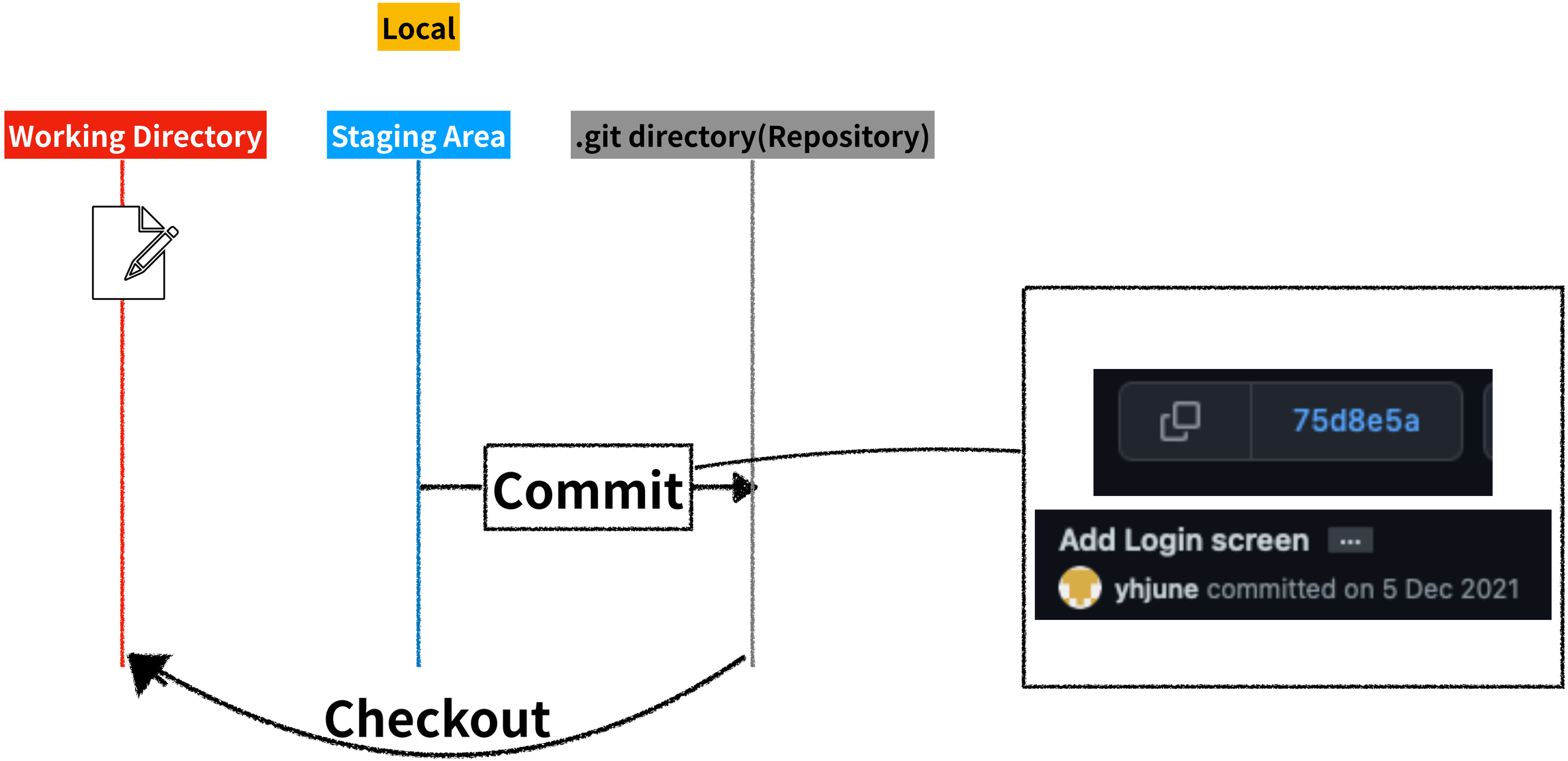


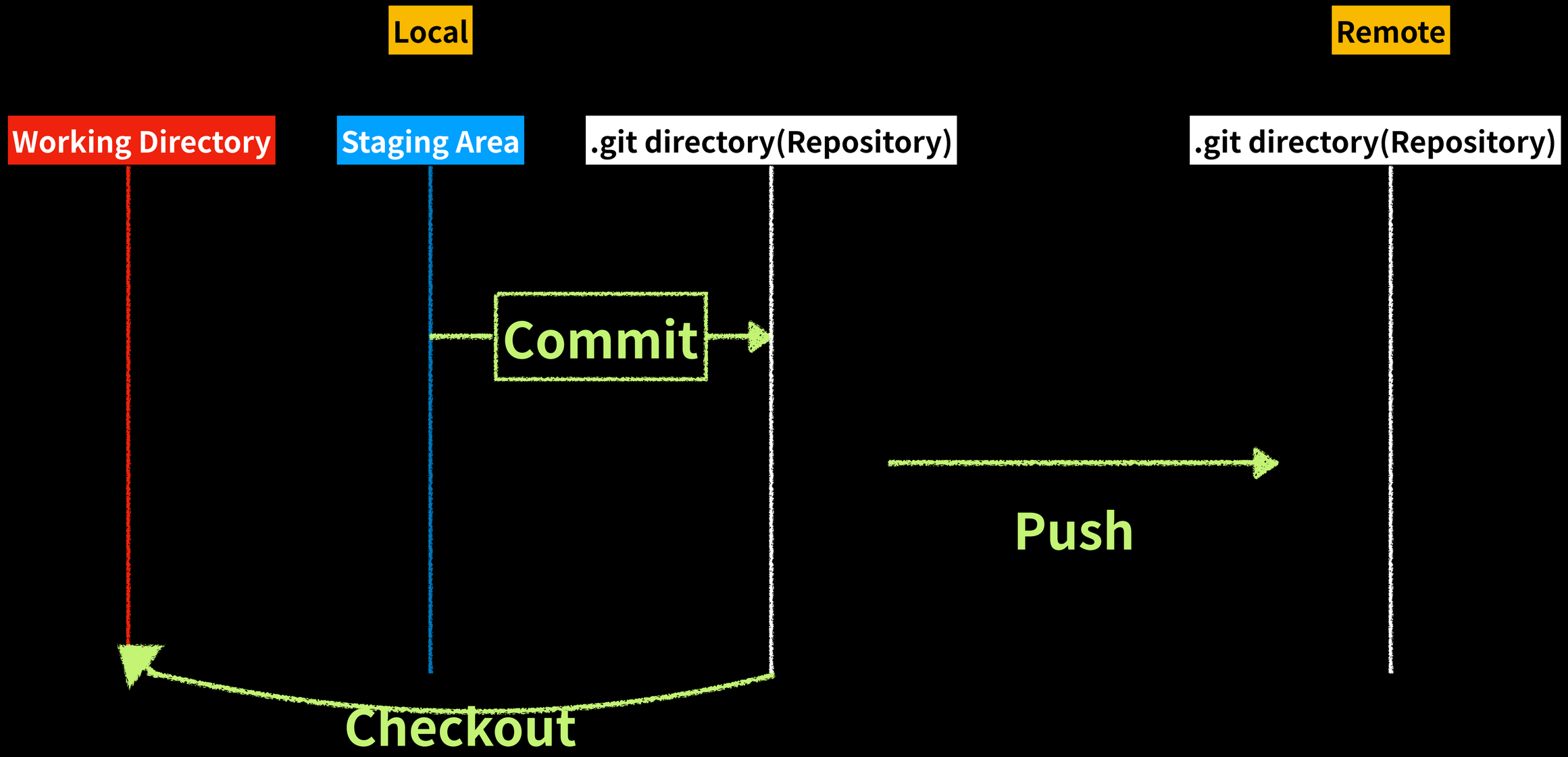
Fig 8. The lifecycle of the status of your files, Chacon, S., & Straub, B. (2014).

SupppotHandsOn

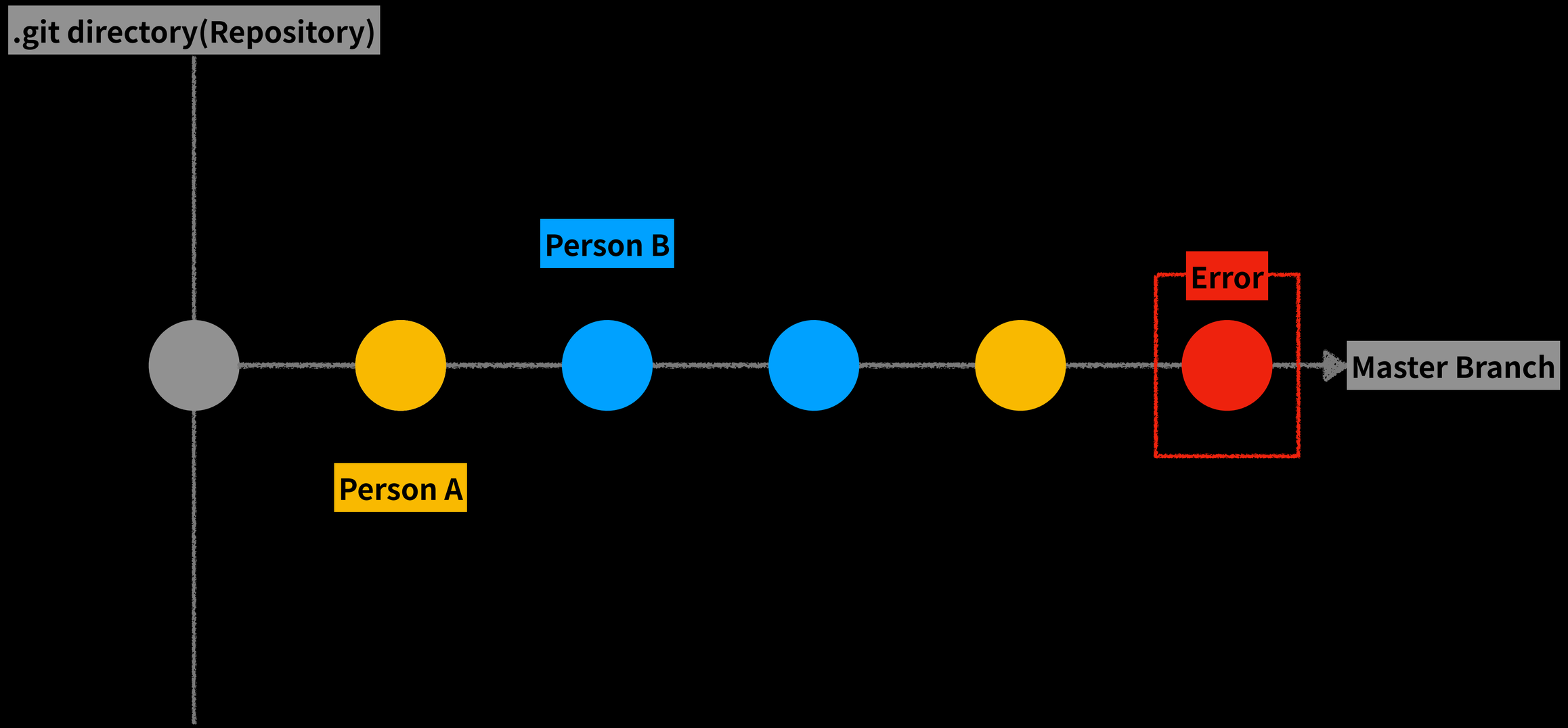
Sections


Local
Working Directory
Staging Area
.git directory(Repository)
Commit
75d8e5a
Add Login screen
yhjune committed on 5 Dec 2021
Checkout

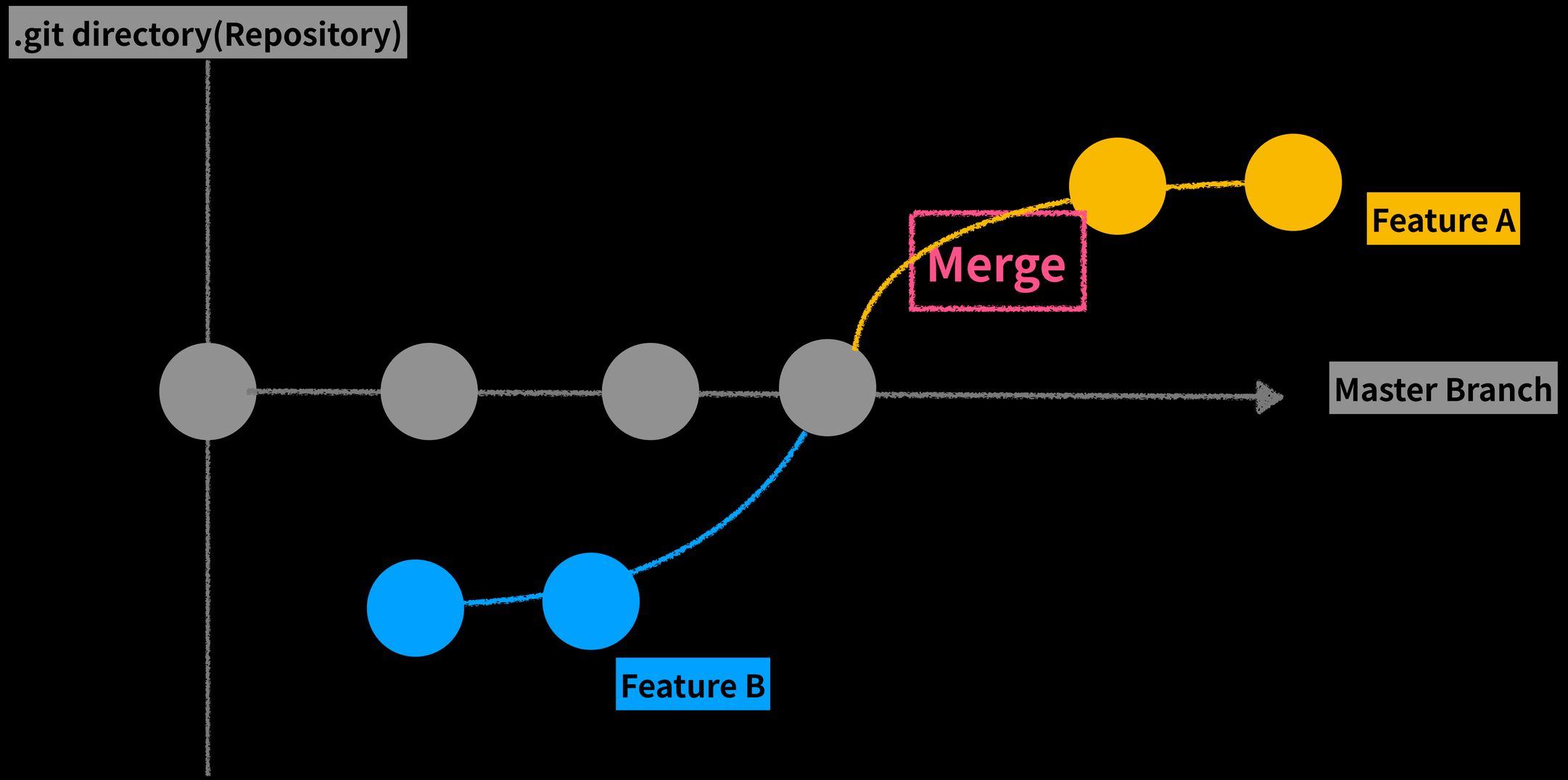
Remote


Local
Remote
Working Directory
Staging Area
.git directory(Repository)
.git directory(Repository)
Commit
Push
Checkout

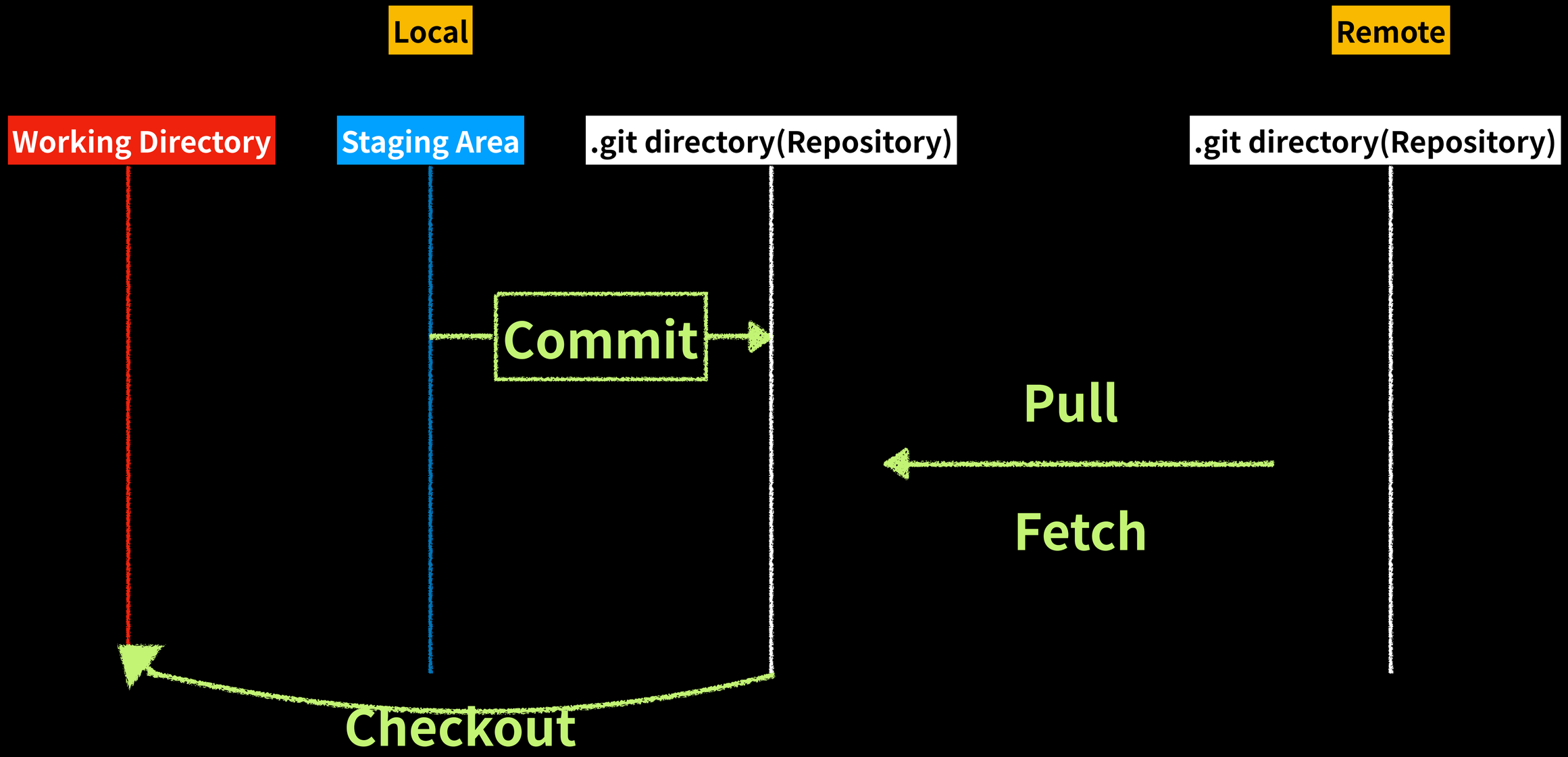
Remote: Work with other



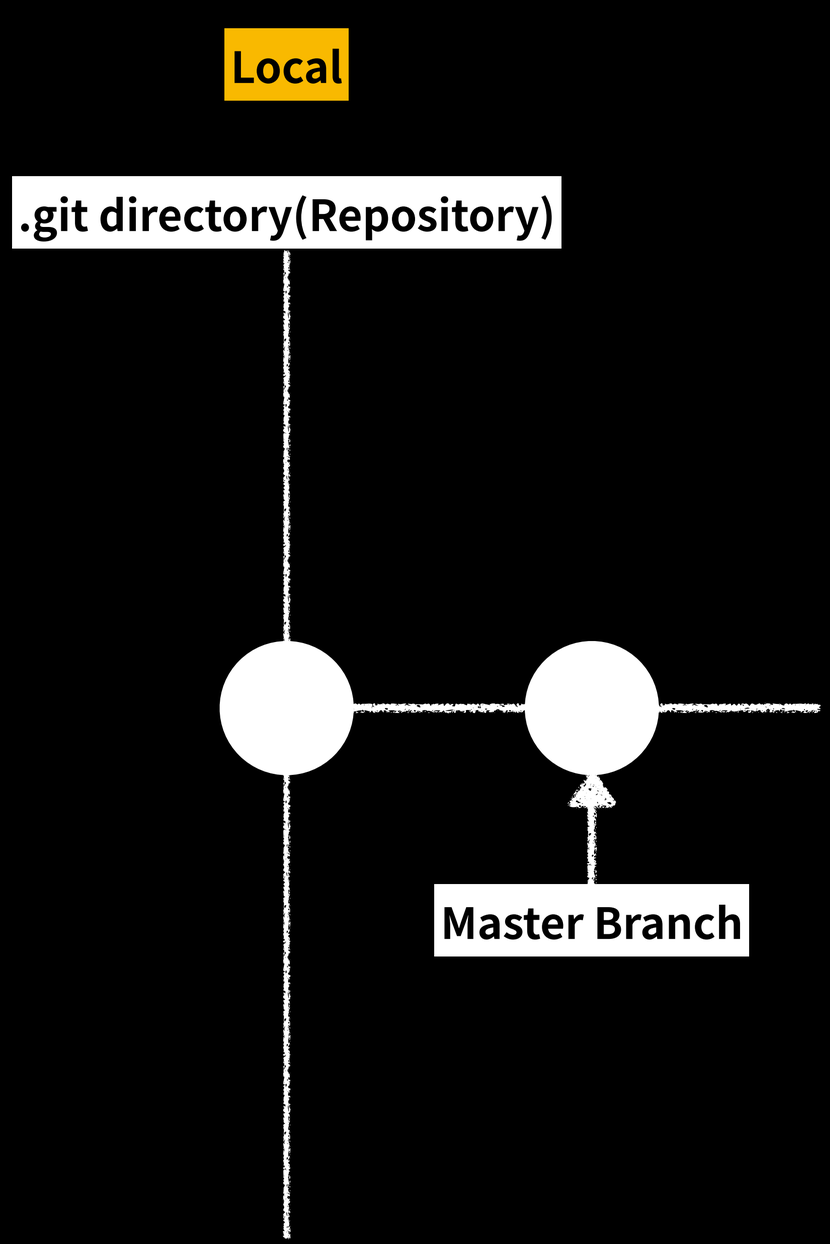
Remote: Work with other

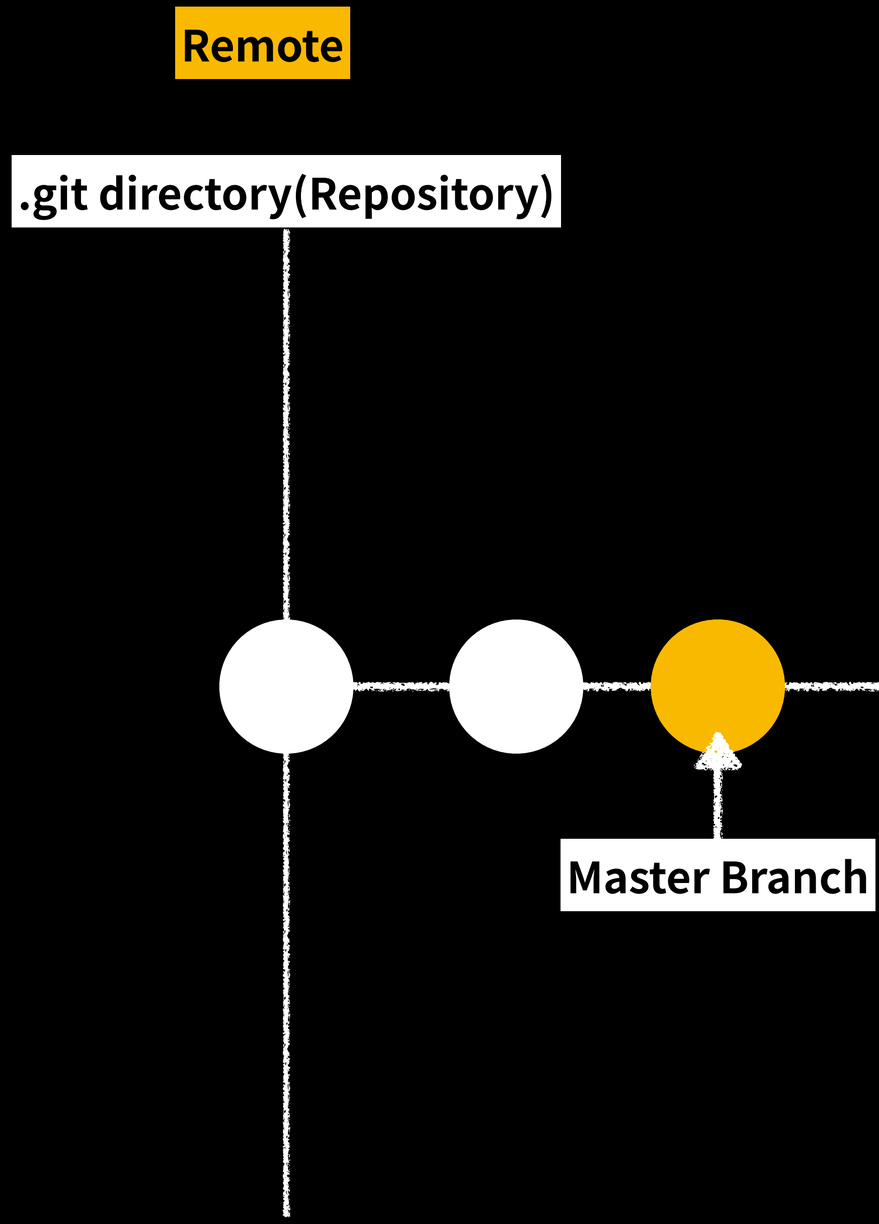


Remote

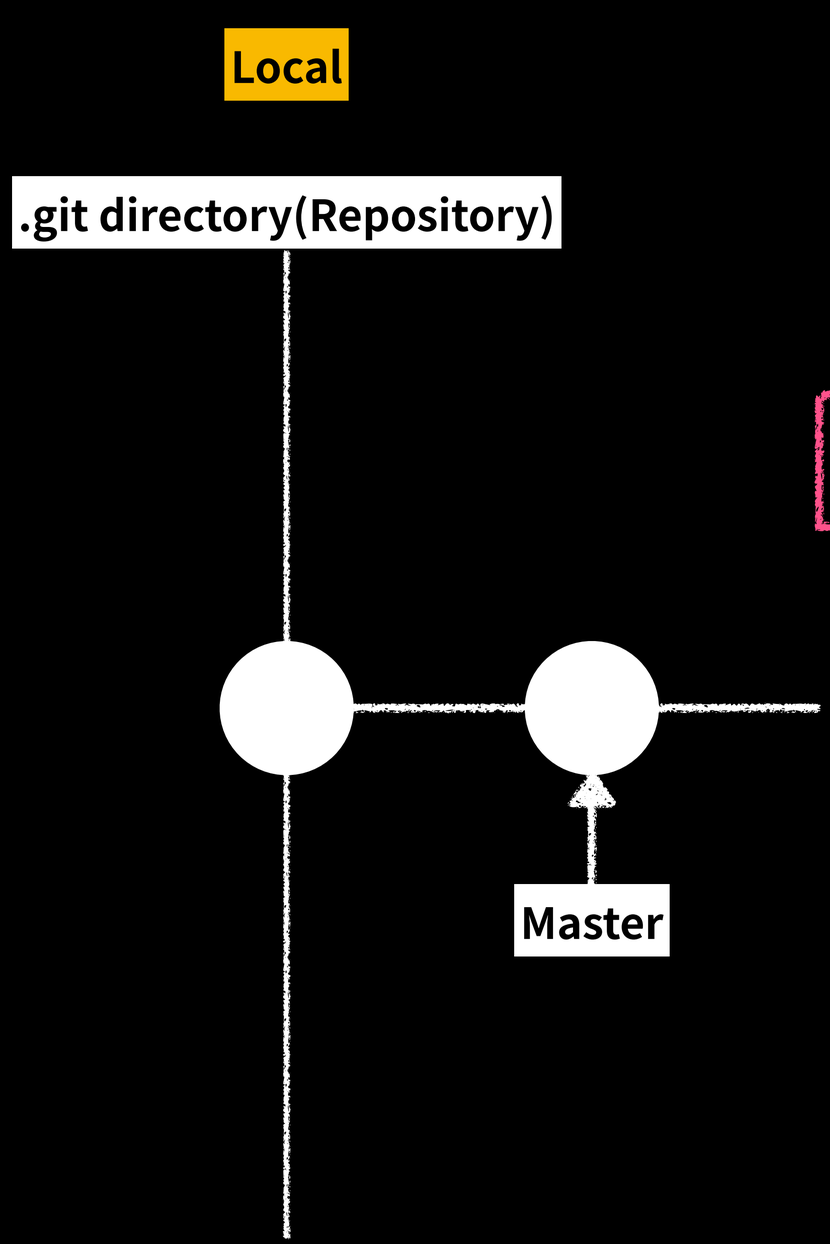

Local
Remote
Working Directory
Staging Area
.git directory(Repository)
.git directory(Repository)
Commit
Pull
Fetch
Checkout

Fetch vs Pull

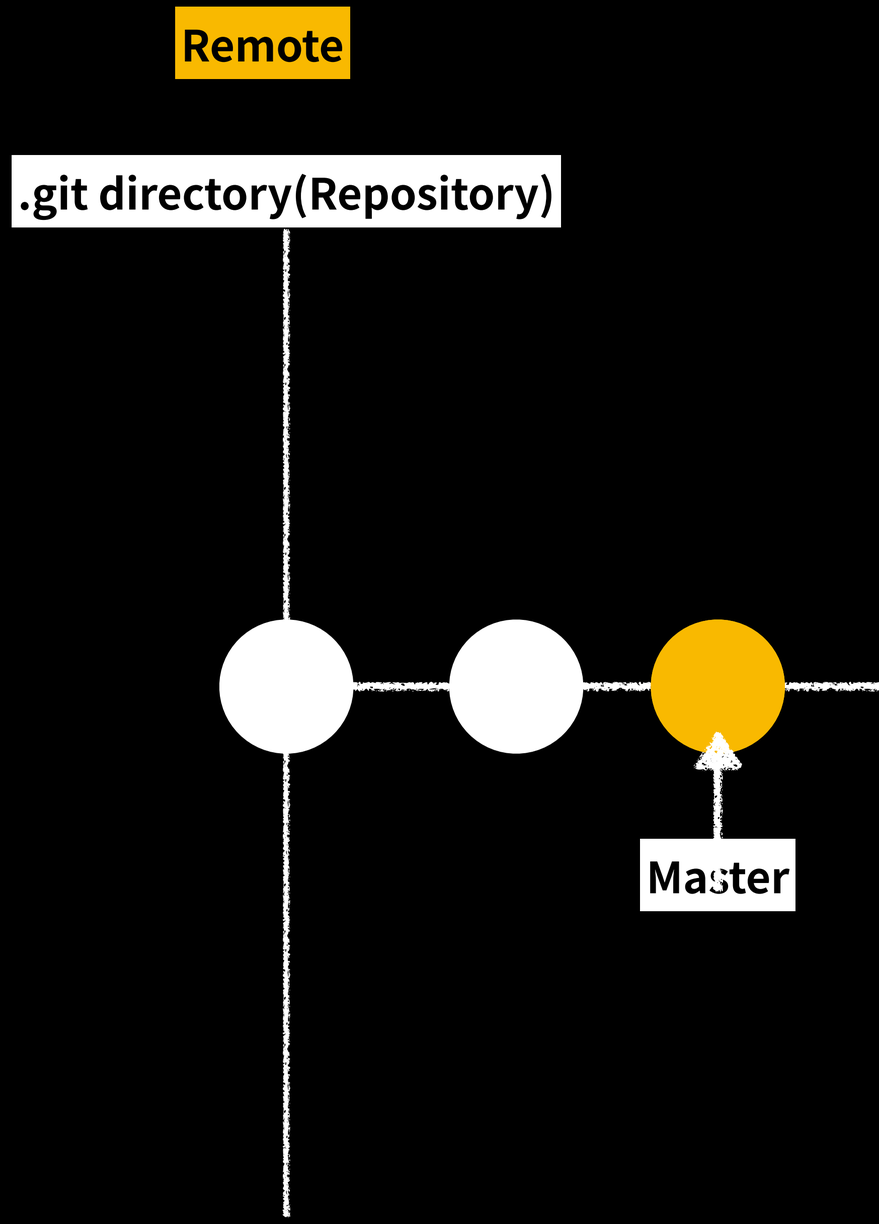




Fetch vs **Pull**



Merge

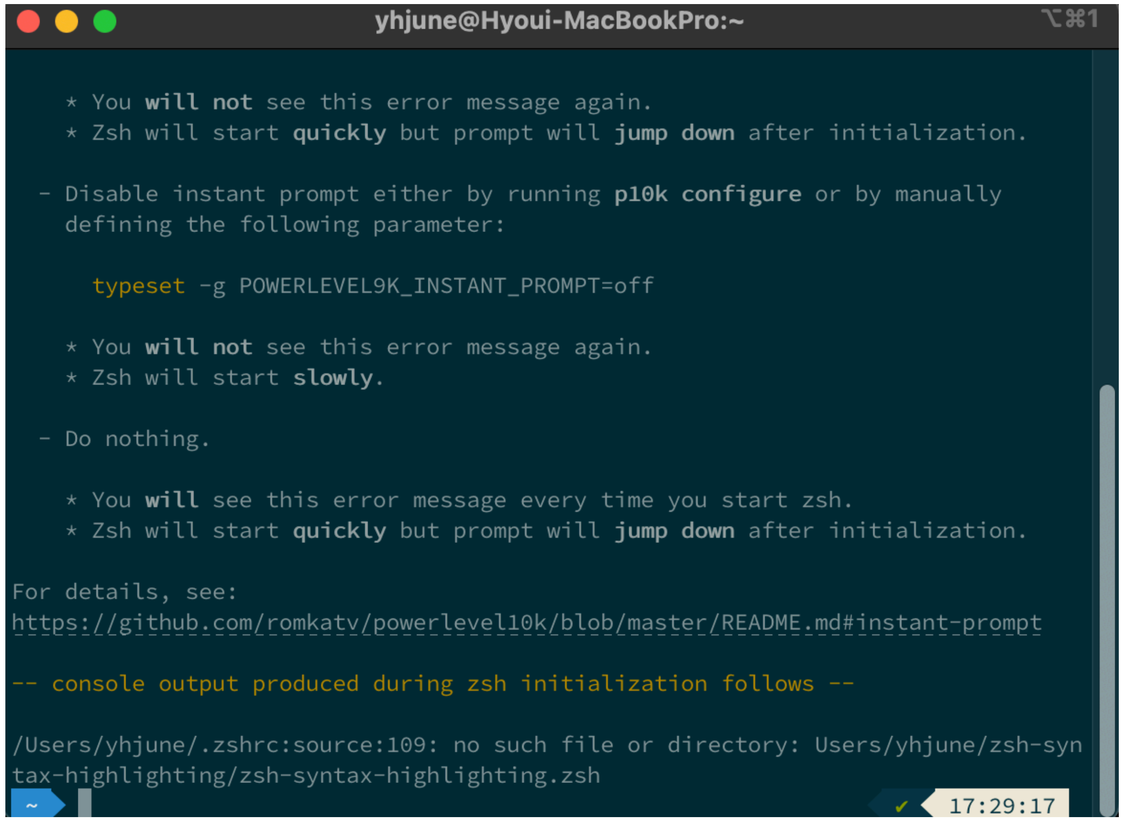


Git Hands-on

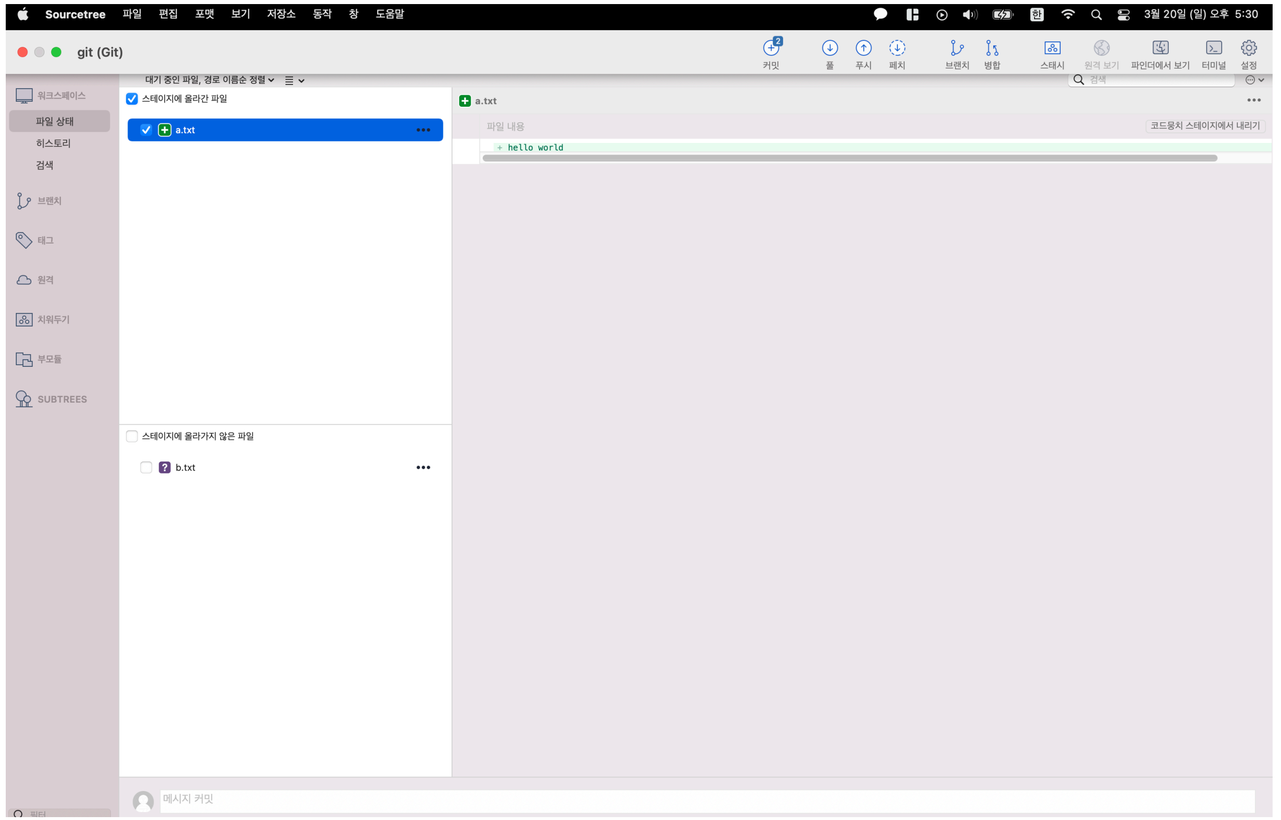
GUI, Setups, and some commands

Git을 사용하기 위한 도구

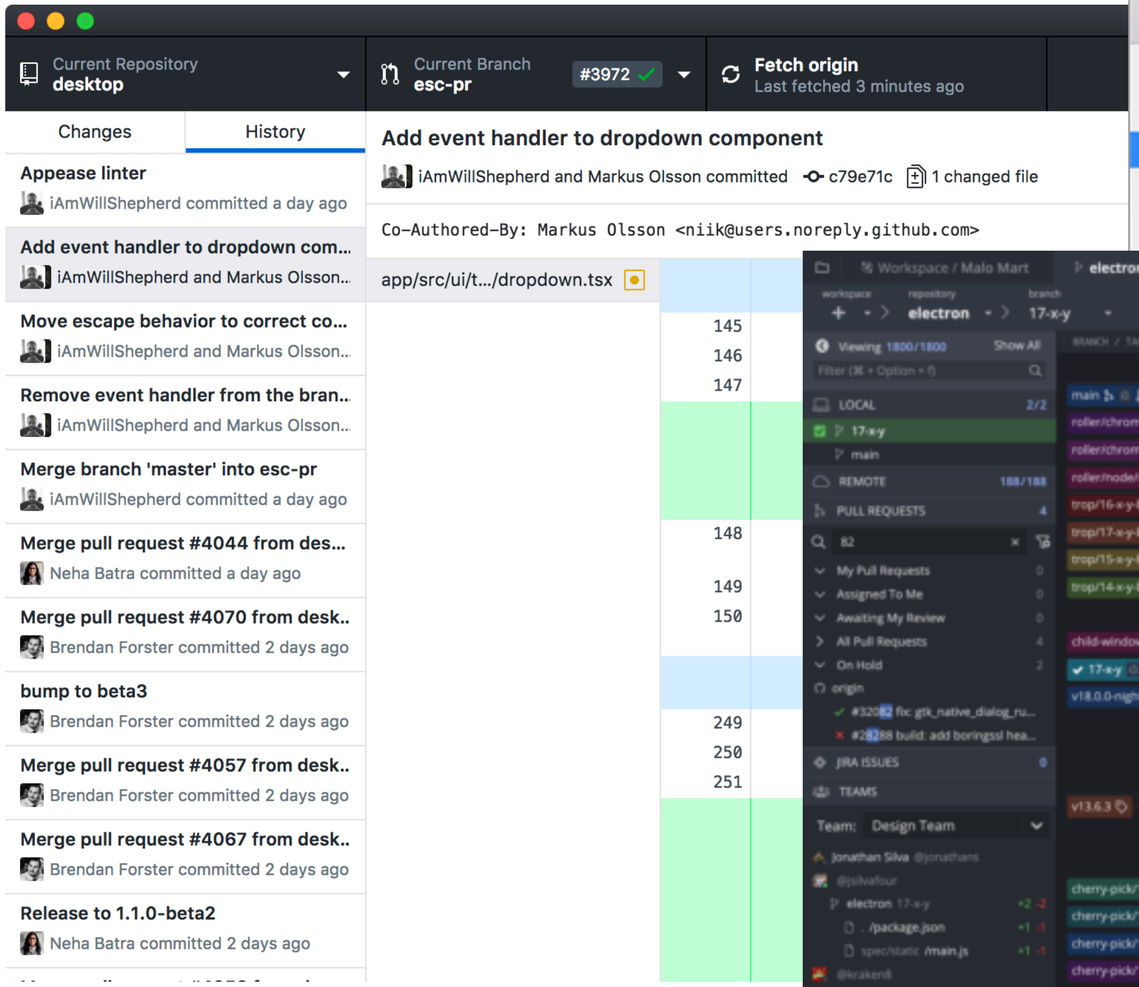
Command-Line Interface

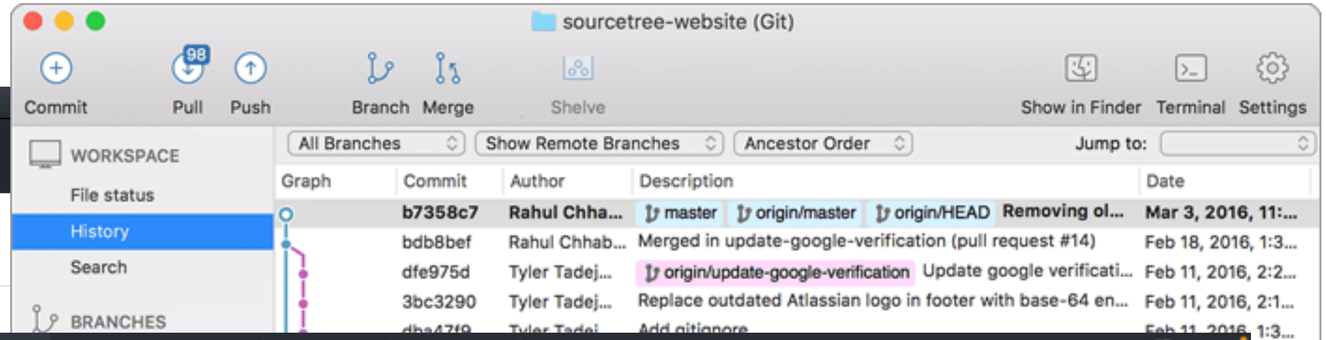


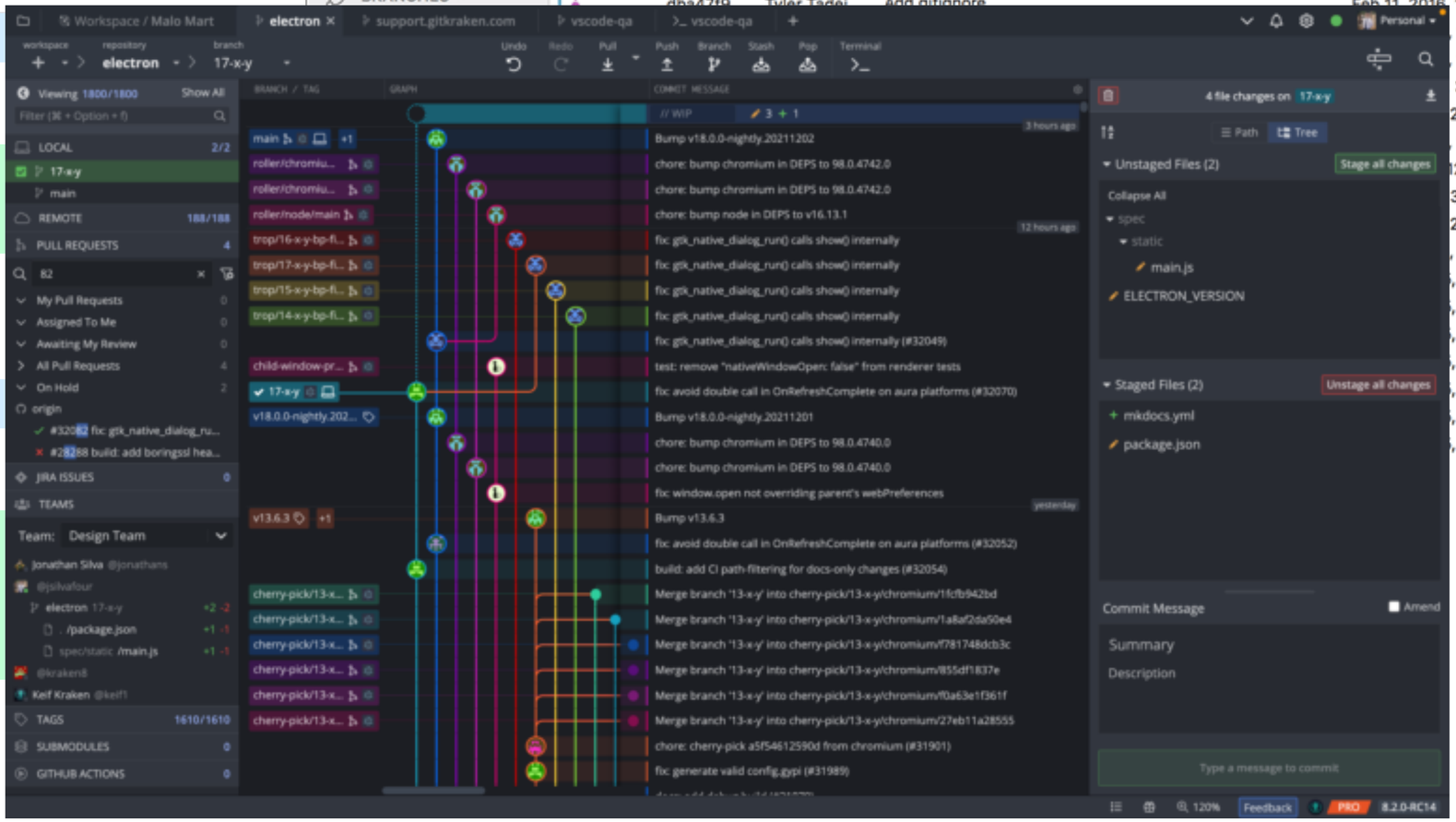
Graphical User Interface



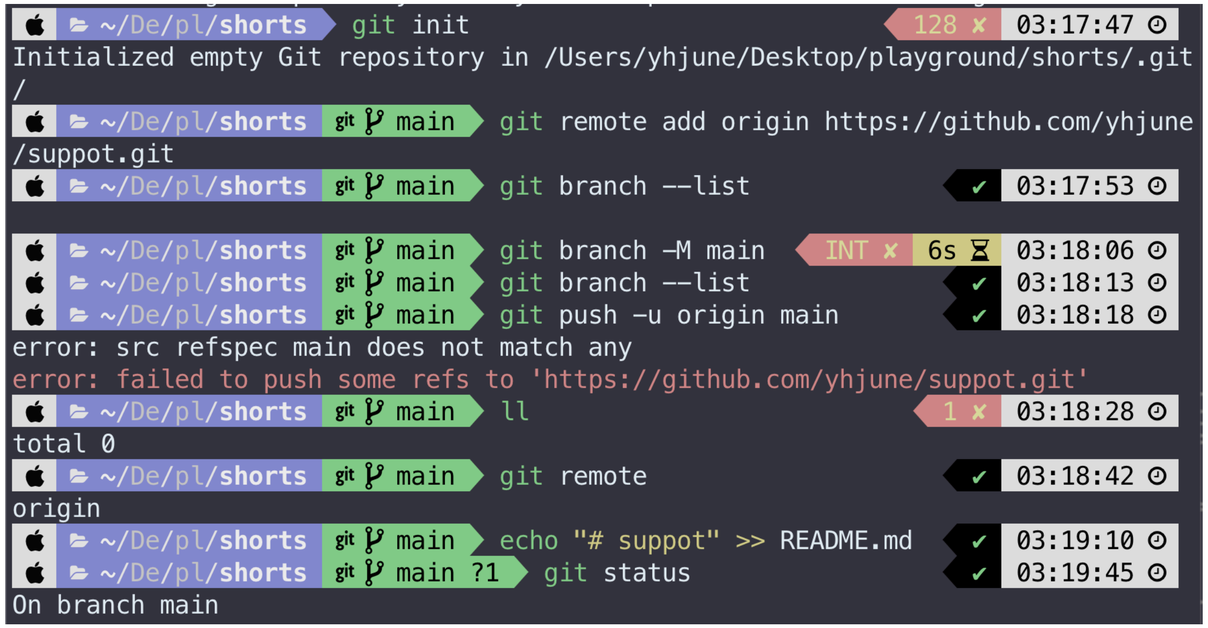
GUI







터미널로 깃을 공부한다면



1. 내가 모르는 명령어가 리눅스인지 깃인지 파악하기

2. 윈도우와 IOS의 명령어가 다른 것을 인지하기

Git Config & Github benefits

Check Config

```
git config --list --show-origin
```

Identity (Just for first setup - ONLY ONCE)

```
git config --global user.name “yourname”

git config --global user.email “email@mail.com”

git config --global init.defaultBranch main
```

Help

```
git help <verb>

Git <verb> --help
```

Usecase

1. 레포지토리 생성 : 2가지 방법
2. Commit
3. Pull
4. Push
5. Staging에서 파일 내리기 : commit message와 맞지 않는 파일
6. Merge
7. Organization
8. Issue, Convention, Wiki

Get a Repository - 2 ways

Initializing

```
git init
```

Remote

```
git remote -v
git remote add <shortname> <url>
git remote remove <shortname>
```

Cloning

```
git clone <url>
```

```
git clone https://{git_token}@github.com/{user_name}/{repository}
```

Recording Changes

Status

```
git status
```

Tracking & Staging

```
git add <filename>
git add .
```

Unstaging a Staged File

```
git reset HEAD <filename> |
git restore --staged <file>
```

Commit

```
git commit -m “your commit message”
git commit -a –m “your commit message”
```

Remove files from git

```
git rm <filename> / git rm --cached <filename>
```

Recording Changes

Branch

```
git branch <branchname> git checkout <branchname>
git branch --set-upstream-to=origin/<branch> <localbranch>
```

Fetch and Pull

```
git fetch <remote>
git pull
```

Push

```
git push <remote> <branch>
```

Revert

```
git log
git revert <commit-has>
git commit
```

Collaborate

Issue, Convention, wiki, and Open-Sources

Reference

Zolkifli, N. N., Ngah, A., & Deraman, A. (2018). Version control system: A review.

Procedia Computer Science, 135, 408-415.

Chacon, S., & Straub, B. (2014). Pro git (p. 456). Springer Nature.

https://git-scm.com/doc

https://learngitbranching.js.org/?locale=ko

https://docs.github.com/en/issues/planning-and-tracking-with-projects/learning-about-projects/best-practices-for-projects

https://docs.github.com/en/issues/tracking-your-work-with-issues/about-issues