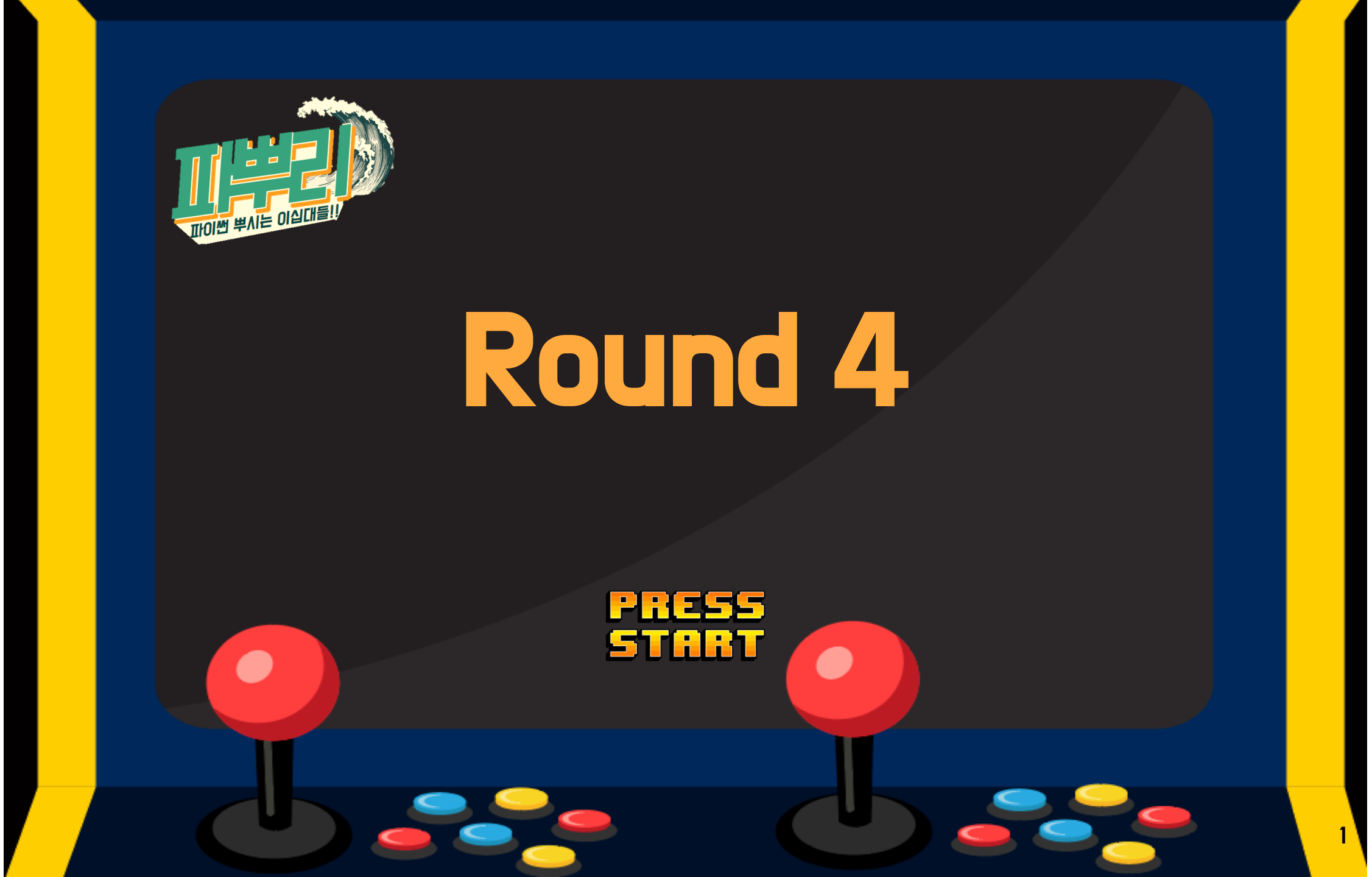

파뿌리
파이썬 뿌시는 이십대들!!
Round 4
PRESS
START

파뿌리
파이썬 뿌시는 이십대들!!

《 Round 4 》

- Git
- Github
- Markdown language

New Assignment

파뿌리
파이썬 뿌시는 이십대들!!

《 Round 4 》

- Git 《
- Github
- Markdown language

Git?

- 분산형 버전 관리 시스템
- 소스 코드 관리에 주로 사용

파뿌리
파이썬 뿌시는 이십대들!!

Why use it?

- 여러 사람이 동일한 코드에 대해 동시에 작업을 할 때
- 상대방의 작업을 방해하지 않으면서
- 변경 이력을 남기면서
- 효율적으로

프로그래밍이 가능해진다.

파뿌리
파이썬 뿌시는 이십대들!!

Why use it?

Project

쇼핑몰 웹 사이트 제작

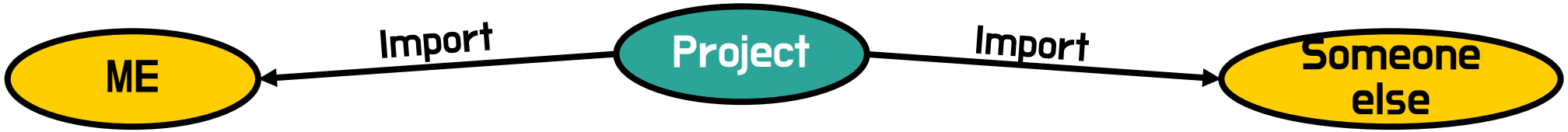
...

각자 할 일
나 + 팀원1 : 로그인, 메인페이지 로직
팀원2 + 팀원3 : 로그인 관련 DB
팀원1 + 팀원3 : 로그인, 메인페이지 프론트엔드
...

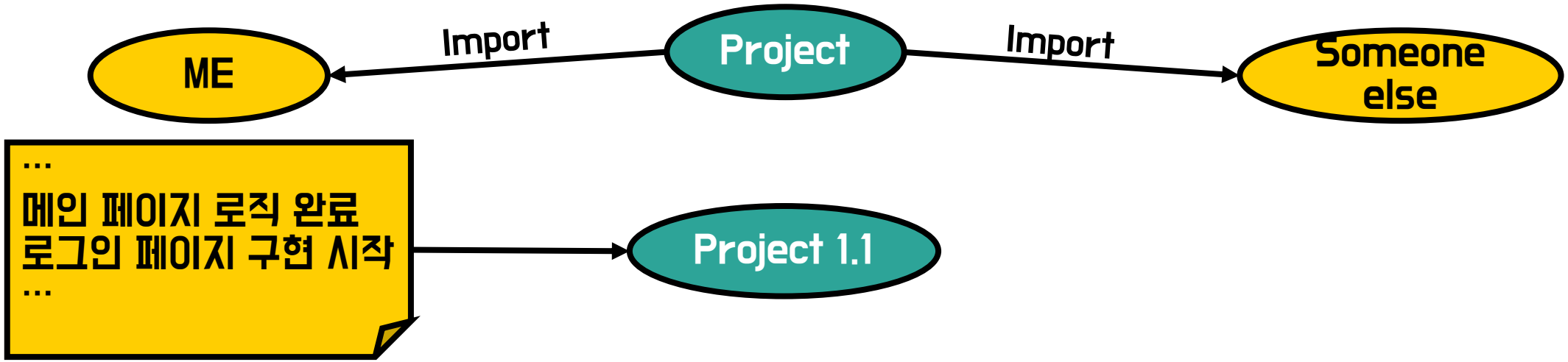
중간 검토 기간 : 이틀 뒤…

파뿌리
파이썬 뿌시는 이십대들!!

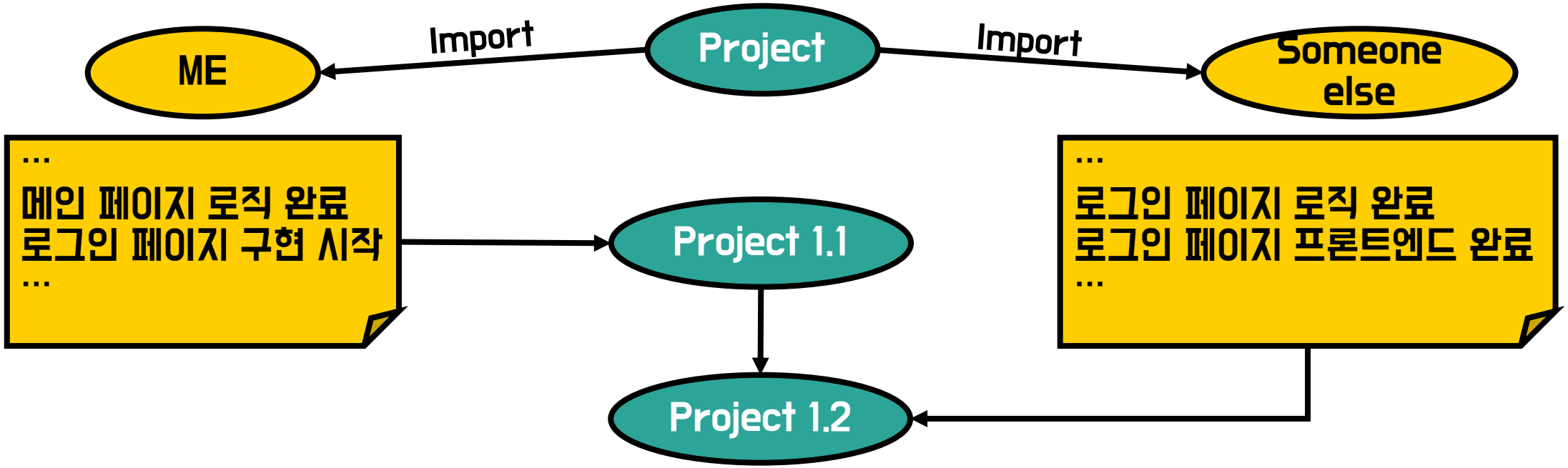
Why use it?



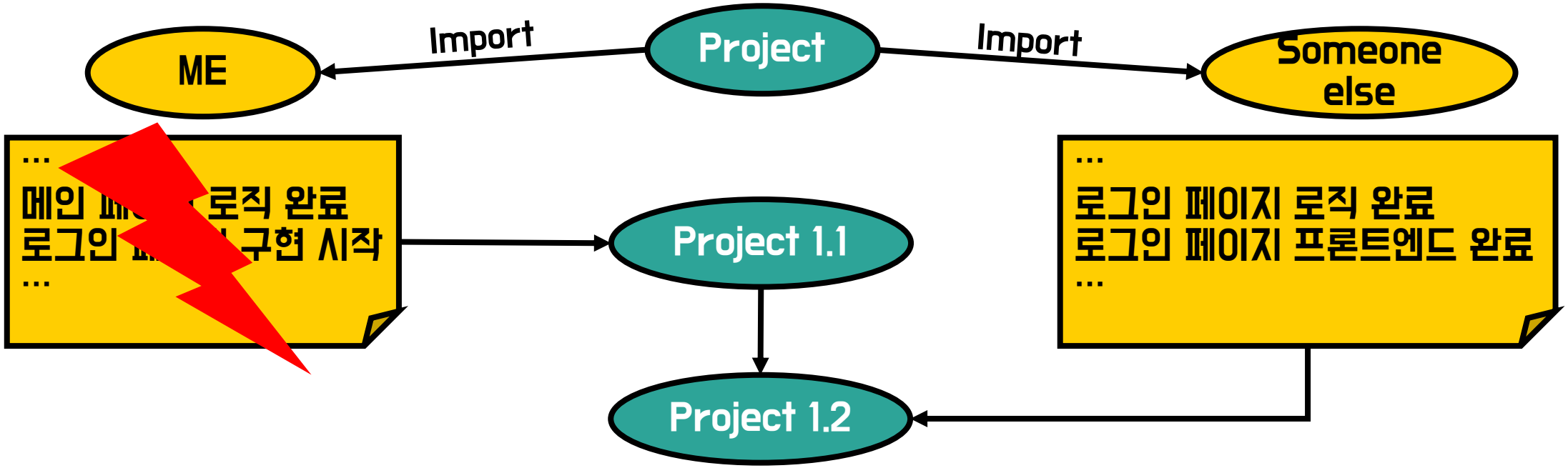
Why use it?


ME
Import
Project
Import
Someone else
...
메인 페이지 로직 완료
로그인 페이지 구현 시작
...
Project 1.1

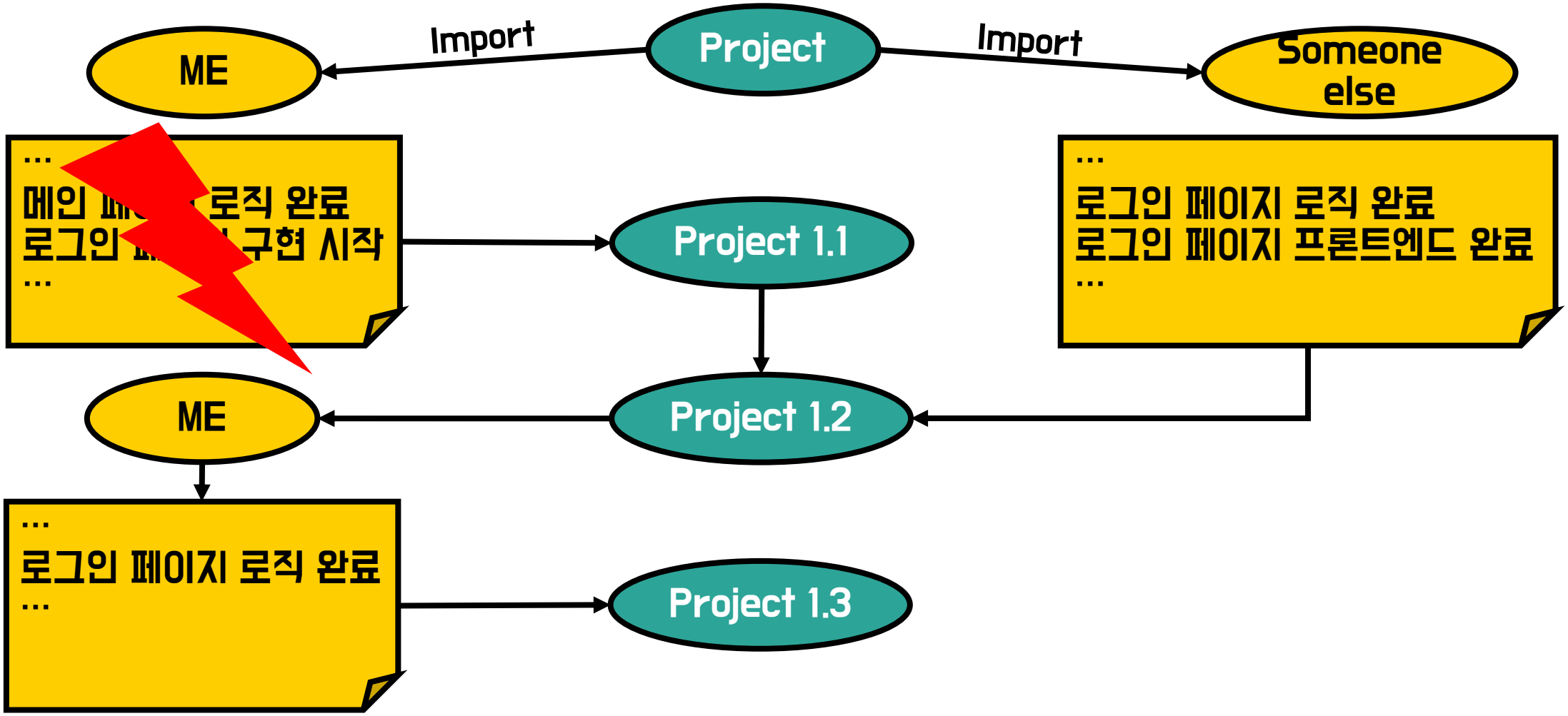
Why use it?


ME
Import
Project
Import
Someone else
...
메인 페이지 로직 완료
로그인 페이지 구현 시작
...
Project 1.1
...
로그인 페이지 로직 완료
로그인 페이지 프론트엔드 완료
...
Project 1.2

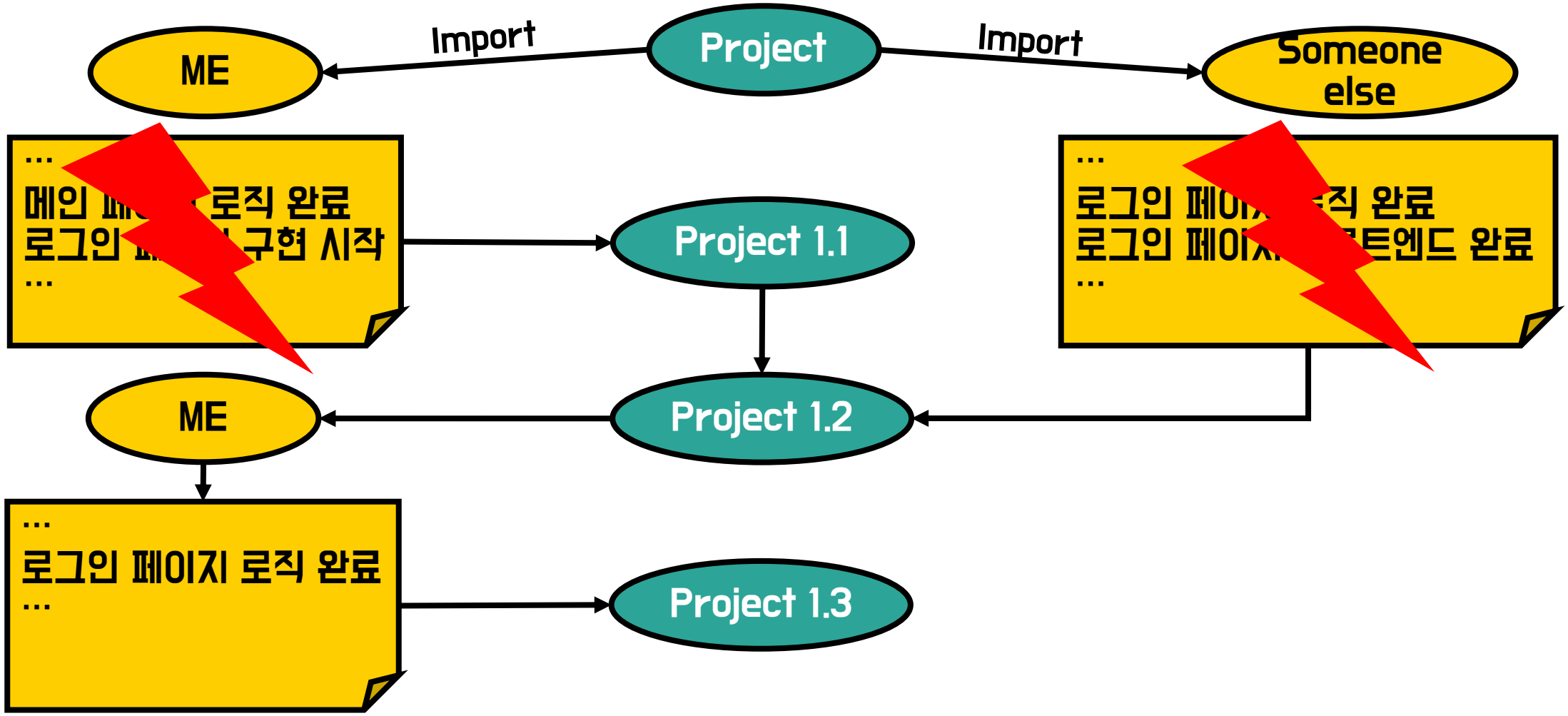
Why use it?


ME
Import
Project
Import
Someone else
...
메인 페 로직 완료
로그인 구현 시작
...
Project 1.1
...
로그인 페이지 로직 완료
로그인 페이지 프론트엔드 완료
...
Project 1.2

Why use it?

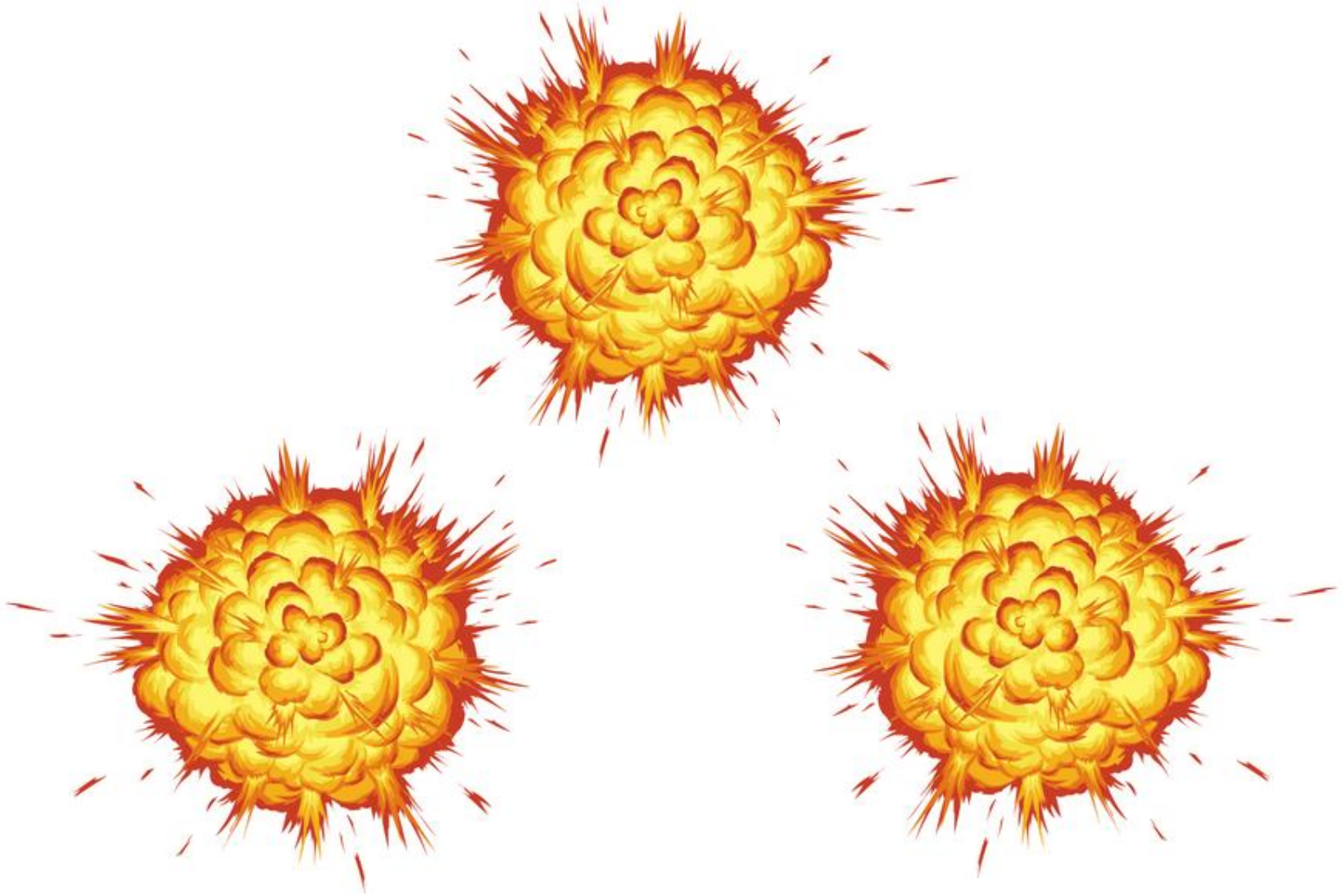

Import
Project
Import
ME
Someone else
...
메인 페
로직 완료
로그인
구현 시작
...
Project 1.1
...
로그인 페이지 로직 완료
로그인 페이지 프론트엔드 완료
...
ME
Project 1.2
...
로그인 페이지 로직 완료
...
Project 1.3

Why use it?

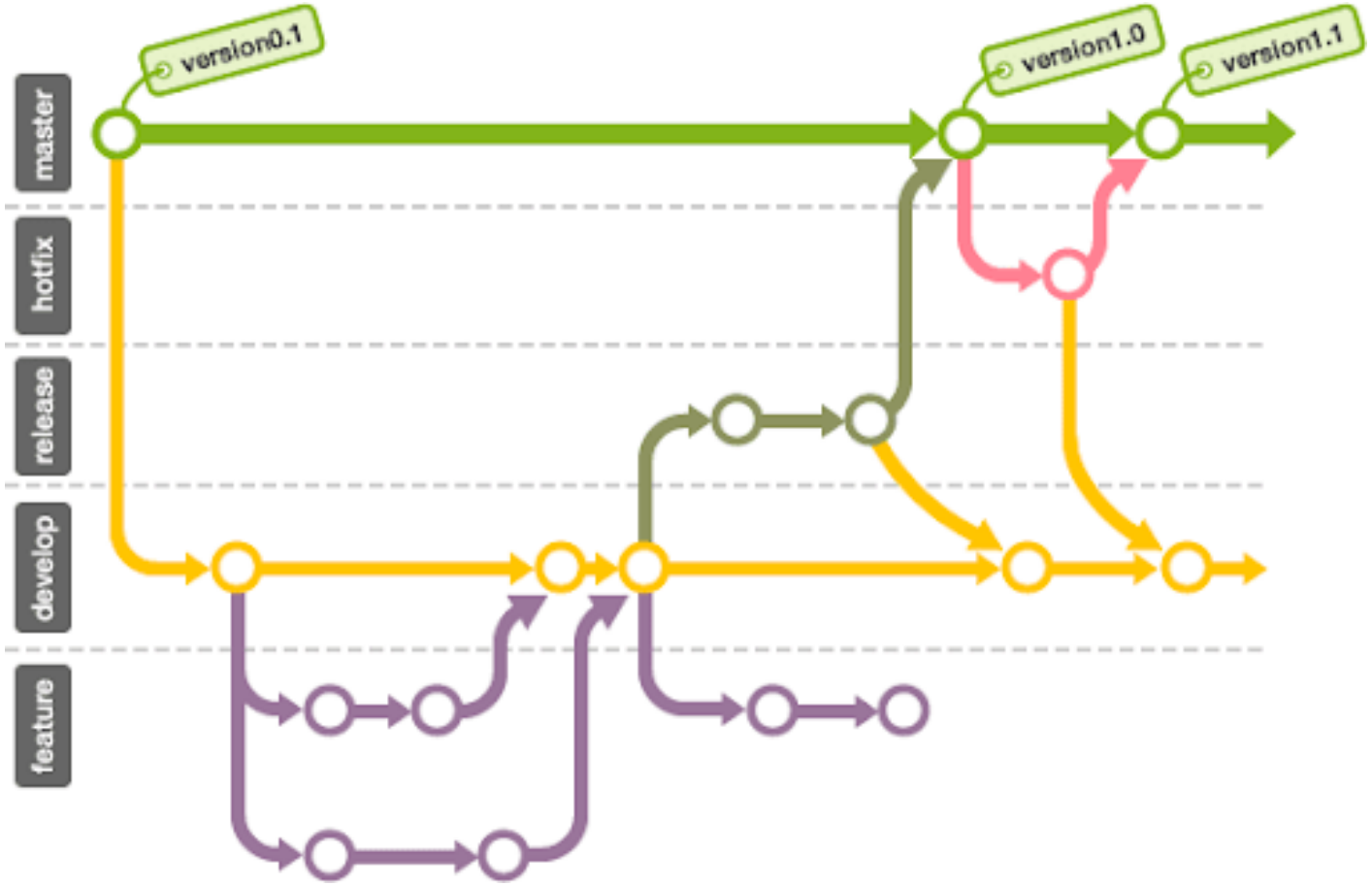

Import
Project
Import
ME
Someone else
...
로그인 구현 시작
...
...
로그인 페이 직 완료
로그인 페이 트엔드 완료
...
Project 1.1
ME
Project 1.2
...
로그인 페이지 로직 완료
...
Project 1.3

파뿌리
파이썬 뿌시는 이십대들!!

Why use it?

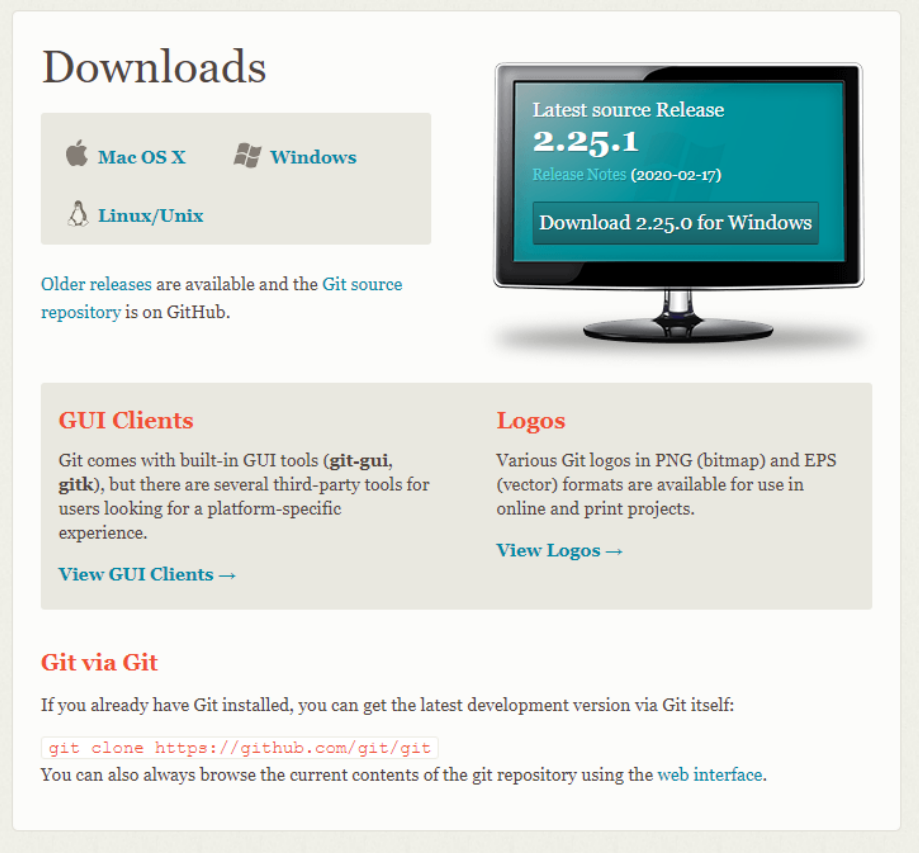


Why use it?

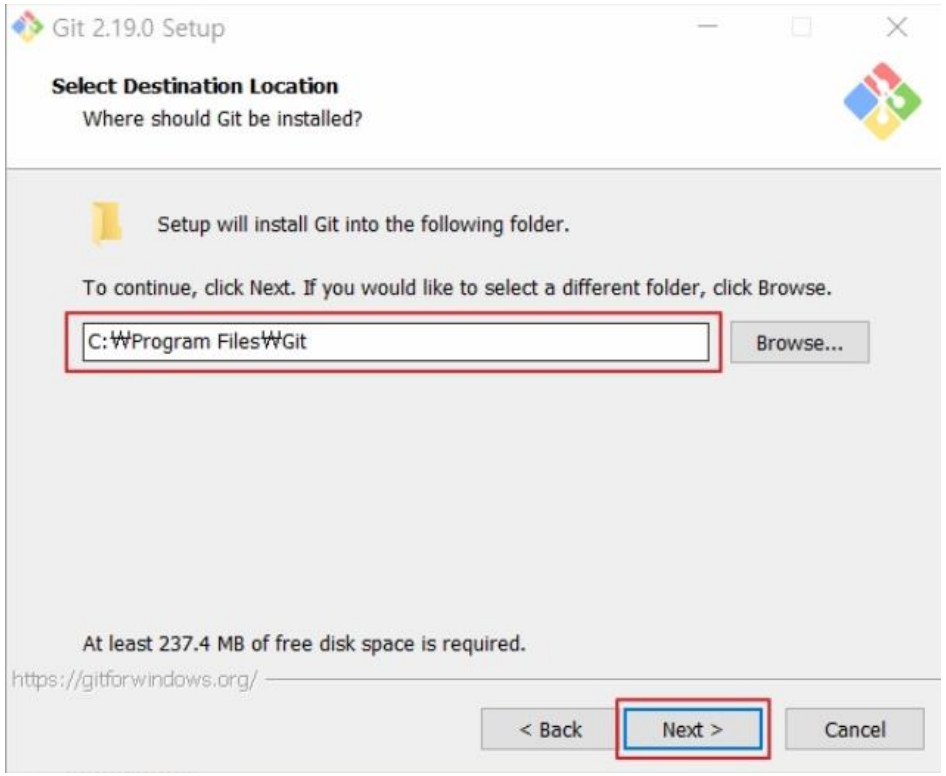

version0.1
version1.0
version1.1
master
hotfix
release
develop
feature

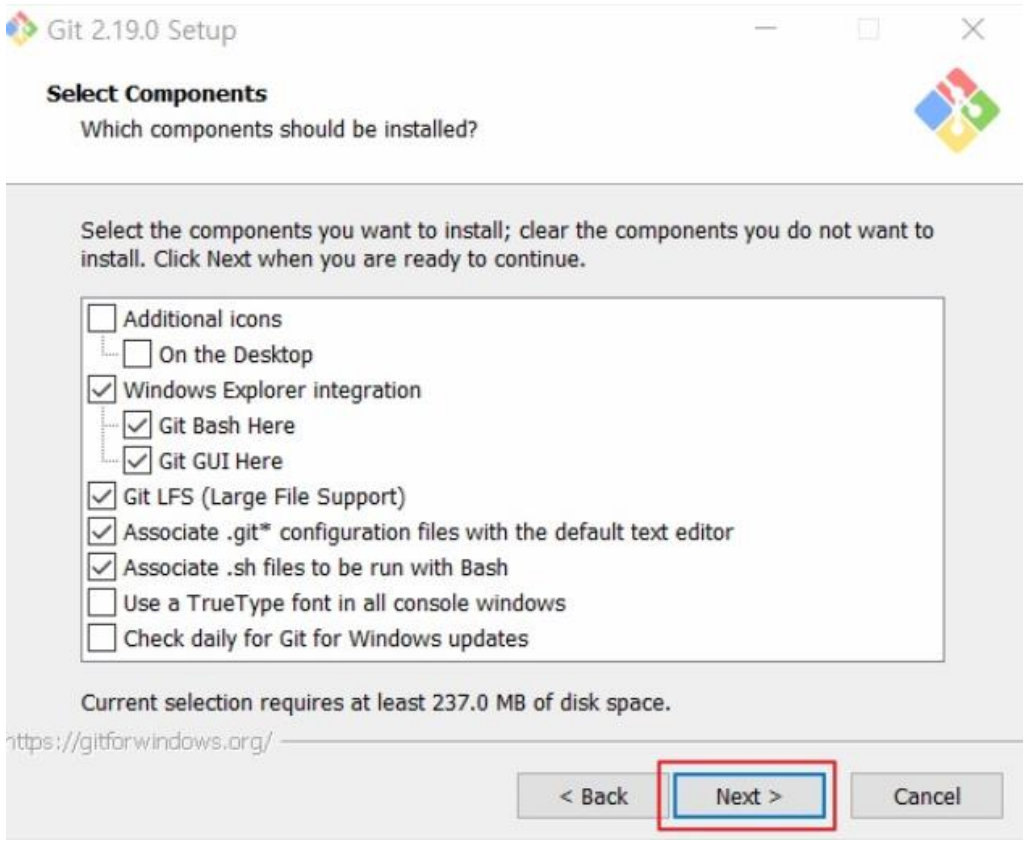
Install git

https://git-scm.com/downloads

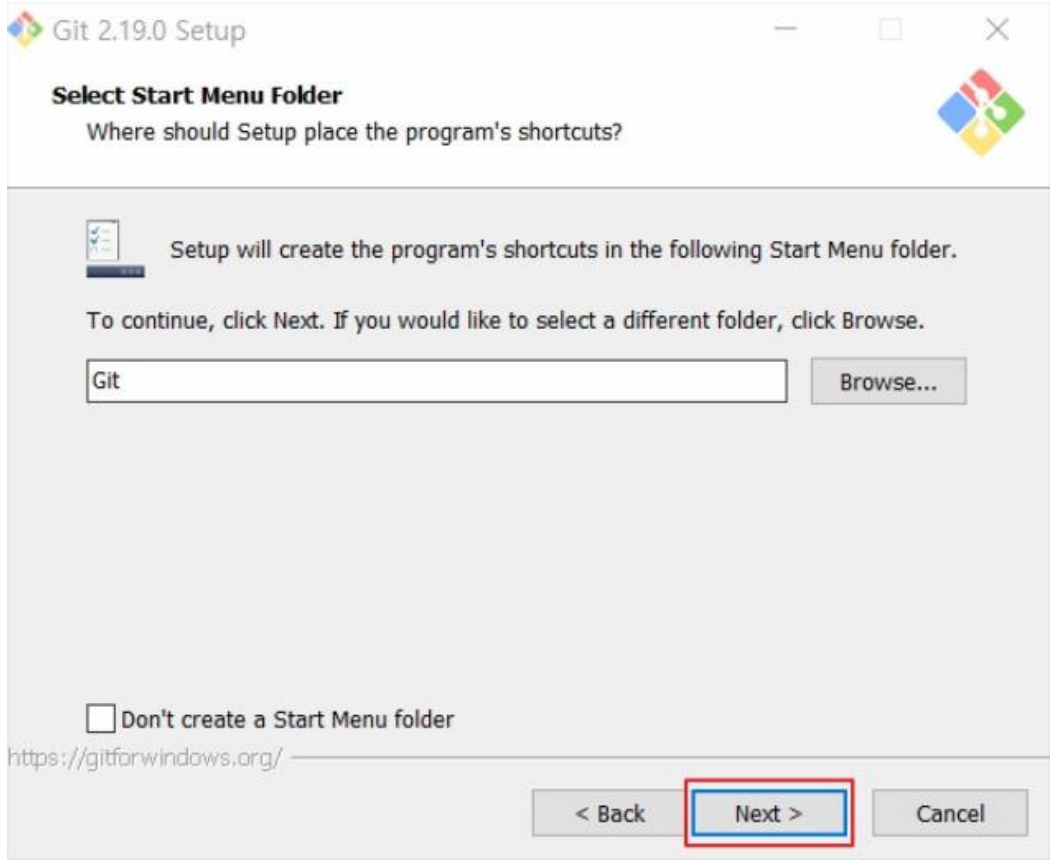


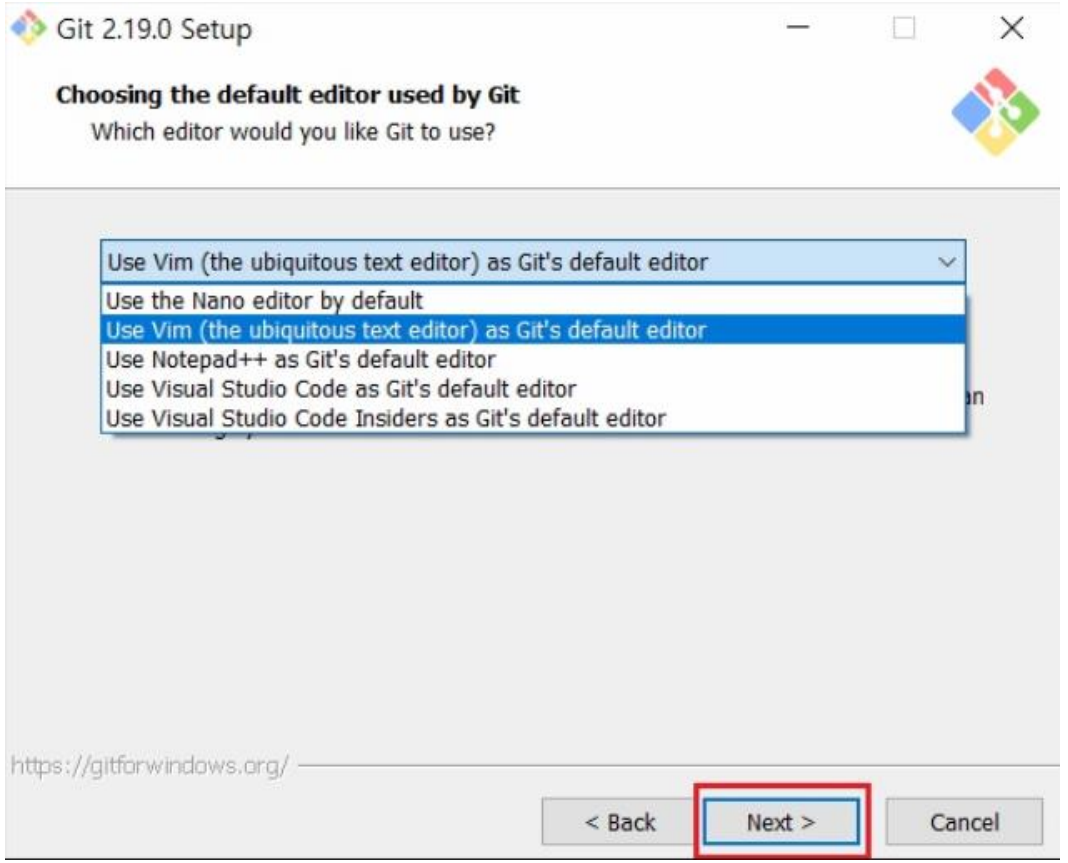
Install git


Git 2.19.0 Setup
Select Destination Location
Where should Git be installed?
Setup will install Git into the following folder.
To continue, click Next. If you would like to select a different folder, click Browse.
C:₩Program Files₩Git
Browse...
At least 237.4 MB of free disk space is required.
https://gitforwindows.org/
< Back
Next >
Cancel

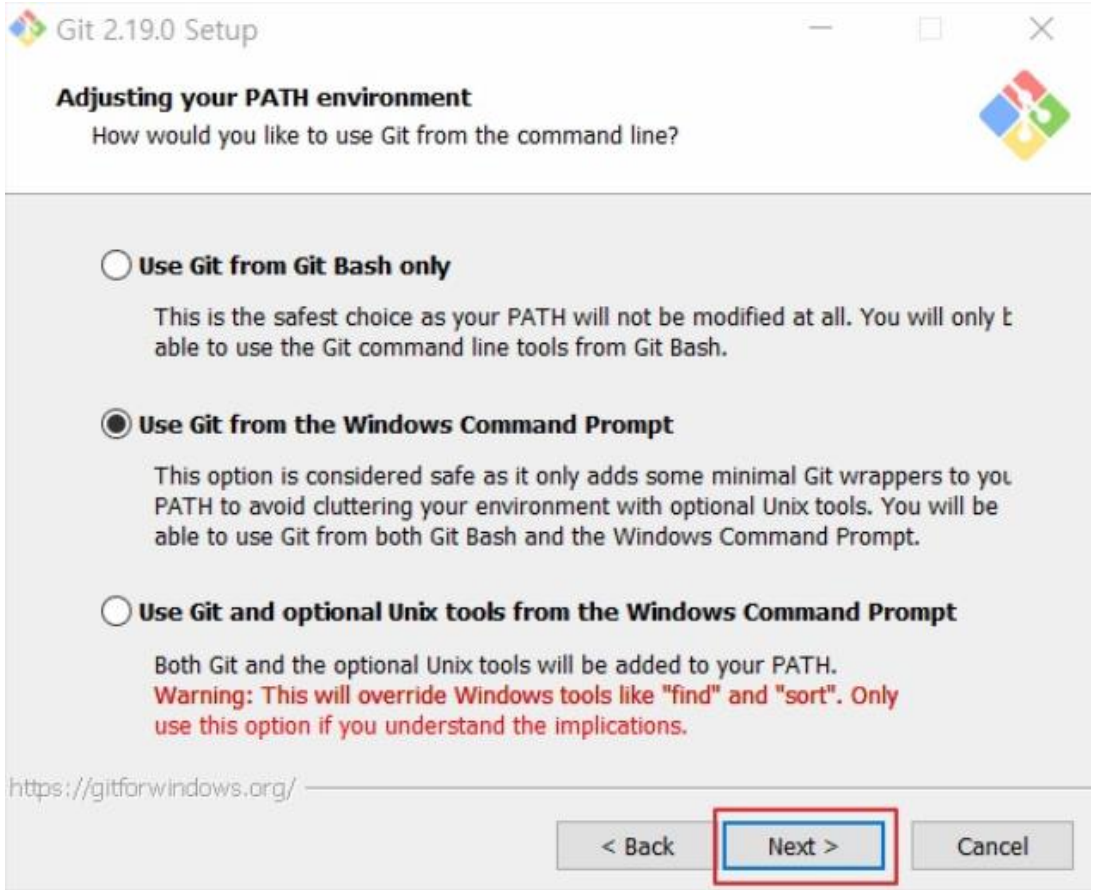

Git 2.19.0 Setup
Select Components
Which components should be installed?
Select the components you want to install; clear the components you do not want to install. Click Next when you are ready to continue.
Additional icons
On the Desktop
Windows Explorer integration
Git Bash Here
Git GUI Here
Git LFS (Large File Support)
Associate .git* configuration files with the default text editor
Associate .sh files to be run with Bash
Use a TrueType font in all console windows
Check daily for Git for Windows updates
Current selection requires at least 237.0 MB of disk space.
https://gitforwindows.org/
< Back
Next >
Cancel

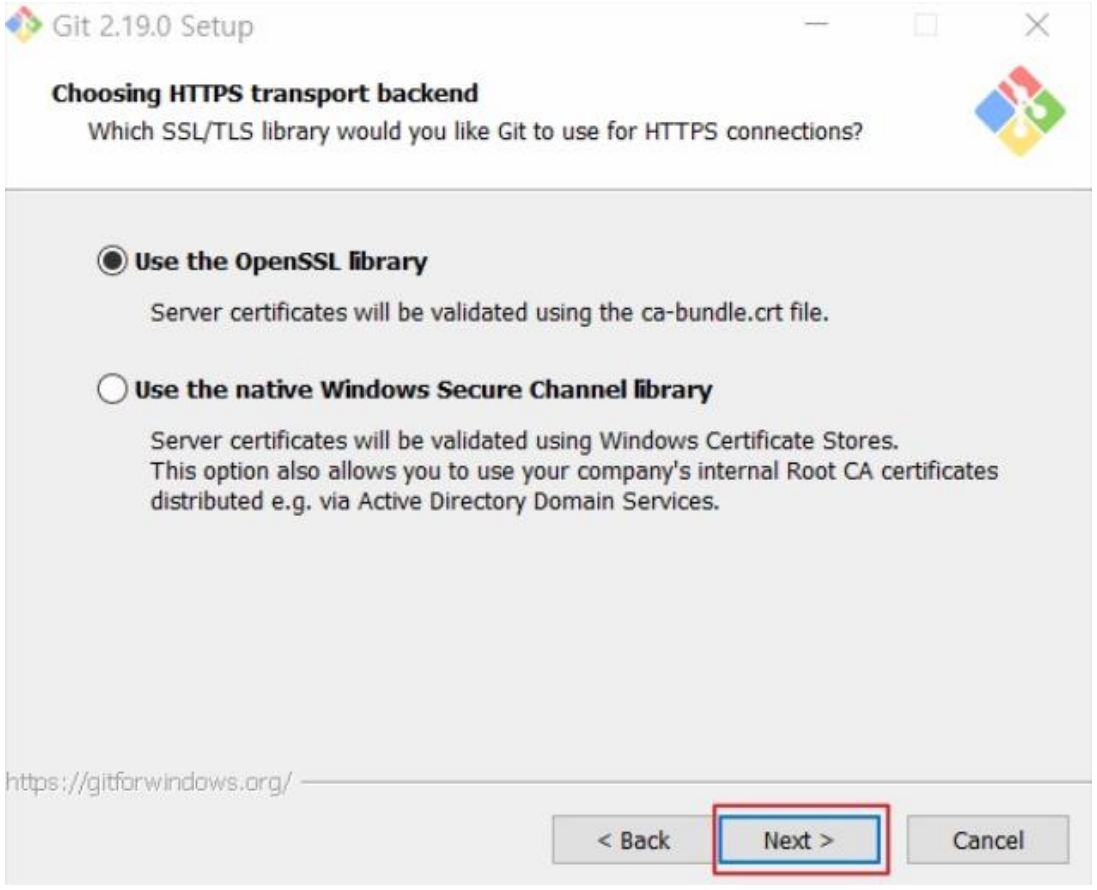
Install git


Git 2.19.0 Setup
Select Start Menu Folder
Where should Setup place the program's shortcuts?
Setup will create the program's shortcuts in the following Start Menu folder.
To continue, click Next. If you would like to select a different folder, click Browse.
Git
Browse...
Don't create a Start Menu folder
https://gitforwindows.org/
< Back
Next >
Cancel

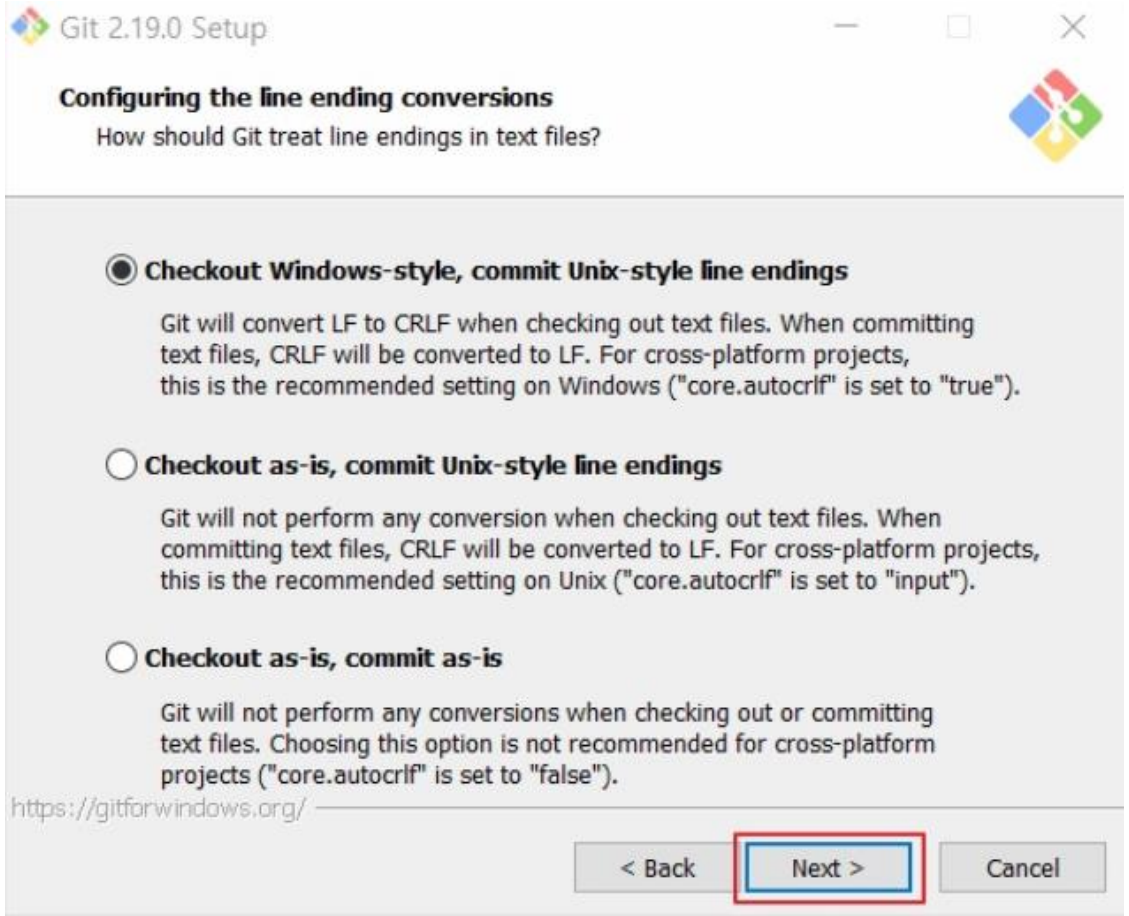

Git 2.19.0 Setup
Choosing the default editor used by Git
Which editor would you like Git to use?
Use Vim (the ubiquitous text editor) as Git's default editor
Use the Nano editor by default
Use Vim (the ubiquitous text editor) as Git's default editor
Use Notepad++ as Git's default editor
Use Visual Studio Code as Git's default editor
Use Visual Studio Code Insiders as Git's default editor
https://gitforwindows.org/
< Back
Next >
Cancel

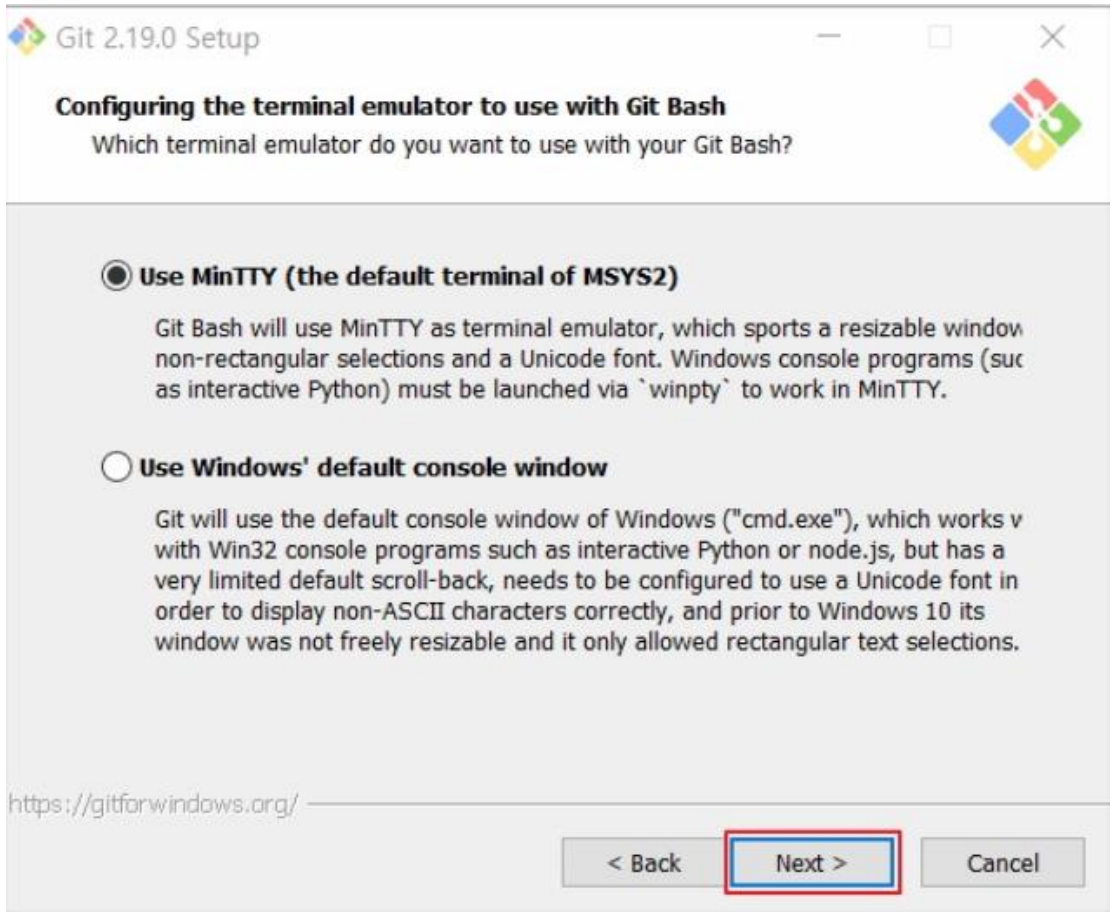
Install git


Git 2.19.0 Setup
Adjusting your PATH environment
How would you like to use Git from the command line?
Use Git from Git Bash only
This is the safest choice as your PATH will not be modified at all. You will only b
able to use the Git command line tools from Git Bash.
Use Git from the Windows Command Prompt
This option is considered safe as it only adds some minimal Git wrappers to you
PATH to avoid cluttering your environment with optional Unix tools. You will be
able to use Git from both Git Bash and the Windows Command Prompt.
Use Git and optional Unix tools from the Windows Command Prompt
Both Git and the optional Unix tools will be added to your PATH.
Warning: This will override Windows tools like "find" and "sort". Only
use this option if you understand the implications.
https://gitforwindows.org/
< Back
Next >
Cancel

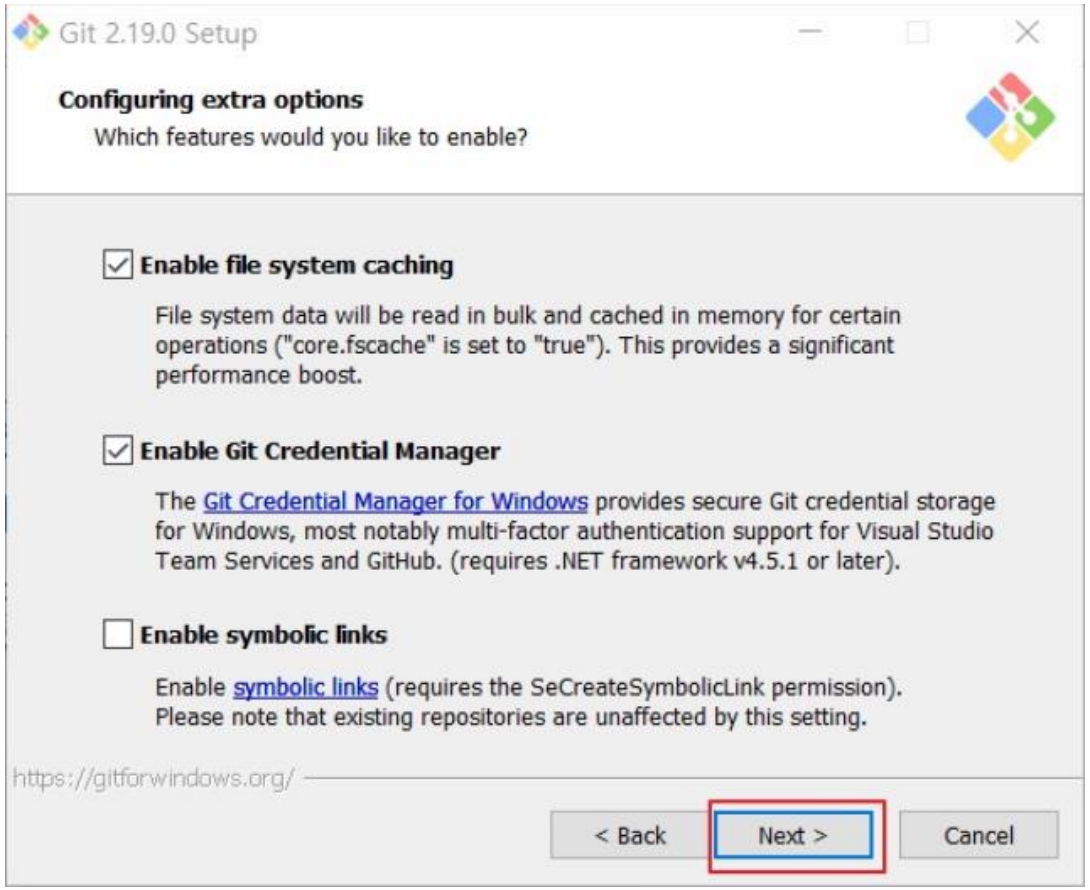

Git 2.19.0 Setup
Choosing HTTPS transport backend
Which SSL/TLS library would you like Git to use for HTTPS connections?
Use the OpenSSL library
Server certificates will be validated using the ca-bundle.crt file.
Use the native Windows Secure Channel library
Server certificates will be validated using Windows Certificate Stores.
This option also allows you to use your company's internal Root CA certificates
distributed e.g. via Active Directory Domain Services.
https://gitforwindows.org/
< Back
Next >
Cancel

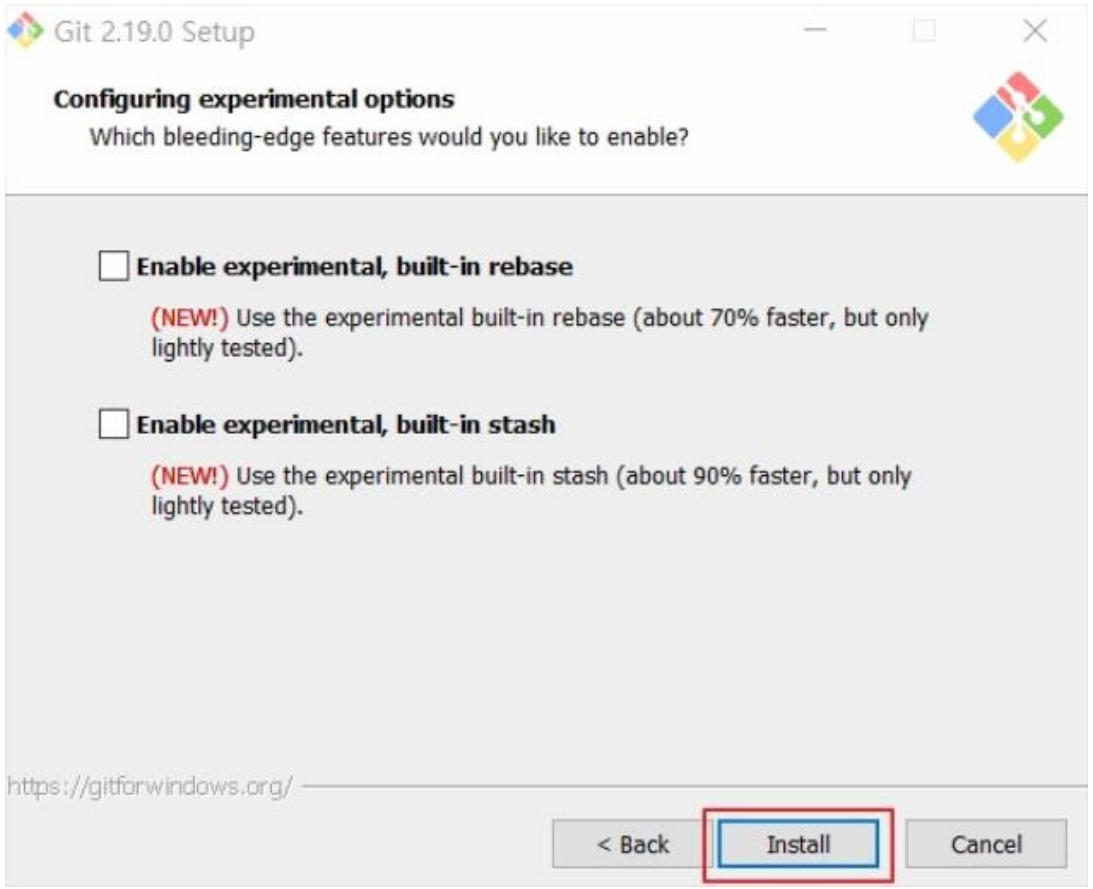
Install git


Git 2.19.0 Setup
Configuring the line ending conversions
How should Git treat line endings in text files?
Checkout Windows-style, commit Unix-style line endings
Git will convert LF to CRLF when checking out text files. When committing text files, CRLF will be converted to LF. For cross-platform projects, this is the recommended setting on Windows ("core.autocrlf" is set to "true").
Checkout as-is, commit Unix-style line endings
Git will not perform any conversion when checking out text files. When committing text files, CRLF will be converted to LF. For cross-platform projects, this is the recommended setting on Unix ("core.autocrlf" is set to "input").
Checkout as-is, commit as-is
Git will not perform any conversions when checking out or committing text files. Choosing this option is not recommended for cross-platform projects ("core.autocrlf" is set to "false").
https://gitforwindows.org/
< Back
Next >
Cancel


Git 2.19.0 Setup
Configuring the terminal emulator to use with Git Bash
Which terminal emulator do you want to use with your Git Bash?
Use MinTTY (the default terminal of MSYS2)
Git Bash will use MinTTY as terminal emulator, which sports a resizable window non-rectangular selections and a Unicode font. Windows console programs (such as interactive Python) must be launched via `winpty` to work in MinTTY.
Use Windows' default console window
Git will use the default console window of Windows ("cmd.exe"), which works well with Win32 console programs such as interactive Python or node.js, but has a very limited default scroll-back, needs to be configured to use a Unicode font in order to display non-ASCII characters correctly, and prior to Windows 10 its window was not freely resizable and it only allowed rectangular text selections.
https://gitforwindows.org/
< Back
Next >
Cancel

Install git


Git 2.19.0 Setup
Configuring extra options
Which features would you like to enable?
Enable file system caching
File system data will be read in bulk and cached in memory for certain operations ("core.fscache" is set to "true"). This provides a significant performance boost.
Enable Git Credential Manager
The Git Credential Manager for Windows provides secure Git credential storage for Windows, most notably multi-factor authentication support for Visual Studio Team Services and GitHub. (requires .NET framework v4.5.1 or later).
Enable symbolic links
Enable symbolic links (requires the SeCreateSymbolicLink permission). Please note that existing repositories are unaffected by this setting.
https://gitforwindows.org/
< Back
Next >
Cancel

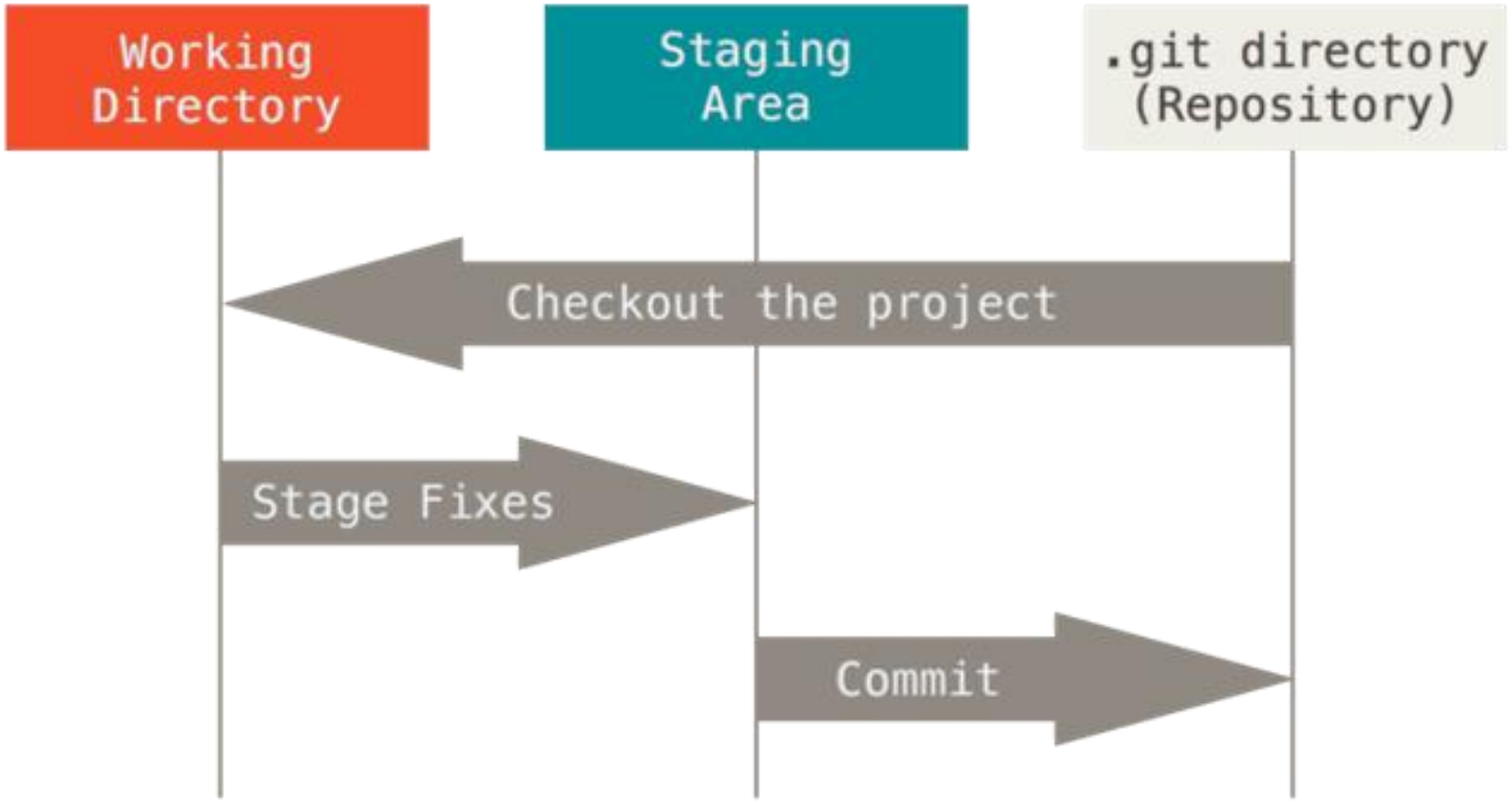

Git 2.19.0 Setup
Configuring experimental options
Which bleeding-edge features would you like to enable?
Enable experimental, built-in rebase
(NEW!) Use the experimental built-in rebase (about 70% faster, but only lightly tested).
Enable experimental, built-in stash
(NEW!) Use the experimental built-in stash (about 90% faster, but only lightly tested).
https://gitforwindows.org/
< Back
Install
Cancel

파뿌리
파이썬 뿌시는 이십대들!!

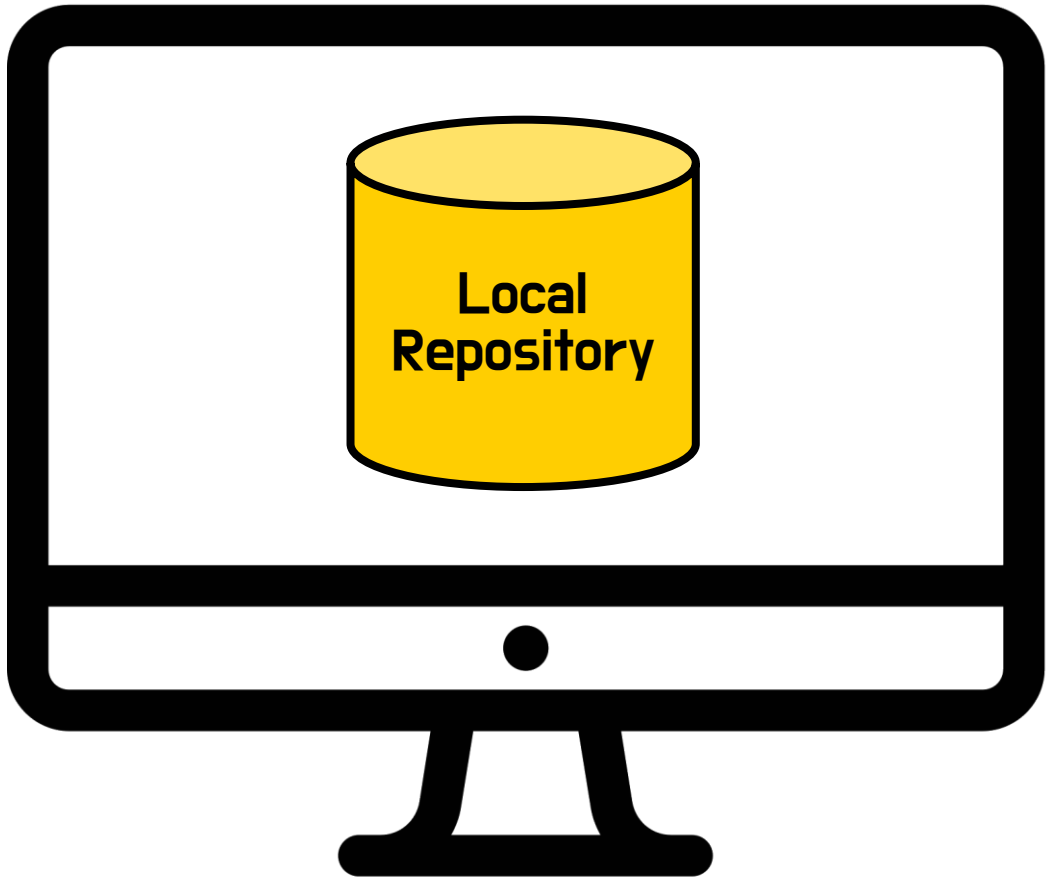
Git status

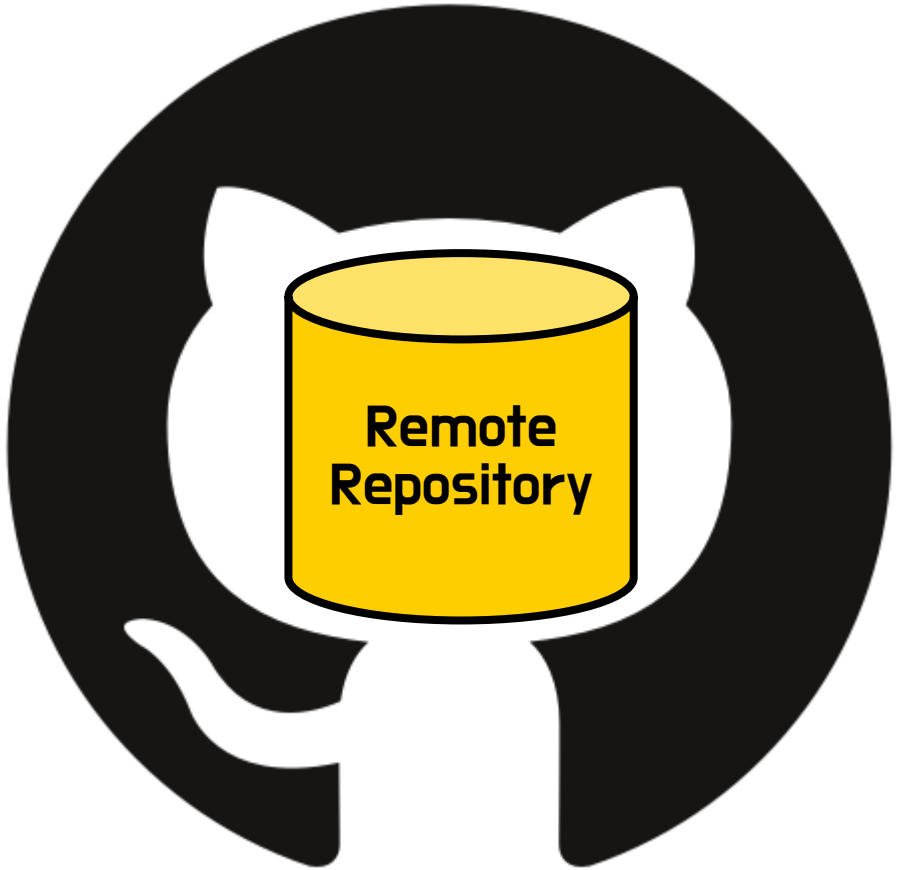
Working Directory	편집된 파일이 저장되어 있음
Staging Area	Repository 로 변경내역을 저장하기 위한 파일들의 목록
.git directory (Repository)	변경 내역이 저장되는 곳 (HEAD라고도 부름)

Git status

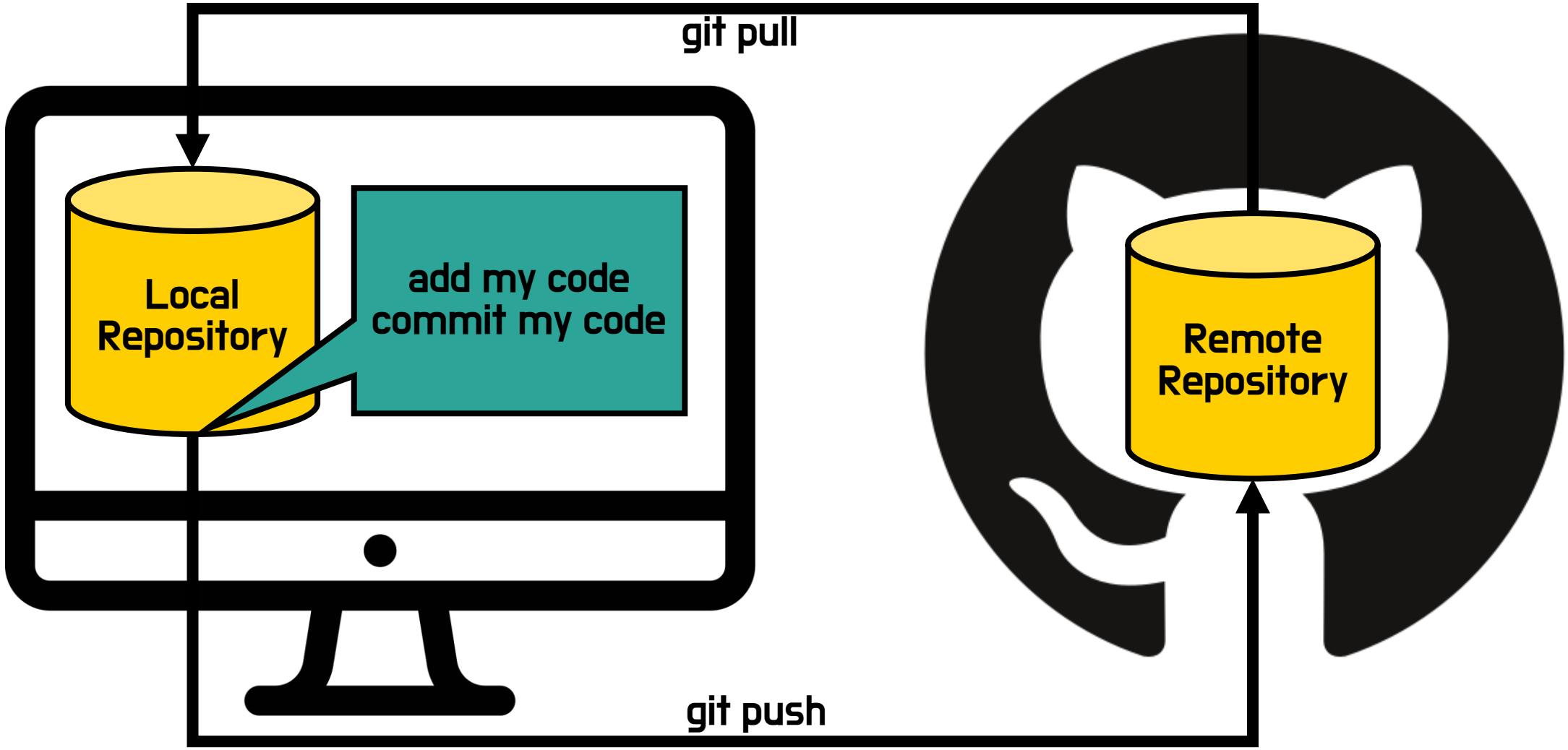

Working Directory
Staging Area
.git directory (Repository)
Checkout the project
Stage Fixes
Commit

Local & Remote Repository





Workflow


git pull
Local Repository
add my code
commit my code
Remote Repository
git push

파뿌리
파이썬 뿌시는 이십대들!!

《 Round 4 》

- Git
- Github 《
- Markup language

Github??

Open Sourse Software를 위한 Remote Repository 사이트

Github??

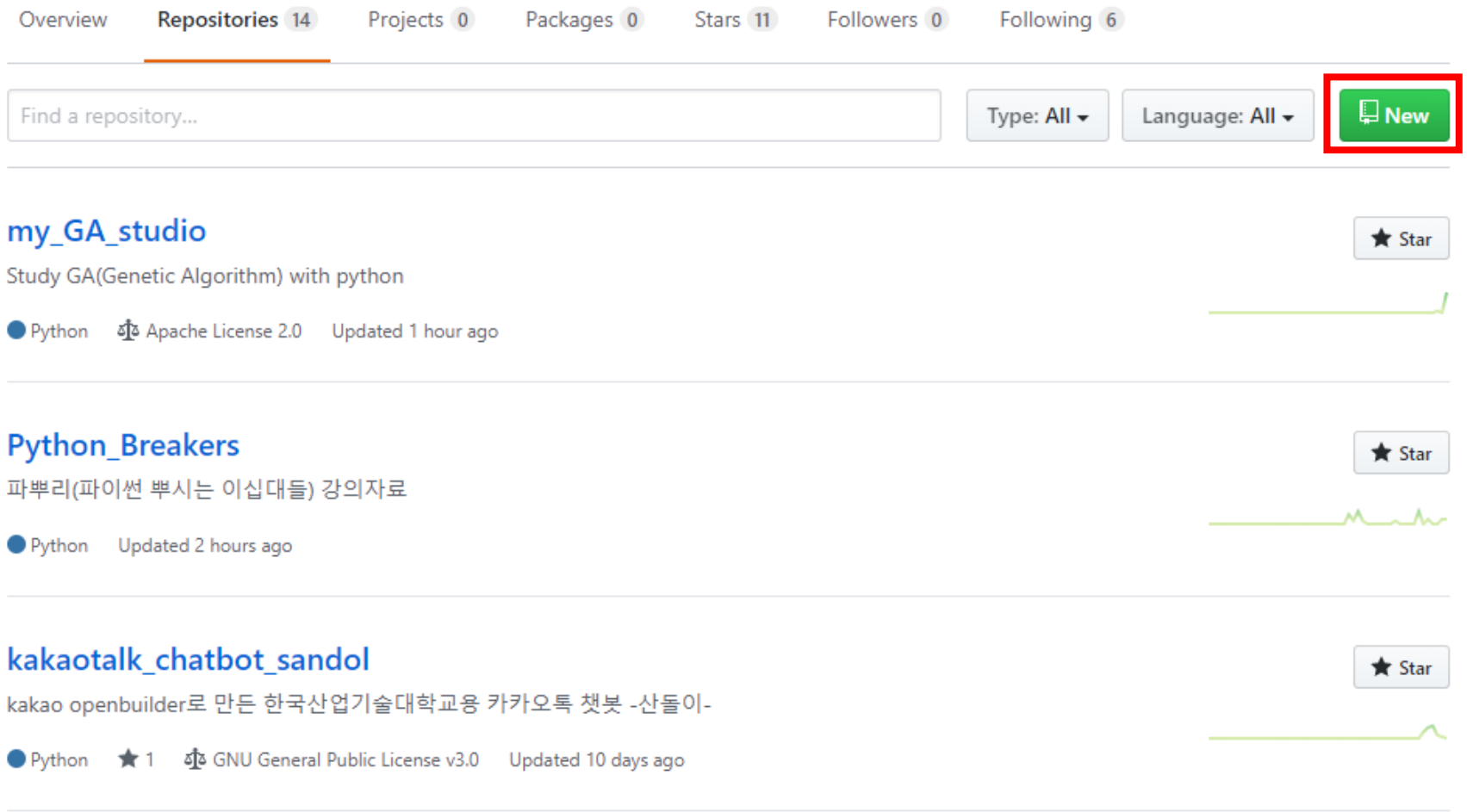
- 무료!

- 모든 소스코드 공개!!
(돈 내면 Extended Private Repository 생성 가능)

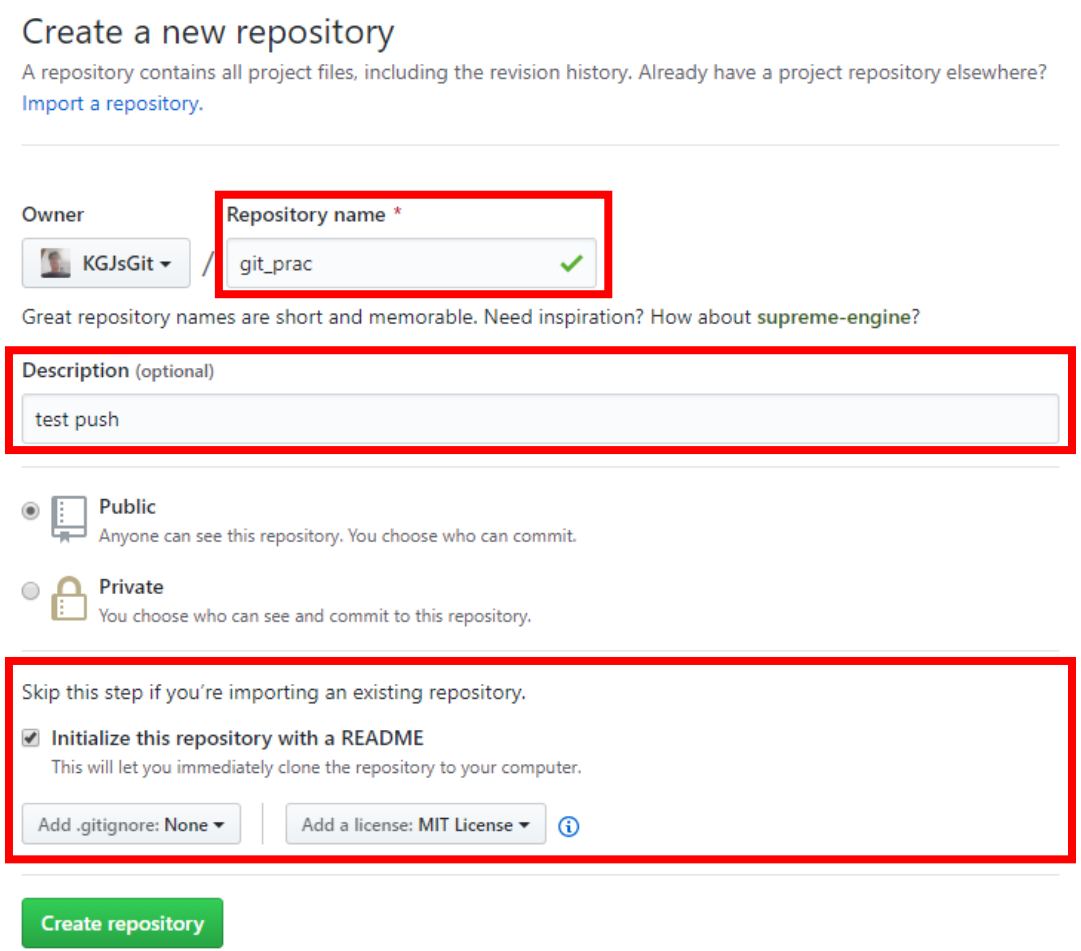
- Github Pages라는 자체 웹 호스팅 서비스 제공
(무료로 프로젝트 홈페이지 제작 가능)

- Open Sourse Software의 성지!!!

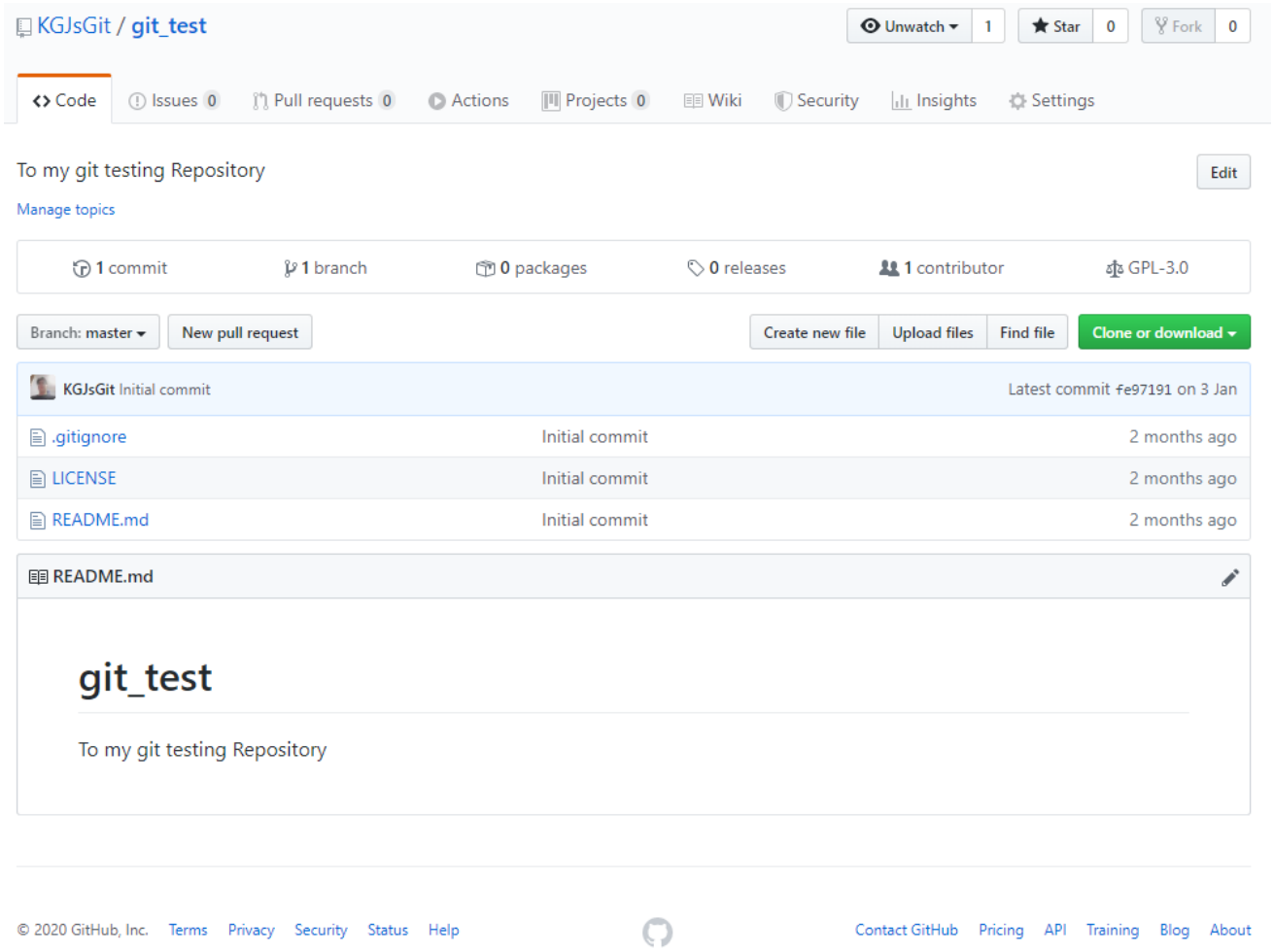
Make a Github repository



Make a Github repository



Make a Github repository



Git settings

- git config --global user.name [자기 이름]

- git config --global user.email [자기 이메일]

파뿌리
파이썬 뿌시는 이십대들!!

Git keyword

- git init
: git repository 생성

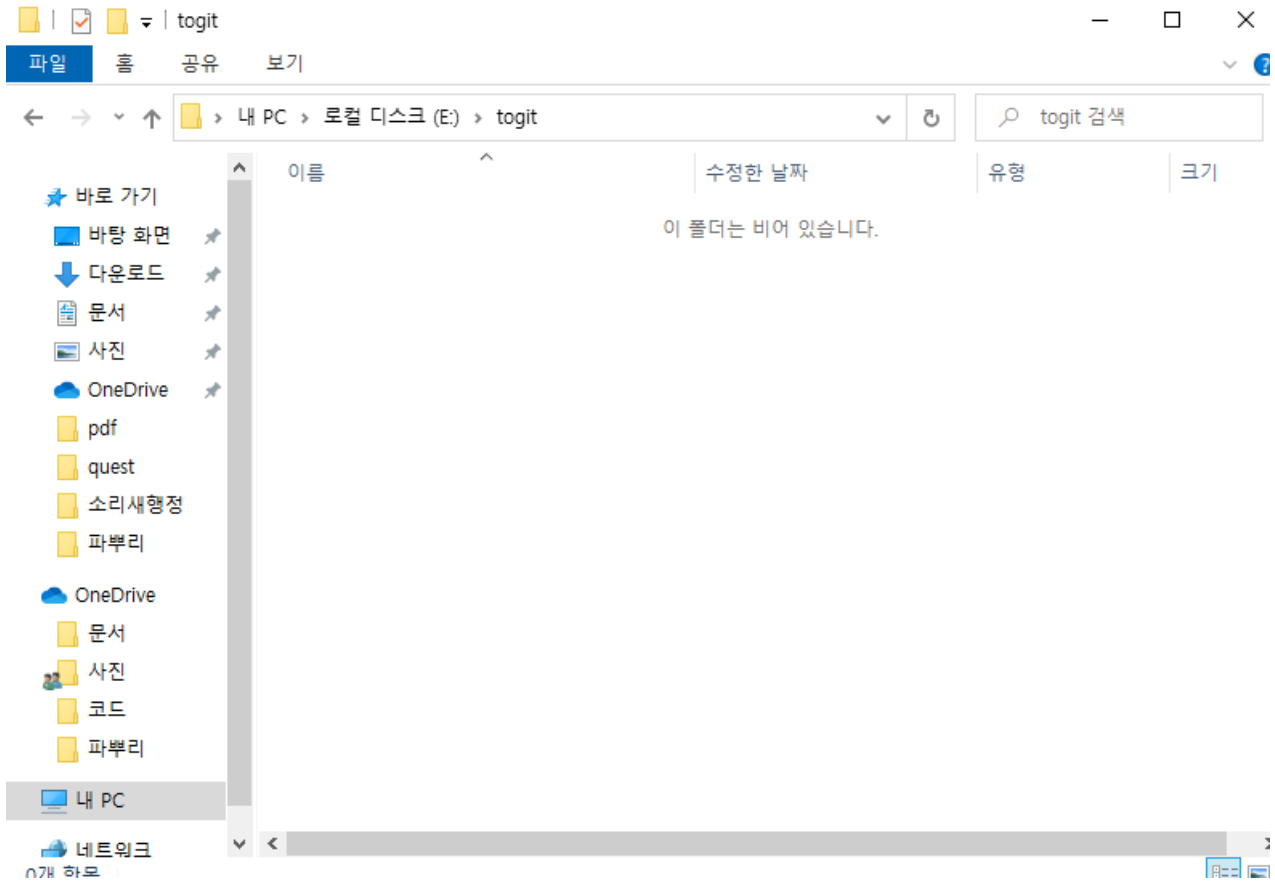
- git add "file name"
: 해당 파일 stage

- git commit -m "comment"
: staged 파일들 commit(repository에 반영)

- git pull "remote repository address"
: 해당 remote repositor를 받아와서 내 local repository에 합병하기

- git push origin master
- git push origin main
: local repository를 remote repository로 밀어넣기

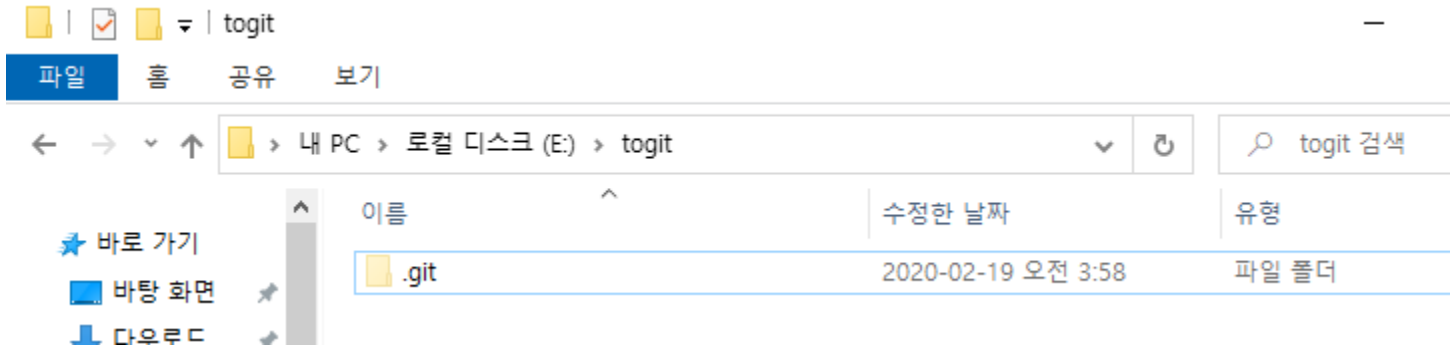
Git training



make a directory (anywhere)

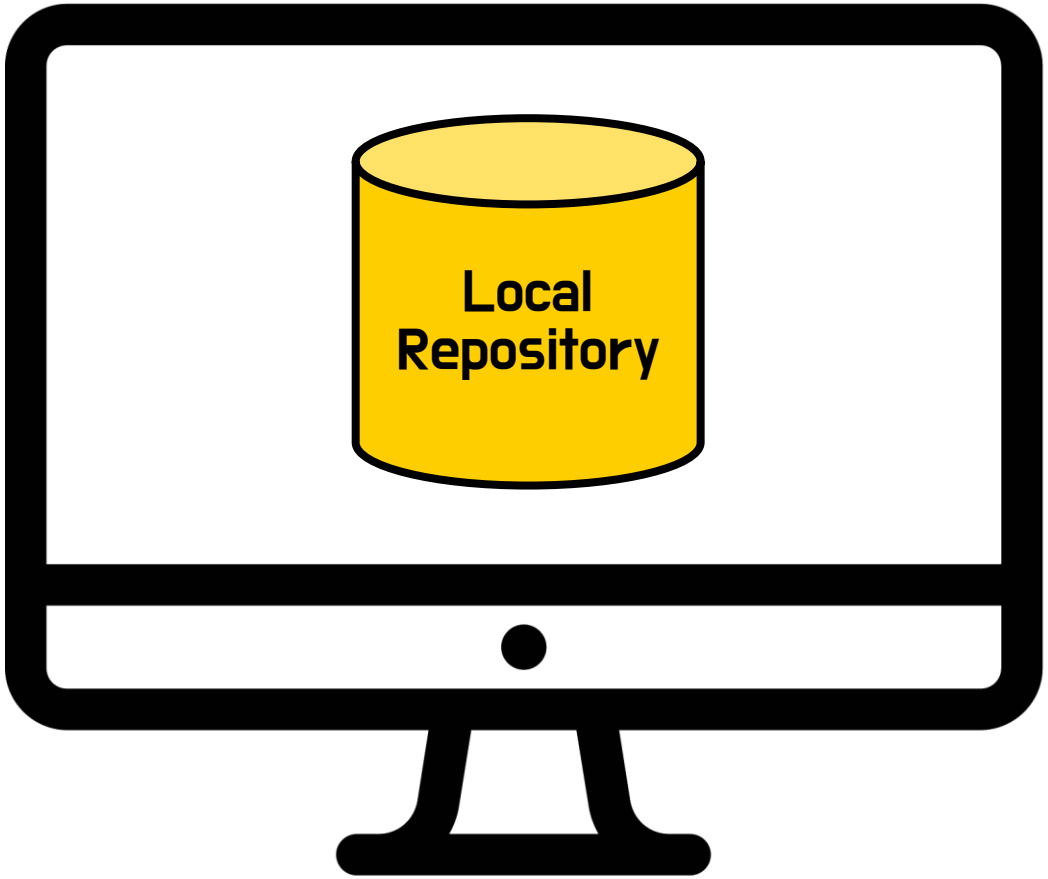
Git training

Go to directory & "git init"



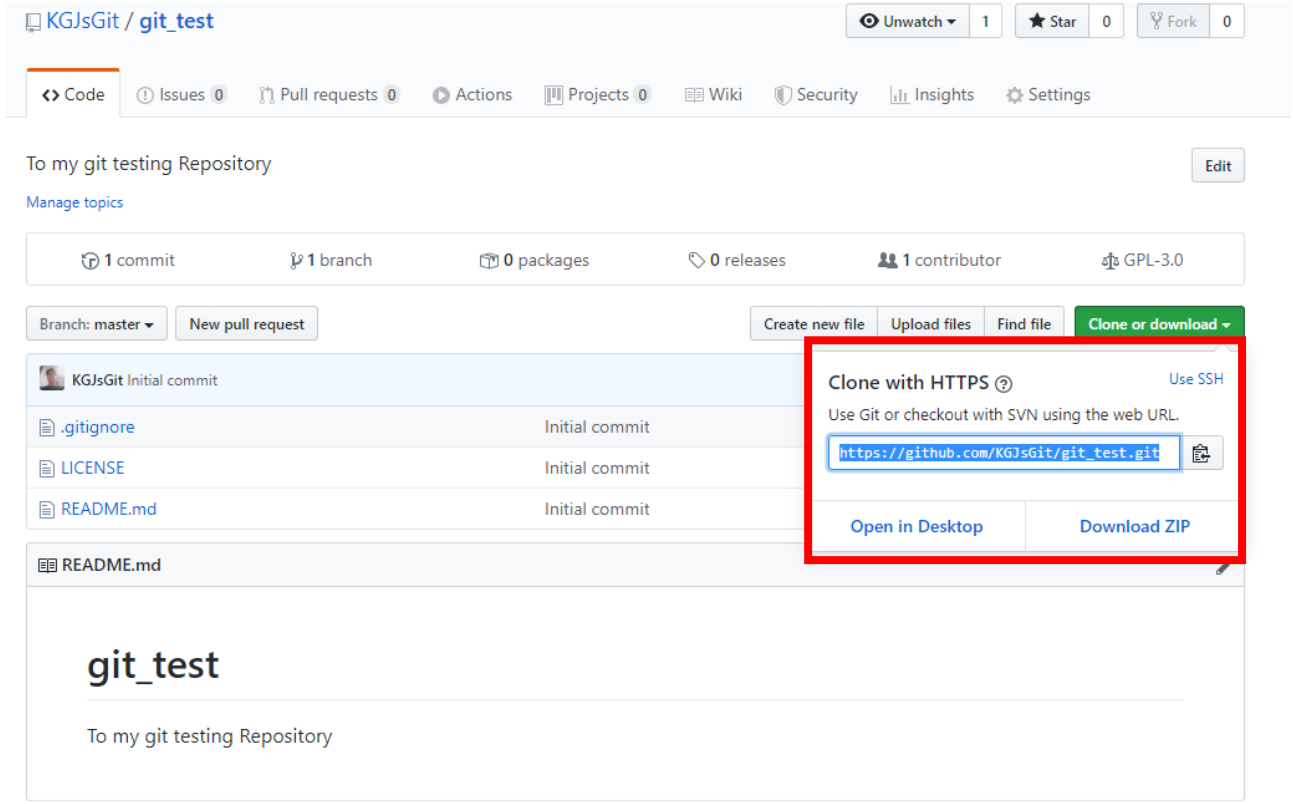
Check your local repository

Git training



"git init"

Git training

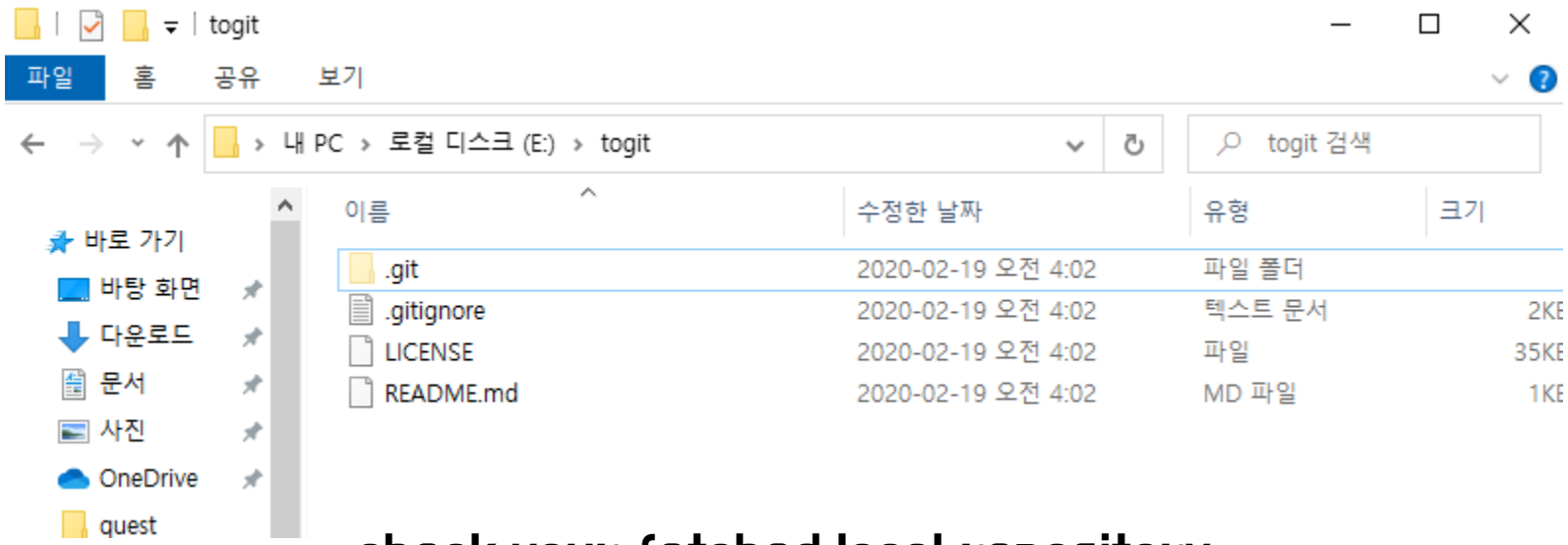


Copy the remote repository URL

Git training

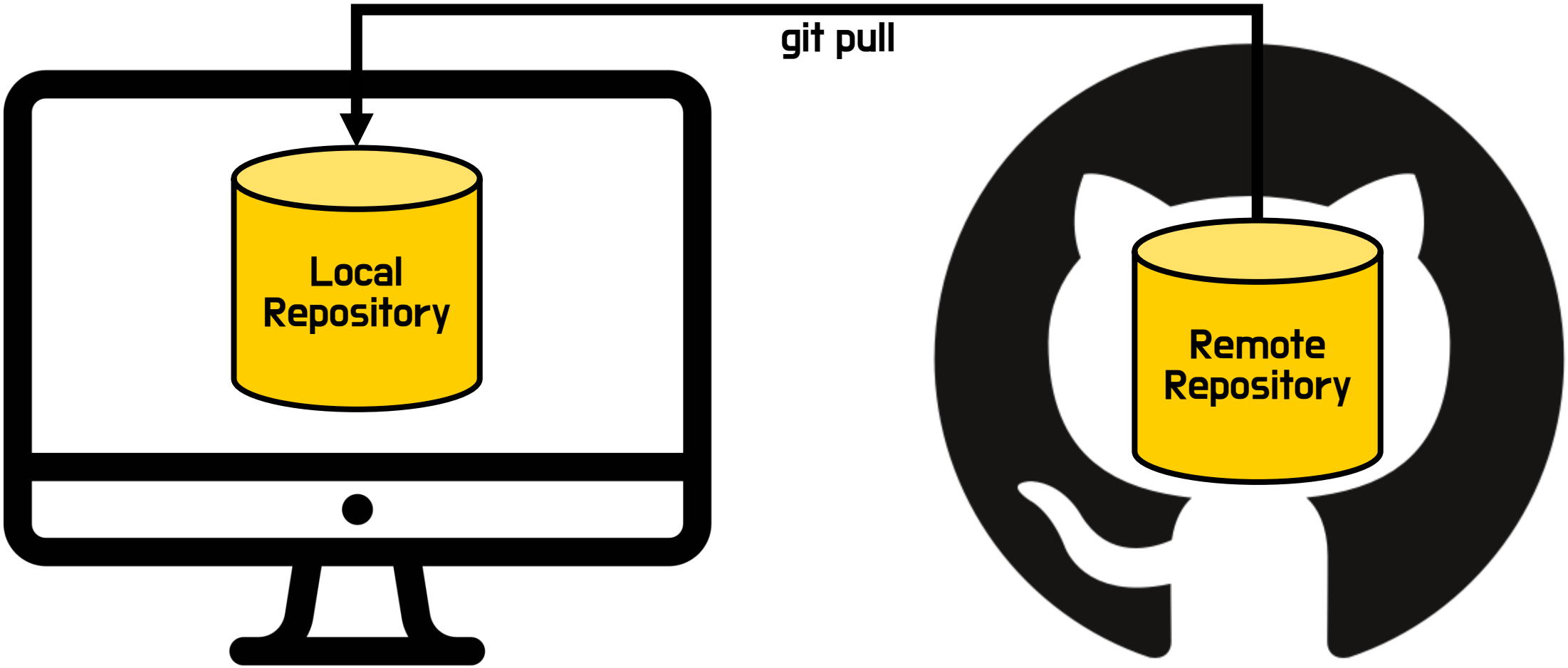
```
E:₩togit>git pull https://github.com/KGJsGit/git_test.git
remote: Enumerating objects: 5, done.
remote: Counting objects: 100% (5/5), done.
remote: Compressing objects: 100% (4/4), done.
remote: Total 5 (delta 0), reused 0 (delta 0), pack-reused 0
Unpacking objects: 100% (5/5), 13.45 KiB | 57.00 KiB/s, done.
From https://github.com/KGJsGit/git_test
 * branch            HEAD       -> FETCH_HEAD

E:₩togit>
```

"git pull 'remote repository URL' "



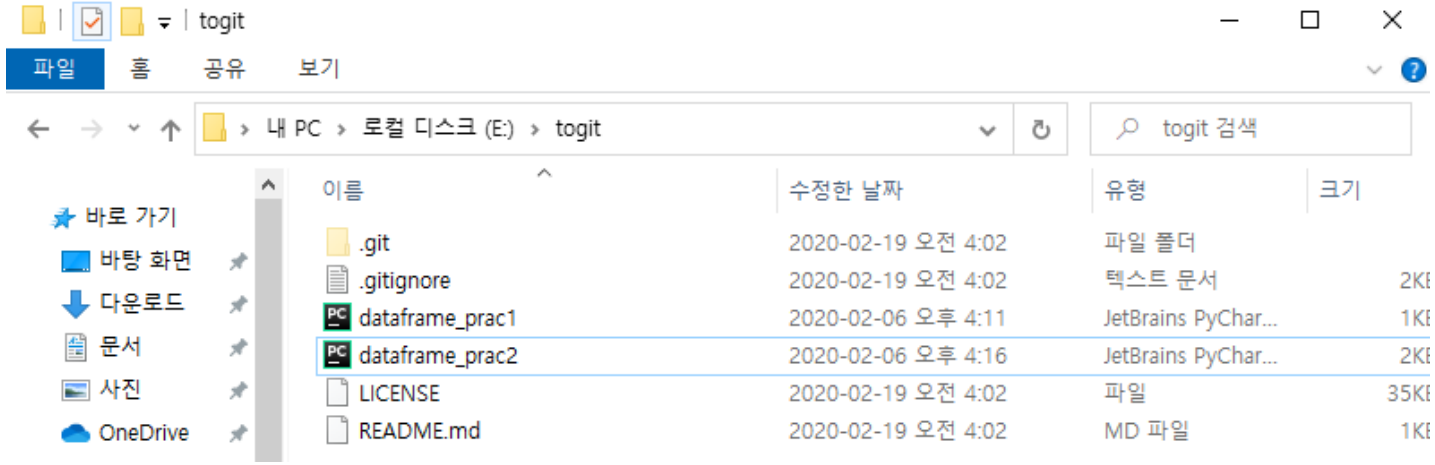
check your fetched local repository

Git training



"git pull 'remote repository URL' "

Git training



make files on the directory

```
E:₩togit>git status
On branch master
Untracked files:
  (use "git add <file>..." to include in what will be committed)
        dataframe_prac1.py
        dataframe_prac2.py

nothing added to commit but untracked files present (use "git add" to track)

E:₩togit>
```

check the git status. "git status"

Git training


```
E:₩togit>git status
On branch master
Untracked files:
  (use "git add <file>..." to include in what will be committed)
        dataframe_prac1.py
        dataframe_prac2.py

nothing added to commit but untracked files present (use "git add" to track)

E:₩togit>
```

check the git status. "git status"
(2 files are untracked)

Git training

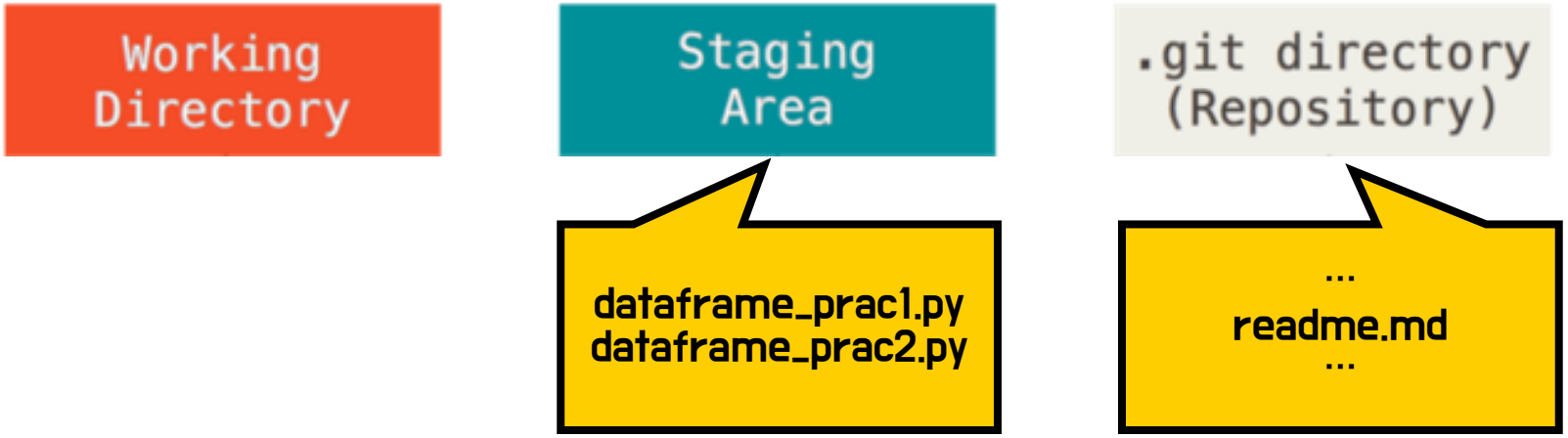
```
E:₩togit>git add dataframe_prac1.py

E:₩togit>git add dataframe_prac2.py
```

"git add 'file name' "

```
E:₩togit>git status
On branch master
Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
        new file:   dataframe_prac1.py
        new file:   dataframe_prac2.py


E:₩togit>
```

check the git status. "git status"

Git training




```
E:₩togit>git status
On branch master
Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
        new file:   dataframe_prac1.py
        new file:   dataframe_prac2.py


E:₩togit>
```

check the git status. "git status"
(2 files are staged)

Git training

```
E:₩togit>git commit -m "add new files"
[master e7c7219] add new files
 2 files changed, 56 insertions(+)
 create mode 100644 dataframe_prac1.py
 create mode 100644 dataframe_prac2.py
```

"git commit -m "Add new files"
(commit message는 명료하고 보기 좋게 쓸 것)

```
E:₩togit>git status
On branch master
nothing to commit, working tree clean

E:₩togit>
```

check the git status. "git status"

Git training

Working Directory

Staging Area

.git directory (Repository)

...
readme.md
dataframe_prac1.py
dataframe_prac2.py
...

```
E:₩togit>git status
On branch master
nothing to commit, working tree clean

E:₩togit>
```

check the git status. "git status"

Git training

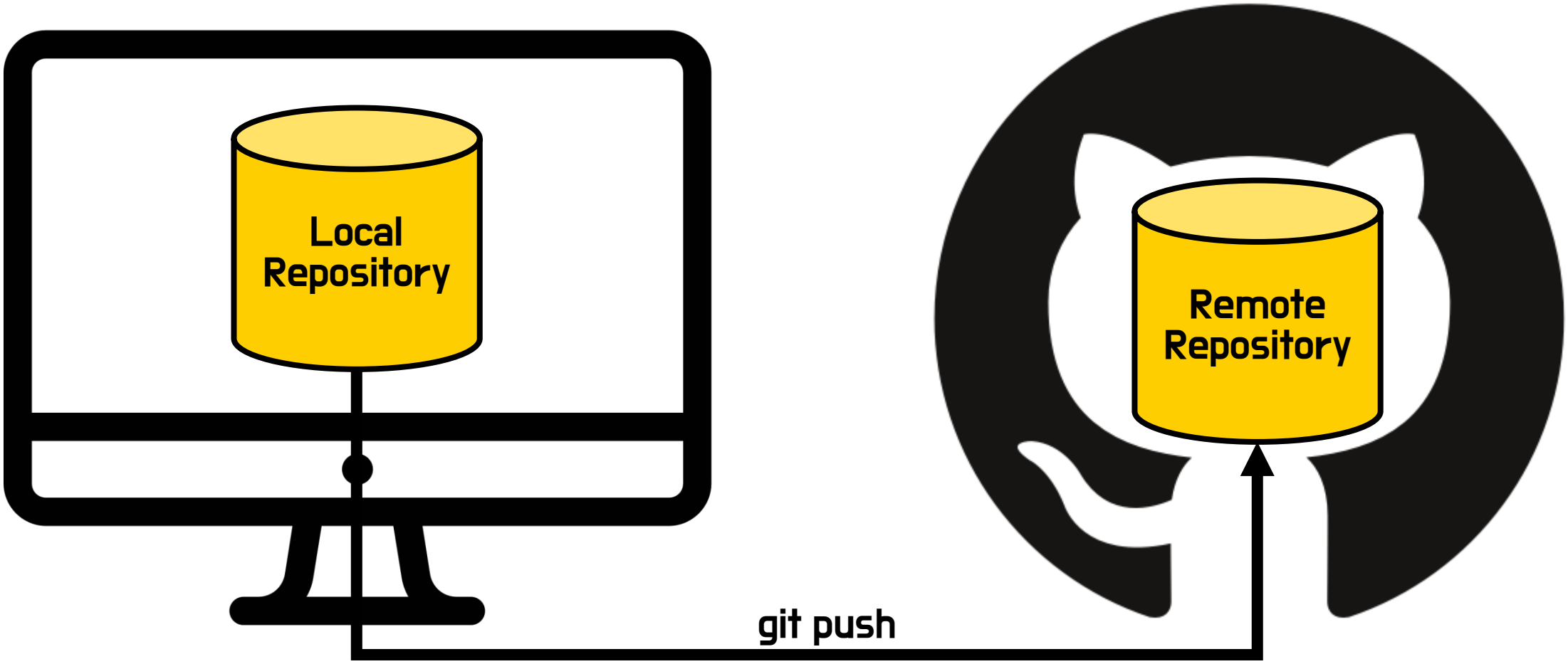
```
E:₩togit>git remote add origin https://github.com/KGJsGit/git_test.git
```

"git remote add origin `remote repository URL'"
(push해줄 remote repository의 URL 명시)

```
E:₩togit>git push origin master
Enumerating objects: 5, done.
Counting objects: 100% (5/5), done.
Delta compression using up to 4 threads
Compressing objects: 100% (4/4), done.
Writing objects: 100% (4/4), 1.31 KiB | 1.31 MiB/s, done.
Total 4 (delta 0), reused 0 (delta 0)
To https://github.com/KGJsGit/git_test.git
   fe97191..9c4acc0  master -> master
```

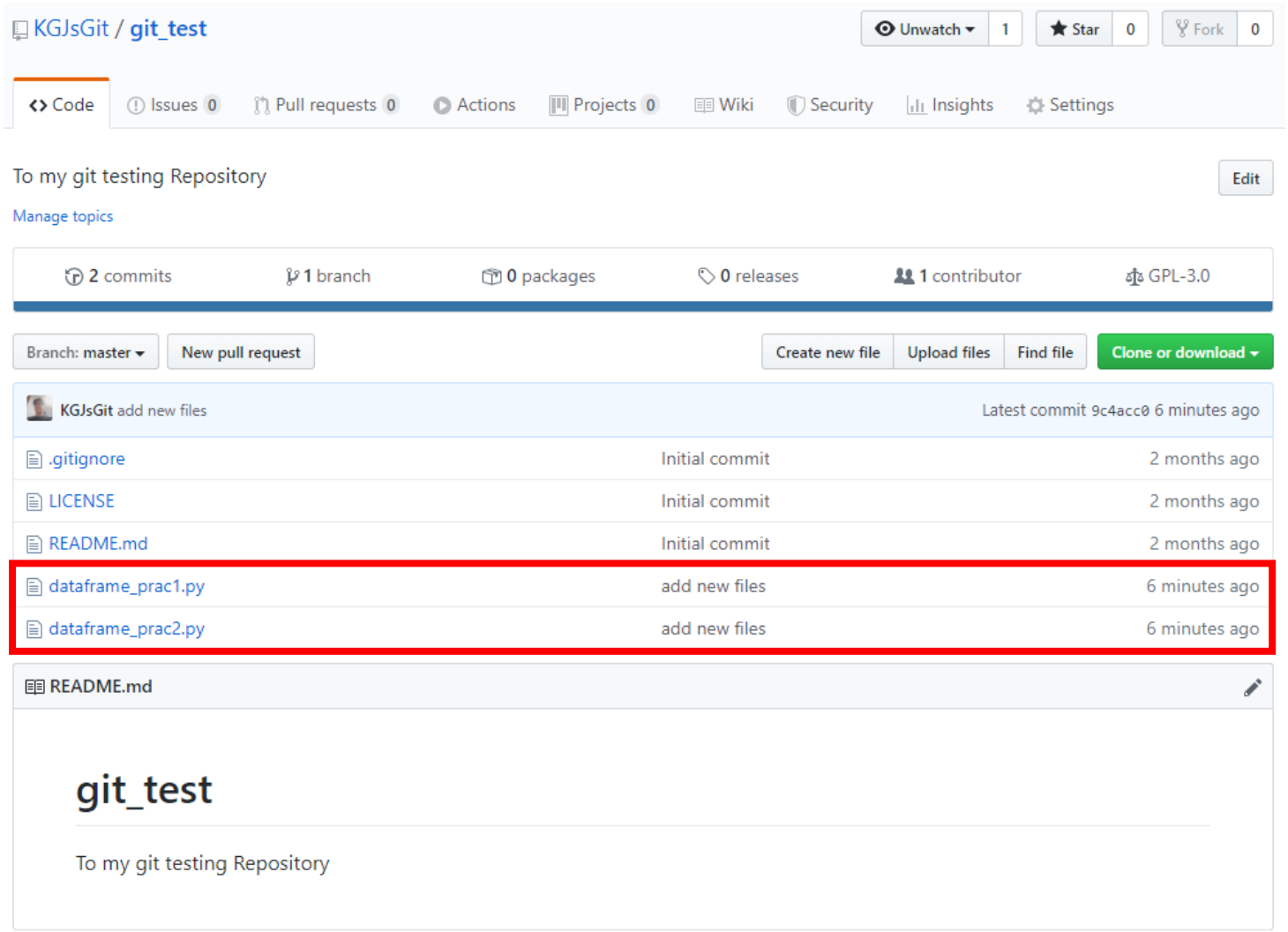
"git push origin master"
of
"git push origin main"

Git training



"git push origin master"

Git training



Additional Git keyword

- git clone
: remote repository의 모든 데이터를 복사. push시 원격저장소 자동 지정

- git rm 'file name'
: 깃의 파일을 삭제. working directory에서도 삭제됩니다. commit필요.

- git push -u origin master
: push의 -u 옵션은 remote repositor로부터 업데이트 받은 후 push하겠다는 의미

- git commit --amend -m 'comment'
: 최근 실수한 commit을 덮어 쓸 수 있음. 다만 되돌린 commit을 다시 되돌릴 수는 없음.

- git reset HEAD 'file name'
: 실수로 staging area에 올린 파일을 unstaged로 바꿔 줌

- git reset hard 'commit num'
: 해당 commit으로 git을 되돌리고 그 이후 모든 이력을 삭제합니다. hard대신 다양한 옵션 사용 가능

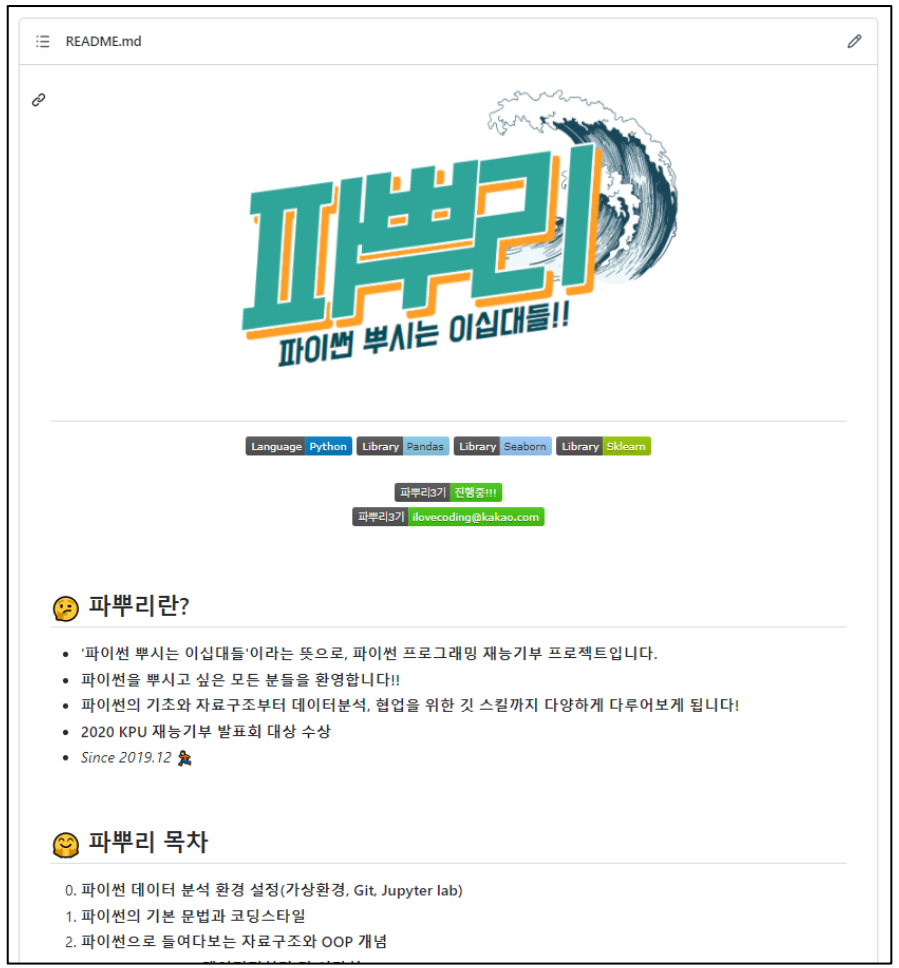
파뿌리
파이썬 뿌시는 이십대들!!

《 Round 4 》

- Git
- Github
- Markdown language 《

Markdown?

- Github repository는 README.md라는 markdown 파일이 얼굴의 역할을 함



README.md

파뿌리

파이썬 뿌시는 이십대들!!

Language Python | Library Pandas | Library Seaborn | Library Sklearn

파뿌리3기 진행중!!!

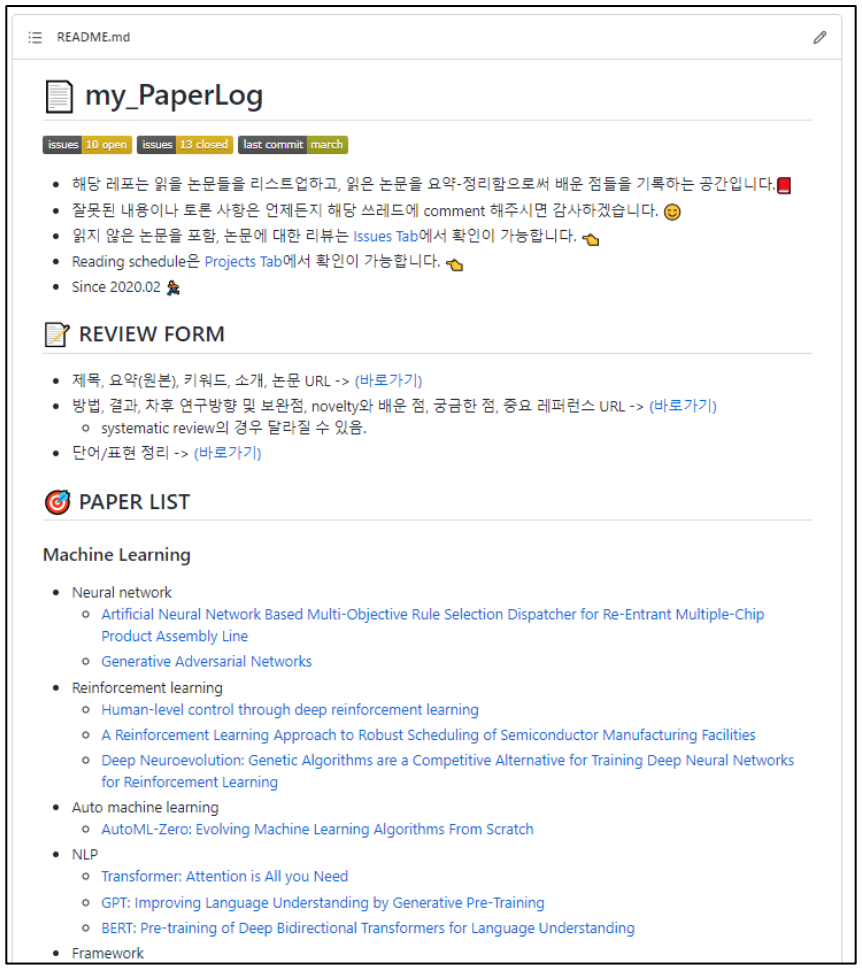
파뿌리3기 ilovecoding@kakao.com

파뿌리란?

- '파이썬 뿌시는 이십대들'이라는 뜻으로, 파이썬 프로그래밍 재능기부 프로젝트입니다.
- 파이썬을 뿌시고 싶은 모든 분들을 환영합니다!!
- 파이썬의 기초와 자료구조부터 데이터분석, 협업을 위한 깃 스킬까지 다양하게 다루어보게 됩니다!
- 2020 KPU 재능기부 발표회 대상 수상
- *Since 2019.12*

파뿌리 목차

0. 파이썬 데이터 분석 환경 설정(가상환경, Git, Jupyter lab)
1. 파이썬의 기본 문법과 코딩스타일
2. 파이썬으로 들여다보는 자료구조와 OOP 개념



README.md

my_PaperLog

issues 10 open | issues 13 closed | last commit march

- 해당 레포는 읽을 논문들을 리스트업하고, 읽은 논문을 요약-정리함으로써 배운 점들을 기록하는 공간입니다.
- 잘못된 내용이나 토론 사항은 언제든지 해당 쓰레드에 comment 해주시면 감사하겠습니다.
- 읽지 않은 논문을 포함, 논문에 대한 리뷰는 Issues Tab에서 확인이 가능합니다.
- Reading schedule은 Projects Tab에서 확인이 가능합니다.
- Since 2020.02

REVIEW FORM

- 제목, 요약(원본), 키워드, 소개, 논문 URL -> (바로가기)
- 방법, 결과, 차후 연구방향 및 보완점, novelty와 배운 점, 궁금한 점, 중요 레퍼런스 URL -> (바로가기)
  - systematic review의 경우 달라질 수 있음.
- 단어/표현 정리 -> (바로가기)

PAPER LIST

Machine Learning

- Neural network
  - Artificial Neural Network Based Multi-Objective Rule Selection Dispatcher for Re-Entrant Multiple-Chip Product Assembly Line
  - Generative Adversarial Networks
- Reinforcement learning
  - Human-level control through deep reinforcement learning
  - A Reinforcement Learning Approach to Robust Scheduling of Semiconductor Manufacturing Facilities
  - Deep Neuroevolution: Genetic Algorithms are a Competitive Alternative for Training Deep Neural Networks for Reinforcement Learning
- Auto machine learning
  - AutoML-Zero: Evolving Machine Learning Algorithms From Scratch
- NLP
  - Transformer: Attention is All you Need
  - GPT: Improving Language Understanding by Generative Pre-Training
  - BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding
- Framework

Markdown?

- 잘 쓰인 README는 이목을 잘 끌 수 있고, 효과적으로 프로젝트의 Abstract를 전달할 수 있음

- 또한 깃헙 내 여러 문서에 마크다운이 사용되며, 이를 활용해 개인 홈페이지나 todo list를 개설할 수 있음