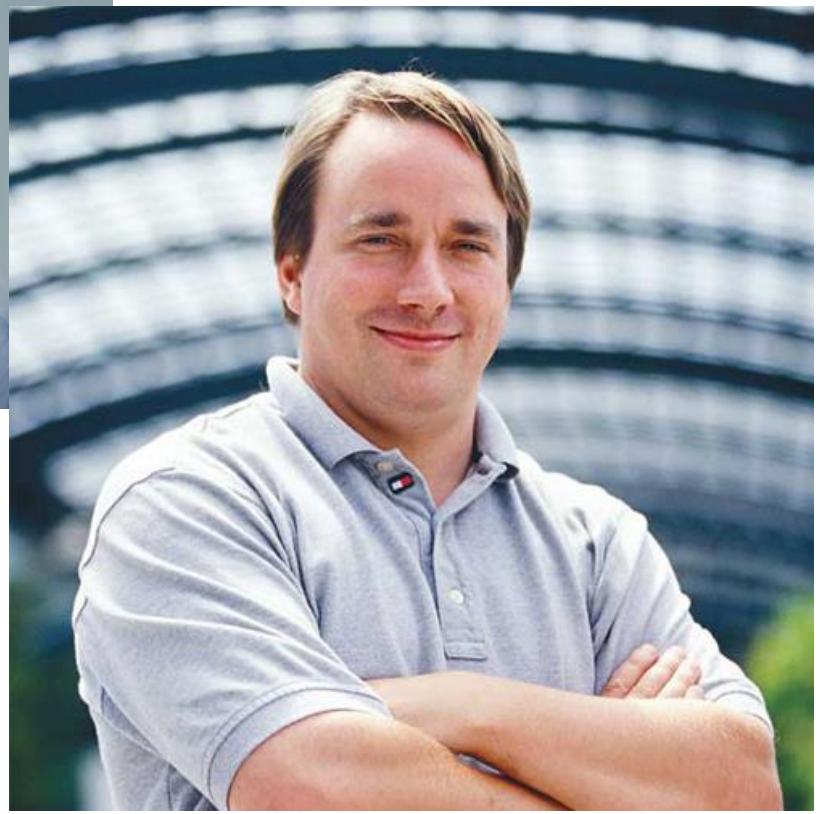

CODINGON
X
posco
K-Digital Training
4기


git

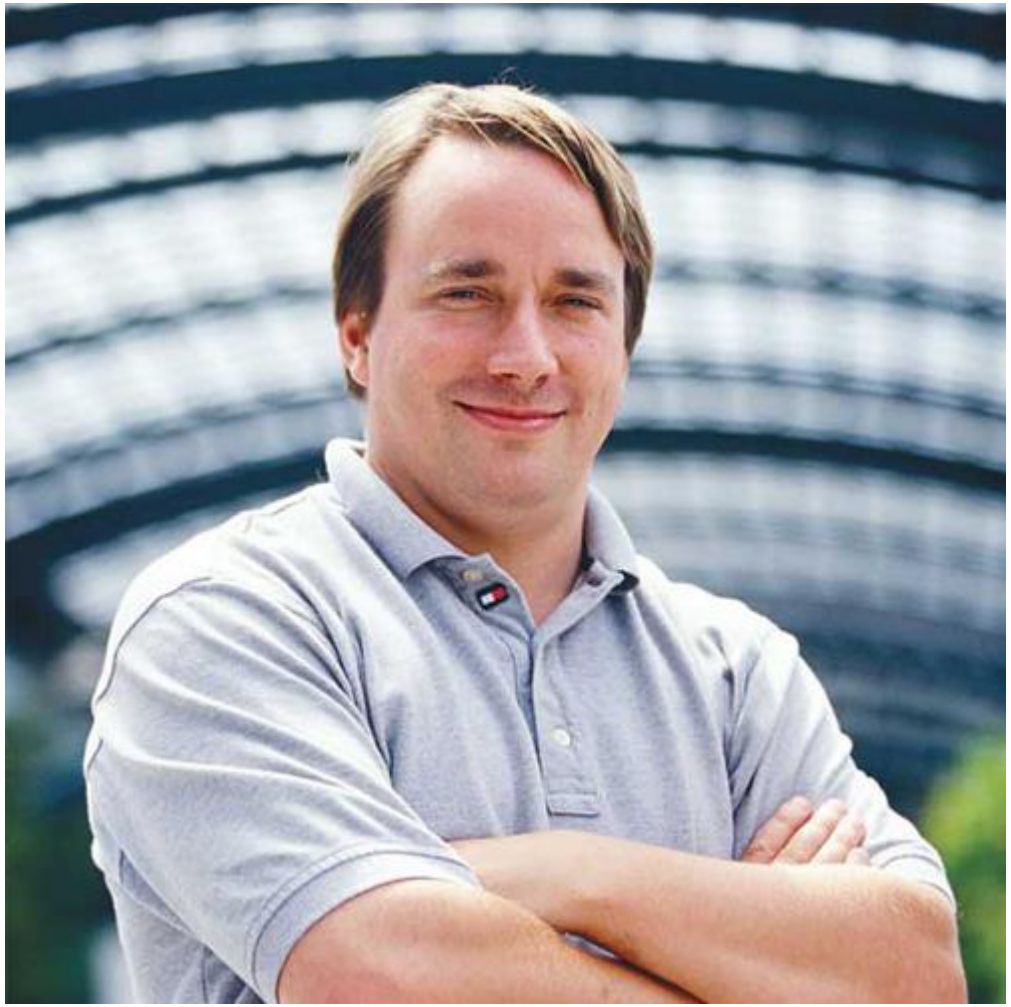
Git(깃)은 컴퓨터 파일의 변경사항을 추적하고
여러 사용자들 간에 해당 파일 작업을 조율하기 위한
대표적인 버전 관리 시스템(VCS)입니다.

VCS(Version Control System)

세계 3대 개발자!?



Linus Benedict Torvalds



Git이 하는 일은!

버전 관리

버전 관리


v_0.1
v_1.0
v_1.0_최종
v_1.3_제발_최종
v_1.5_더 이상은Naver


Commit
Commit
Commit
Commit
Commit

Git의 특징!

- 폴더 단위로 관리! → 하나의 프로젝트 단위로 관리

- Git에 대한 모든 정보는 .git 폴더에 있습니다
  - 일종의 블랙 박스!
- 커밋은 로컬(= 여러분의 컴퓨터)에서만 일어나는 행위입니다!
- 따라서, 깃 허브는 여러분의 로컬에 있는 깃 정보를 온라인에 저장해 주는 서비스를 제공하는 것입니다!
- 깃 허브의 repository는 여러분 로컬의 폴더의 개념과 같아요!

Commit?

project

로컬에 설치

사용자
(local, 로컬)

Git, 설치하기(Windows)

https://git-scm.com/

Download for Windows

Click here to download the latest (**2.36.1**) **64-bit** version of **Git for Windows**. This is the most recent maintained build. It was released **24 days ago**, on 2022-05-09.

Other Git for Windows downloads

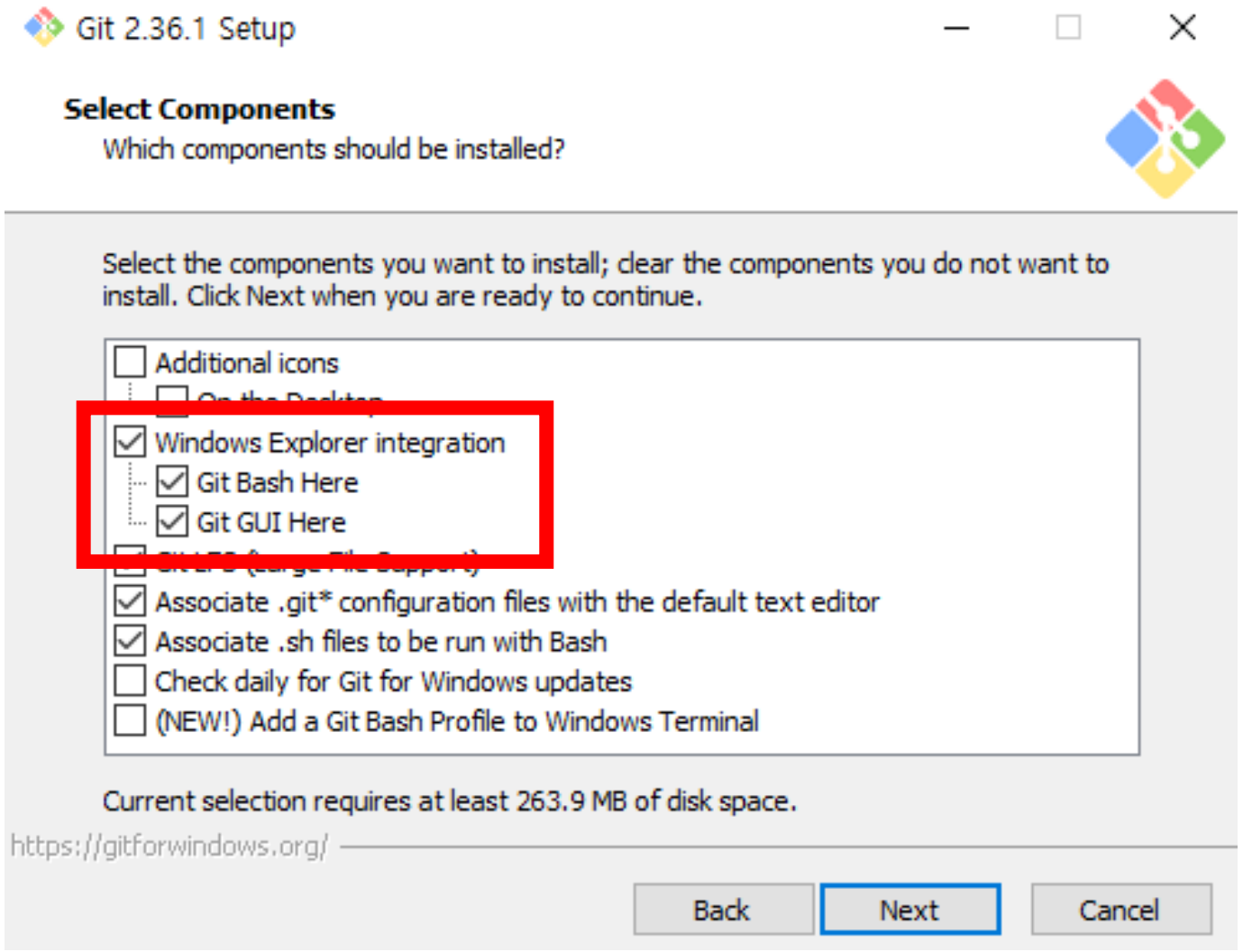
Standalone Installer
32-bit Git for Windows Setup.

64-bit Git for Windows Setup.

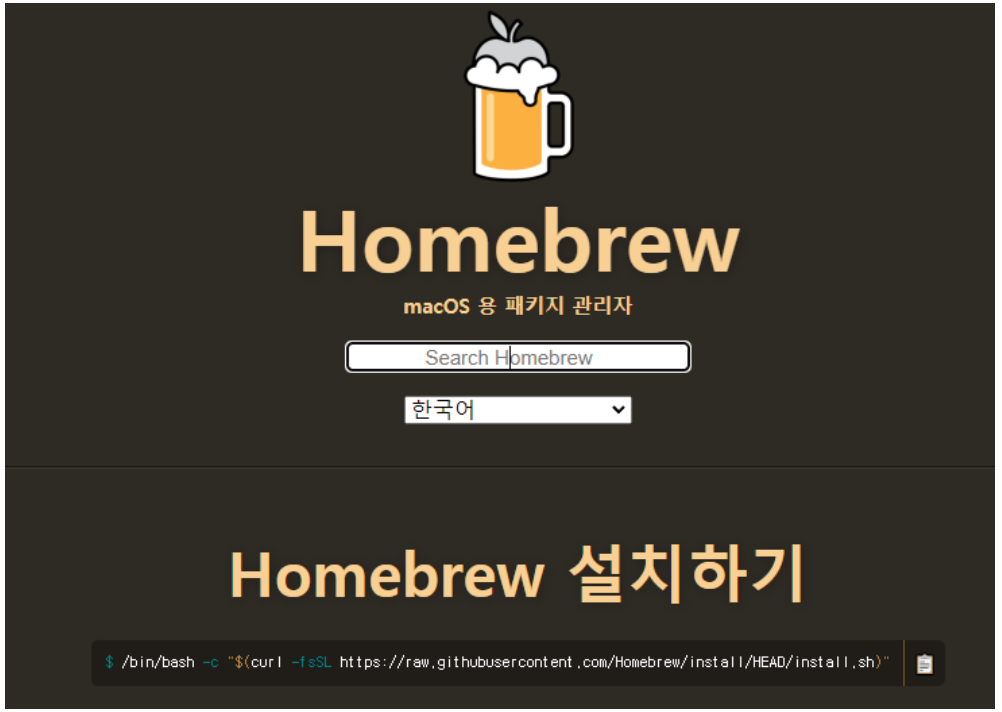
Portable ("thumbdrive edition")
32-bit Git for Windows Portable.

64-bit Git for Windows Portable.

Git, 설치하기(Windows)



Git, 설치하기(Mac)



https://brew.sh/index_ko

brew install git

Git hub 사용을 위한 세팅 시작!

- git init
- git config --global init.defaultBranch main
- 개행 문자 관련 처리(윈도우는 CR, LF / 맥은 LF 만)
  - git config --global core.autocrlf true (Window, CRLF → LF)
  - git config --global core.autocrlf input (Mac, LF 만 사용)
- git config --global user.name “프로필 이름”
- git config --global user.email “이메일 주소”
- git config --lobal --list

```
# 개행 문자(Newline) 설정
## macOS
$ git config --global core.autocrlf input
## Windows
$ git config --global core.autocrlf true

# 사용자 정보
## 커밋(버전 생성)을 위한 정보 등록
$ git config --global user.name 'YOUR_NAME'
$ git config --global user.email 'YOUR_EMAIL'

# 구성 확인
## Q키를 눌러서 종료!
$ git config --global --list
```

project

로컬에 설치

사용자
(local, 로컬)

```
$ git init
# 현재 프로젝트에서 변경사항 추적(버전 관리)을 시작.
```

git

사용자
(local, 로컬)

master

index.html

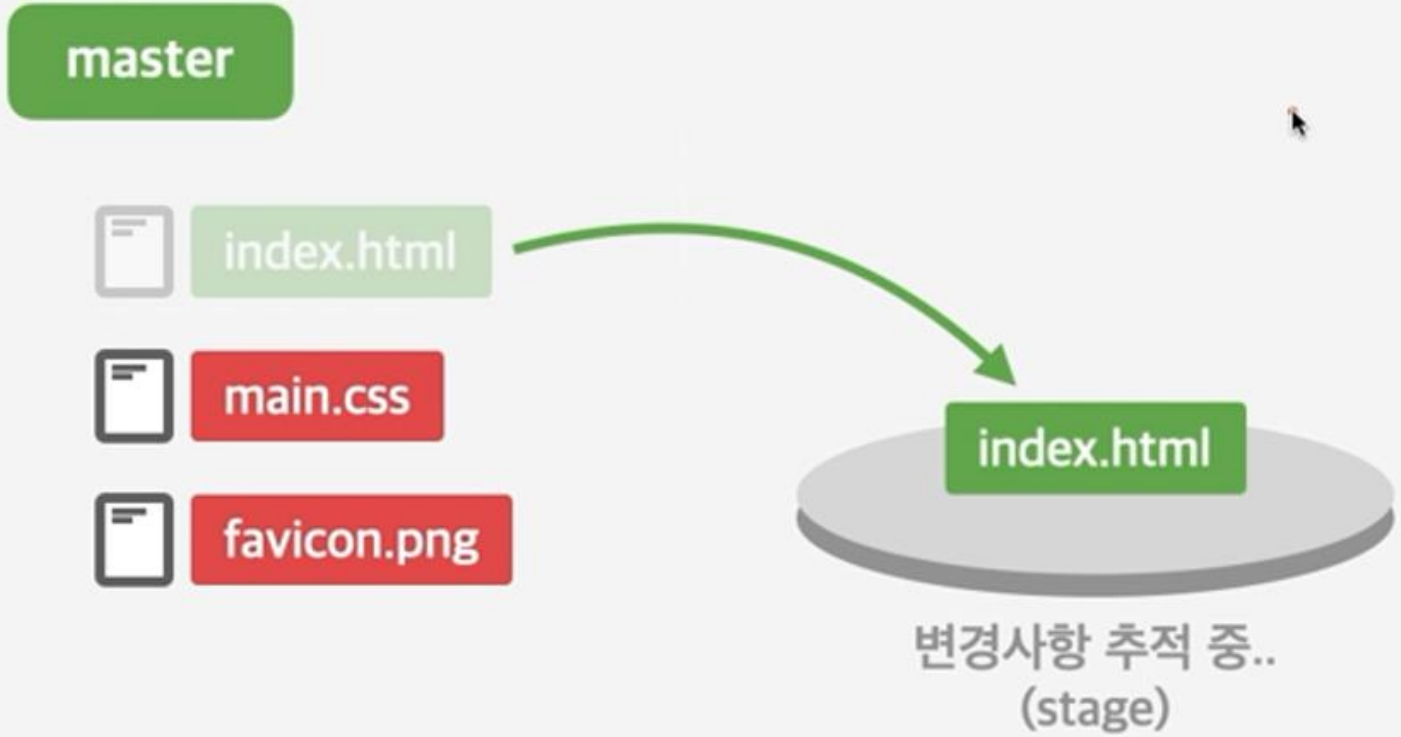
main.css

favicon.png

변경사항 추적 중..
(stage)

```
$ git add index.html
# 변경사항을 추적할 특정 파일(index.html)을 지정.
```

git 사용자 (local, 로컬)



```
$ git add .
# 모든 파일의 변경사항을 추적하도록 지정.
```

git 사용자 (local, 로컬)

master

```
$ git commit -m '프로젝트 생성'
# 메시지(-m)와 함께 버전을 생성.
```

git

사용자
(local, 로컬)

master

- index.html
- main.css
- favicon.png

● 프로젝트 생성

```
$
```

git 사용자 (local, 로컬)

master

index.html 수정함

main.css 수정함

favicon.png

main.js

main.js 추가

프로젝트 생성

```
$ git add .
# 모든 파일의 변경사항을 추적하도록 지정.
```

git 사용자 (local, 로컬)

master

- index.html
- main.css
- favicon.png
- main.js

main.js 추가

프로젝트 생성

```
$ git commit -m 'index.html 수정'
# 메시지(-m)와 함께 버전을 생성.
```

git

사용자
(local, 로컬)

master

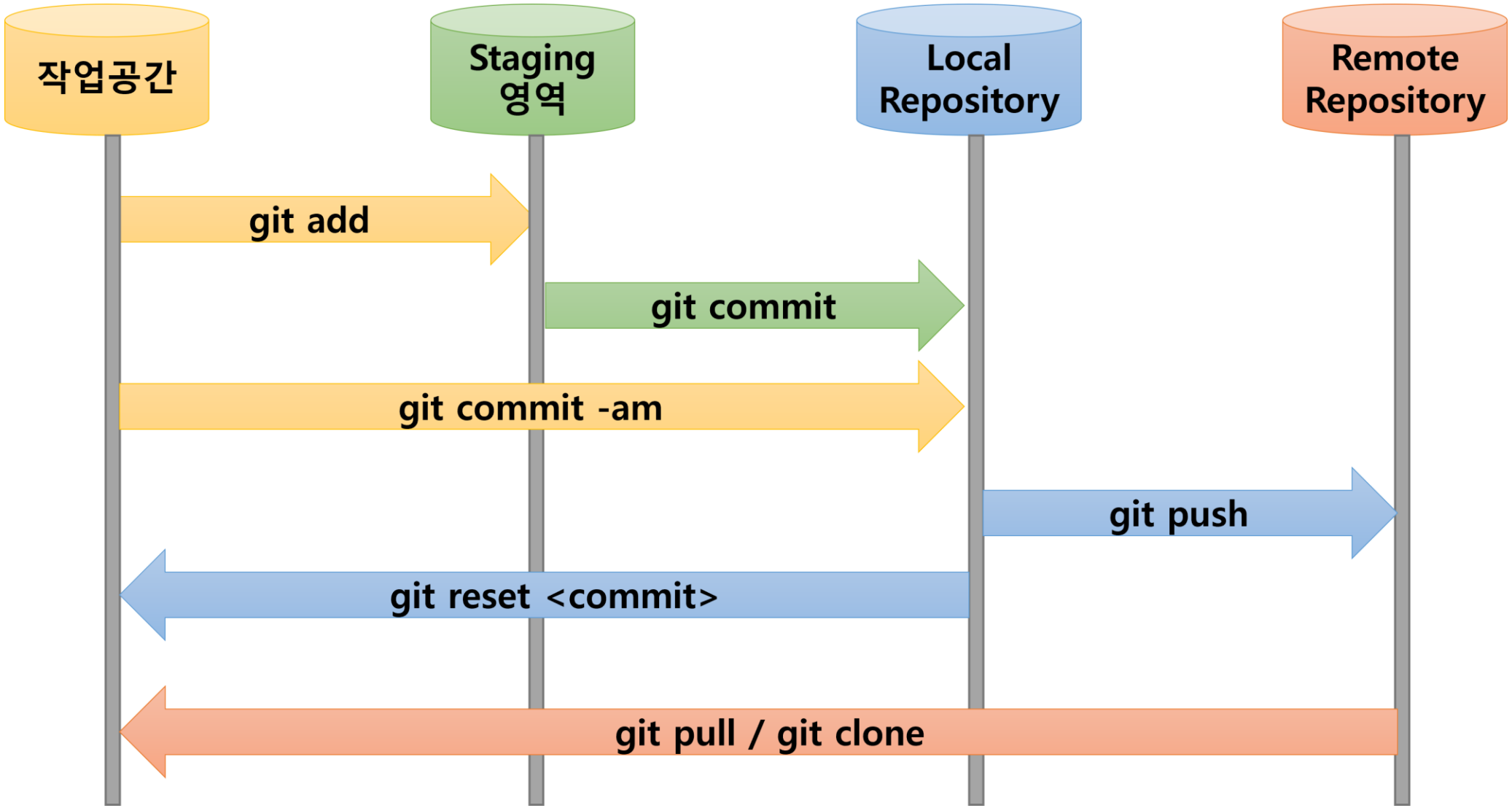
- index.html
- main.css
- favicon.png
- main.js

- index.html 수정
- main.js 추가
- 프로젝트 생성

Git, 커밋하기 on CLI

- **git status**
- **Untracked files?**
- **git add 가 추천 되네요?**
- **git add .**
- **git status**
- **git commit -m “커밋 메세지”**
- **git log**

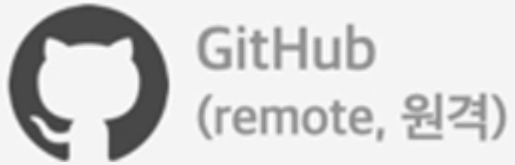
Push!?


작업공간
Staging 영역
Local Repository
Remote Repository
git add
git commit
git commit -am
git push
git reset <commit>
git pull / git clone

GitHub

GitHub
(remote, 원격)

원격 저장소
(Repository)




```
$ git remote add origin https://github.c...
# origin이란 별칭으로 원격 저장소를 연결.
```

git 사용자 (local, 로컬)

GitHub (remote, 원격)

```
$ git push origin master
# origin이란 별칭의 원격 저장소로 버전 내역 전송.
```

master

- index.html
- main.css
- favicon.png
- main.js

원격 저장소 (Repository)

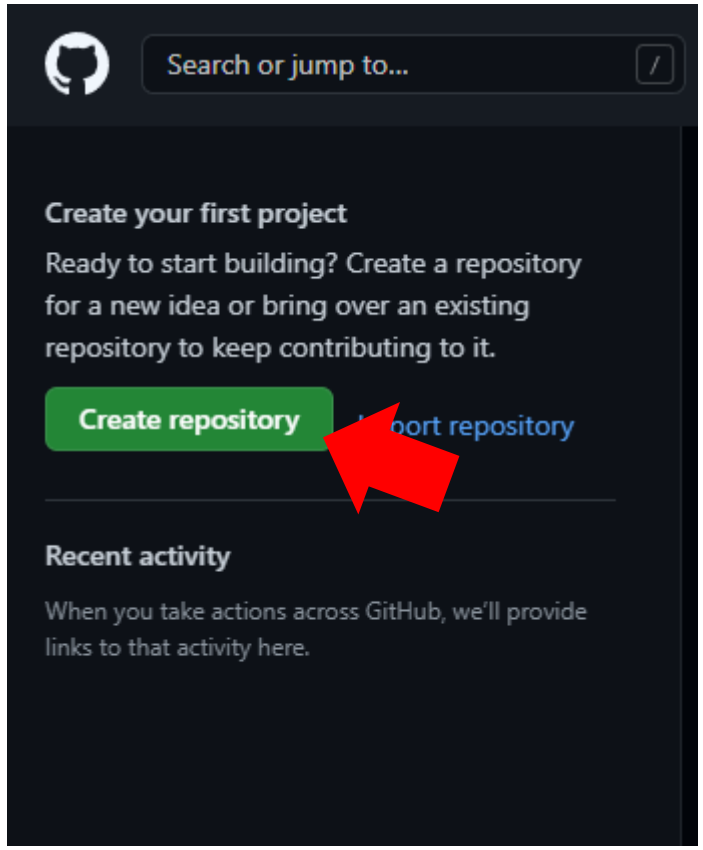
Push, 온라인에 깃을 저장하자!

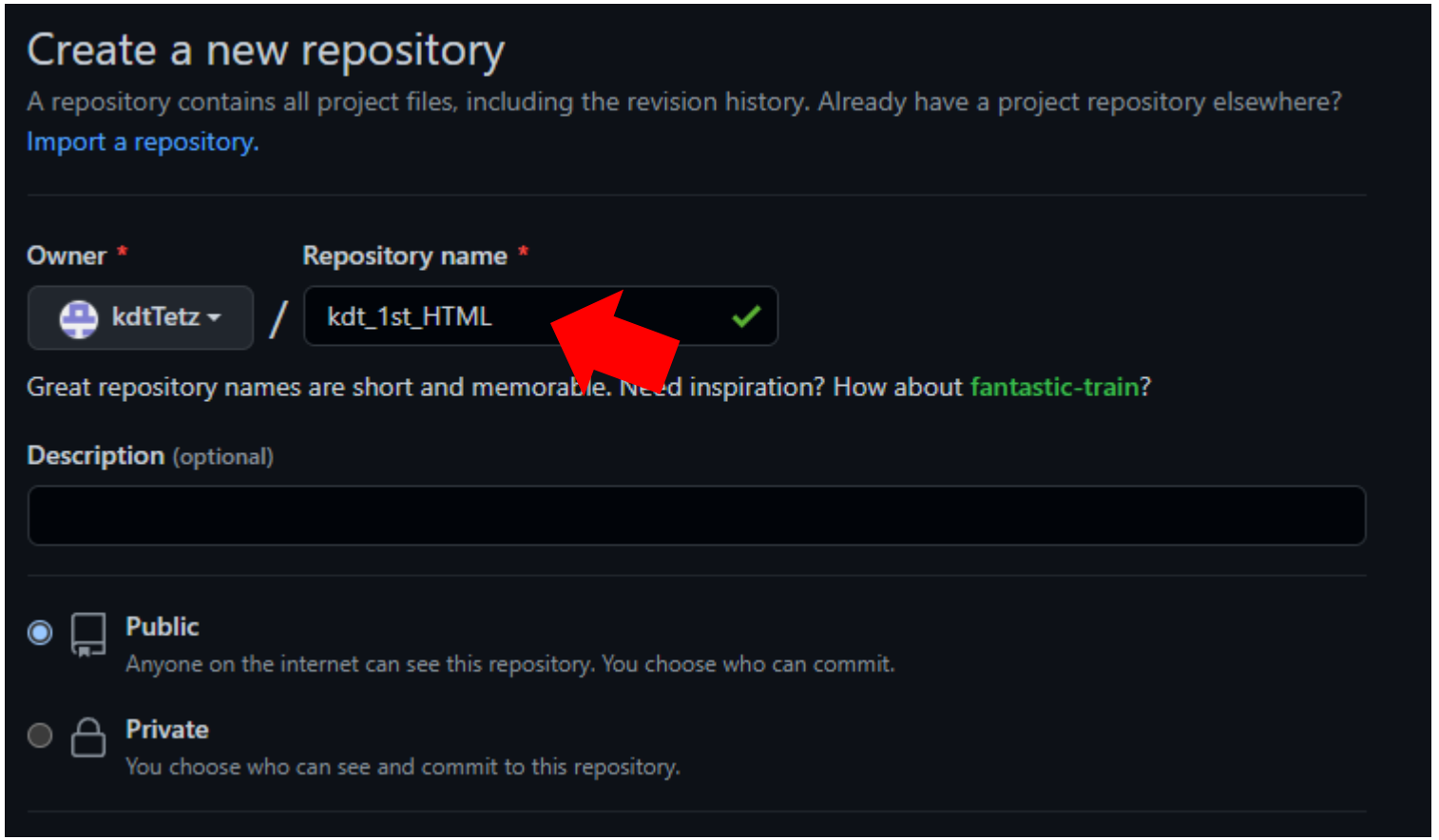
- Commit 은 여러분의 컴퓨터(로컬)에서만 일어나는 일입니다!
- 즉, 온라인에 해당 내역을 올리지 않으면 github은 아무것도 모르는 거죠

Push, 온라인에 깃을 저장하자!

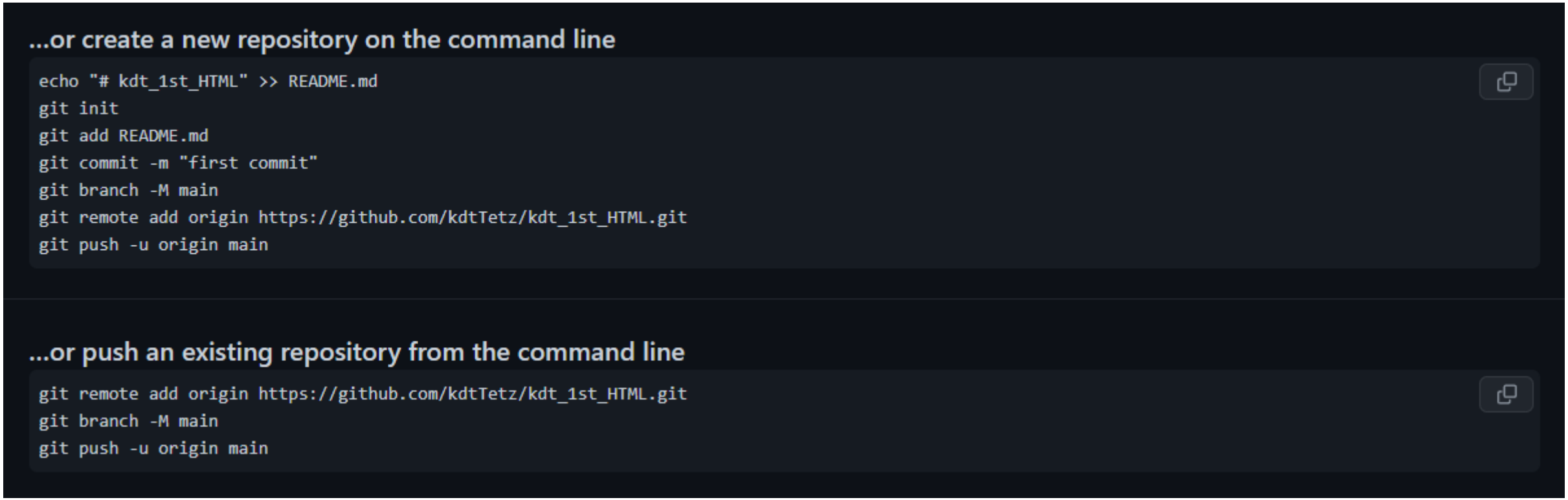
- Github에 여러분의 커밋 내역을 올리려면 반드시 push를 해주어야 합니다!
- Git이 관리하는 폴더 하나(=프로젝트) → github 리포지토리 하나
  - 두 개를 하나에 올리면, 코드의 변화를 추적하는 깃의 특성상 황당하겠죠?
  - 자동차 2대의 블랙 박스를 하나의 블랙 박스에 넣는 것과 비슷합니다!

Github 에 올리기! - Repo 생성





Github 에 올리기! - 안내 따라하기!



- **상황에 맞는 코드를 복사해서 Vscode 에 붙여 넣기!**

Github 에 올리기!(Push)

- echo "# ss" >> README.md
- git init
- git add .
- git commit -m “커밋 메세지"
- git branch -M main
- git remote add origin https://github.com/TetzSpreatics/ss.git
- git push -u origin main

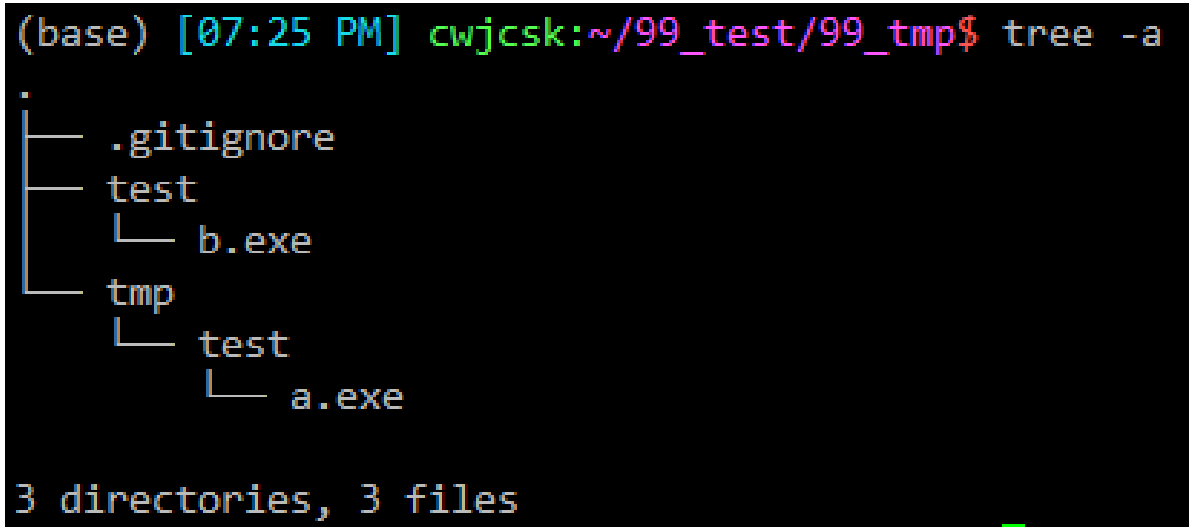
.gitignore

.gitignore?

- Git 버전 관리에서 제외할 파일 목록을 지정하는 파일

- Git 관리에서 특정 파일을 제외하기 위해서는 git에 올리기 전에 .gitignore에 파일 목록을 미리 추가해야 한다.

.gitignore



*.txt → 확장자가 txt로 끝나는 파일 모두 무시

!test.txt → test.txt는 무시되지 않음.

test/ → test 폴더 내부의 모든 파일을 무시 (b.exe와 a.exe 모두 무시)

/test → (현재 폴더) 내에 존재하는 폴더 내부의 모든 파일 무시 (b.exe무시)