Git !(1)

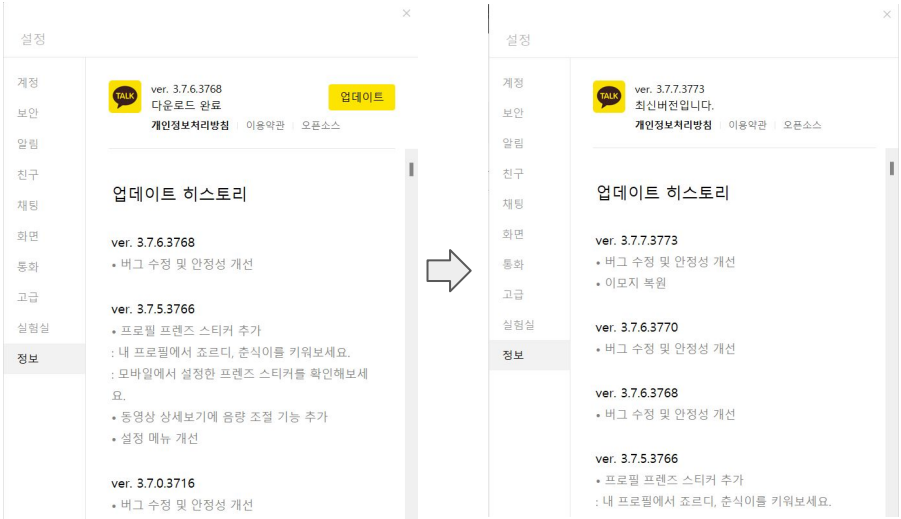
20203312 김현우


git


GitHub

VCS란 ?

버전 관리 시스템
Version Control System



과제_#1

```c
#include <stdio.h>

int main() {
    char operator;
    float num1, num2, result;

    // 연산자와 피연산자 입력 받기
    printf("Enter an operator (+, -, *, /): ");
    scanf("%c", &operator);
    printf("Enter two operands: ");
    scanf("%f %f", &num1, &num2);

    // 연산 수행
    switch(operator) {
        case '+':
            result = num1 + num2;
            printf("%.2f + %.2f = %.2f", num1, num2, result);
            break;
        case '-':
            result = num1 - num2;
            printf("%.2f - %.2f = %.2f", num1, num2, result);
            break;
        case '*':
            result = num1 * num2;
            printf("%.2f * %.2f = %.2f", num1, num2, result);
            break;
        case '/':
            if(num2 != 0) {
                result = num1 / num2;
                printf("%.2f / %.2f = %.2f", num1, num2, result);
            } else {
                printf("Error! Division by zero.");
            }
            break;
        default:
            printf("Error! Invalid operator.");
    }

    return 0;
}
```

1st request

과제_#2

```c
#include <stdio.h>
#include <math.h> // sqrt 함수를 사용하기 위해 필요한 헤더 파일

int main() {
    char operator;
    float num1, num2, result;

    // 연산자와 피연산자 입력 받기
    printf("Enter an operator (+, -, *, ^, <): ");
    scanf(" %c", &operator); // 공백을 포함하여 문자 하나만 입력받기
    printf("Enter two operands: ");
    scanf("%f %f", &num1, &num2);

    // 연산 수행
    switch(operator) {
        case '+':
            result = num1 + num2;
            printf("%.2f + %.2f = %.2f", num1, num2, result);
            break;
        case '-':
            result = num1 - num2;
            printf("%.2f - %.2f = %.2f", num1, num2, result);
            break;
        case '^':
            result = pow(num1, num2); // pow 함수를 사용하여 제곱 연산 수행
            printf("%.2f ^ %.2f = %.2f", num1, num2, result);
            break;
        case '<':
            if(num1 >= 0) {
                result = sqrt(num1); // sqrt 함수를 사용하여 루트 연산 수행
                printf("sqrt(%.2f) = %.2f", num1, result);
            } else {
                printf("Error! Cannot compute square root of a negative number.");
            }
            break;
        default:
            printf("Error! Invalid operator.");
    }

    return 0;
}
```

2nd request

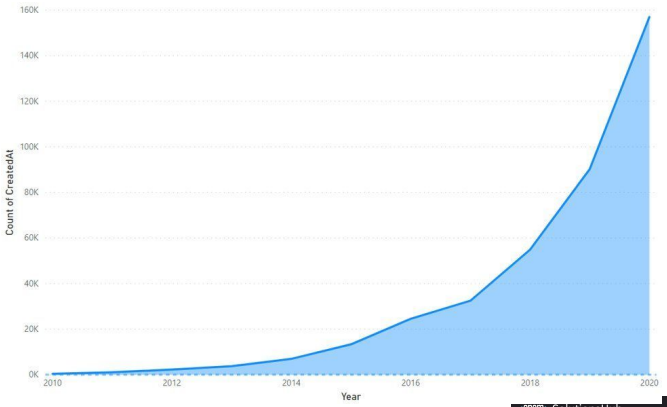
과제_#3

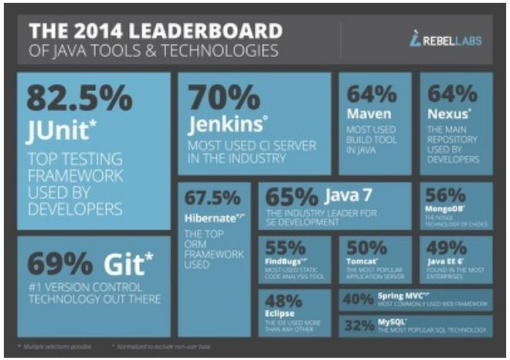
```c
#include <stdio.h>

int main() {
    char operator;
    float num1, num2, result;

    // 연산자와 피연산자 입력 받기
    printf("Enter an operator (+, -, *, /): ");
    scanf("%c", &operator);
    printf("Enter two operands: ");
    scanf("%f %f", &num1, &num2);

    // 연산 수행
    switch(operator) {
        case '+':
            result = num1 + num2;
            printf("%.2f + %.2f = %.2f", num1, num2, result);
            break;
        case '-':
            result = num1 - num2;
            printf("%.2f - %.2f = %.2f", num1, num2, result);
            break;
        case '*':
            result = num1 * num2;
            printf("%.2f * %.2f = %.2f", num1, num2, result);
            break;
        case '/':
            if(num2 != 0) {
                result = num1 / num2;
                printf("%.2f / %.2f = %.2f", num1, num2, result);
            } else {
                printf("Error! Division by zero.");
            }
            break;
        default:
            printf("Error! Invalid operator.");
    }

    return 0;
}
```

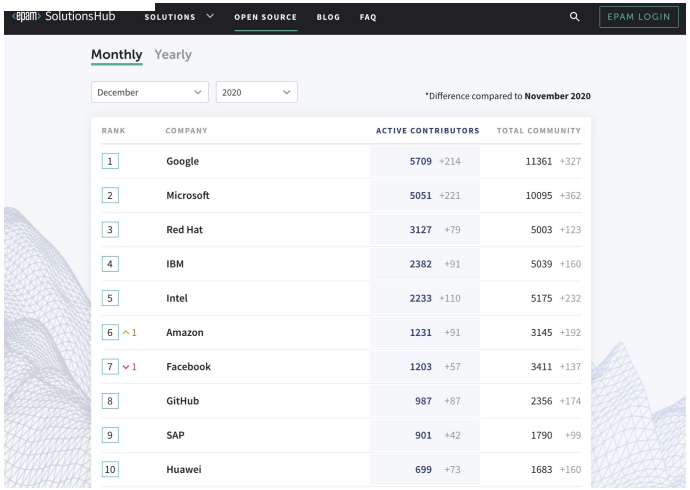
3rd request

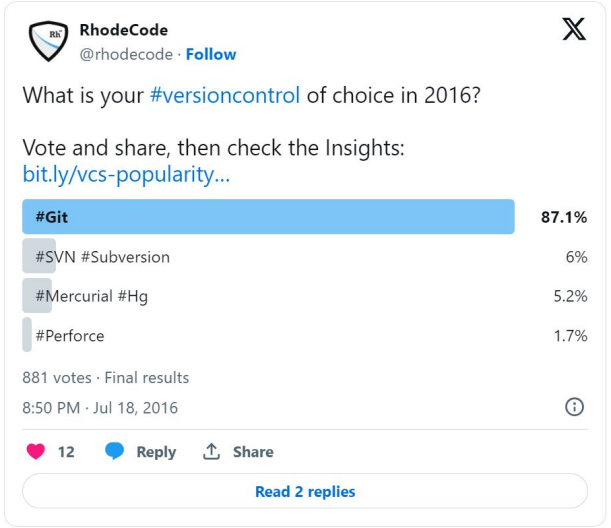
시간

Git을 알아야 하는 이유!

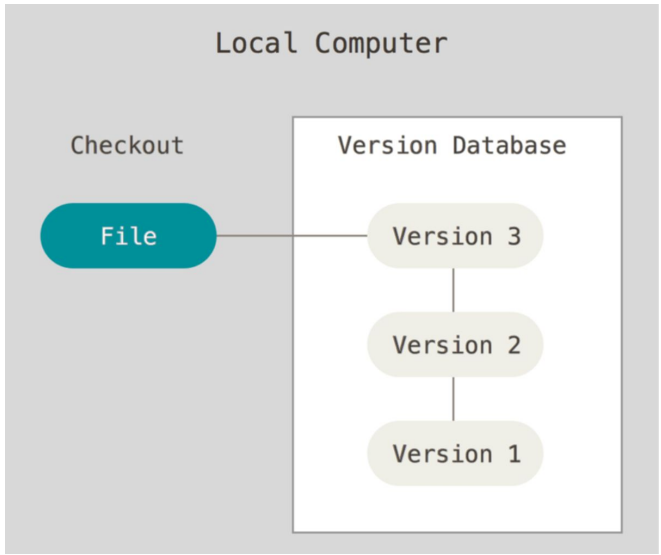






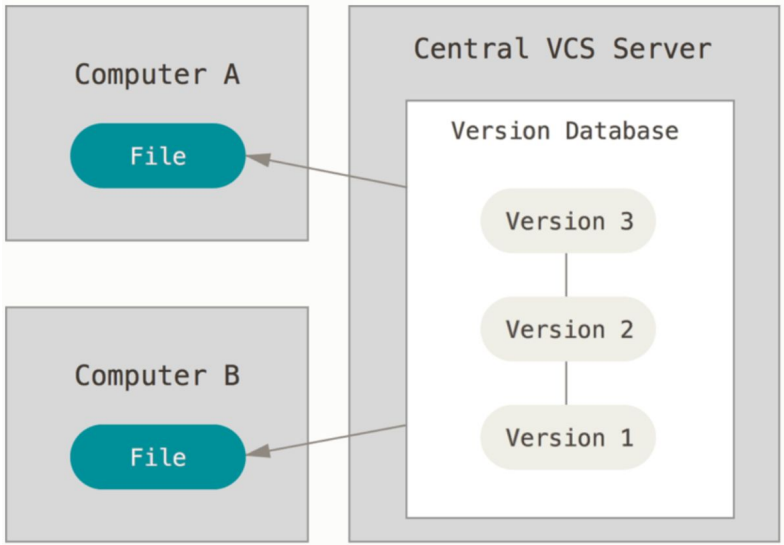


VCS의 구조

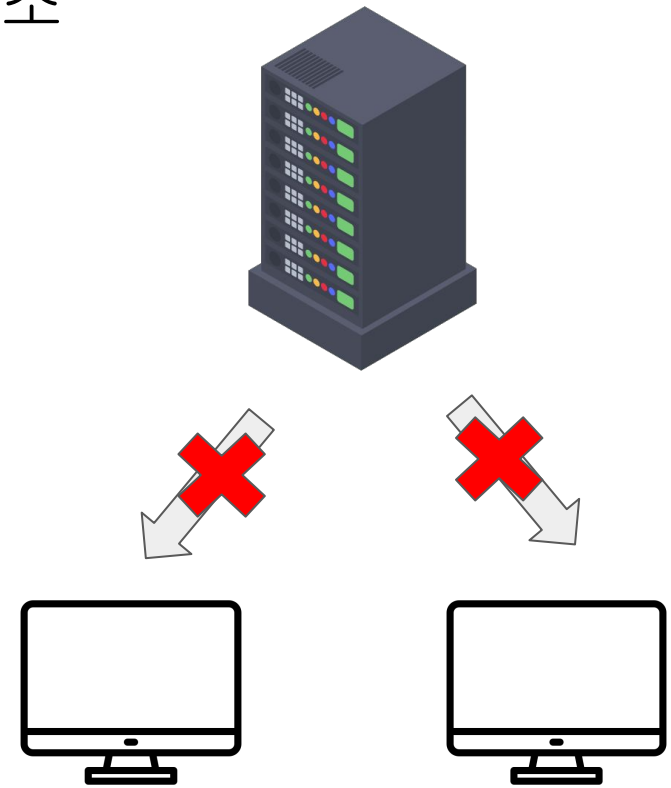


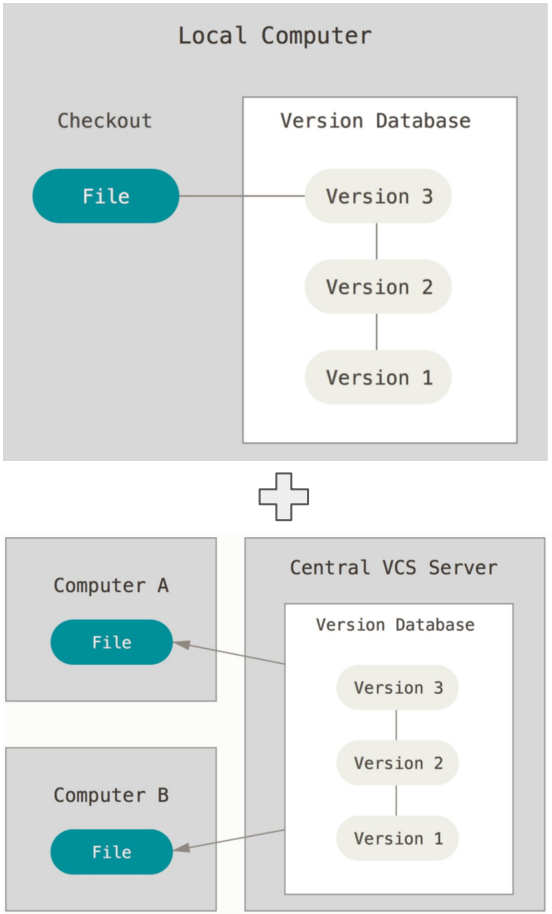
로컬 버전 관리 시스템

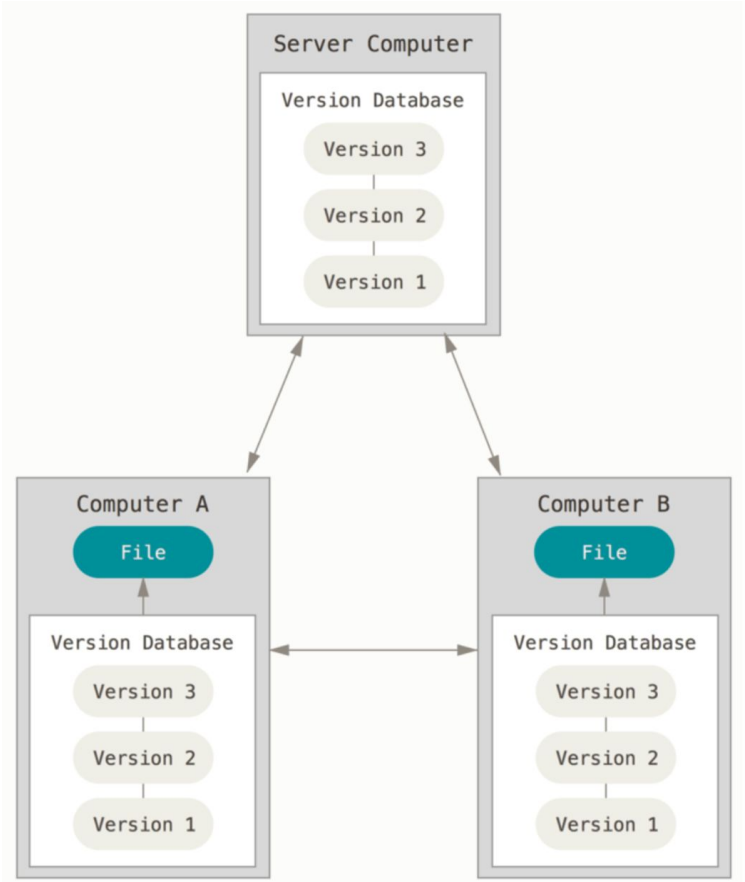
VCS의 구조



중앙 집중식 버전 관리 시스템



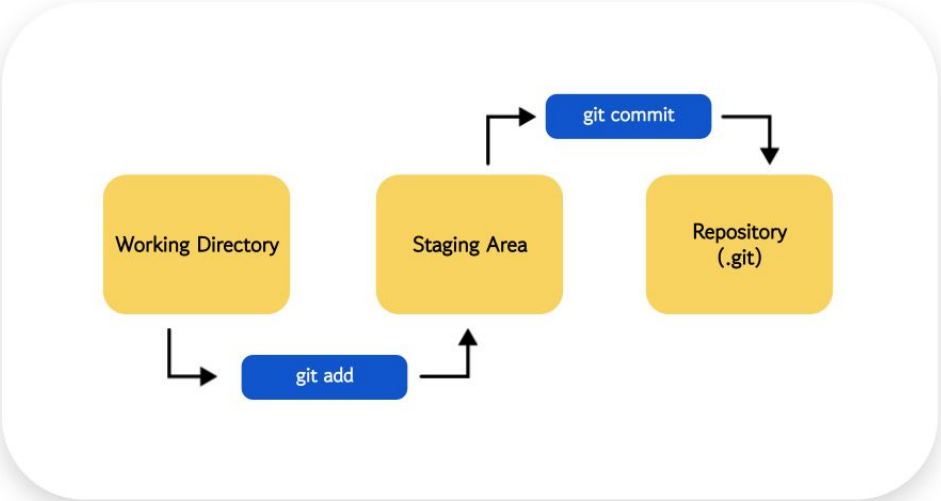

Local Computer
Checkout
File
Version Database
Version 3
Version 2
Version 1
Computer A
File
Computer B
File
Central VCS Server
Version Database
Version 3
Version 2
Version 1


Server Computer
Version Database
Version 3
Version 2
Version 1
Computer A
File
Version Database
Version 3
Version 2
Version 1
Computer B
File
Version Database
Version 3
Version 2
Version 1

분산 버전 관리 시스템

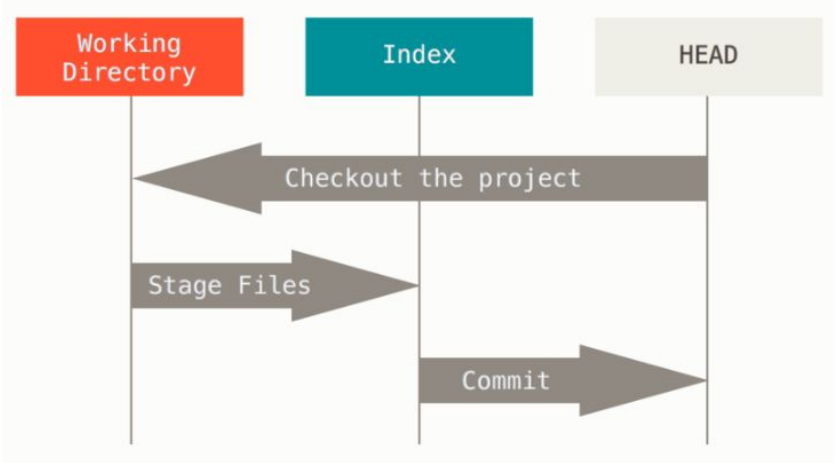
Git의 구조 (Local)

Local Computer
(Working Directory)

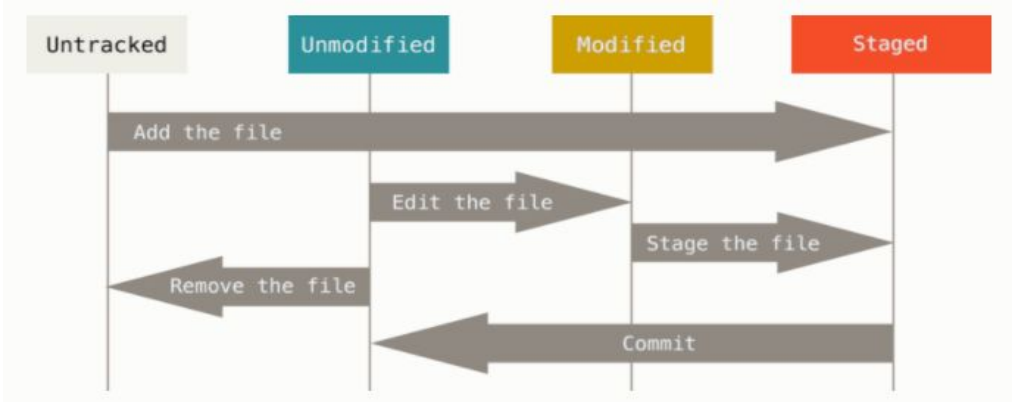


원격 저장소(GitHub)
Remote Repository

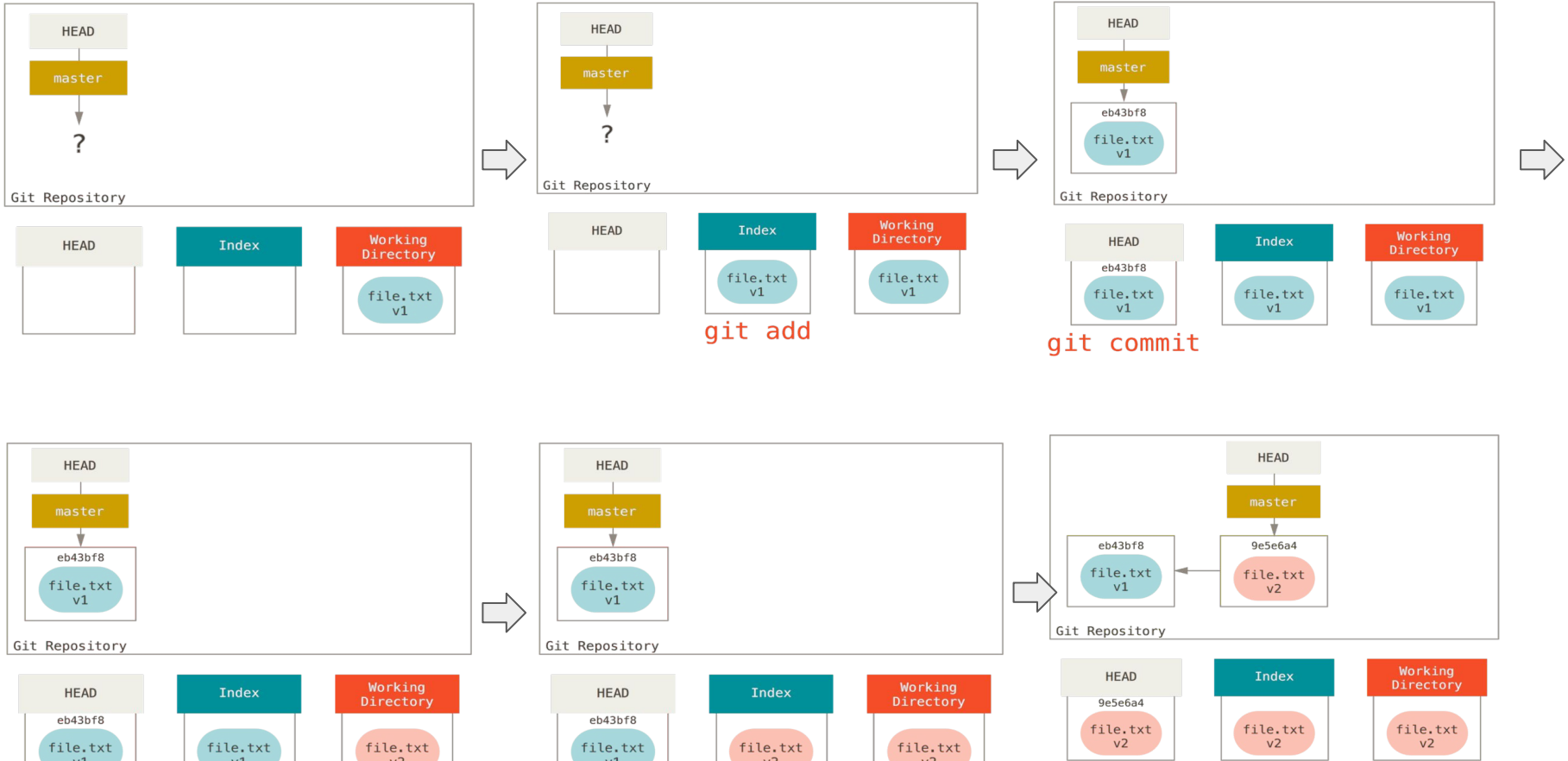
Git의 파일 공간, 파일 상태



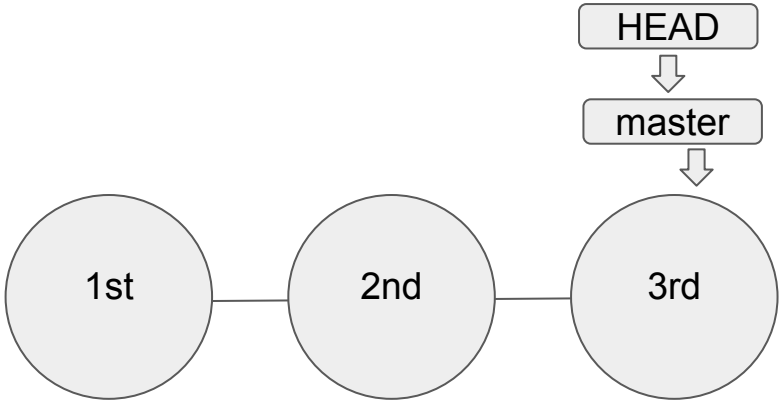
Working Directory 내부



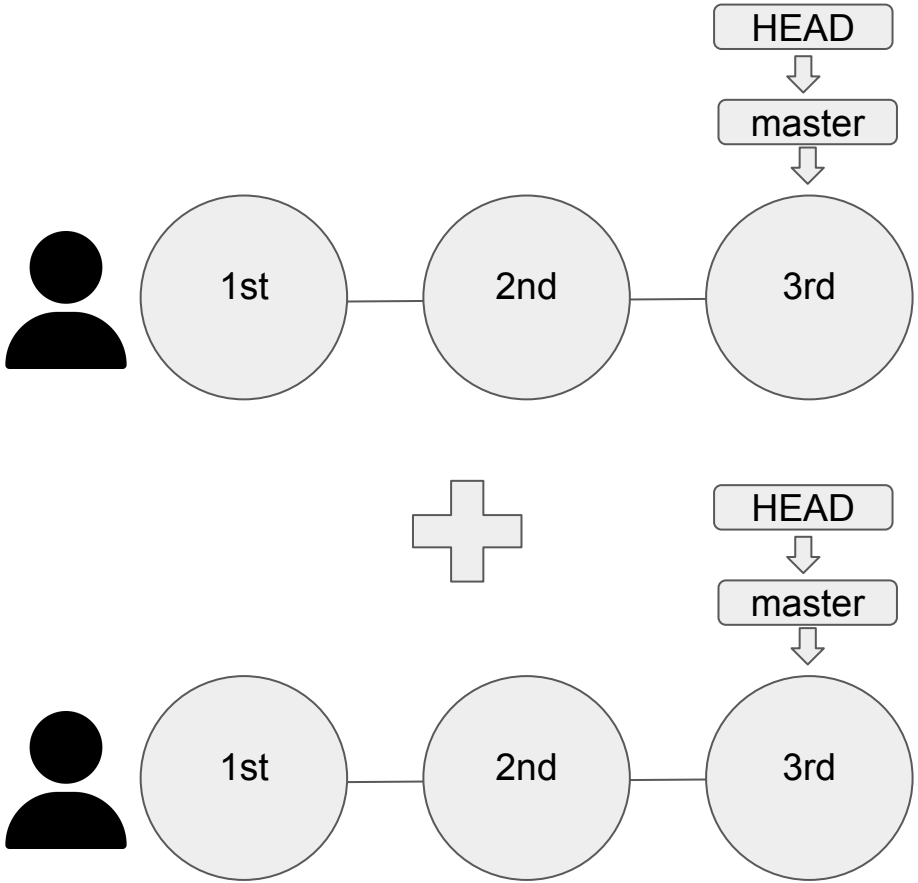
- Untracked
- Tracked
  - Unmodified
  - Modified
  - Staged


HEAD
master
?
Git Repository
HEAD
Index
Working Directory
file.txt v1
git add
eb43bf8
git commit
edit file
9e5e6a4
file.txt v2

Git commit 정리

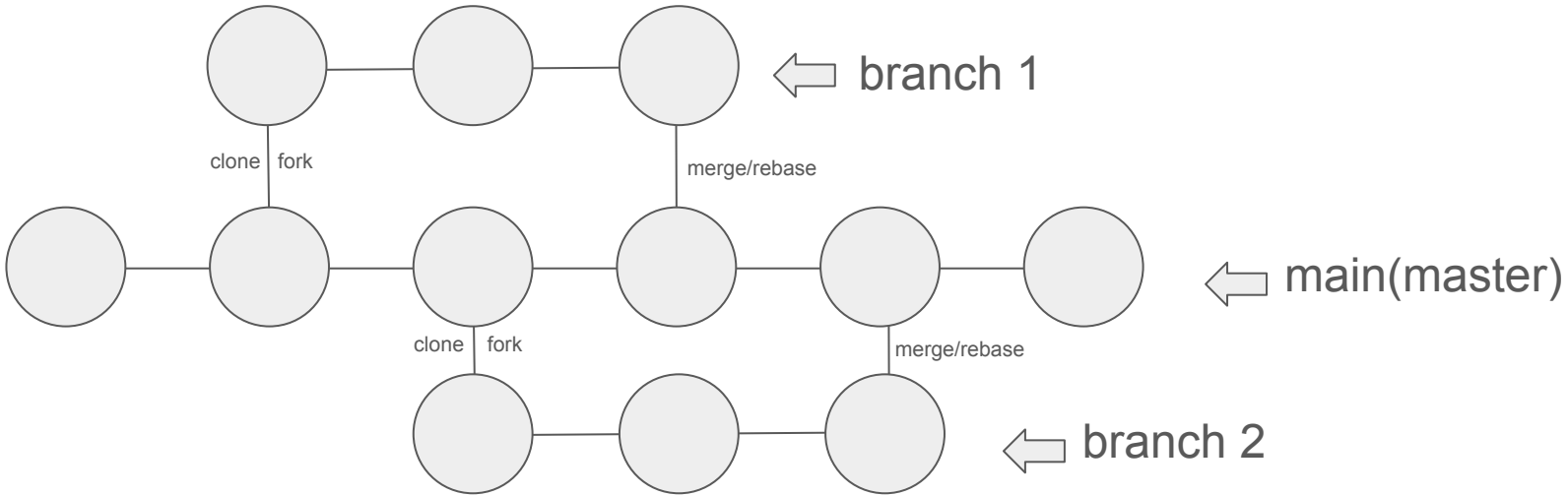


local

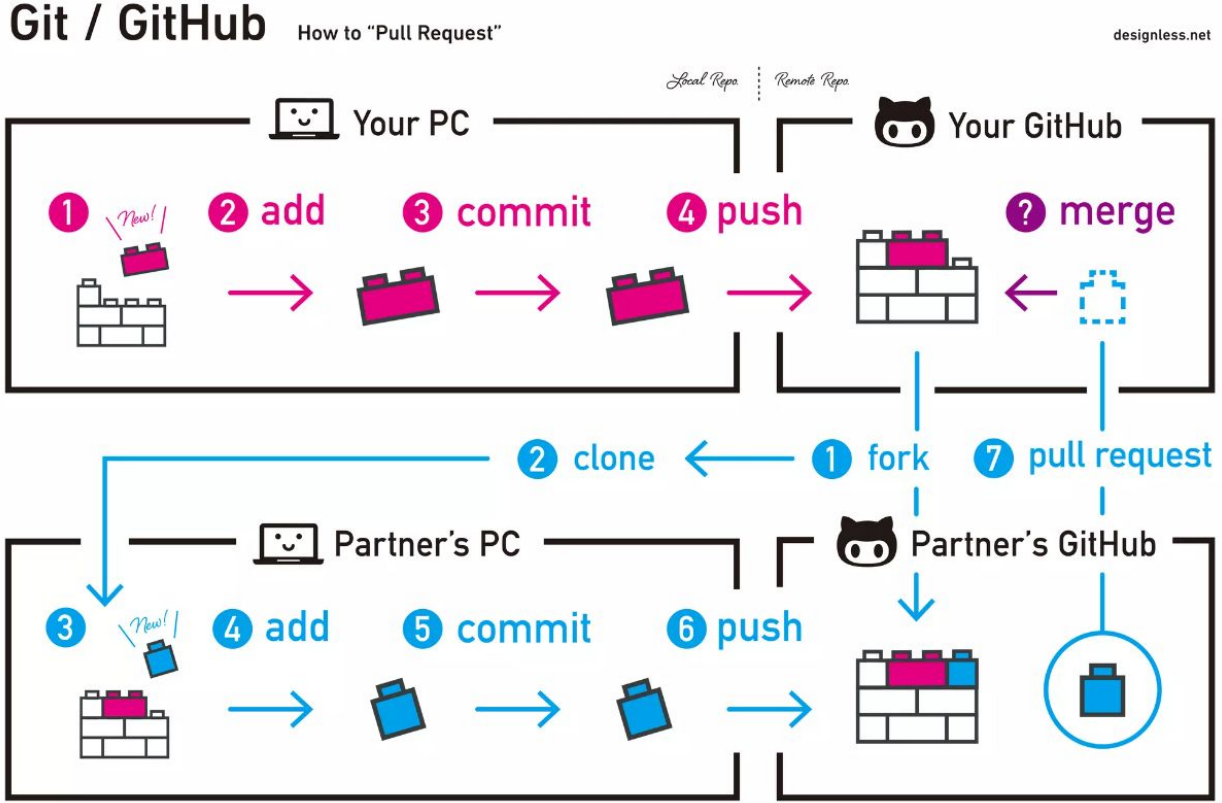


Git branch 개념

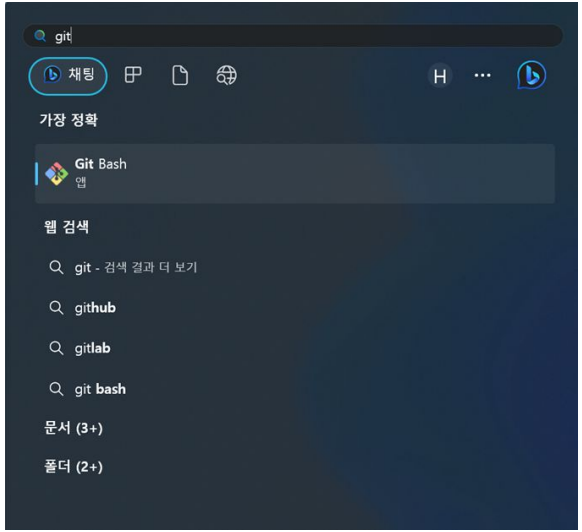
branch란 가지, 분기

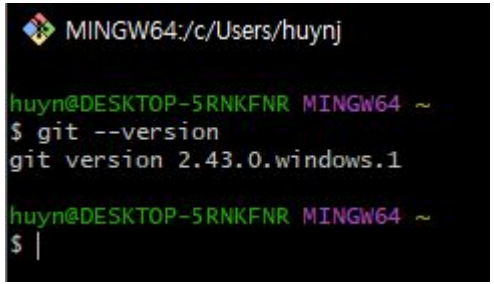


Git 의 구조 (Local + Remote)



Git 설치하기

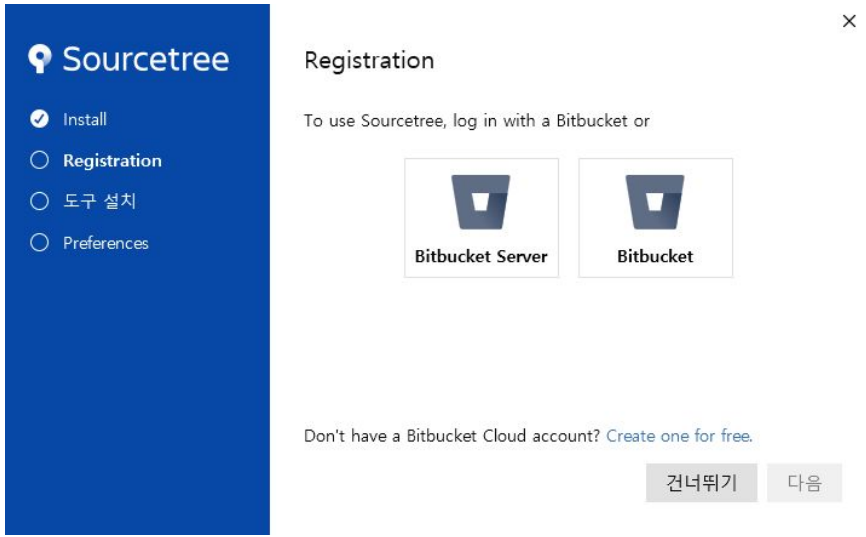


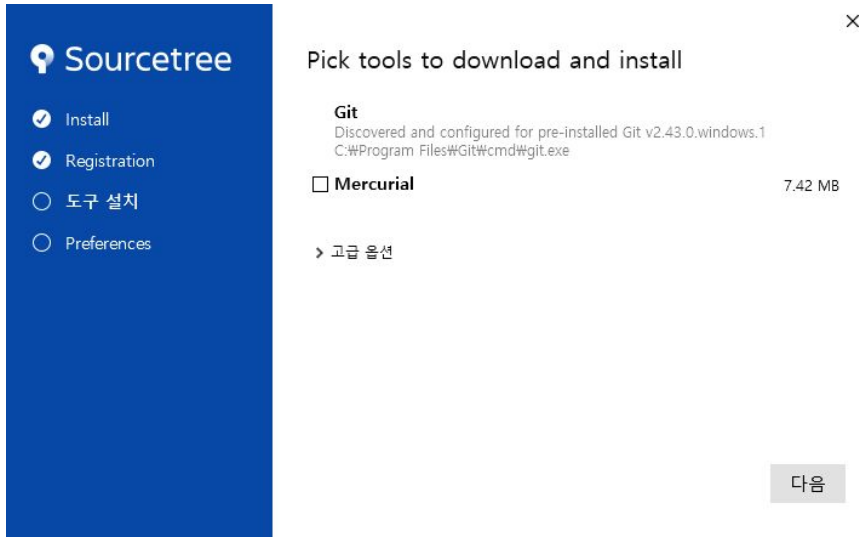


https://git-scm.com/

git –version (대쉬 2개)

Sourcetree 설치

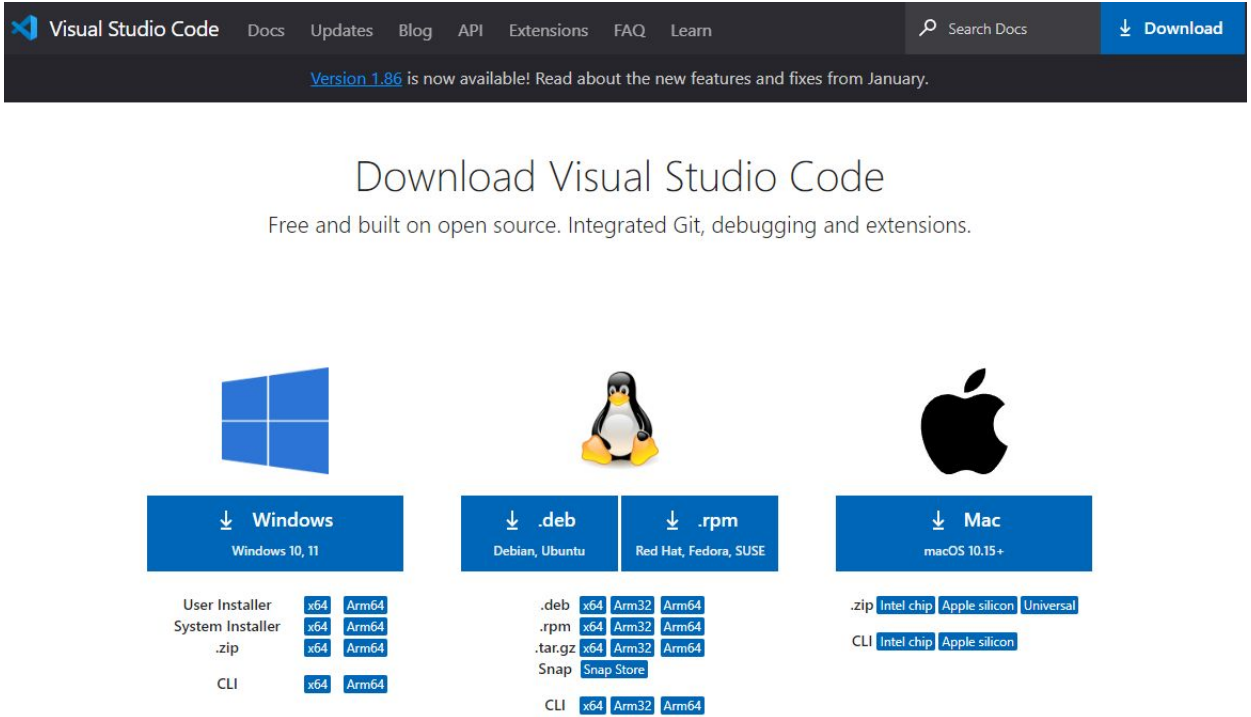




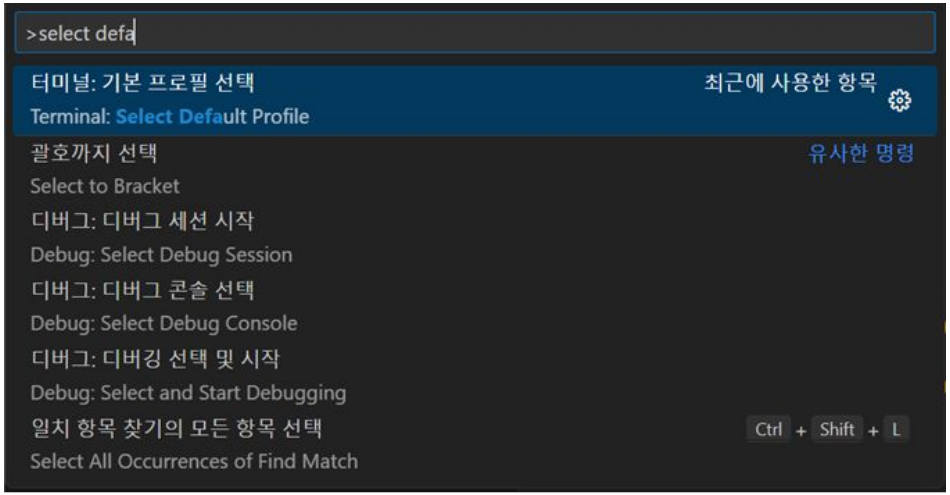
건너뛰기 https://www.sourcetreeapp.com/

체크 해제

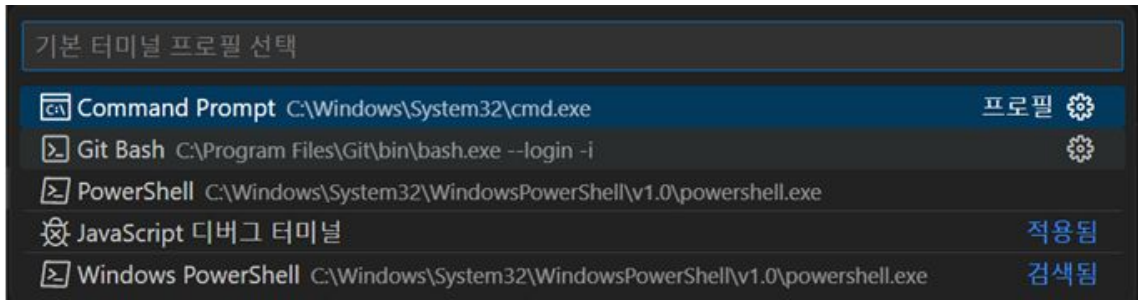
VS Code 설치


Visual Studio Code
Docs
Updates
Blog
API
Extensions
FAQ
Learn
Search Docs
Download
Version 1.86 is now available! Read about the new features and fixes from January.
Download Visual Studio Code
Free and built on open source. Integrated Git, debugging and extensions.
Windows
Windows 10, 11
User Installer x64 Arm64
System Installer x64 Arm64
.zip x64 Arm64
CLI x64 Arm64
.deb
Debian, Ubuntu
.rpm
Red Hat, Fedora, SUSE
.deb x64 Arm32 Arm64
.rpm x64 Arm32 Arm64
.tar.gz x64 Arm32 Arm64
Snap Snap Store
CLI x64 Arm32 Arm64
Mac
macOS 10.15+
.zip Intel chip Apple silicon Universal
CLI Intel chip Apple silicon

VS Code 환경설정

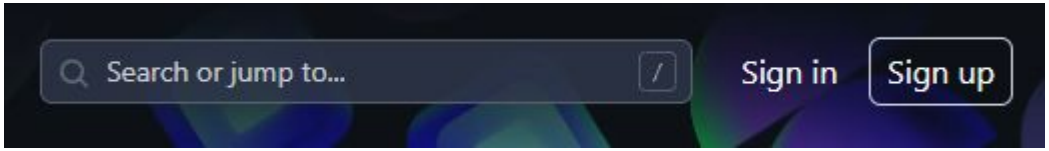


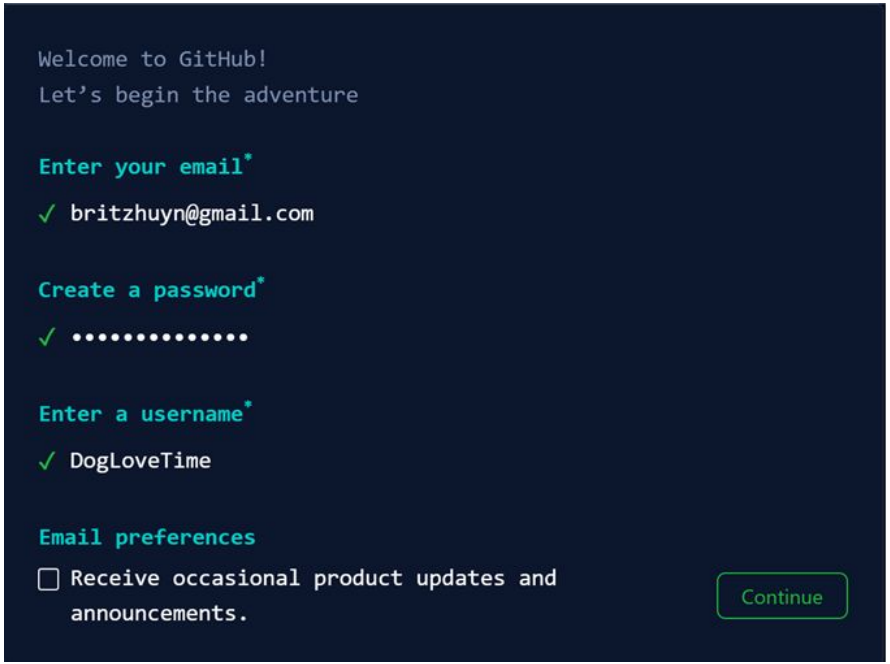
f1 + select default profile

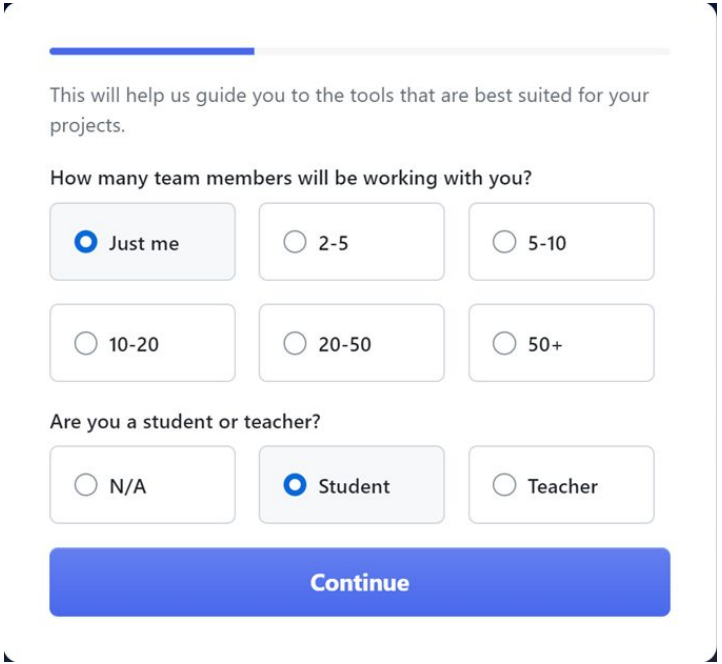


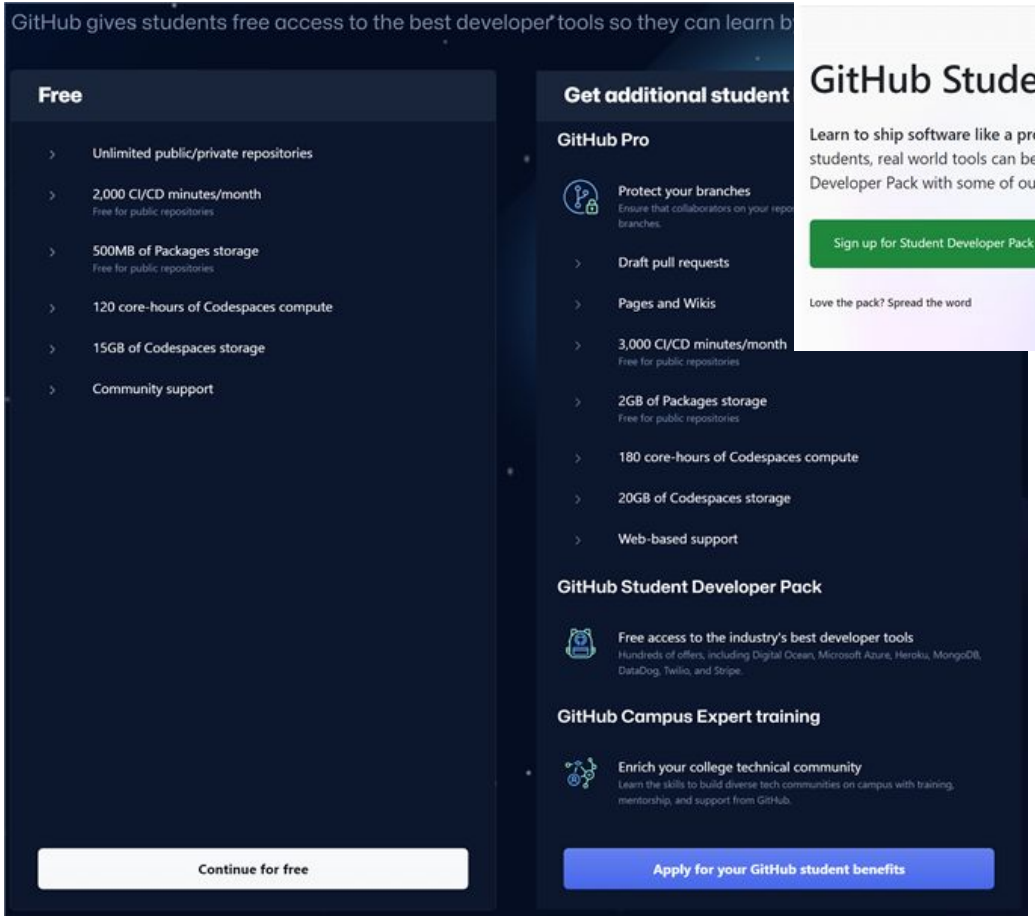
git bash

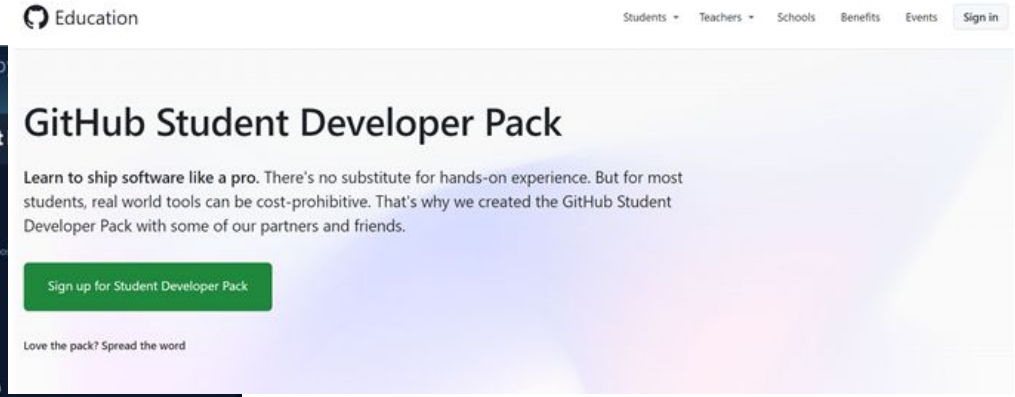
GitHub 가입

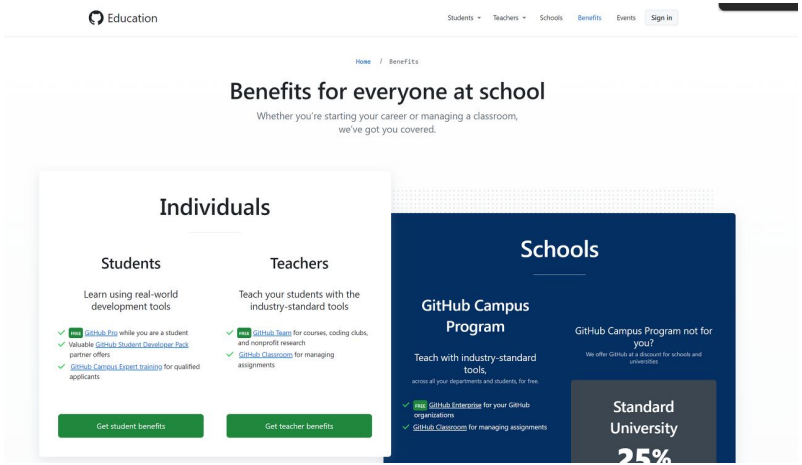

Search or jump to...
/
Sign in
Sign up

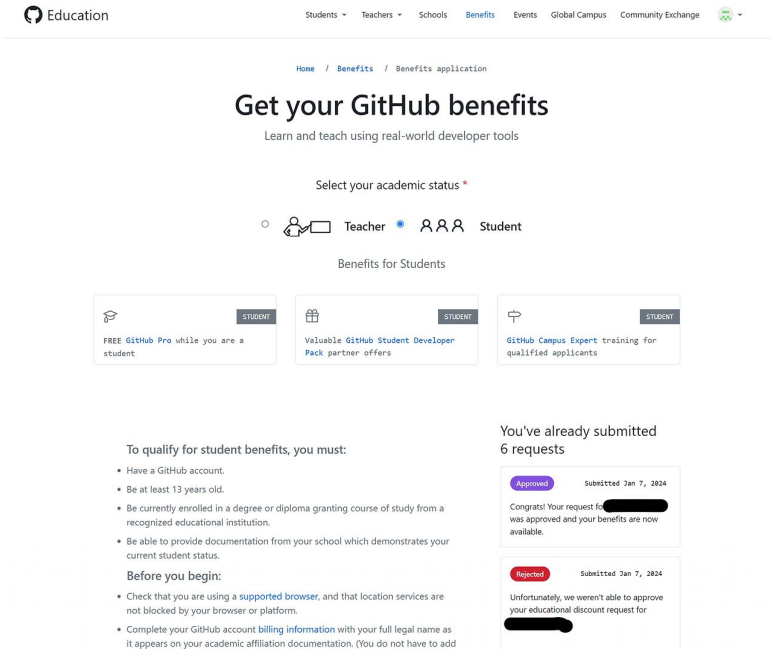

Welcome to GitHub!
Let's begin the adventure
Enter your email*
✓ britzhuyn@gmail.com
Create a password*
✓ ••••••••••••••
Enter a username*
✓ DogLoveTime
Email preferences
Receive occasional product updates and announcements.
Continue

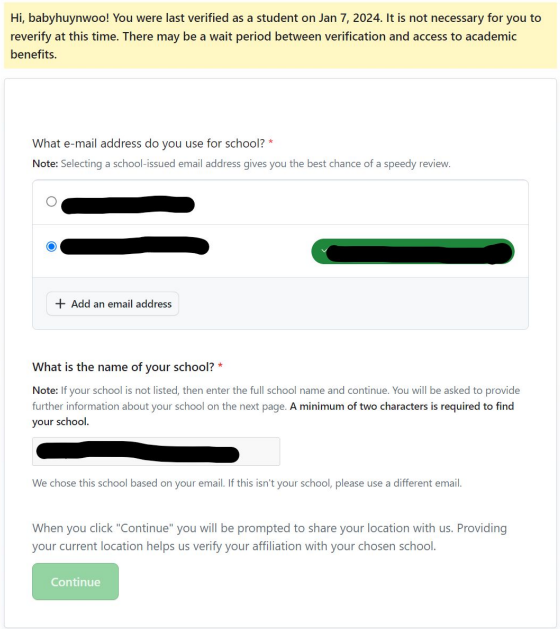

This will help us guide you to the tools that are best suited for your projects.
How many team members will be working with you?
Just me
2-5
5-10
10-20
20-50
50+
Are you a student or teacher?
N/A
Student
Teacher
Continue


GitHub gives students free access to the best developer tools so they can learn b
Free
Unlimited public/private repositories
2,000 CI/CD minutes/month
Free for public repositories
500MB of Packages storage
Free for public repositories
120 core-hours of Codespaces compute
15GB of Codespaces storage
Community support
Continue for free
Get additional student
GitHub Pro
Protect your branches
Draft pull requests
Pages and Wikis
3,000 CI/CD minutes/month
Free for public repositories
2GB of Packages storage
Free for public repositories
180 core-hours of Codespaces compute
20GB of Codespaces storage
Web-based support
GitHub Student Developer Pack
Free access to the industry's best developer tools
Hundreds of offers, including Digital Ocean, Microsoft Azure, Heroku, MongoDB, DataDog, Twilio, and Stripe.
GitHub Campus Expert training
Enrich your college technical community
Learn the skills to build diverse tech communities on campus with training, mentorship, and support from GitHub.
Apply for your GitHub student benefits


Education
Students
Teachers
Schools
Benefits
Events
Sign in
GitHub Student Developer Pack
Learn to ship software like a pro. There's no substitute for hands-on experience. But for most students, real world tools can be cost-prohibitive. That's why we created the GitHub Student Developer Pack with some of our partners and friends.
Sign up for Student Developer Pack
Love the pack? Spread the word


Education
Students
Teachers
Schools
Benefits
Events
Sign in
Home / Benefits
Benefits for everyone at school
Whether you're starting your career or managing a classroom, we've got you covered.
Individuals
Students
Learn using real-world development tools
GitHub Pro while you are a student
Valuable GitHub Student Developer Pack partner offers
GitHub Campus Expert training for qualified applicants
Get student benefits
Teachers
Teach your students with the industry-standard tools
GitHub Team for courses, coding clubs, and nonprofit research
GitHub Classroom for managing assignments
Get teacher benefits
Schools
GitHub Campus Program
Teach with industry-standard tools,
across all your departments and students, for free.
GitHub Enterprise for your GitHub organizations
GitHub Classroom for managing assignments
GitHub Campus Program not for you?
We offer GitHub at a discount for schools and universities
Standard University
25%



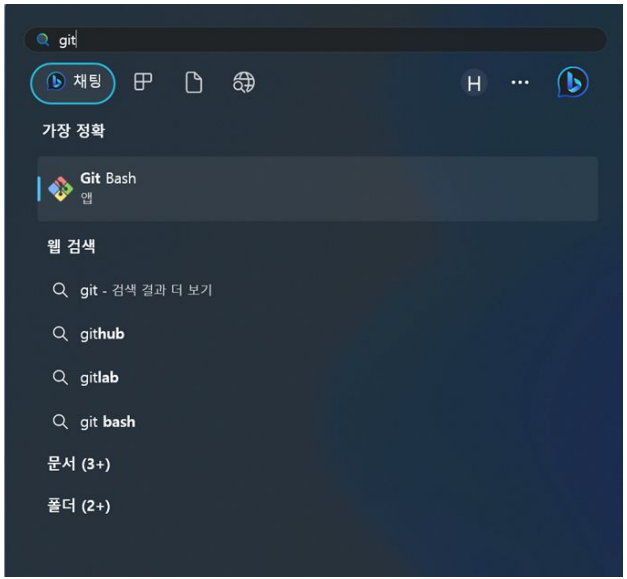


https://ttodday.tistory.com/36

Git !(2)

20203312 김현우

Git 설정



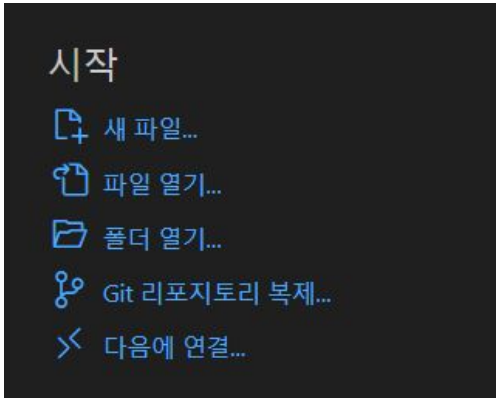
커밋: b522825387a7ffaa767cb3906b8eea1d5b24ef61 [b522825]
상위 항목: ddaa95f109
작성자: khw <huynjoo01@naver.com>
날짜: 2024년 2월 19일 월요일 오후 6:22:43
커밋한 사람: khw

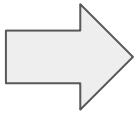
1st commit

커밋: 573a3e965df14d2c3f51d23171227c0f36338283 [573a3e9]
상위 항목: b522825387
작성자: khw <huynjoo01@naver.com>
날짜: 2024년 2월 19일 월요일 오후 6:35:31
커밋한 사람: khw

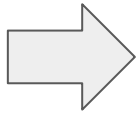
2nd commit

Git - CLI - 시작

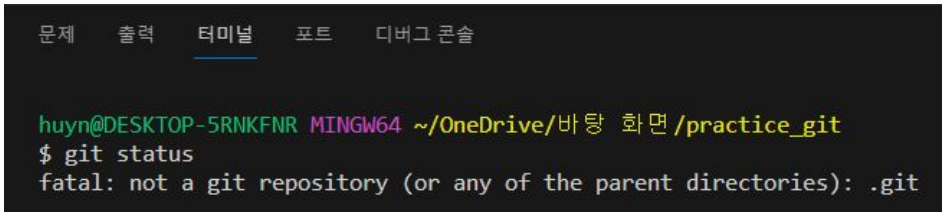

시작
새 파일...
파일 열기...
폴더 열기...
Git 리포지토리 복제...
다음에 연결...

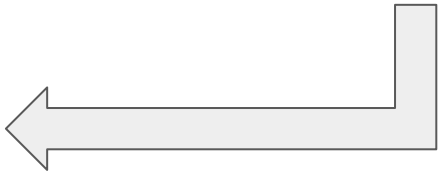


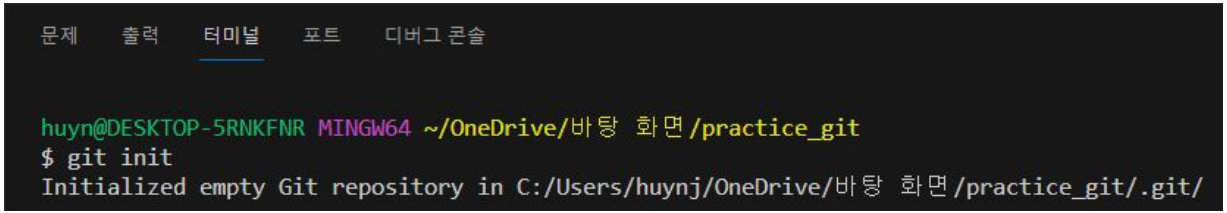

practice_git

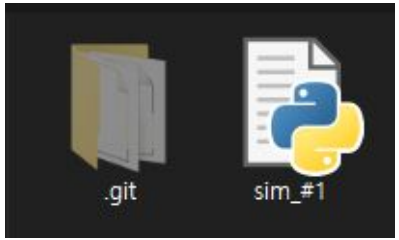



sim_#1


문제
출력
터미널
포트
디버그 콘솔
huyn@DESKTOP-5RNKFNR MINGW64 ~/OneDrive/바탕 화면/practice_git
$ git status
fatal: not a git repository (or any of the parent directories): .git

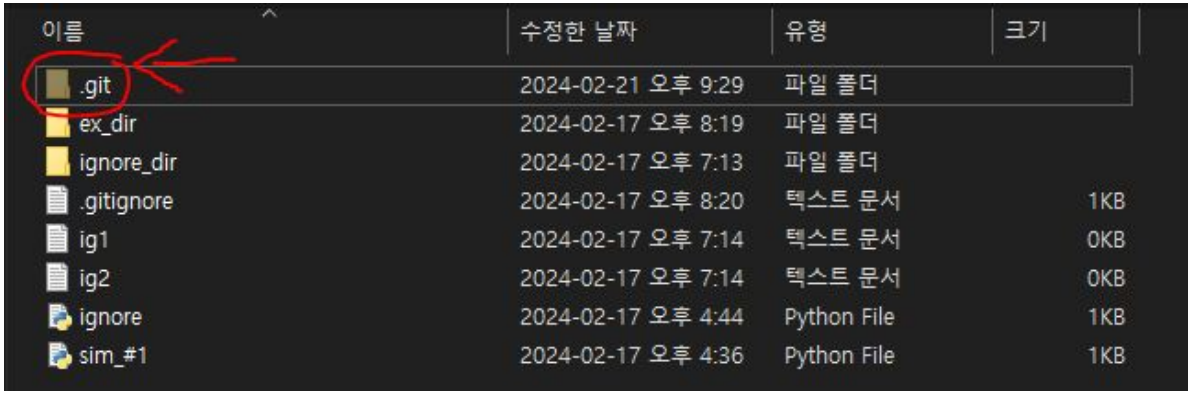



문제
출력
터미널
포트
디버그 콘솔
huyn@DESKTOP-5RNKFNR MINGW64 ~/OneDrive/바탕 화면/practice_git
$ git init
Initialized empty Git repository in C:/Users/huynj/OneDrive/바탕 화면/practice_git/.git/


.git
sim_#1

Git - CLI - init

```
huyn@DESKTOP-5RNKFNR MINGW64 ~/OneDrive/바탕 화면/practice_git_CLI
$ git init
Initialized empty Git repository in C:/Users/huynj/OneDrive/바탕 화면/practice_git_CLI/.git/
```



이름	수정한 날짜	유형	크기
.git	2024-02-21 오후 9:29	파일 폴더	
ex_dir	2024-02-17 오후 8:19	파일 폴더	
ignore_dir	2024-02-17 오후 7:13	파일 폴더	
.gitignore	2024-02-17 오후 8:20	텍스트 문서	1KB
ig1	2024-02-17 오후 7:14	텍스트 문서	0KB
ig2	2024-02-17 오후 7:14	텍스트 문서	0KB
ignore	2024-02-17 오후 4:44	Python File	1KB
sim_#1	2024-02-17 오후 4:36	Python File	1KB

Git - CLI - 파일 제외

ignore.py > ...

```python
"""
this is my secret file
i want ignore this file
"""
name = "김현우"
id = 20203312
major = "컴퓨터 공학과"
address = "서울특별시 동대문구 회기동 90-1번지 3층 301호"

print("my naem is",name)
print("my school id is ",id)
print("my major is ",major)
print("my address is ", address)
```

ignore.py

.gitignore

문제 출력 터미널 포트 디버그 콘솔

```
huyn@DESKTOP-5RNKFNR MINGW64 ~/OneDrive/바탕 화면/practice_git_CLI (master)
$ git status
On branch master

No commits yet

Untracked files:
  (use "git add <file>..." to include in what will be committed)
        .gitignore
        sim_#1.py

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

Git - CLI - 파일 제외

DESCRIPTION

A `gitignore` file specifies intentionally untracked files that Git should ignore. Files already tracked by Git are not affected; see the NOTES below for details.

Each line in a `gitignore` file specifies a pattern. When deciding whether to ignore a path, Git normally checks `gitignore` patterns from multiple sources, with the following order of precedence, from highest to lowest (within one level of precedence, the last matching pattern decides the outcome):

- Patterns read from the command line for those commands that support them.
- Patterns read from a `.gitignore` file in the same directory as the path, or in any parent directory (up to the top-level of the working tree), with patterns in the higher level files being overridden by those in lower level files down to the directory containing the file. These patterns match relative to the location of the `.gitignore` file. A project normally includes such `.gitignore` files in its repository, containing patterns for files generated as part of the project build.
- Patterns read from `$GIT_DIR/info/exclude`.
- Patterns read from the file specified by the configuration variable `core.excludesFile`.

Which file to place a pattern in depends on how the pattern is meant to be used.

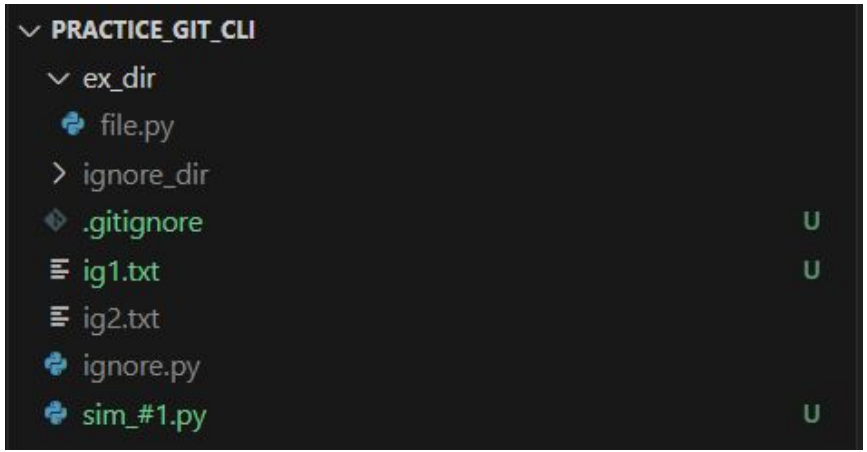
- Patterns which should be version-controlled and distributed to other repositories via clone (i.e., files that all developers will want to ignore) should go into a `.gitignore` file.
- Patterns which are specific to a particular repository but which do not need to be shared with other related repositories (e.g., auxiliary files that live inside the repository but are specific to one user's workflow) should go into the `$GIT_DIR/info/exclude` file.
- Patterns which a user wants Git to ignore in all situations (e.g., backup or temporary files generated by the user's editor of choice) generally go into a file specified by `core.excludesFile` in the user's `~/.gitconfig`. Its default value is $XDG_CONFIG_HOME/git/ignore. If $XDG_CONFIG_HOME is either not set or empty, $HOME/.config/git/ignore is used instead.

The underlying Git plumbing tools, such as `git ls-files` and `git read-tree`, read `gitignore` patterns specified by command-line options, or from files specified by command-line options. Higher-level Git tools, such as `git status` and `git add`, use patterns from the sources specified above.

https://git-scm.com/docs/gitignore

.gitignore

```
1  # 특정 파일 이름으로 ignore 하기 (디렉토리와 하위 디렉토리 모두 적용)
2  ignore.py
3  # 특정 디렉토리 이름으로 ignore 하기
4  ignore_dir/
5  # 특정 디렉토리 내부 특정 파일 ignore 하기
6  ex_dir/file.py
7  # 특정 확장자 파일들을 ignore 하기
8  *.txt
9  # 전체 확장자 파일 중 특정 파일만 ignore 해제 하기 (예외 지정하기)
10 !ig1.txt
```



Git - CLI - add

문제 출력 포트 터미널

스테이지에 올라간 파일

ig1.txt

스테이지에 올라가지 않은 파일

.gitignore

sim_#1.py

```
huyn@DESKTOP-5RNKFNR MINGW64 ~/OneDrive/바탕 화면/practice_git_CLI (master)
$ git add ig1.txt
```

문제 출력 포트 터미널 디버그 콘솔

```
huyn@DESKTOP-5RNKFNR MINGW64 ~/OneDrive/바탕 화면/practice_git_CLI (master)
$ git status
On branch master

No commits yet

Changes to be committed:
  (use "git rm --cached <file>..." to unstage)
        new file:   ig1.txt

Untracked files:
  (use "git add <file>..." to include in what will be cc
        .gitignore
        sim_#1.py
```

git add ‘file_name’

git add . -> 숨김 파일 포함 (일반적)

git add * -> 숨김 파일 제외

```
huyn@DESKTOP-5RNKFNR MINGW64 ~/OneDrive/바탕 화면/practice_git_CLI (master)
$ git add .

huyn@DESKTOP-5RNKFNR MINGW64 ~/OneDrive/바탕 화면/practice_git_CLI (master)
$ git status
On branch master

No commits yet

Changes to be committed:
  (use "git rm --cached <file>..." to unstage)
        new file:   .gitignore
        new file:   ig1.txt
        new file:   sim_#1.py
```

스테이지에 올라간 파일

.gitignore

ig1.txt

sim_#1.py

Git - CLI - commit

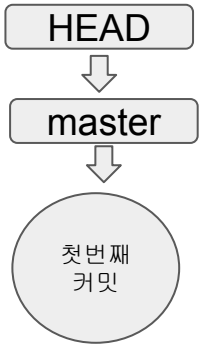
커밋(commit)?

git repository에 잘 포장된 꾸러미

git repository에 체크포인트 중 하나

HASH

꾸러미는 여러 개 만들 수 있지만, 이미지처럼 굴비 엮이듯 연결되어 있다.



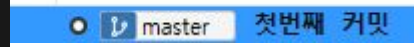
3b9fadd …

입력

```
$ git commit
```

```
첫번째  커밋
# Please enter the commit message for your changes. Lines starting
# with '#' will be ignored, and an empty message aborts the commit.
#
# On branch master
#
# Initial commit
#
# Changes to be committed:
#       new file:   .gitignore
#       new file:   ig2.txt
#       new file:   sim_#1.py
```

```
$ git commit
[master (root-commit) 3b9fadd] 첫번째  커밋
 3 files changed, 20 insertions(+)
 create mode 100644 .gitignore
 create mode 100644 ig2.txt
 create mode 100644 sim_#1.py
```



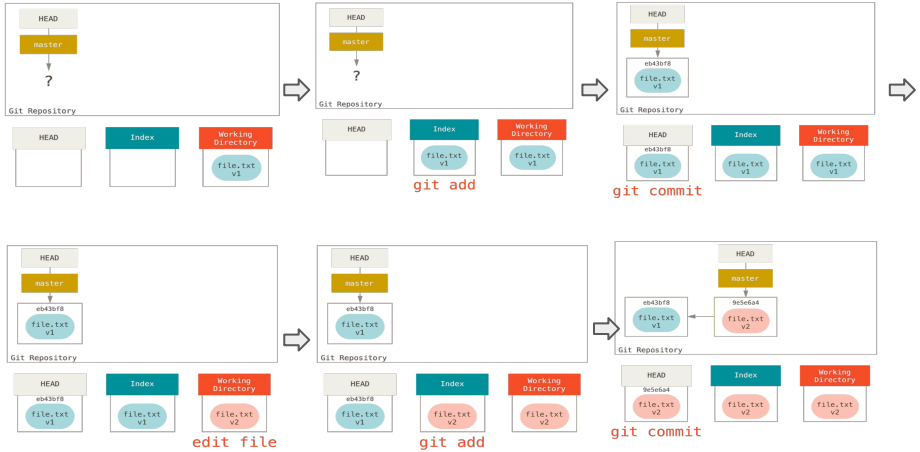
HASH

```
$ git log
commit 3b9faddb8a31ccc0df59066fe6c74f25d8bdee0c (HEAD -> master)
Author: khw <huynjoo01@naver.com>
Date:   Wed Feb 28 20:45:32 2024 +0900

    첫번째  커밋
```

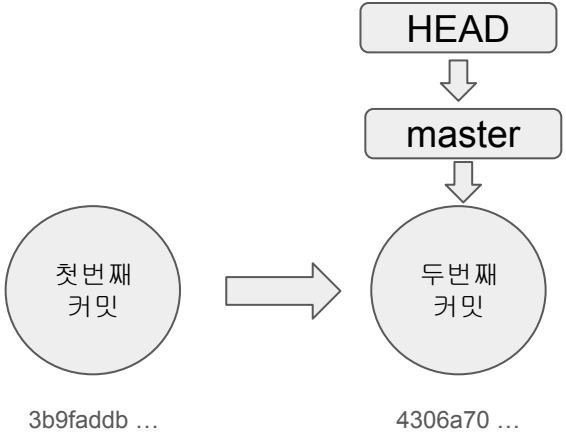
Git - CLI - commit

```
'''

깃을 실습해봅시다

첫번째 커밋


'''

print("first commit")
```

```
'''

깃을 실습해봅시다

두번째 커밋


'''

print("second commit")
```




```
$ git log
commit 4306a708c6dde30faff666c12f6cd487bcb1a9bc (HEAD -> master)
Author: khw <huynjoo01@naver.com>
Date:   Wed Feb 28 20:48:42 2024 +0900

    두번째 커밋

commit 3b9faddb8a31ccc0df59066fe6c74f25d8bdee0c
Author: khw <huynjoo01@naver.com>
Date:   Wed Feb 28 20:45:32 2024 +0900

    첫번째  커밋
```



git 버전 되돌리기(commit 취소)

1. revert

1st.py

```
"""

첫번째 커밋

"""

print("first commit")
```

2nd.py

```
"""

두번째 커밋

"""

print("second commit")
```

3rd.py

```
"""

세번째 커밋

"""

print("third commit")
```

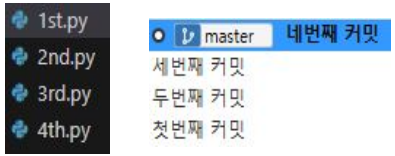
4th.py

```
"""

네번째 커밋

"""

print("fourth commit")
```

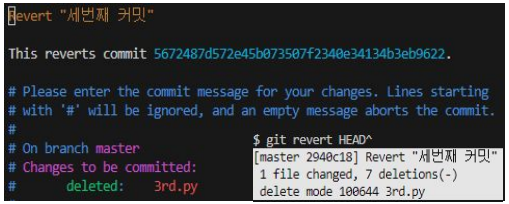
git 버전 되돌리기(commit 취소)

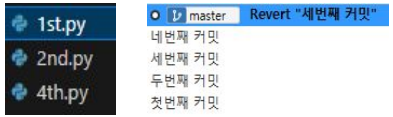
1. revert

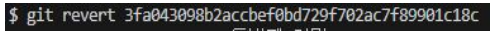
```
$ git log
commit 9598f306b5591b32a115bcd5976178877cffb3ff (HEAD -> master)
Author: khw <huynjoo01@naver.com>
Date:   Wed Feb 28 21:41:31 2024 +0900

    네번째 커밋

commit 5672487d572e45b073507f2340e34134b3eb9622
Author: khw <huynjoo01@naver.com>
Date:   Wed Feb 28 21:40:58 2024 +0900

    세번째 커밋

commit 3fa043098b2accbef0bd729f702ac7f89901c18c
Author: khw <huynjoo01@naver.com>
Date:   Wed Feb 28 21:40:43 2024 +0900

    두번째 커밋

commit 4544b0689f197b8a2b422405294a5e7d2b1aef3e
Author: khw <huynjoo01@naver.com>
Date:   Wed Feb 28 21:40:03 2024 +0900

    첫번째 커밋
```



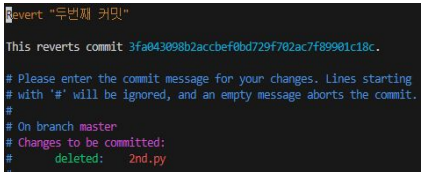
git revert HEAD^



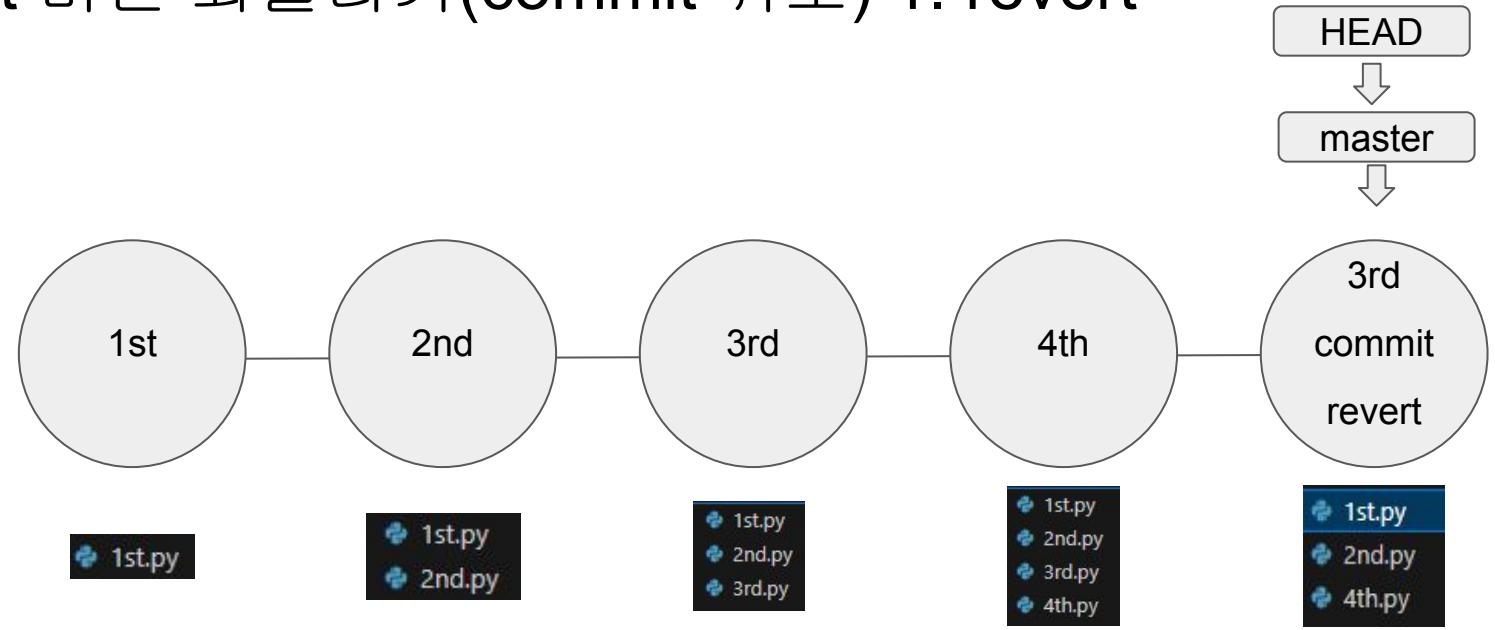




git revert [hash]



git 버전 되돌리기(commit 취소) 1. revert

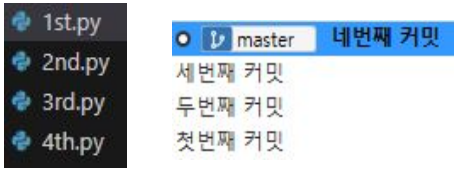


git revert HEAD^

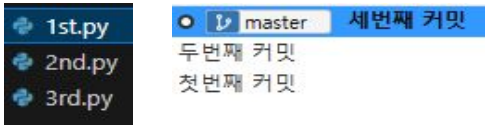
git 버전 되돌리기(commit 취소)

```
$ git log
commit 9598f306b5591b32a115bcd5976178877cffb3ff (HEAD -> master)
Author: khw <huynjoo01@naver.com>
Date:   Wed Feb 28 21:41:31 2024 +0900

    네번째 커밋

commit 5672487d572e45b073507f2340e34134b3eb9622
Author: khw <huynjoo01@naver.com>
Date:   Wed Feb 28 21:40:58 2024 +0900

    세번째 커밋

commit 3fa043098b2accbef0bd729f702ac7f89901c18c
Author: khw <huynjoo01@naver.com>
Date:   Wed Feb 28 21:40:43 2024 +0900

    두번째 커밋

commit 4544b0689f197b8a2b422405294a5e7d2b1aef3e
Author: khw <huynjoo01@naver.com>
Date:   Wed Feb 28 21:40:03 2024 +0900

    첫번째 커밋
```

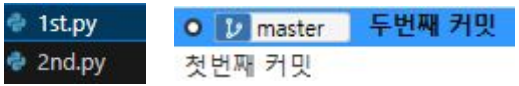
2. reset



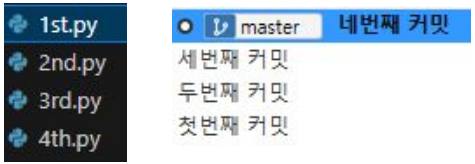

```
$ git reset --hard HEAD^
HEAD is now at 5672487 세번째 커밋
```



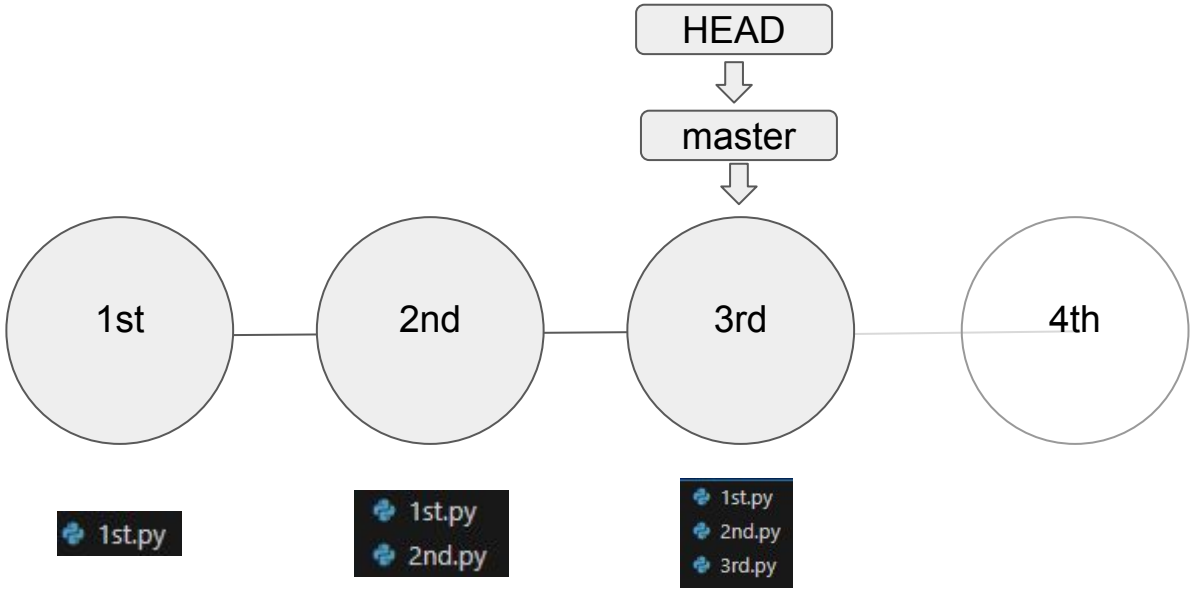

```
$ git reset --hard 3fa043098b2acc
HEAD is now at 3fa0430 두번째 커밋
```

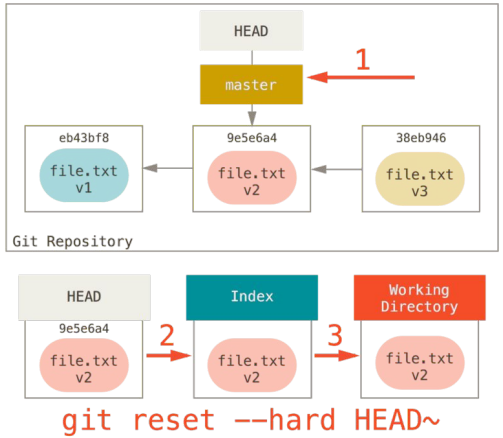



```
$ git reset --hard 9598f306b5591b32a115bcd5976178877cffb3ff
HEAD is now at 9598f30 네번째 커밋
```



git 버전 되돌리기(commit 취소) 2. reset

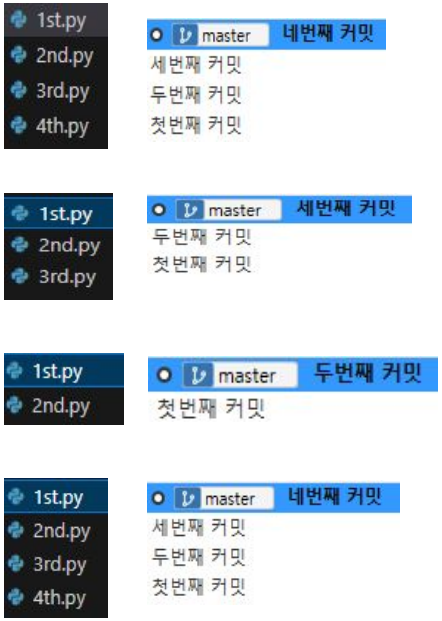




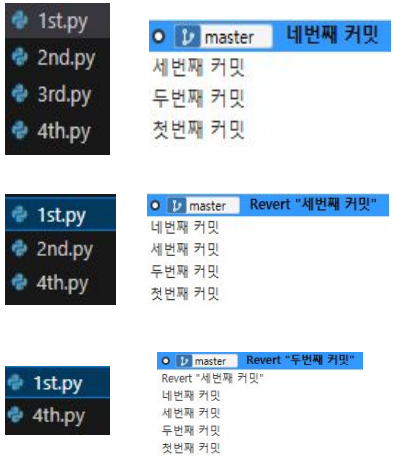
git reset HEAD^

git 버전 되돌리기(commit 취소)

1. reset



2. revert



reset은 HEAD를 조정 and
(커밋 히스토리를 지움)

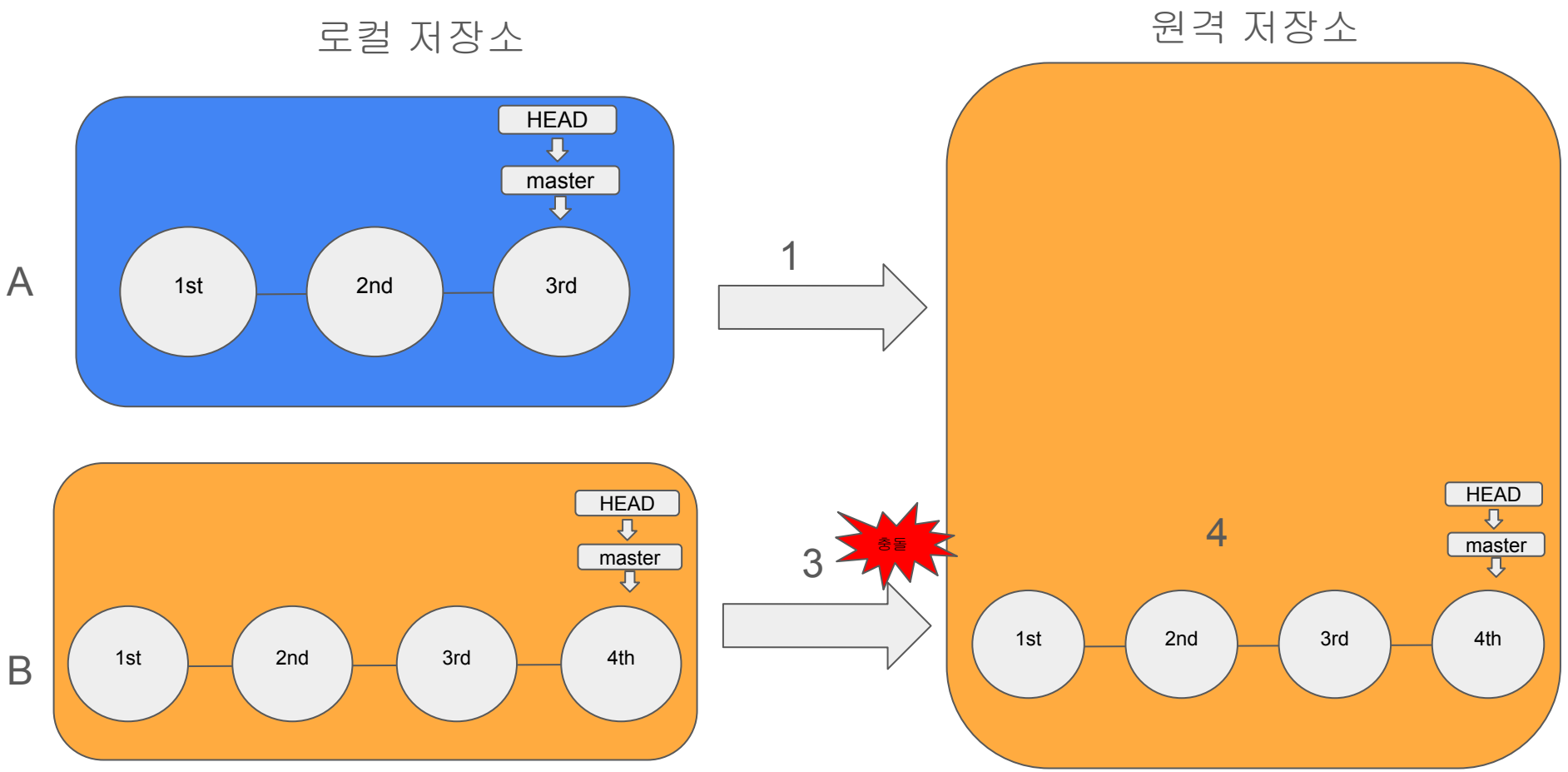
과거의 특정 커밋으로 돌아감

revert는 새로운 커밋을 생성
(커밋 히스토리 유지)

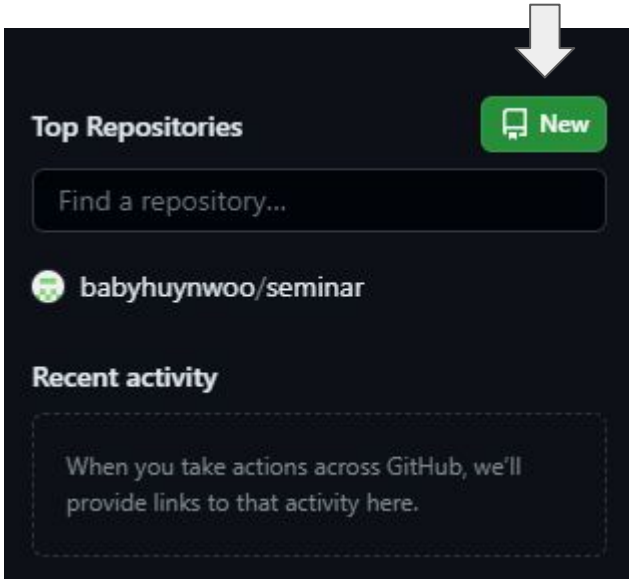
현재에 있으면서 과거의 커밋을 삭제

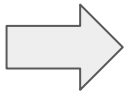
따라서, 혼자 작업할 때에는 reset

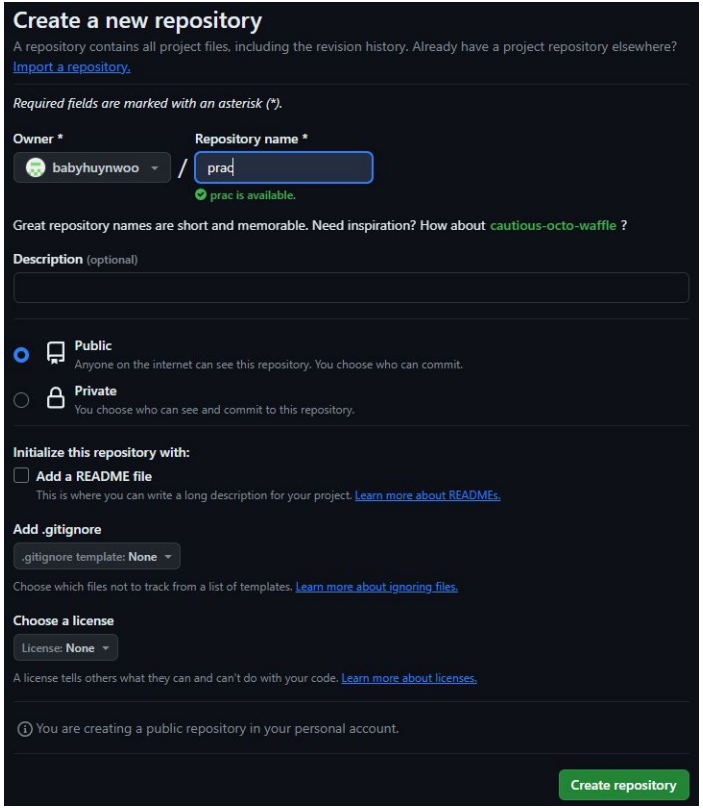
파트너와 같이 작업할 때는 revert


로컬 저장소
원격 저장소
A
HEAD
master
1st
2nd
3rd
1
B
HEAD
master
1st
2nd
3rd
4th
3
충돌
4
HEAD
master
1st
2nd
3rd
4th

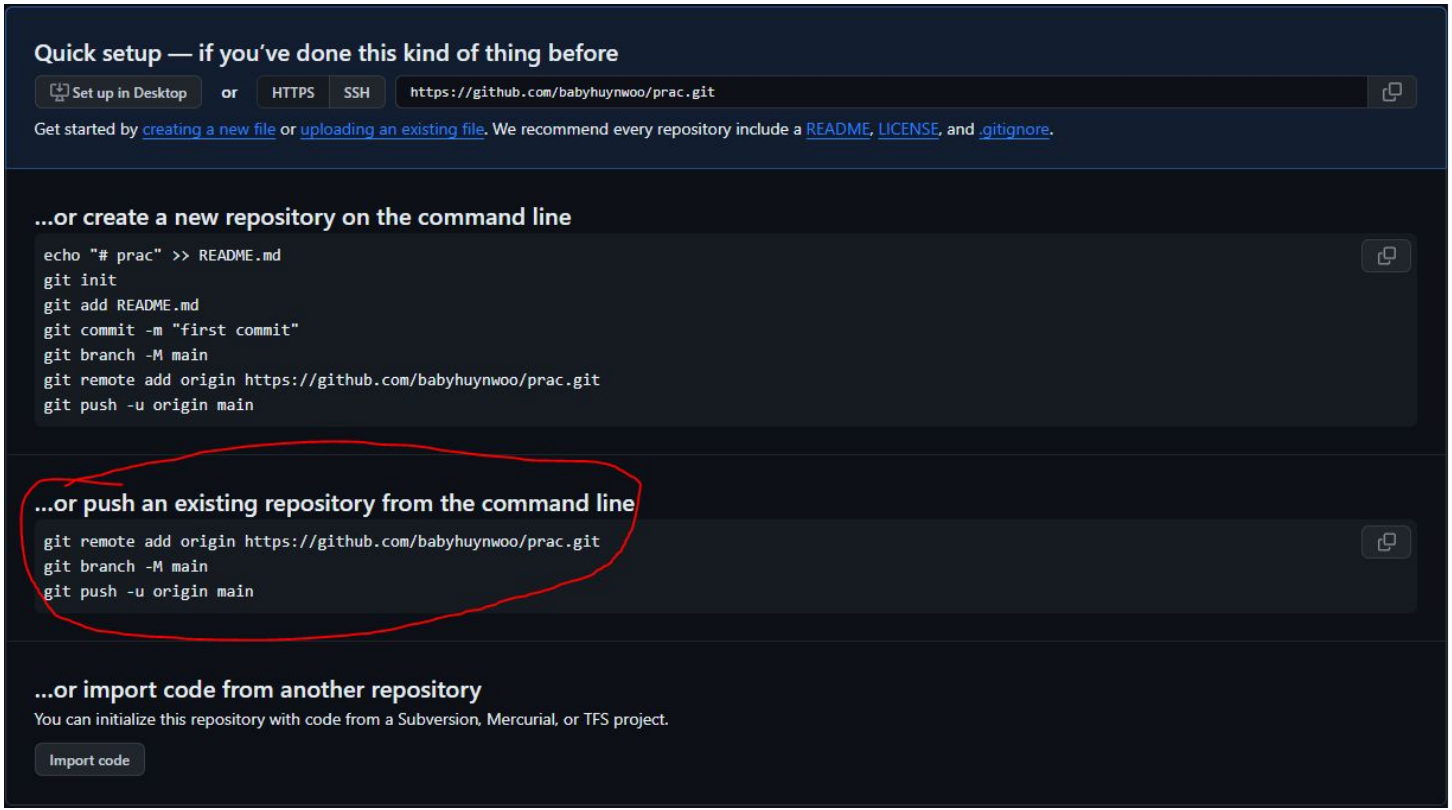
GitHub 연동


Top Repositories
New
Find a repository...
babyhuynwoo/seminar
Recent activity
When you take actions across GitHub, we'll provide links to that activity here.




Create a new repository
A repository contains all project files, including the revision history. Already have a project repository elsewhere?
Import a repository.
Required fields are marked with an asterisk (*).
Owner *
Repository name *
babyhuynwoo
prac
prac is available.
Great repository names are short and memorable. Need inspiration? How about cautious-octo-waffle ?
Description (optional)
Public
Anyone on the internet can see this repository. You choose who can commit.
Private
You choose who can see and commit to this repository.
Initialize this repository with:
Add a README file
This is where you can write a long description for your project. Learn more about READMEs.
Add .gitignore
.gitignore template: None
Choose which files not to track from a list of templates. Learn more about ignoring files.
Choose a license
License: None
A license tells others what they can and can't do with your code. Learn more about licenses.
You are creating a public repository in your personal account.
Create repository

GitHub 연동


Quick setup — if you've done this kind of thing before
Set up in Desktop
or
HTTPS
SSH
https://github.com/babyhuynwoo/prac.git
Get started by creating a new file or uploading an existing file. We recommend every repository include a README, LICENSE, and .gitignore.
...or create a new repository on the command line
echo "# prac" >> README.md
git init
git add README.md
git commit -m "first commit"
git branch -M main
git remote add origin https://github.com/babyhuynwoo/prac.git
git push -u origin main
...or push an existing repository from the command line
git remote add origin https://github.com/babyhuynwoo/prac.git
git branch -M main
git push -u origin main
...or import code from another repository
You can initialize this repository with code from a Subversion, Mercurial, or TFS project.
Import code

GitHub remote, branch, push

```
huyn@DESKTOP-5RNKFNR MINGW64 ~/OneDrive/바탕 화면/practive_git2 (master)
$ git remote add origin https://github.com/babyhuynwoo/prac.git

huyn@DESKTOP-5RNKFNR MINGW64 ~/OneDrive/바탕 화면/practive_git2 (master)
$ git branch -M main

huyn@DESKTOP-5RNKFNR MINGW64 ~/OneDrive/바탕 화면/practive_git2 (main)
$ git push -u origin main
Enumerating objects: 12, done.
Counting objects: 100% (12/12), done.
Delta compression using up to 12 threads
Compressing objects: 100% (9/9), done.
Writing objects: 100% (12/12), 1.08 KiB | 552.00 KiB/s, done.
Total 12 (delta 2), reused 0 (delta 0), pack-reused 0
remote: Resolving deltas: 100% (2/2), done.
To https://github.com/babyhuynwoo/prac.git
 * [new branch]      main -> main
branch 'main' set up to track 'origin/main'.
```

git remote add origin [github 주소]

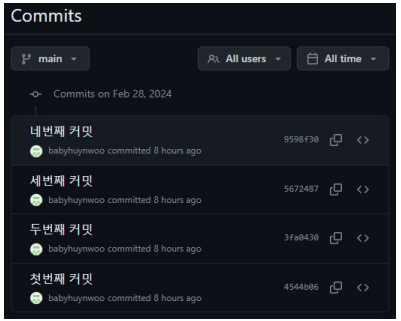
git remote add [원격 저장소 이름] [원격 저장소 주소]

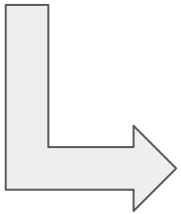
-> 원격 저장소의 이름을 지정

git branch -M [이름] : 메인 branch 이름을 이름으로 변경

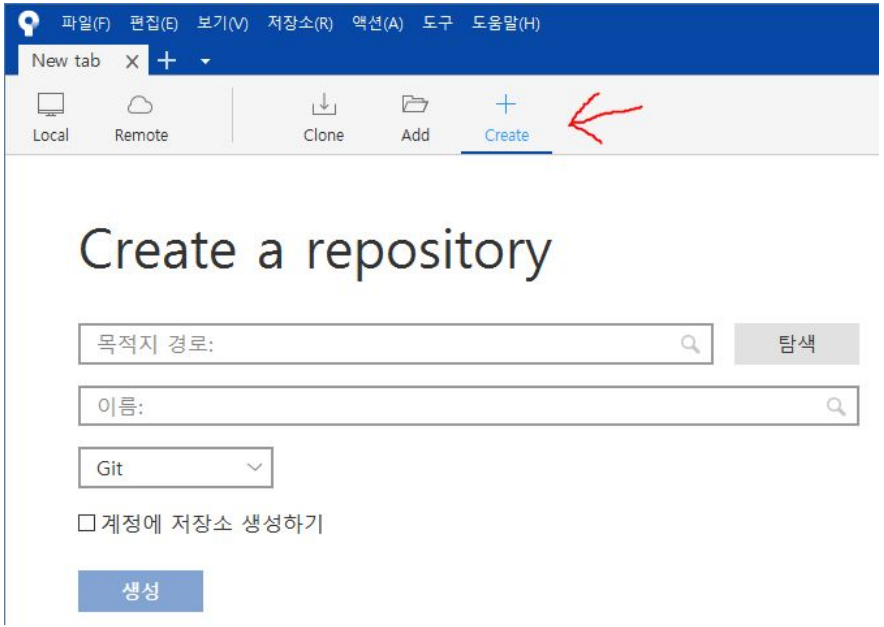
git push (원격 저장소 명) (브랜치 명)

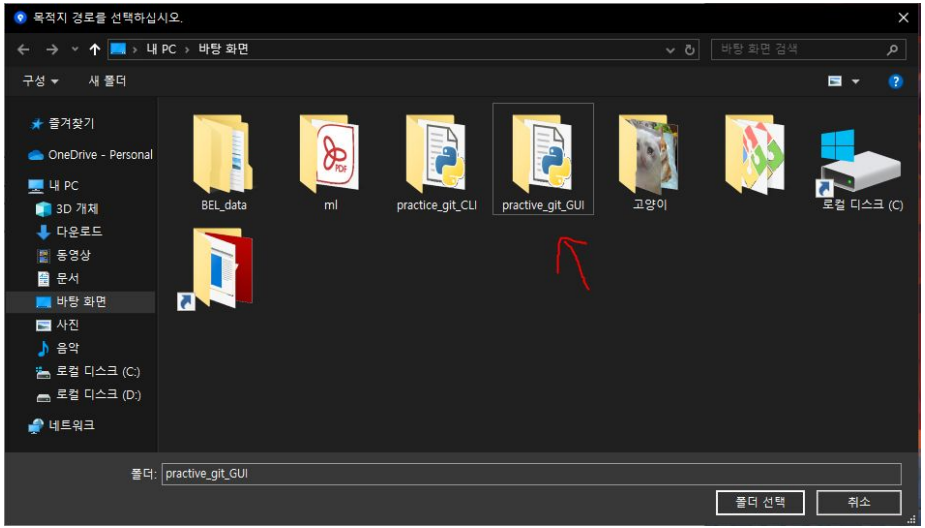
-> 원격 저장소에 push, 원격 저장소에 올리겠다는 뜻





Git 시작 - GUI

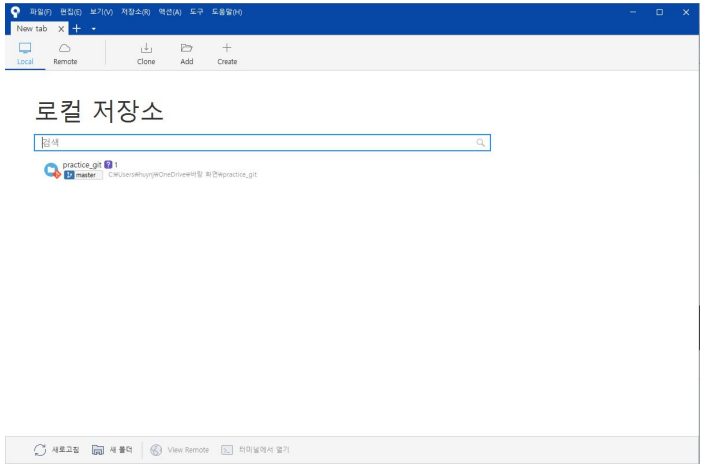

파일(F) 편집(E) 보기(V) 저장소(R) 액션(A) 도구 도움말(H)
New tab
Local
Remote
Clone
Add
Create
Create a repository
목적지 경로:
탐색
이름:
Git
계정에 저장소 생성하기
생성

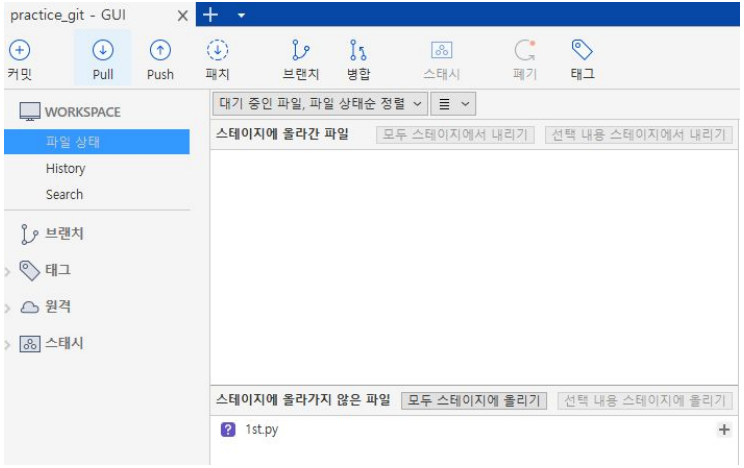

목적지 경로를 선택하십시오.
내 PC › 바탕 화면
바탕 화면 검색
구성
새 폴더
즐겨찾기
OneDrive - Personal
내 PC
3D 개체
다운로드
동영상
문서
바탕 화면
사진
음악
로컬 디스크 (C:)
로컬 디스크 (D:)
네트워크
BEL_data
ml
practice_git_CLI
practive_git_GUI
고양이
로컬 디스크 (C)
폴더: practive_git_GUI
폴더 선택
취소


.git
1st

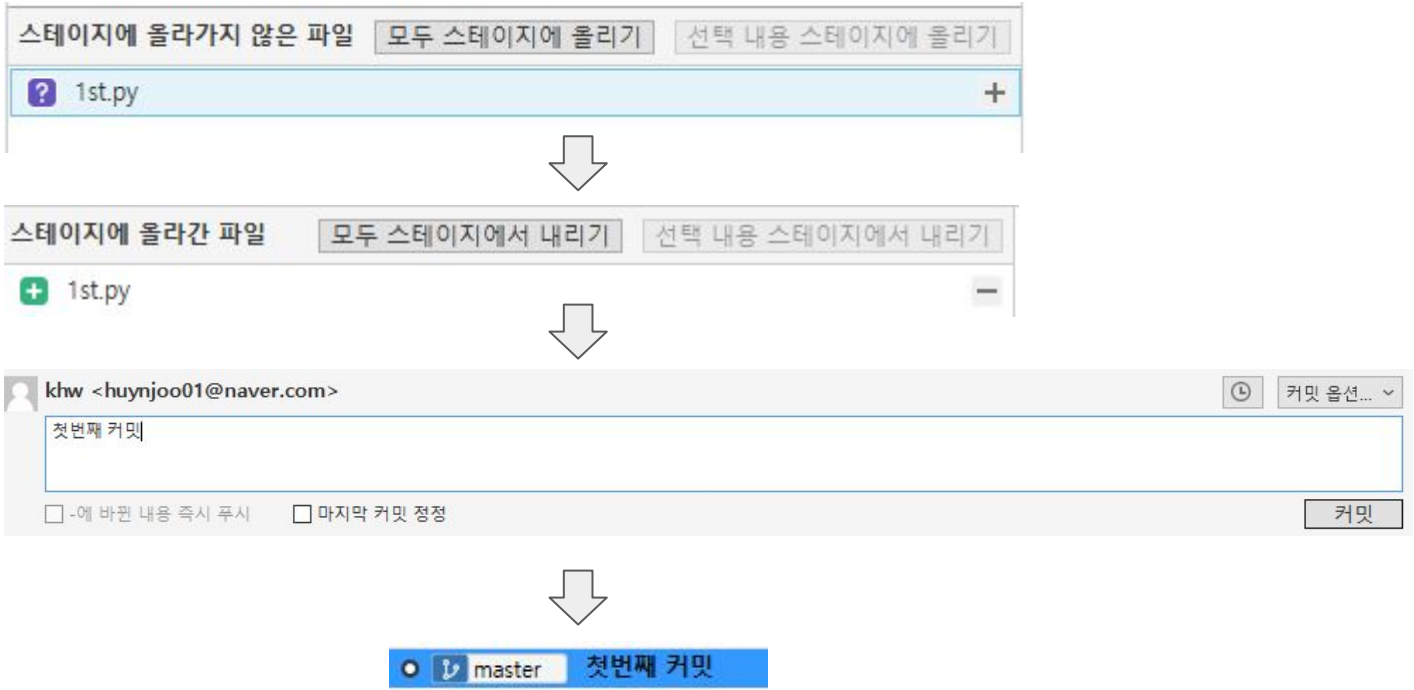
Git 시작 - GUI

드래그



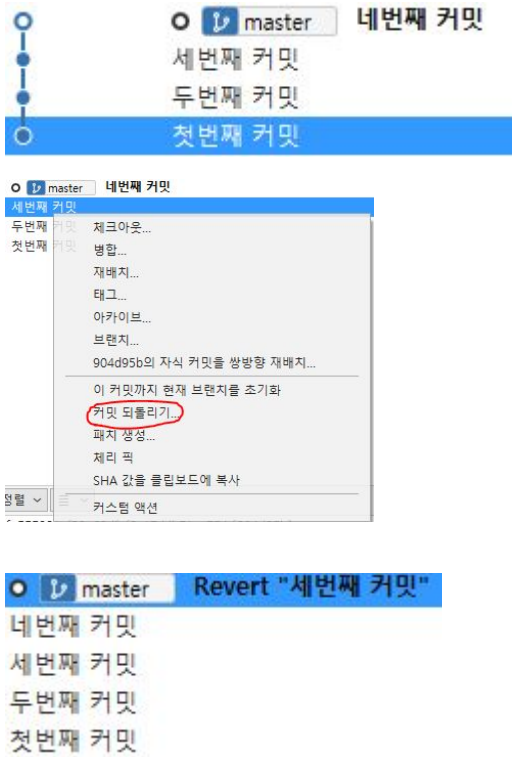


Git 시작 - GUI - add - commit

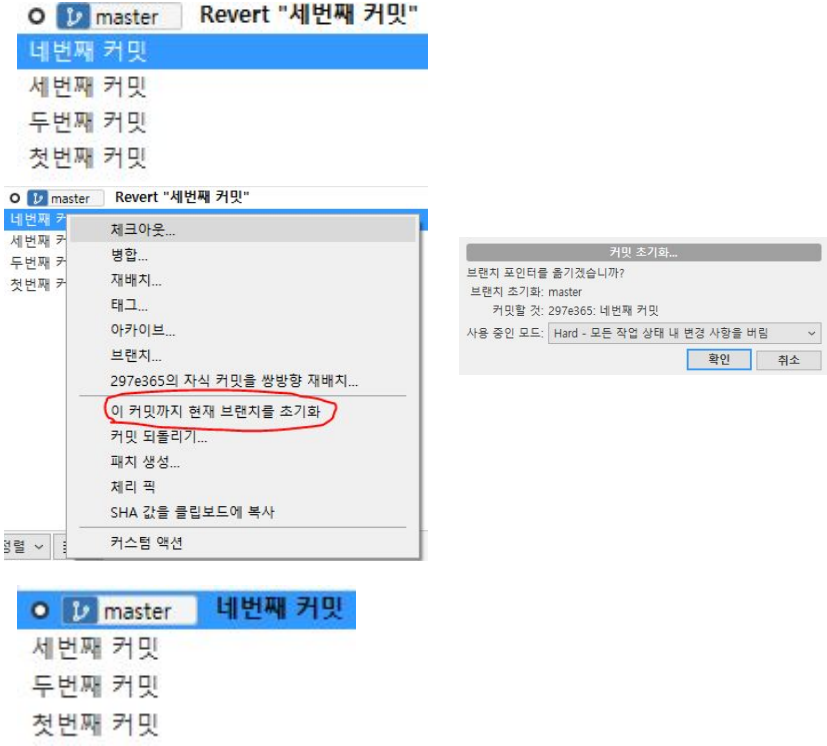


Git 시작 - GUI - revert - reset

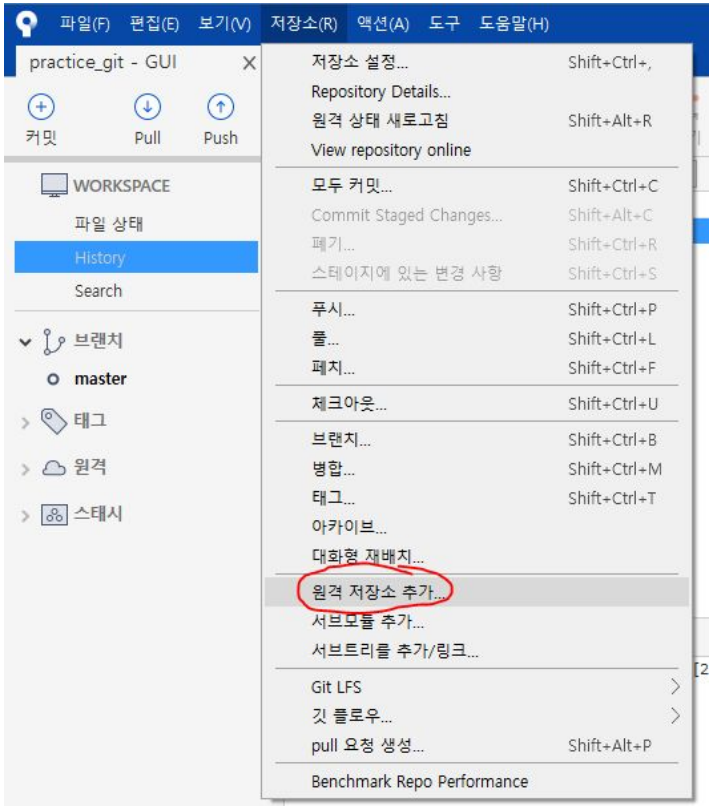
revert

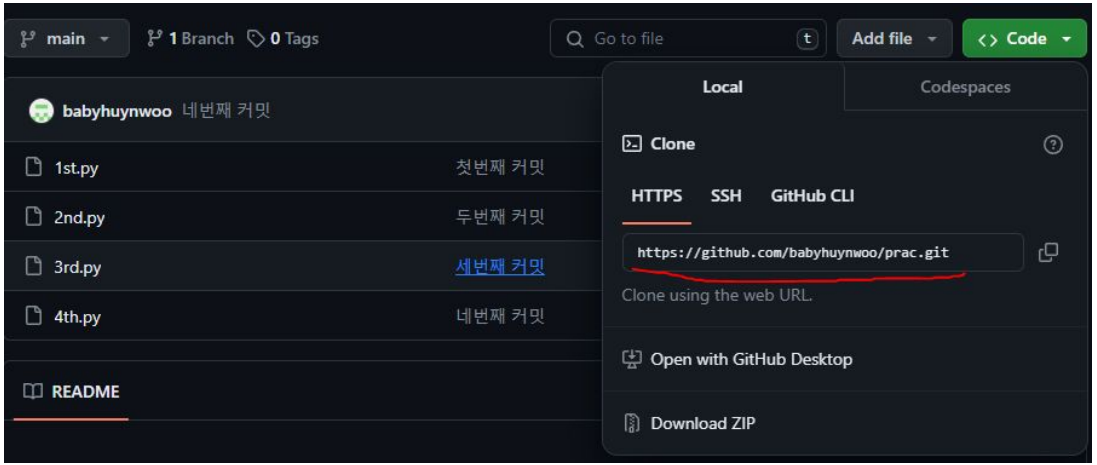


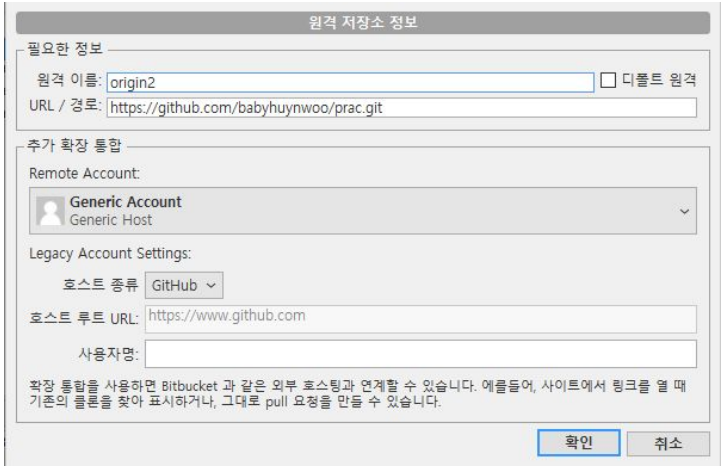
reset



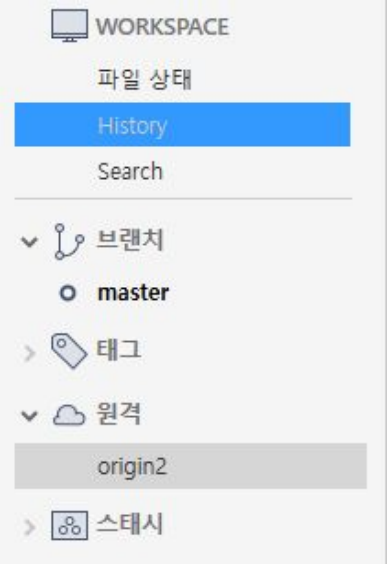
GitHub 연동

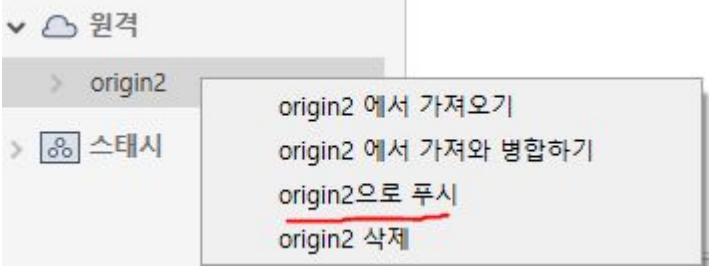

파일(F) 편집(E) 보기(V) 저장소(R) 액션(A) 도구 도움말(H)
practice_git - GUI
커밋
Pull
Push
WORKSPACE
파일 상태
History
Search
브랜치
master
태그
원격
스태시
저장소 설정... Shift+Ctrl+,
Repository Details...
원격 상태 새로고침 Shift+Alt+R
View repository online
모두 커밋... Shift+Ctrl+C
Commit Staged Changes... Shift+Alt+C
폐기... Shift+Ctrl+R
스테이지에 있는 변경 사항 Shift+Ctrl+S
푸시... Shift+Ctrl+P
풀... Shift+Ctrl+L
페치... Shift+Ctrl+F
체크아웃... Shift+Ctrl+U
브랜치... Shift+Ctrl+B
병합... Shift+Ctrl+M
태그... Shift+Ctrl+T
아카이브...
대화형 재배치...
원격 저장소 추가...
서브모듈 추가...
서브트리를 추가/링크...
Git LFS
깃 플로우...
pull 요청 생성... Shift+Alt+P
Benchmark Repo Performance

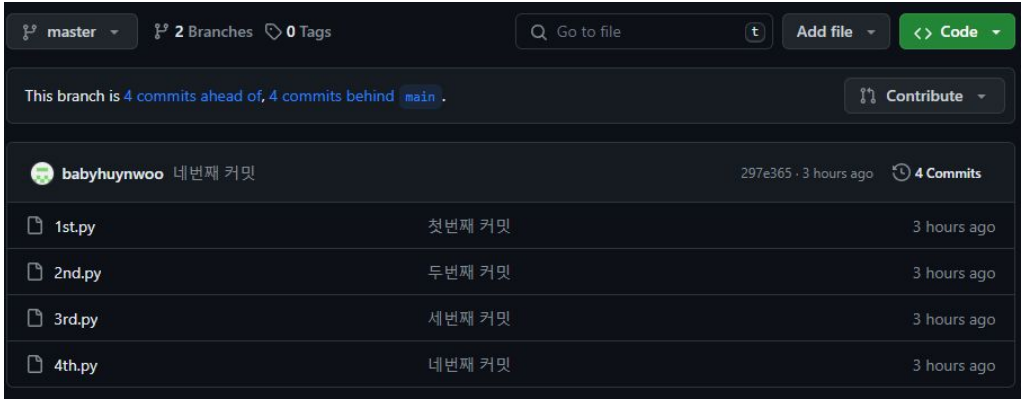

main
1 Branch
0 Tags
Go to file
t
Add file
Code
babyhuynwoo 네번째 커밋
1st.py 첫번째 커밋
2nd.py 두번째 커밋
3rd.py 세번째 커밋
4th.py 네번째 커밋
README
Local
Codespaces
Clone
HTTPS
SSH
GitHub CLI
https://github.com/babyhuynwoo/prac.git
Clone using the web URL.
Open with GitHub Desktop
Download ZIP


원격 저장소 정보
필요한 정보
원격 이름: origin2
디폴트 원격
URL / 경로: https://github.com/babyhuynwoo/prac.git
추가 확장 통합
Remote Account:
Generic Account
Generic Host
Legacy Account Settings:
호스트 종류 GitHub
호스트 루트 URL: https://www.github.com
사용자명:
확장 통합을 사용하면 Bitbucket 과 같은 외부 호스팅과 연계할 수 있습니다. 예를들어, 사이트에서 링크를 열 때 기존의 클론을 찾아 표시하거나, 그대로 pull 요청을 만들 수 있습니다.
확인
취소

GitHub 에 push



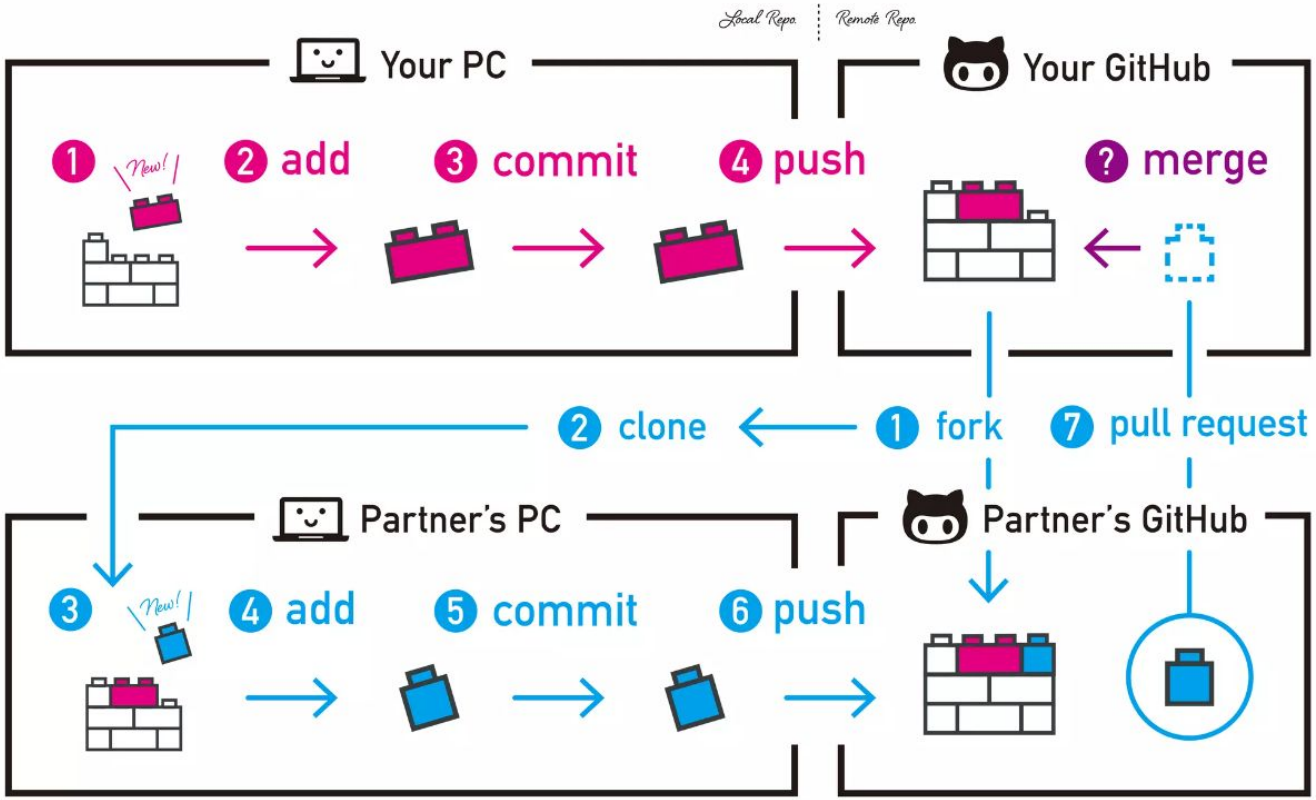




Git / GitHub

How to “Pull Request”

designless.net



Git 의 구조 (Local + Remote)