

# 버전관리 Git, Github 이해와 활용

---

## 4. 로컬저장소, 원격저장소 연계 및 Github 활용

유재명(wesleyok@jnu.ac.kr)

1. Git 원리
2. 개발환경 만들기
3. 로컬에서 Git 사용하기
4. 로컬저장소, 원격저장소 연계 및 Github 활용
  - 1) Github 특징 및 핵심기능 소개
  - 2) 원격저장소 만들기
  - 3) 로컬저장소에 복제 및 원격저장소 연계
5. 실습(1) - Python 패키지 배포 및 활용
6. 실습(2) - Node.js 패키지 배포 및 활용
7. 브랜치 이해 및 활용

## ★ Github 특징

- Git 기반의 분산 버전 관리 시스템이며, 원격저장소로 활용됨
- 개인 작업 뿐 아니라, 여러 개발자와 동시에 협업 가능
- Github는 전 세계 오픈 소스 프로젝트의 주요 플랫폼
- 전 세계 가장 많은 개발자들이 참여하는 커뮤니티로 코드 관리 및 배포, 문서화, 버그 리포팅, 토론 등으로 활성화되어 있음

# 1) Github 특징 및 핵심기능 소개

## ★ Github 핵심기능

### ■ Github Repository

<> Code Issues Pull requests Actions Projects Wiki Security Insights

- Code(원격저장소), Issues(버그,기능요청,질문 및 토론 등), Wiki(문서작성), Insights(저장소의 데이터 시각화) 등

### ■ Github Packages : 각 언어의 패키지 관리를 위한 저장소로 활용

- Npm(javascript), pip(python), maven(java), docker이미지 등
- Public(공개용) : 무료[무제한]
- Private(비공개용) : 유료[500MB,2GB(pro, team),50GB(enterprise)]

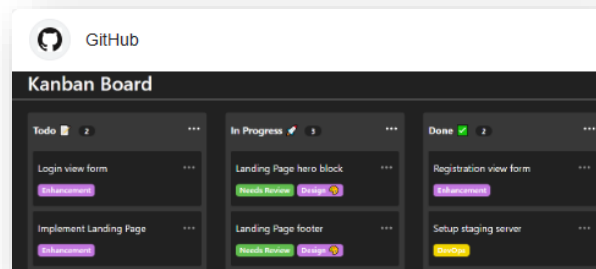
### ■ Github Actions : 개발 워크플로(빌드, 테스트, 배포) 수행(CI/CD)

### ■ Github Fork

- 다른 사용자의 저장소를 복사하여 자신의 Github 계정으로 가져오는 기능으로 복사본 생성하여 독립적인 작업이 가능 ( 예: easydiffusion 복제 )

### ■ Github Projects

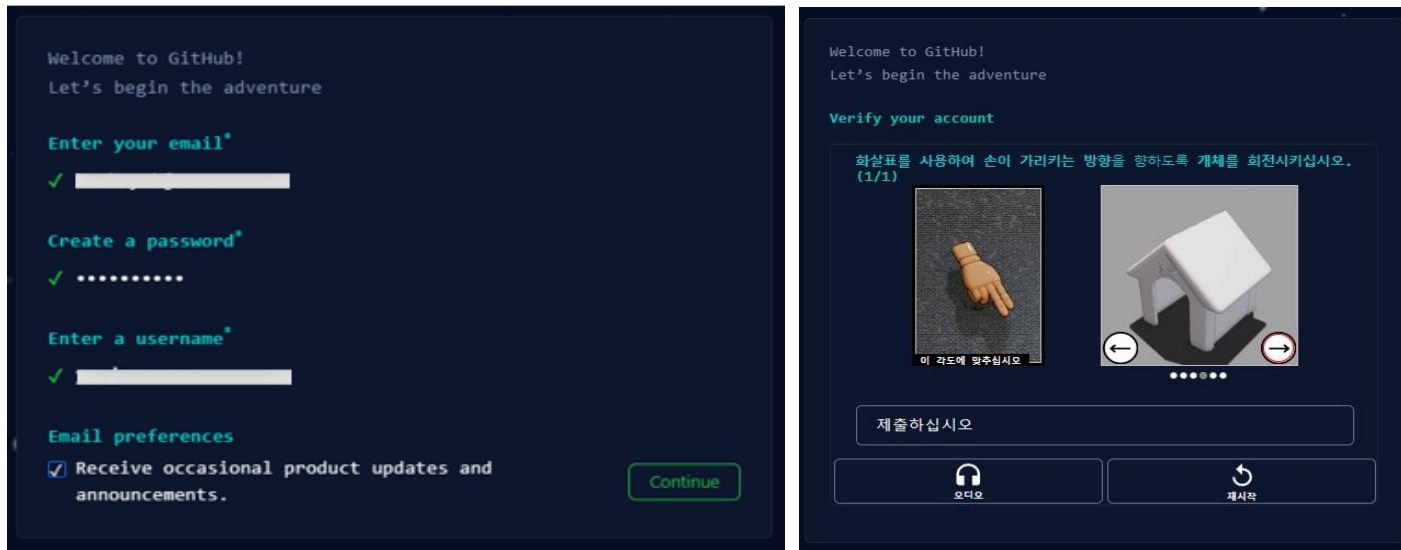
- 프로젝트 관리 도구로 팀이나 개인의 개발 작업 관리 및 추적 지원
- 칸반 보드 스타일( )의 인터페이스 제공



# 1) Github 특징 및 핵심기능 소개

## ★ Github 계정 만들기

- ① <https://www.github.com> 회원가입 (Sign up)
- ② 사용가능한 이메일/password/username 등록 및 간단한 정보확인 진행



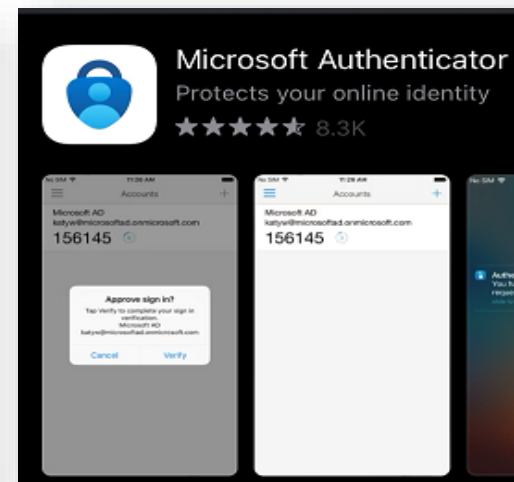
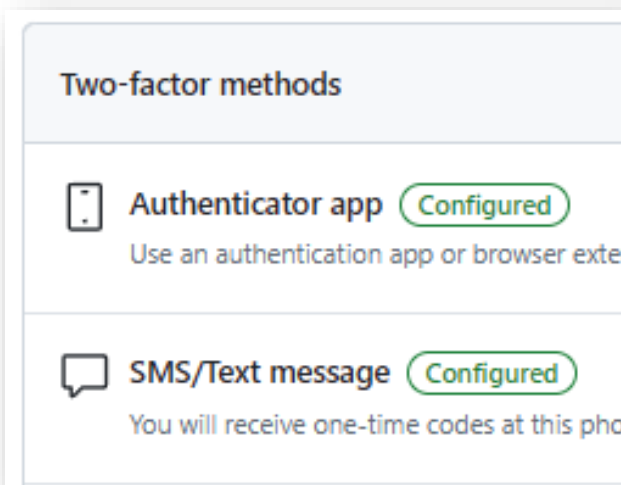
Left Screenshot: Welcome to GitHub! Let's begin the adventure. Enter your email\*, Create a password\*, Enter a username\*, Email preferences (Receive occasional product updates and announcements), Continue.

Right Screenshot: Welcome to GitHub! Let's begin the adventure. Verify your account. 화살표를 사용하여 손이 가리키는 방향을 향하도록 개체를 회전시키십시오. (1/1). 이 각도에 맞추십시오. 제출하십시오. 오디오, 재시작.

- ③ Github 회원가입시 등록한 메일의 수신함에서 Github의 메일 확인하여 완성
- ④ 회원가입완료 후 개인정보보호를 위한 추가 인증 작업
  - ⓐ 2FA(Two-factor authentication) (개인정보보호 강화 목적)
  - ⓑ 개인 액세스 토큰 Github API를 활용하거나 명령창에서 작업시 필수

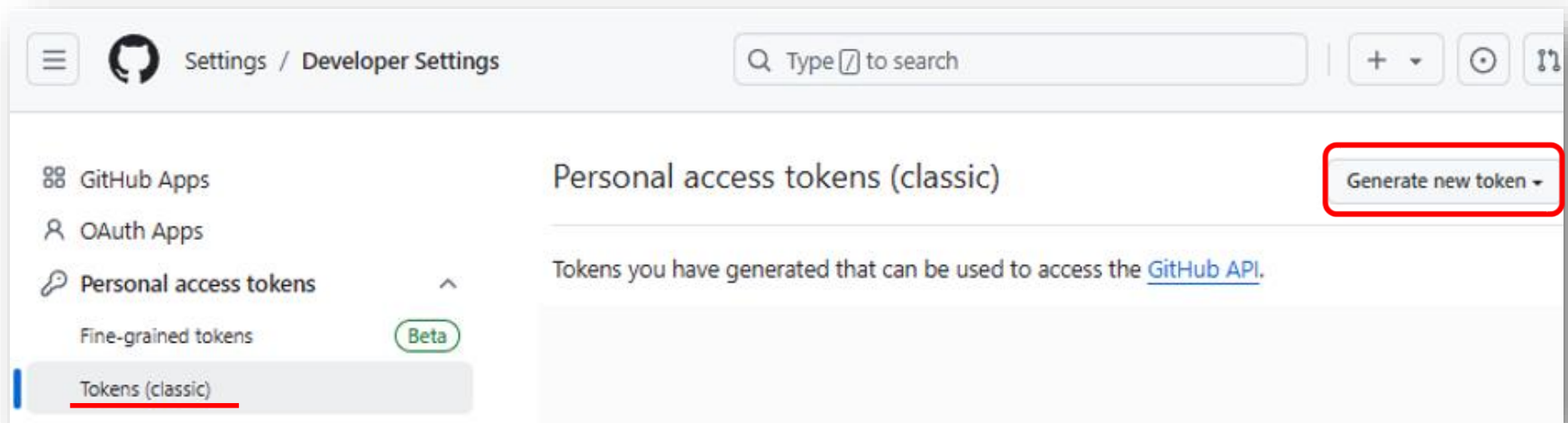
## ㉠ 2FA(Two-factor authentication, 2단계 인증)

- 2023.3월 기준으로 더욱 안전한 개인 정보 보호를 위해 모든 사용자에게 2FA(2단계 인증) 요구
  - Github 로그인 > Settings > Password and authentication
    - Authenticator app : 시간 기반 일회용 비밀번호(TOTP, Time-based One-Time Password)
    - SMS/Text message
  - 2FA를 활성화하지 않는 경우
    - 새 브라우저나 새 기기에서 본인 확인을 위한 2FA 과정을 추가하도록 요청



## ㉞ 개인 액세스 토큰(personal access token)

- 개발자로서 [Github API 활용](#)하거나 [명령줄을 사용](#)할 때 반드시 필요
- 토큰은 소유자가 생성한 원격저장소의 리소스에 대한 접근 권한 부여
- 2가지 유형
  - personal access tokens(classic) ← 명령창에서 버전관리 작업하기 위해
  - fine-grained personal access token



# 1) Github 특징 및 핵심기능 소개

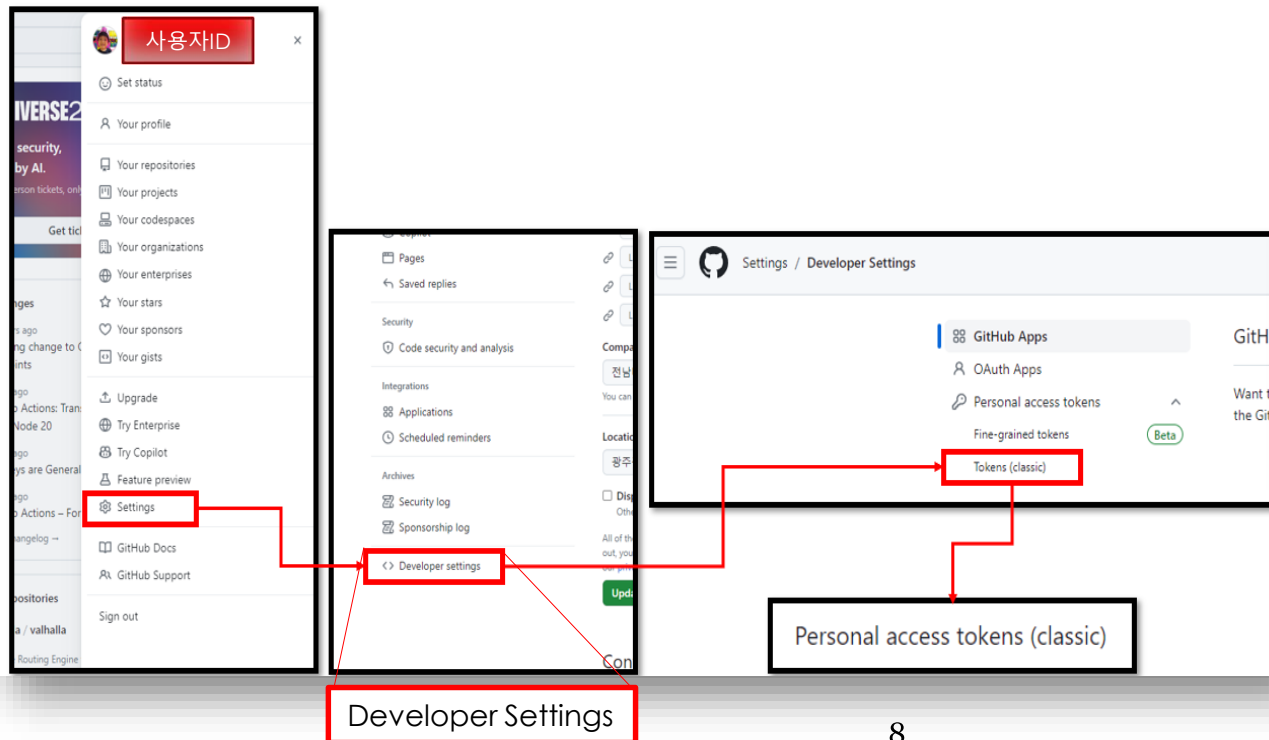
## ✦ personal access tokens(classic) 만들기

### ■ Github로그인 > Settings > Developer settings 클릭

- 아래 그림과 같이 **Generate new token**을 생성
- **토큰 이름**, **만료일** 지정, **저장소 액세스 범위** 선택
- **생성된 토큰**을 필요에 따라 사용하기 위해 **안전하게 보관 및 사용**

Git Bash와 같은 환경에서  
작업할 때 보안 토큰 필수

github 사이트에 로그인



[Note] : my-first-access-token

[Expiration] : No expiration

일단 모든 박스 체크 후 [Generate token] 클릭

[Note] 및 [토큰]을 안전한 곳(파일)에 저장하기



# 1) Github 특징 및 핵심기능 소개

## ✦ 토큰 사용 방법

- **Git Bash** 작업에서 HTTPS를 통해 원격저장소로 push할 때 ID/암호 대신 **토큰** 요구

```
$ git push https://github.com/USERNAME/REPO.git
Username: YOUR-USERNAME
Password: YOUR-PERSONAL-ACCESS-TOKEN
```

- **윈도우 자격증명(Credential)**에 **토큰**을 저장하여 사용

- ① **제어판** - **자격 증명 관리자**에서 **Windows 자격 증명** 탭으로 이동
- ② **git:https://github.com**의 자격 정보 찾아 편집버튼 클릭
- ③ 발급받은 **access token**을 암호 입력창에 붙여 넣고 저장

The screenshot shows the Windows Credential Manager interface. On the left, a search bar contains '자격 증명' (Credentials). Below it, a list of credentials is shown, with '자격 증명 관리자' (Credential Manager) selected. The main area displays the '자격 증명 관리자' (Credential Manager) window, which has a '웹 자격 증명' (Web Credentials) tab selected. This tab shows a list of stored web credentials. The first entry is for 'git:https://github.com' with a '수정한 날짜' (Last Modified) of '2023-10-04'. The second entry is for 'git:https://yoojaemyeong@github.com' with a '수정한 날짜' (Last Modified) of '2023-10-04'. A red arrow points from the search bar to the '자격 증명 관리자' window.

Credential Name	Last Modified
git:https://github.com	수정한 날짜: 2023-10-04
git:https://yoojaemyeong@github.com	수정한 날짜: 2023-10-04

## ★ 자격 증명 상태 보기

- `git config credential.helper`  
`manager`

- 서로 다른 시스템을 연결하고자 할 때 자격증명이 필요

## ★ 윈도우 자격증명(credential) 관리

- 윈도우 [시작] - [검색창] "자격증명"



자격 증명 관리자  
제어판

계정 정보가 없는 상태에서 `git push` 실행하면 로그인창이 떠서 ID/패스워드를 묻는다.  
계정이 있다면 아래와 같은 정보를 볼 수 있다.

```
$ git push
```

Everything up-to-date

✓ 맥OS에서 인증관련 명령어는 `osxkeychain`

## ✦ 자격증명 관리 3가지 옵션

1. 윈도우 운영체제에서 제공하는 자격증명관리자에 사용자 인증 정보 저장

```
PS F:\Projects\test2> git config credential.helper  
manager
```

2. 지정한 시간동안 ID와 패스워드 저장(예: 30초)

```
git config credential.helper "cache --timeout=30"
```

3. 파일을 이용하여 ID와 패스워드 저장

```
git config credential.helper "store --file ~/.git-credentials"
```

```
git push origin main
```

```
Username: <type your username>
```

```
Password: <type your password>
```

\* 위와 같이 등록하면 추후 영구적으로 증명(credentials) 저장됨

- \* 공용PC에서 실습 종료한 후, 윈도우 자격증명관리자에서 자격증명 삭제

## 1. ssh-keygen 유틸리티 사용하여 ssh키 생성



- 작업위치 : 홈폴더/.ssh
- 개인키 이름 변경: `mv id_rsa ~/.ssh/github-id_rsa`
- 환경파일작성 : `config`

```
Host github.com
  Hostname github.com
  IdentityFile ~/.ssh/github-id_rsa
```

```
$ ls -al ~/.ssh/id_rsa*
-rw-r--r-- 1 master 197121 2610 Nov 21 15:01 /c/Users/master/.ssh/id_rsa
-rw-r--r-- 1 master 197121 576 Nov 21 15:01 /c/Users/master/.ssh/id_rsa.pub
```

- 로컬에서 `ssh-keygen` 사용하여 생성 후 공개키를 github에 저장

```
$ cat id_rsa.pub
ssh-rsa AAAAB3NzaC1yc2E...
AAAA...zXVi5v1uc=...
master@DESKTOP-S3CV8RN
```

이 파일을 복사하여 [github.com > Settings > SSH and GPG Keys](#)

SSH keys New SSH key

This is a list of SSH keys associated with your account. Remove any keys that you do not recognize.

Add new SSH Key

Title  
git-practice

Key type  
Authentication Key

Key  
ssh-rsa AAAAB3NzaC1yc2E...  
AAAA...zXVi5v1uc=...  
master@DESKTOP-S3CV8RN

Add SSH key

- 로컬에서 SSH원격저장소 추가
  - `git remote add ssh-origin git@github.com:[user-id]/[git-저장소].git`

## 2) 원격저장소 만들기

### ✦ 작업 시나리오

#### 1. Github에 로그인하여 원격저장소 만들기

- 원격저장소 이름은 `git-practice`
- 원격저장소에 README.md, calculator.py 2개의 파일을 생성.

#### 2. 로컬PC에 Projects 폴더를 만들고 원격저장소 내용을 복제

- 로컬PC 복제 장소: Projects\`git-practice`

#### 3. 작업폴더(git-practice)에서 README.md, calculator.py 파일을 편집

#### 4. 작업폴더의 변경된 파일 -> Staging Area (git add) 버전관리 작업목록 추가

#### 5. Staging Area -> 로컬저장소로 커밋(git commit 작업일지에 저장)

#### 6. 원격저장소로 전송하여 로컬저장소와 원격저장소 동기화

#### 7. 원격저장소에 변경된 내용 확인

## 2) 원격저장소 만들기

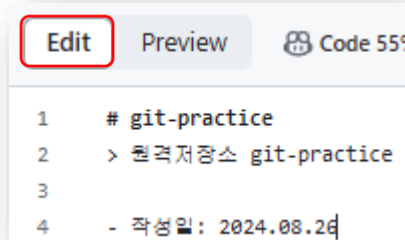
### 1. Github에 로그인하여 원격저장소 만들기

① Github 로그인 [ + ▼ ] 버튼 클릭 > [New repository] 클릭

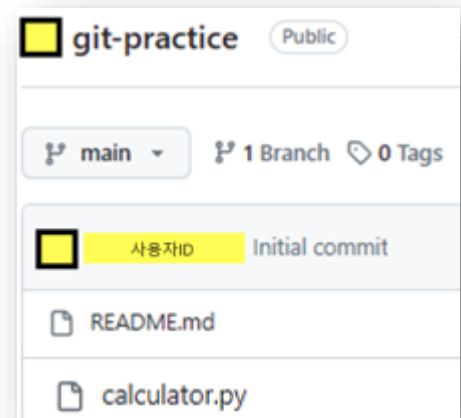
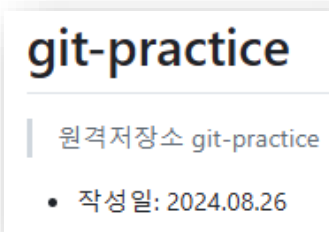
② [Create a new repository]에서

- Repository name\*에 `git-practice` 입력
- Description: “원격저장소 `git-practice`” 입력
- **Public** 선택
- Add a README file 체크 후 [Create repository] 클릭

③ README.md 파일 편집 후 [Commit changes...] 클릭

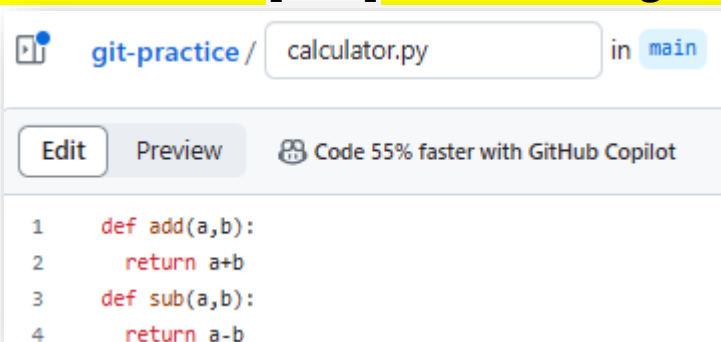


```
1 # git-practice
2 > 원격저장소 git-practice
3
4 - 작성일: 2024.08.26
```



④ [Add file ▼] > [+ Create new file] > [Commit changes...]

- 파일명: `calculator.py`
- 내용추가



### 3) 로컬저장소에 복제(1)

#### 2. 로컬PC에 Projects 폴더를 만들고 원격저장소 내용을 복제

- ① Projects 폴더 생성(`mkdir` 폴더명)

```
F:\>mkdir Projects
```

- ② 생성된 폴더(Projects)의 git 상태 정보(`git status`) 확인

```
F:\Projects>git status  
fatal: not a git repository (or any of the parent directories): .git
```

- ③ 로컬저장소에서 원격저장소 복제(`git clone`)

```
F:\Projects>git clone https://github.com/[사용자ID]/git-practice.git  
Cloning into 'git-practice'...  
remote: Enumerating objects: 9, done.  
remote: Counting objects: 100% (9/9), done.  
remote: Compressing objects: 100% (6/6), done.  
remote: Total 9 (delta 0), reused 0 (delta 0), pack-reused 0 (from 0)  
Receiving objects: 100% (9/9), done.
```

- ④ 작업폴더(`git-practice`)로 이동

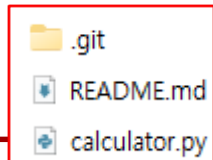
```
F:\Projects>cd git-practice
```



## 2) 로컬저장소에 복제(2)

### 3. 로컬PC의 작업폴더(git-practice)에서 내용수정

#### 1. 메모장을 이용하여 파일 편집

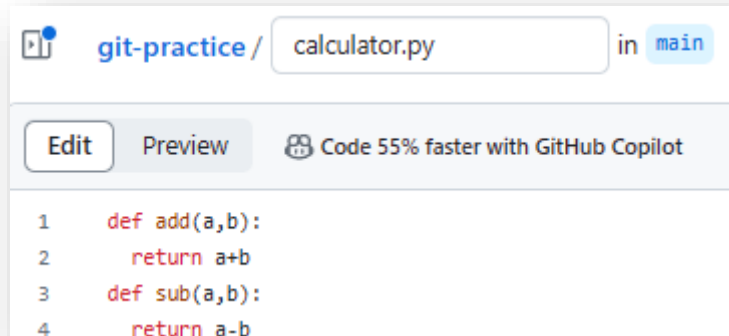


#### git-practice

원격저장소 git-practice  
• 작성일: 2024.08.26

```
F:\Projects\git-practice>notepad README.md
```

```
F:\Projects\git-practice>notepad calculator.py
```

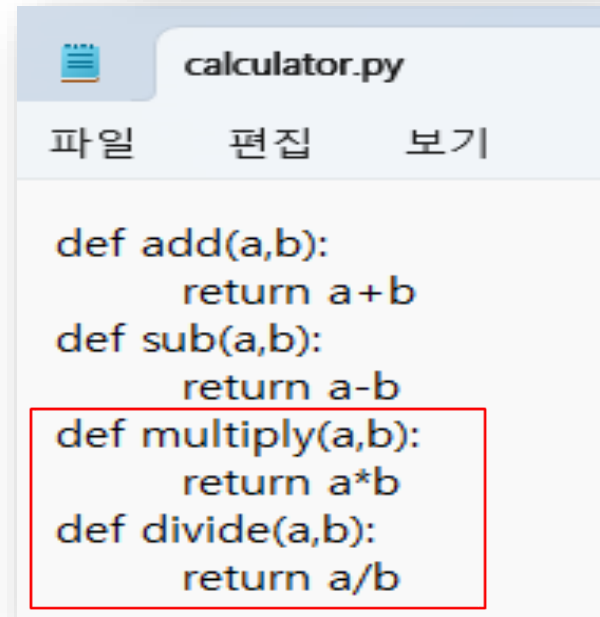


```
# git-practice  
> 원격저장소 git-practice
```

---

```
* 로컬PC에서 작업  
> calculator.py 파일 업데이트  
> multiply 함수 추가  
> divide 함수 추가
```

- 작성일: 2024.08.26



### 3) 로컬저장소에 복제(3)

#### 3 ~ 4. 작업완료 후 버전관리 git add, commit 작업

```
F:\Projects\git-practice>git log --oneline --all
089b7ff (HEAD -> main, origin/main, origin/HEAD) Create calculator.py
cf992e3 Update README.md
f2429ee Initial commit
```

복제본의 최초 로그 정보

```
F:\Projects\git-practice>git add . 로컬에서 수행한 작업, Staging Area에 추가
```

```
F:\Projects\git-practice\git-practice>git status 로컬 git 작업 상태 확인
On branch main
Your branch is up to date with 'origin/main'.
```

```
Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
        modified:   README.md
        modified:   calculator.py
```

로컬에서 최초 git commit 실행(로컬저장소에 저장)

```
F:\Projects\git-practice>git commit -m "update README, calculator.py"
[main ca63e93] update README, calculator.py
2 files changed, 8 insertions(+)
```

```
F:\Projects\git-practice>git status 로컬 git 작업 상태 확인
On branch main
Your branch is ahead of 'origin/main' by 1 commit.
  (use "git push" to publish your local commits)
```

```
nothing to commit, working tree clean
```

### 3) 로컬저장소와 원격저장소 연계(1)

#### 6. 로컬저장소의 git 로컬 환경변수(.git\config) 확인

```
F:\Projects\git-practice>notepad .git\config
```

```
[core]
  repositoryformatversion = 0
  filemode = false
  bare = false
  logallrefupdates = true
  symlinks = false
  ignorecase = true
[remote "origin"]
  url = https://github.com/[User ID]/git-practice.git
  fetch = +refs/heads/*:refs/remotes/origin/*
[branch "main"]
  remote = origin
  merge = refs/heads/main
```

```
F:\Projects\git-practice>git config --local --list
core.repositoryformatversion=0
core.filemode=false
core.bare=false
core.logallrefupdates=true
core.symlinks=false
core.ignorecase=true
remote.origin.url=https://github.com/[User ID]/git-practice.git
remote.origin.fetch=+refs/heads/*:refs/remotes/origin/*
branch.main.remote=origin
branch.main.merge=refs/heads/main
```

★ 원격저장소 별칭 'origin'

★ 원격저장소 별칭 만드는 방법 : git remote add [별칭] [Remote URL]

```
git remote add r_origin https://github.com/[User ID]/git-practice.git
```

### 3) 로컬저장소와 원격저장소 연계(2)

#### 7. 로컬저장소와 원격저장소 연계

- 원격저장소 상태 보기 : `git remote -v`

```
F:\Projects\git-practice>git remote -v
origin https://github.com/[User ID]/git-practice.git (fetch)
origin https://github.com/[User ID]/git-practice.git (push)
r_origin https://github.com/[User ID]/git-practice.git (fetch)
r_origin https://github.com/[User ID]/git-practice.git (push)
```

- 로컬저장소 => 원격저장소 동기화

```
F:\Projects\git-practice>git push origin main
Enumerating objects: 7, done.
Counting objects: 100% (7/7), done.
Delta compression using up to 24 threads
Compressing objects: 100% (4/4), done.
Writing objects: 100% (4/4), 529 bytes | 529.00 KiB/s, done.
Total 4 (delta 0), reused 0 (delta 0), pack-reused 0
To https://github.com/[User ID]/git-practice.git
089b7ff..ca63e93 main -> main
```

git-practice

원격저장소 git-practice

- 작성일: 2024.08.26

README

git-practice

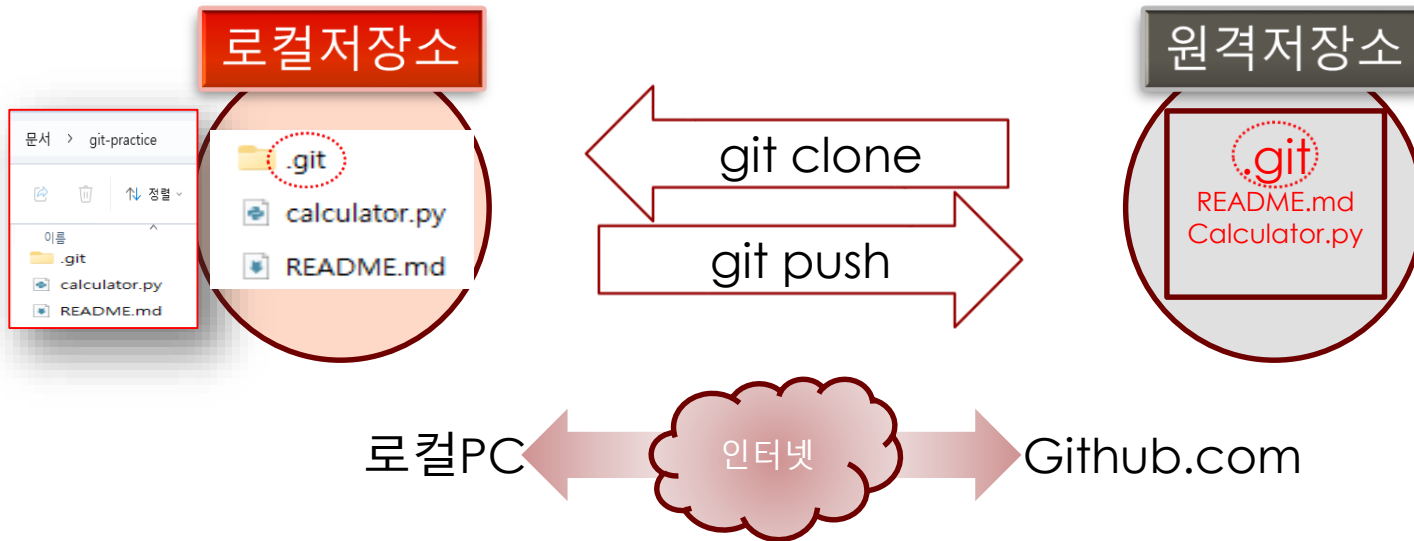
원격저장소 git-practice

- 로컬PC에서 작업

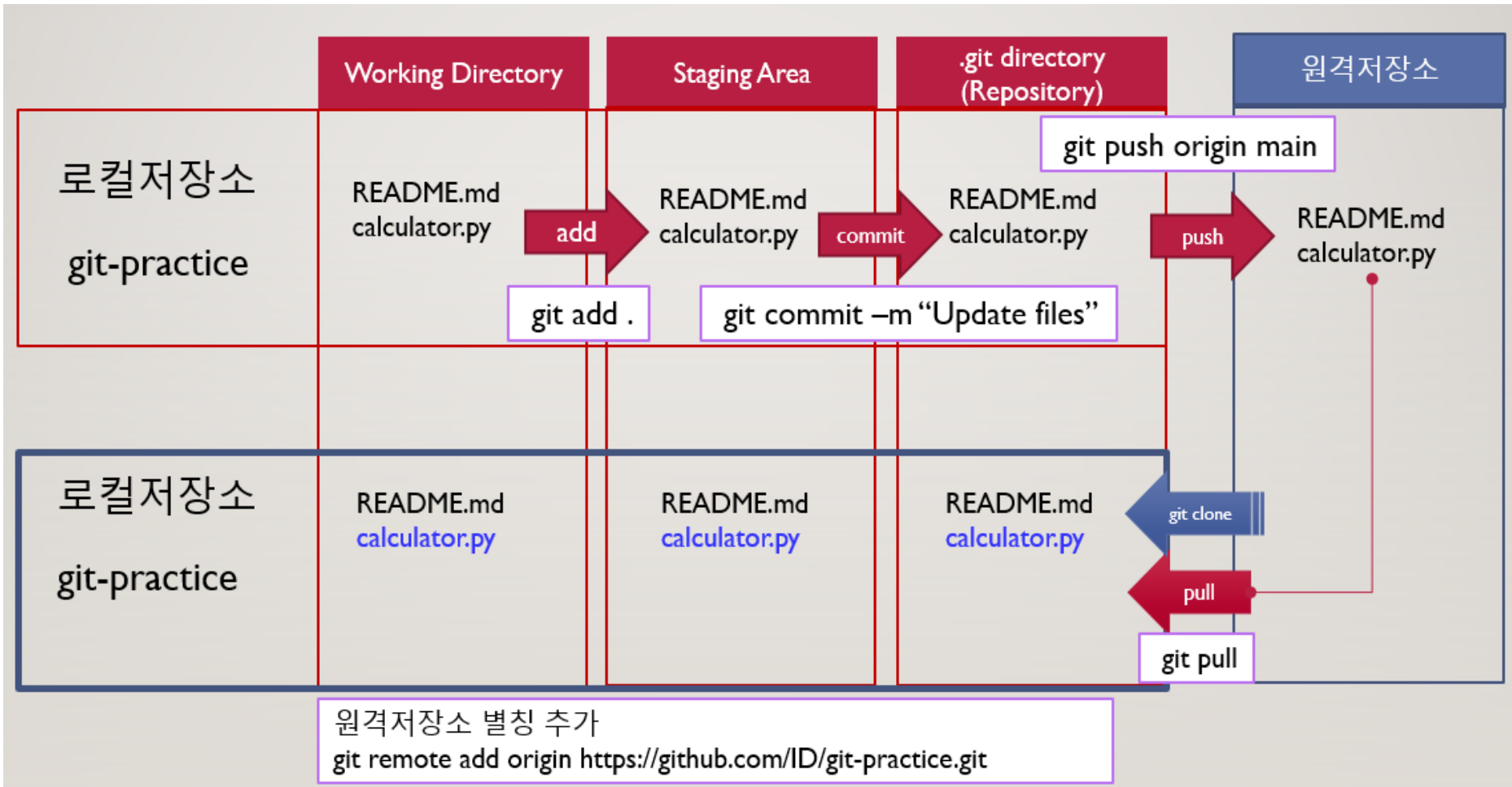
calculator.py 파일 업데이트  
multiply 함수 추가  
divide 함수 추가

- 작성일: 2024.08.26

### 3) 로컬저장소와 원격저장소 연계(3)



### 3) 로컬저장소와 원격저장소 연계(4)



감사합니다.