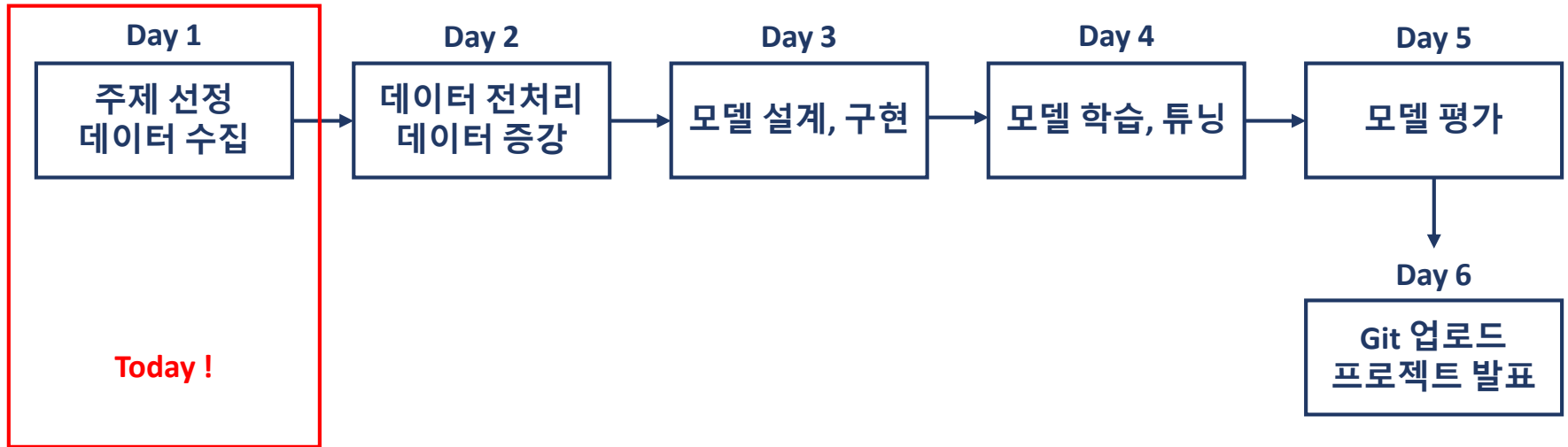


AI 기반 영상 데이터 분석 실습

- Day 1 -

index



index

1. Git & Github



2. classification 주제 선정

- 어떤 대상을 분류해보고 싶은지?

1. Classification

- 이미지 자체를 판단



Output : CAT

2. Object Detection

- Bounding Box 안에 있는 객체가 어떤 객체인지를 판별



CAT, DOG, DUCK

3. Image Segmentation

- Pixel 단위로 객체를 탐지



CAT, DOG, DUCK

3. 데이터 수집

Git & Github

Git & Github : 주요 기능

Git : 기본 개념

Git 사용법 :

- 설치
- 기초 명령어
- 프로젝트 관리 명령어
- 명령어

Github 사용법 :

- Local to repositories
- Repositories to local

Git & Github : 주요 기능

주요 기능 : git

- 프로젝트 내 문서 수정 이력 관리 가능(특정 시점으로 되돌리기 등)
- 하나의 프로젝트를 여러 사람이 함께 수행할 때 각자 개발 버전 관리 가능

주요 기능 : github

- 깃 저장소 호스팅 지원
- 온라인 상에 저장소 만들기 가능

Git : 기본 개념

주로 사용하는 인터페이스 (실습 시 사용)

: CLI (Command – Line Interface)

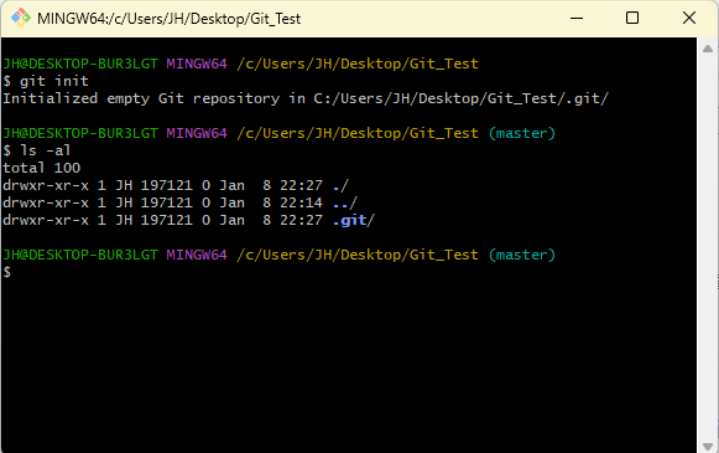
- 깃만 설치하면 바로 사용할 수 있는 기본적인 방법
- 깃의 모든 기능을 지원하는 인터페이스

Git의 세가지 관리영역

1. Working Directory (이하 WD) :
프로젝트 폴더 (작업 하는 폴더)

2. Staging Area (이하 SA) :
수정 이력 기록할 파일 대기 장소

3. Repository (이하 R):
staging area에서 대기중인 파일이 최종적으로 기록(커밋)되는 영역



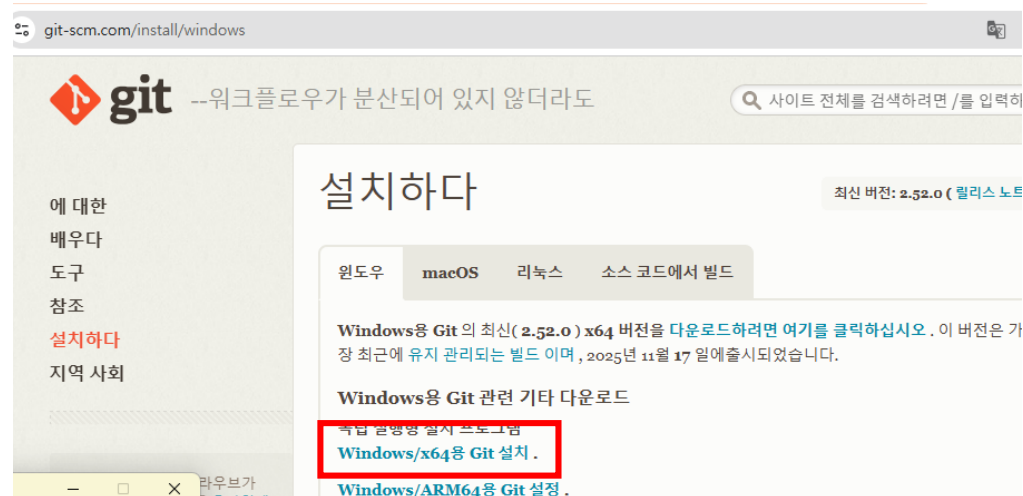
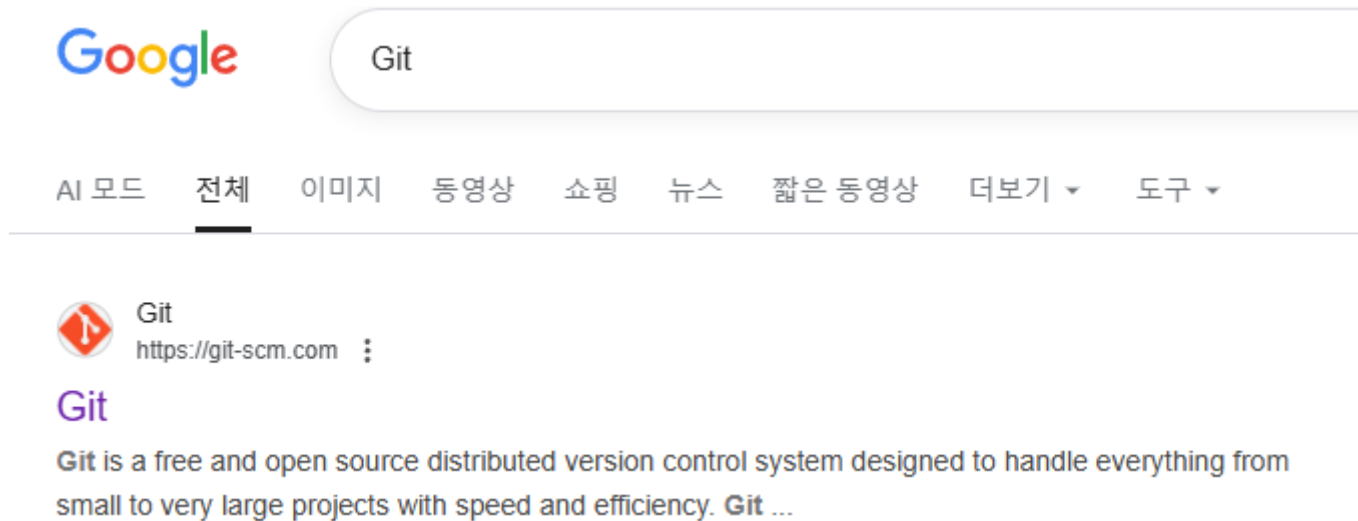
```
MINGW64/c:/Users/JH/Desktop/Git_Test
JH@DESKTOP-BUR3LGT MINGW64 /c:/Users/JH/Desktop/Git_Test
$ git init
Initialized empty Git repository in C:/Users/JH/Desktop/Git_Test/.git/

JH@DESKTOP-BUR3LGT MINGW64 /c:/Users/JH/Desktop/Git_Test (master)
$ ls -al
total 100
drwxr-xr-x 1 JH 197121 0 Jan  8 22:27 ./
drwxr-xr-x 1 JH 197121 0 Jan  8 22:14 ../
drwxr-xr-x 1 JH 197121 0 Jan  8 22:27 .git/

JH@DESKTOP-BUR3LGT MINGW64 /c:/Users/JH/Desktop/Git_Test (master)
$
```

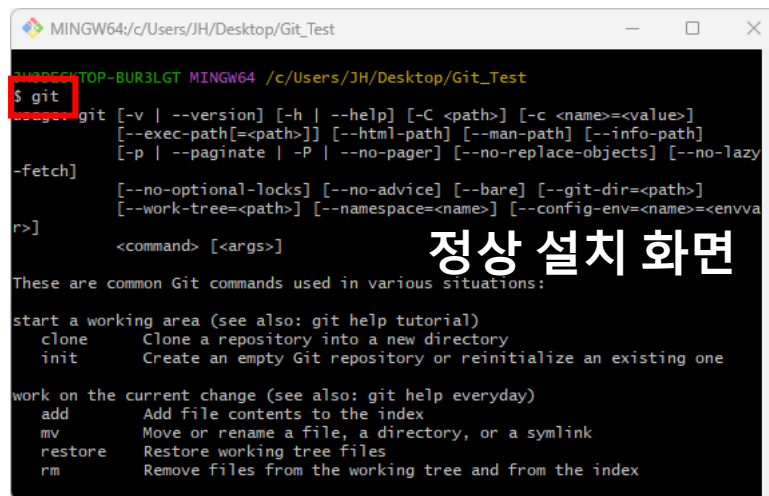
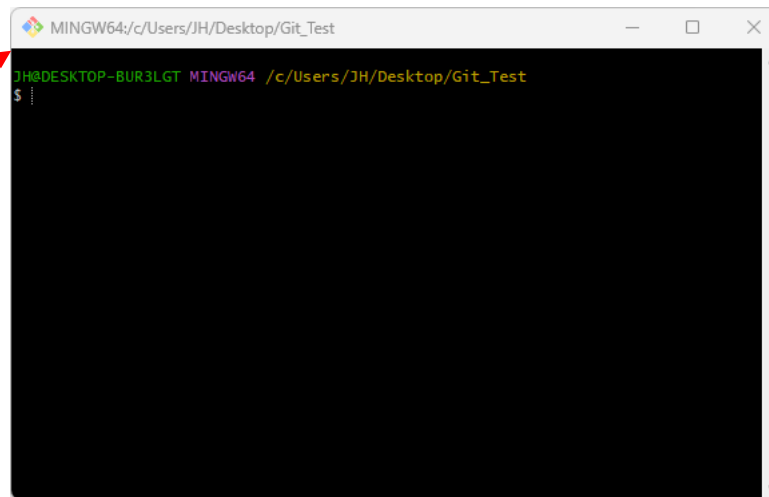
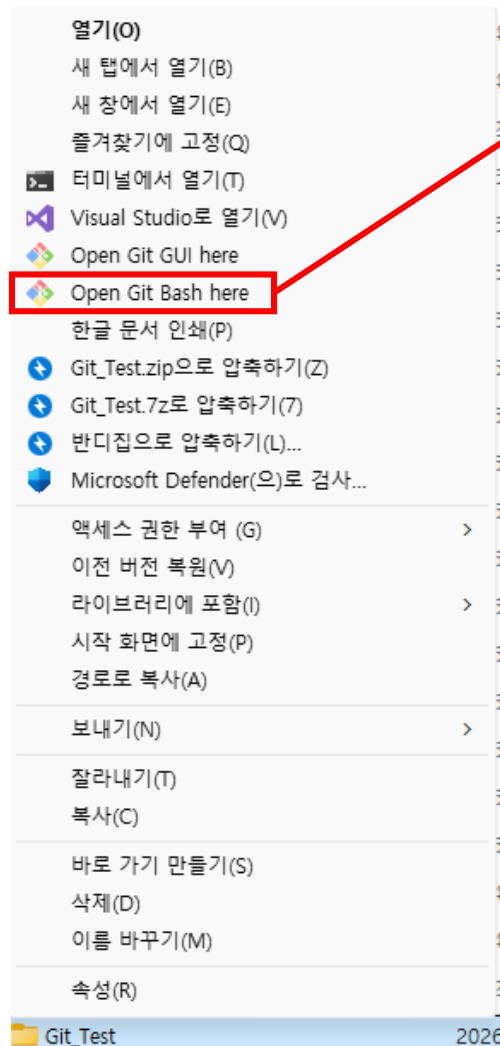
Git 사용법 : 설치

1. Git 설치



Git 사용법 : 설치

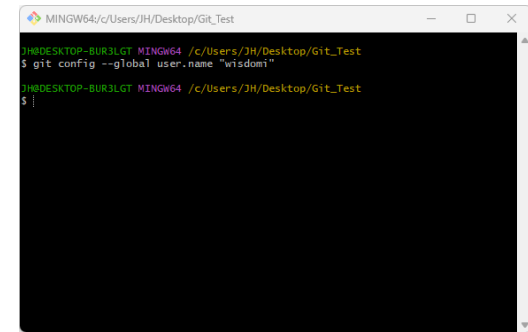
2. 터미널 열기 : 깃 또는 시스템 명령어 사용 가능 & 설치 확인



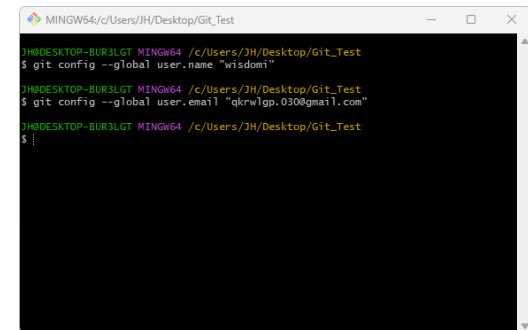
Git 사용법 : 기초 명령어

3. 기초 명령어

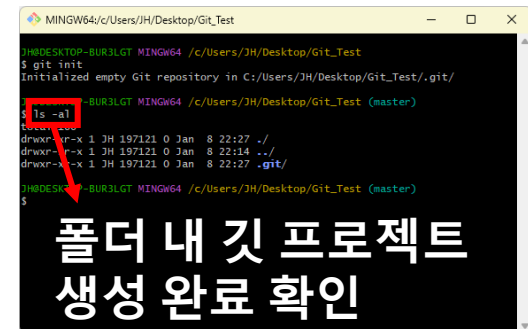
- \$git config --global user.name "사용자명"
-> 사용자명 설정
- \$git config --global user.email
-> 사용자 메일 주소 설정
- \$git init
-> 터미널이 열려있는 폴더를 깃 저장소로
초기화하는 깃 명령어
(이 폴더를 깃에서 관리하겠다)



```
MINGW64/c/Users/JH/Desktop/Git_Test
JHDESKTOP-BUR3LGT MINGW64 /c/Users/JH/Desktop/Git_Test
$ git config --global user.name "wisdom1"
JHDESKTOP-BUR3LGT MINGW64 /c/Users/JH/Desktop/Git_Test
$
```



```
MINGW64/c/Users/JH/Desktop/Git_Test
JHDESKTOP-BUR3LGT MINGW64 /c/Users/JH/Desktop/Git_Test
$ git config --global user.name "wisdom1"
JHDESKTOP-BUR3LGT MINGW64 /c/Users/JH/Desktop/Git_Test
$ git config --global user.email "dkrwlgp.030@gmail.com"
JHDESKTOP-BUR3LGT MINGW64 /c/Users/JH/Desktop/Git_Test
$
```



```
MINGW64/c/Users/JH/Desktop/Git_Test
JHDESKTOP-BUR3LGT MINGW64 /c/Users/JH/Desktop/Git_Test
$ git init
Initialized empty Git repository in C:/Users/JH/Desktop/Git_Test/.git/
JHDESKTOP-BUR3LGT MINGW64 /c/Users/JH/Desktop/Git_Test (master)
$ ls -al
total 16
drwxr-xr-x 1 JH 197121 0 Jan 8 22:27 ./
drwxr-xr-x 1 JH 197121 0 Jan 8 22:14 ../
drwxr-xr-x 1 JH 197121 0 Jan 8 22:27 .git/
JHDESKTOP-BUR3LGT MINGW64 /c/Users/JH/Desktop/Git_Test (master)
$
```

**폴더 내 깃 프로젝트
생성 완료 확인**

Git 사용법 : 프로젝트 관리 명령어

- \$git status
-> 깃 프로젝트 상태 확인

```
MINGW64/c/Users/JH/Desktop/Git_Test
JH@DESKTOP-BUR3LGT MINGW64 /c/Users/JH/Desktop/Git_Test (master)
$ ls
Git_Test_Text.txt
JH@DESKTOP-BUR3LGT MINGW64 /c/Users/JH/Desktop/Git_Test (master)
$ git status
On branch master
No commits yet

Untracked files:
  (use "git add <file>..." to include in what will be committed)
    Git_Test_Text.txt

nothing added to commit but untracked files present (use "git add" to track)
JH@DESKTOP-BUR3LGT MINGW64 /c/Users/JH/Desktop/Git_Test (master)
$
```

- \$git add
-> WD영역 문서를 SA에 추가

```
MINGW64/c/Users/JH/Desktop/Git_Test
JH@DESKTOP-BUR3LGT MINGW64 /c/Users/JH/Desktop/Git_Test (master)
$ git add Git_Test_Text.txt
JH@DESKTOP-BUR3LGT MINGW64 /c/Users/JH/Desktop/Git_Test (master)
$ git status
On branch master

No commits yet

Changes to be committed:
  (use "git rm --cached <file>..." to unstage)
    new file:   Git_Test_Text.txt
JH@DESKTOP-BUR3LGT MINGW64 /c/Users/JH/Desktop/Git_Test (master)
$
```

- \$git commit
-> SA영역 문서를 R에 추가 (커밋)

```
MINGW64/c/Users/JH/Desktop/Git_Test
Git_Test_commit_ver1
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:   Git_Test_Text.txt
JH@DESKTOP-BUR3LGT MINGW64 /c/Users/JH/Desktop/Git_Test (master)

[master (root-commit) 8384bfe] Git_Test_commit_ver1
1 file changed, 1 insertion(+)
create mode 100644 Git_Test_Text.txt
JH@DESKTOP-BUR3LGT MINGW64 /c/Users/JH/Desktop/Git_Test (master)
$
```

저장 후 나가기 (gvim 과 동일)

```
git /COMMIT_EDITMSG[+] [unix] (22:52 08/01/2026) 1,20 All
!wq
```

- \$git log
-> 커밋한 수정 이력 확인

```
MINGW64/c/Users/JH/Desktop/Git_Test
JH@DESKTOP-BUR3LGT MINGW64 /c/Users/JH/Desktop/Git_Test (master)
$ git log
commit 8384bfe2f00bb7dfbe0f41534ffacdF5da196aa4 (HEAD -> master)
Author: wisdomi <qkrwlgp.030@gmail.com>
Date: Thu Jan 8 22:52:45 2026 +0900

    Git_Test_commit_ver1
JH@DESKTOP-BUR3LGT MINGW64 /c/Users/JH/Desktop/Git_Test (master)
$
```

Git 사용법 : 명령어

6. 명령어

- `$git log -p`
->커밋 변경 사항까지 함께 출력
- `$git log --oneline`
->각 커밋 요약해 한줄씩 출력
- `$git checkout` 커밋해시
->문서 내용을 특정 커밋으로 되돌림
- `$git reflog`
->HEAD 포인터 참조 이력 출력

Git 사용법 : 명령어

6. 명령어

- `$git log -p`
-> 커밋 변경 사항까지 함께 출력
- `$git log --oneline`
-> 각 커밋 요약해 한줄씩 출력
- `$git checkout 커밋해시12jknek21jh`
-> 문서 내용을 특정 커밋으로 되돌림
- `$git reflog`
-> HEAD 포인터 참조 이력 출력
- `$git reset`
-> commit 이력 되돌리기

Git 사용법 : 명령어

6. 명령어

- \$git ignore

->수정 이력에서 제외, 수정 사항이 있어도 감지 안함

```
MINGW64/c/Users/JH/Desktop/Git_Test
JH@DESKTOP-BUR3LGT MINGW64 /c/Users/JH/Desktop/Git_Test (master)
$ touch .gitignore

JH@DESKTOP-BUR3LGT MINGW64 /c/Users/JH/Desktop/Git_Test (master)
$ ls
Git_Test_Text.txt  ignoreTest.txt

JH@DESKTOP-BUR3LGT MINGW64 /c/Users/JH/Desktop/Git_Test (master)
$ ls -al
total 105
drwxr-xr-x 1 JH 197121 0 Jan 8 23:16 ./
drwxr-xr-x 1 JH 197121 0 Jan 8 22:14 ../
drwxr-xr-x 1 JH 197121 0 Jan 8 23:10 .git/
-rw-r--r-- 1 JH 197121 0 Jan 8 23:16 .gitignore
-rw-r--r-- 1 JH 197121 33 Jan 8 23:07 Git_Test_Text.txt
-rw-r--r-- 1 JH 197121 0 Jan 8 23:16 ignoreTest.txt

JH@DESKTOP-BUR3LGT MINGW64 /c/Users/JH/Desktop/Git_Test (master)
$ nano .gitignore

GNU nano 8.7 .gitignore Modified
ignoreTest.txt

[ Read 0 Lines ]
^G Help  ^O Write Out  ^F Where Is  ^K Cut      ^C Location
^X Exit  ^R Read File  ^\ Replace   ^U Paste    ^J Justify   ^_ Go To Line
```

```
MINGW64/c/Users/JH/Desktop/Git_Test
JH@DESKTOP-BUR3LGT MINGW64 /c/Users/JH/Desktop/Git_Test (master)
$ git commit -m "add .gitignore"
[master 7469b3f] add .gitignore
1 file changed, 1 insertion(+)
create mode 100644 .gitignore

JH@DESKTOP-BUR3LGT MINGW64 /c/Users/JH/Desktop/Git_Test (master)
$ git log
commit 7469b3f37339a66007db4d9f54e83fa0c06c7ee8 (HEAD -> master)
Author: wisdomi <qkrwlgp.030@gmail.com>
Date: Thu Jan 8 23:21:50 2026 +0900

    add .gitignore

commit a5ae22e644dec7a89774891bbc8eb989c8a09b35
Author: wisdomi <qkrwlgp.030@gmail.com>
Date: Thu Jan 8 23:08:34 2026 +0900

    Git_Test_Text_ver2

commit 8384bfe2f00bb7dfbe0f41534ffacdf5da196aa4
Author: wisdomi <qkrwlgp.030@gmail.com>
Date: Thu Jan 8 22:52:45 2026 +0900

    Git_Test_commit_ver1

JH@DESKTOP-BUR3LGT MINGW64 /c/Users/JH/Desktop/Git_Test (master)
$
```

Commit 해야함

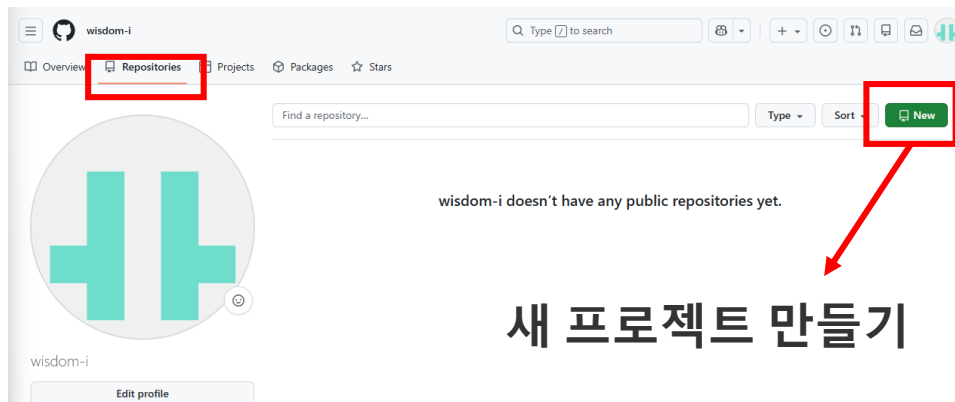
Github 사용법 : Local to repositories

1. Local - repositories

- `$git remote -v`
-> 현재 깃 프로젝트에 등록된 원격 저장소 확인
- `$git remote add` 원격저장소이름 원격저장소주소
-> 현재 깃 프로젝트에 원격 저장소 등록 & 이름 붙이기
- `$git push`
-> 로컬 저장소 내용을 원격 저장소에 공유
- `$git pull`
-> 원격 저장소 내용을 로컬 저장소에 공유

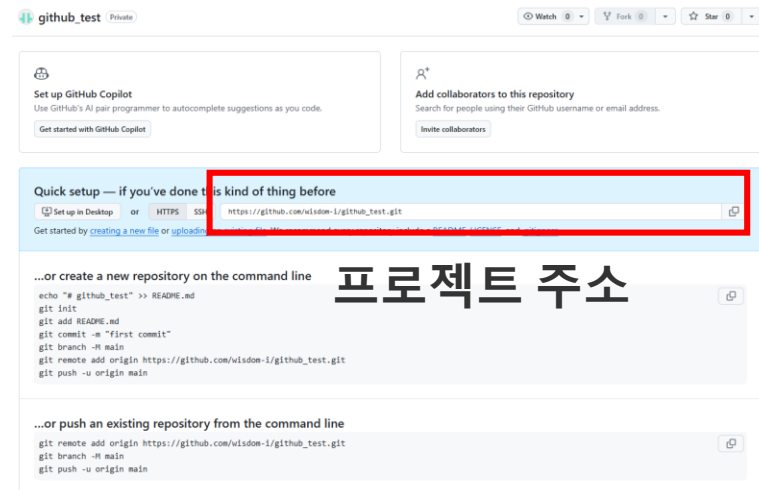
Github 사용법 : Local to repositories

1. Local - repositories



새 프로젝트 만들기

만든 후 안내 페이지

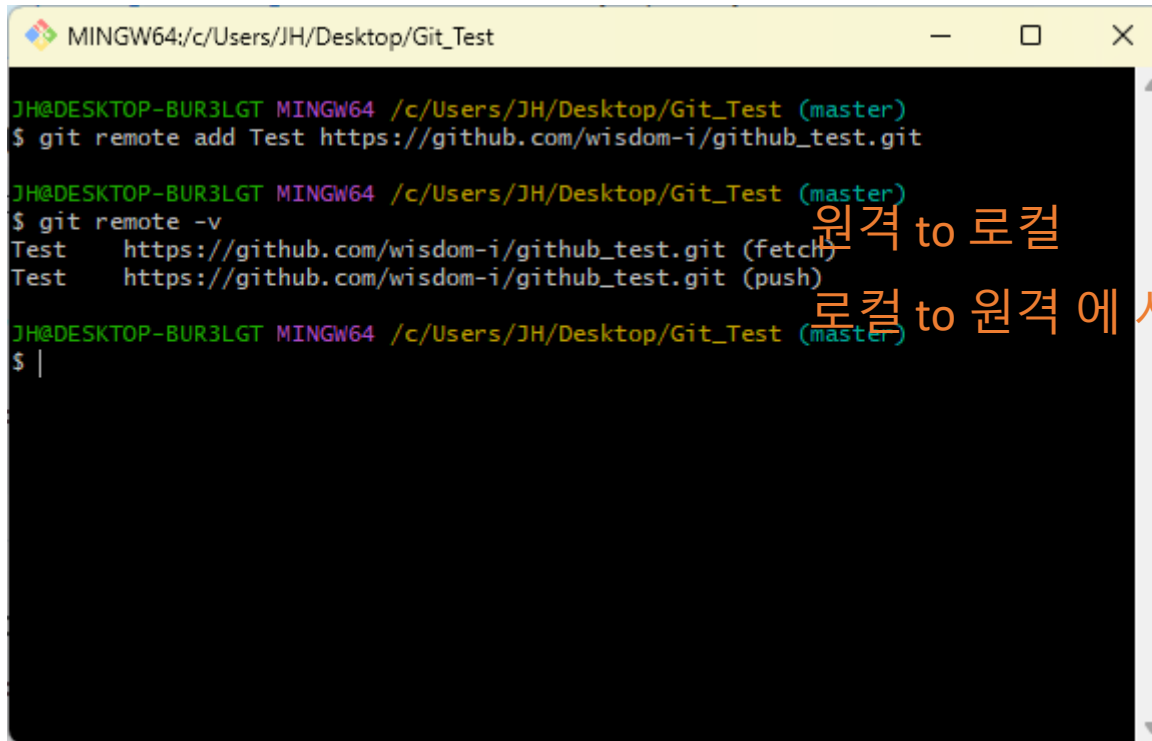


Github 사용법 : Local to repositories

2. Local – repositories : 원격 저장소와 연결

\$git remote add 이름지정 주소

-> 원격 저장소와 연결



```
MINGW64:/c/Users/JH/Desktop/Git_Test
JH@DESKTOP-BUR3LGT MINGW64 /c/Users/JH/Desktop/Git_Test (master)
$ git remote add Test https://github.com/wisdom-i/github_test.git

JH@DESKTOP-BUR3LGT MINGW64 /c/Users/JH/Desktop/Git_Test (master)
$ git remote -v
Test      https://github.com/wisdom-i/github_test.git (fetch)
Test      https://github.com/wisdom-i/github_test.git (push)

JH@DESKTOP-BUR3LGT MINGW64 /c/Users/JH/Desktop/Git_Test (master)
$ |
```

원격 to 로컬

로컬 to 원격 에 사용하는 주소

Github 사용법 : Local to repositories

2. Local – repositories : 원격 저장소와 연결

- `$git push -u` 원격저장소이름 브랜치이름
- 이후 같은 브랜치에서 `$git push`로만 사용 가능

```
MINGW64:/c/Users/JH/Desktop/Git_Test
JH@DESKTOP-BUR3LGT MINGW64 /c/Users/JH/Desktop/Git_Test (master)
$ git push
fatal: The current branch master has no upstream branch.
To push the current branch and set the remote as upstream, use

    git push --set-upstream Test master

To have this happen automatically for branches without a tracking
upstream, see 'push.autoSetupRemote' in 'git help config'.
```

브랜치의 첫 push 실패

```
JH@DESKTOP-BUR3LGT MINGW64 /c/Users/JH/Desktop/Git_Test (master)
$ git push -u Test master
```

원격저장소이름 브랜치(폴더) 이름

```
MINGW64:/c/Users/JH/Desktop/Git_Test
To push the current branch and set the remote as upstream, use

    git push --set-upstream Test master

To have this happen automatically for branches without a tracking
upstream, see 'push.autoSetupRemote' in 'git help config'.

JH@DESKTOP-BUR3LGT MINGW64 /c/Users/JH/Desktop/Git_Test (master)
$ git push -u Test master
info: please complete authentication in your browser...
Enumerating objects: 12, done.
Counting objects: 100% (12/12), done.
Delta compression using up to 16 threads
Compressing objects: 100% (6/6), done.
Writing objects: 100% (12/12), 951 bytes | 951.00 KiB/s, done.
Total 12 (delta 1), reused 0 (delta 0), pack-reused 0 (from 0)
remote: Resolving deltas: 100% (1/1), done.
To https://github.com/wisdom-i/github_test.git
 * [new branch]      master -> master
branch 'master' set up to track 'Test/master'.

JH@DESKTOP-BUR3LGT MINGW64 /c/Users/JH/Desktop/Git_Test (master)
$
```

Github 사용법 : Local to repositories

3. Local – repositories : 확인

The screenshot displays the GitHub interface for a repository named 'github_test' under the user 'wisdom-i'. The repository is private and has 0 forks and 0 stars. The main view shows the 'master' branch with 1 branch and 0 tags. The file list includes '.gitignore', 'Git_Test_Text.txt', and 'test.txt'. A red box highlights these files, with an arrow pointing to the text '마스터 브랜치 (폴더) 내 항목 모두 업로드 완료' (All items in the master branch (folder) are uploaded). Another red box highlights the '4 Commits' link, with an arrow pointing to the 'Commit 이력' (Commit history) section. A third red box highlights the 'Add a README' button, with an arrow pointing to the 'Commit 메시지' (Commit message) section. The commit history shows three commits: 'Test_upload' (1cddea2), 'add .gitignore' (7469bf3), and 'Git_Test_Text_ver2' (a5ae22e).

마스터 브랜치 (폴더) 내 항목
모두 업로드 완료

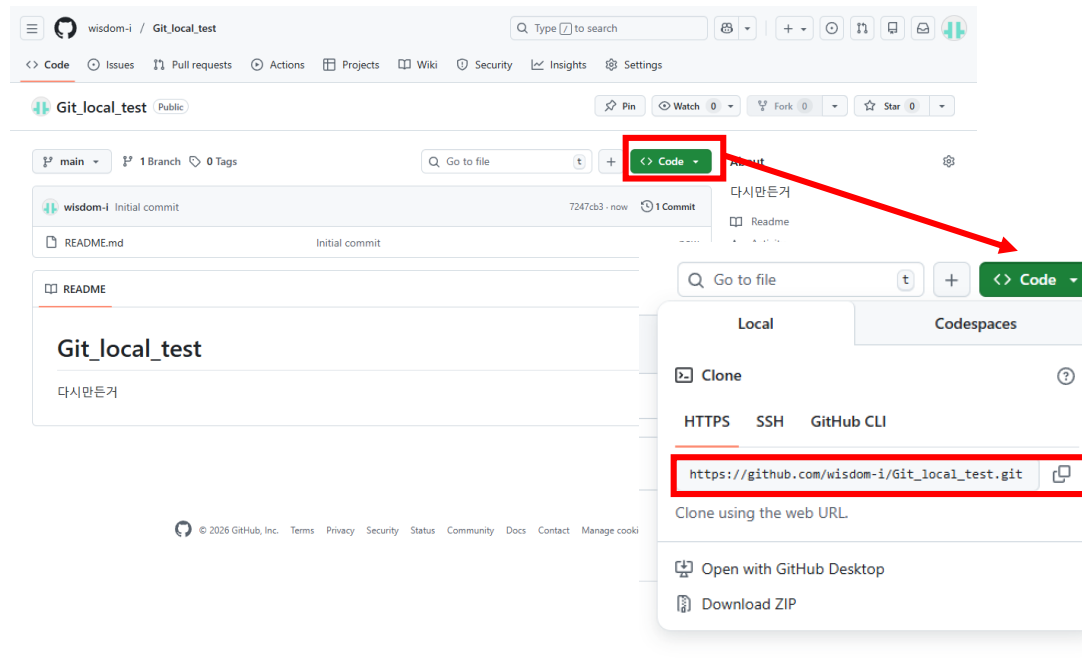
Commit 이력

Commit 메시지

Github 사용법 : Repositories to local

4. Repositories – local

- 새 프로젝트 생성

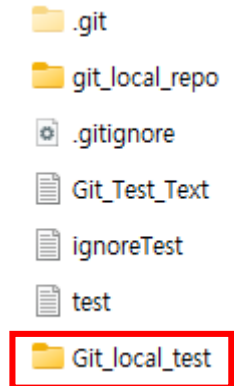


- 복사하고싶은 브랜치(폴더)에서 `$git clone` 주소 입력

```
JH@DESKTOP-BUR3LGT MINGW64 /c/Users/JH/Desktop/Git_Test (master)
$ git clone https://github.com/wisdom-i/Git_local_test.git
```

Github 사용법 : Repositories to local

5. Repositories – local : 확인



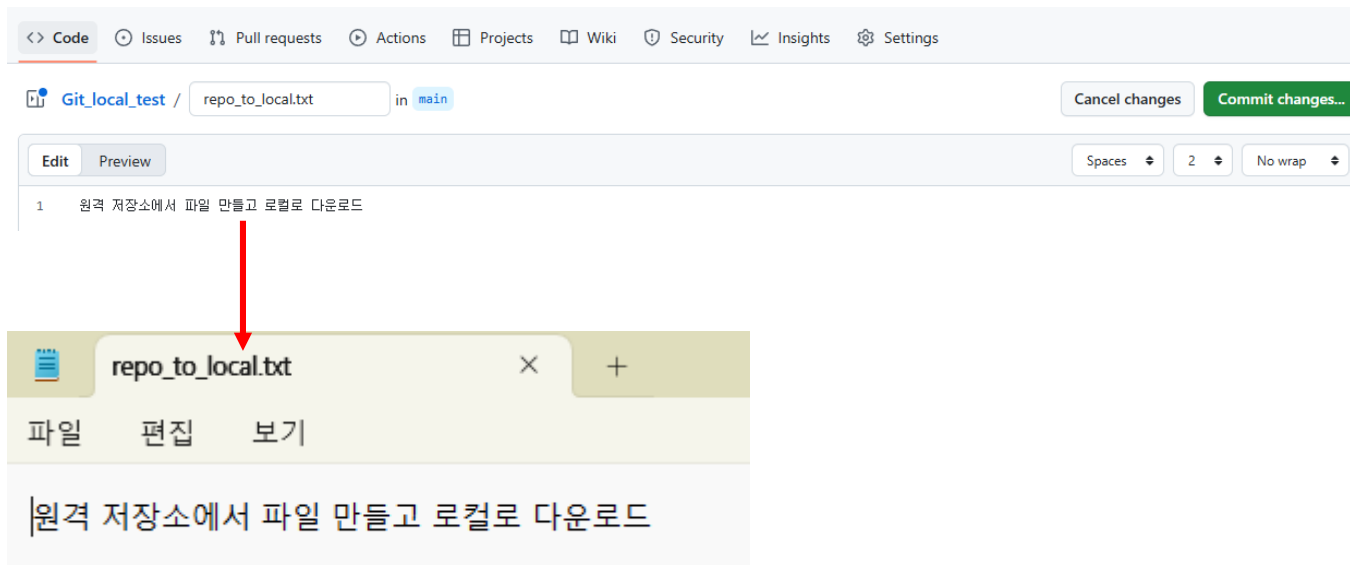
Github 사용법 : Repositories to local

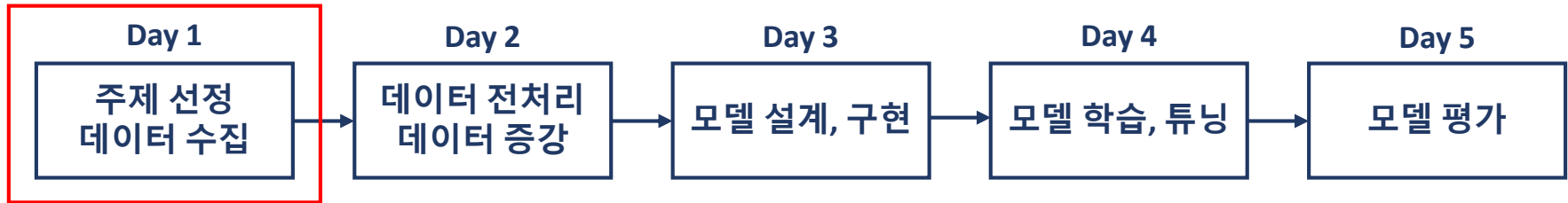
5. Repositories – local : 확인

The screenshot shows the GitHub interface for a repository named **Git_local_test**. The repository is public and has a main branch. The file `repo_to_local.txt` is selected for editing. The commit message field contains the text "원격 저장소에서 파일 만들고 로컬로 다운로드". The commit message field also has a placeholder text "Add instruction for downloading from remote repository". The extended description field is empty. The commit message field has a "Commit changes..." button. The extended description field has a "Commit changes..." button. The commit message field has a "Commit changes..." button. The extended description field has a "Commit changes..." button.

Github 사용법 : Repositories to local

5. Repositories – local : 확인





1. 주제 선정 및 데이터 수집

- 데이터 선정
 - 어떤 데이터를 분류해보고 싶은지
 - 예시 : 개 vs 고양이, 비행기 vs 자동차
- 데이터 수집
 - 수집 기준 : 라벨 당 200장 이상, 최소 2라벨 수집
 - 방법 : 직접 촬영 혹은 구글 서칭



?



Thank you