

# Introduction to Git

---

Hansol Choi ([hschoi95@kw.ac.kr](mailto:hschoi95@kw.ac.kr))

2020-04-21

KW-VIP 2020-1

# Contents

---

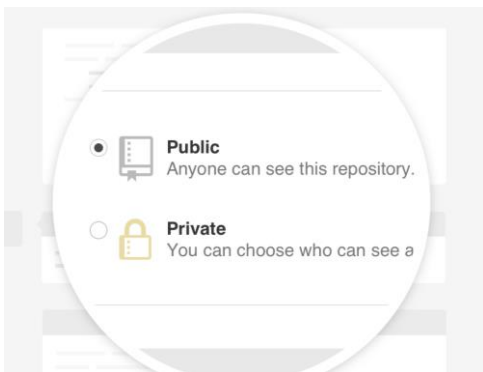
- Introduction
- Local/remote repository
- 단독 개발을 위한 git working flow 예제
- 공동 개발을 위한 git working flow 예제
- 3차 과제

# INTRODUCTION

---

# git과 GitHub

- git
  - 소스 코드 관리를 위한 분산 버전 관리 시스템 도구
- GitHub
  - git을 기반으로 소스 코드를 호스팅하고 협업 기능을 지원하는 서비스 및 원격 저장소
  - GitHub외에 Git을 서비스하는 곳
    - Bitbucket, Gitlab, Gogs



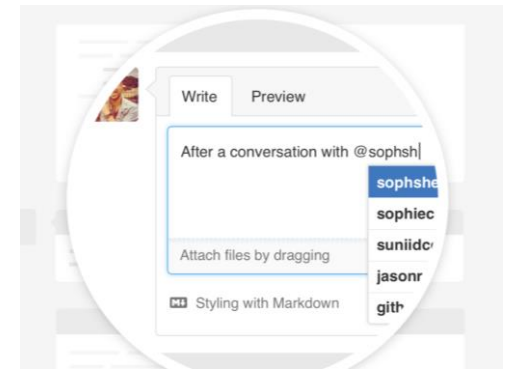
## Host a project

Collaborate with other developers in public, or invite them to join you in unlimited private repositories.



## Build and learn

You can save every version of your code, so you are free to experiment without losing your work.

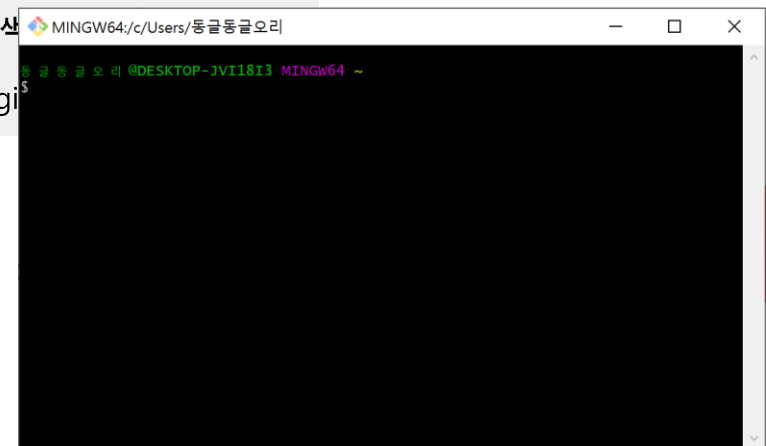
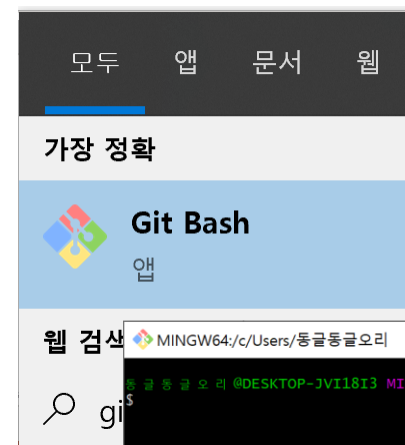
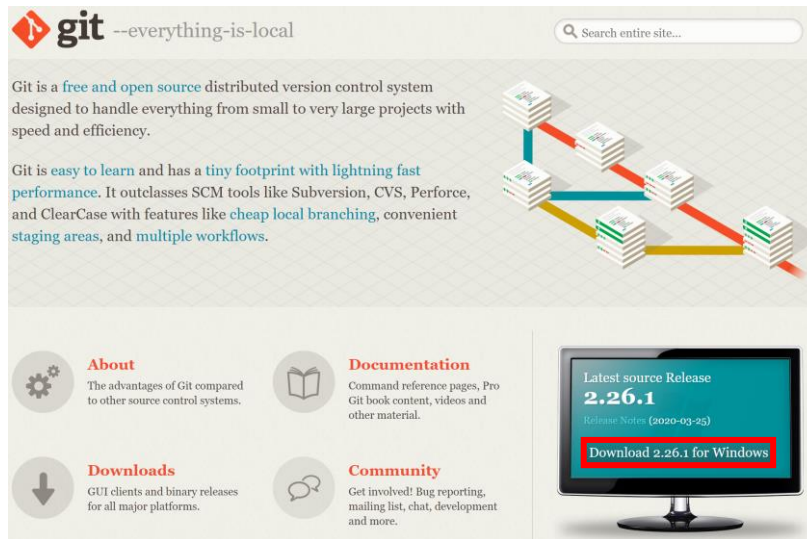


## Work together

Once your code is on GitHub, you can invite others to join in with a link or an @mention.

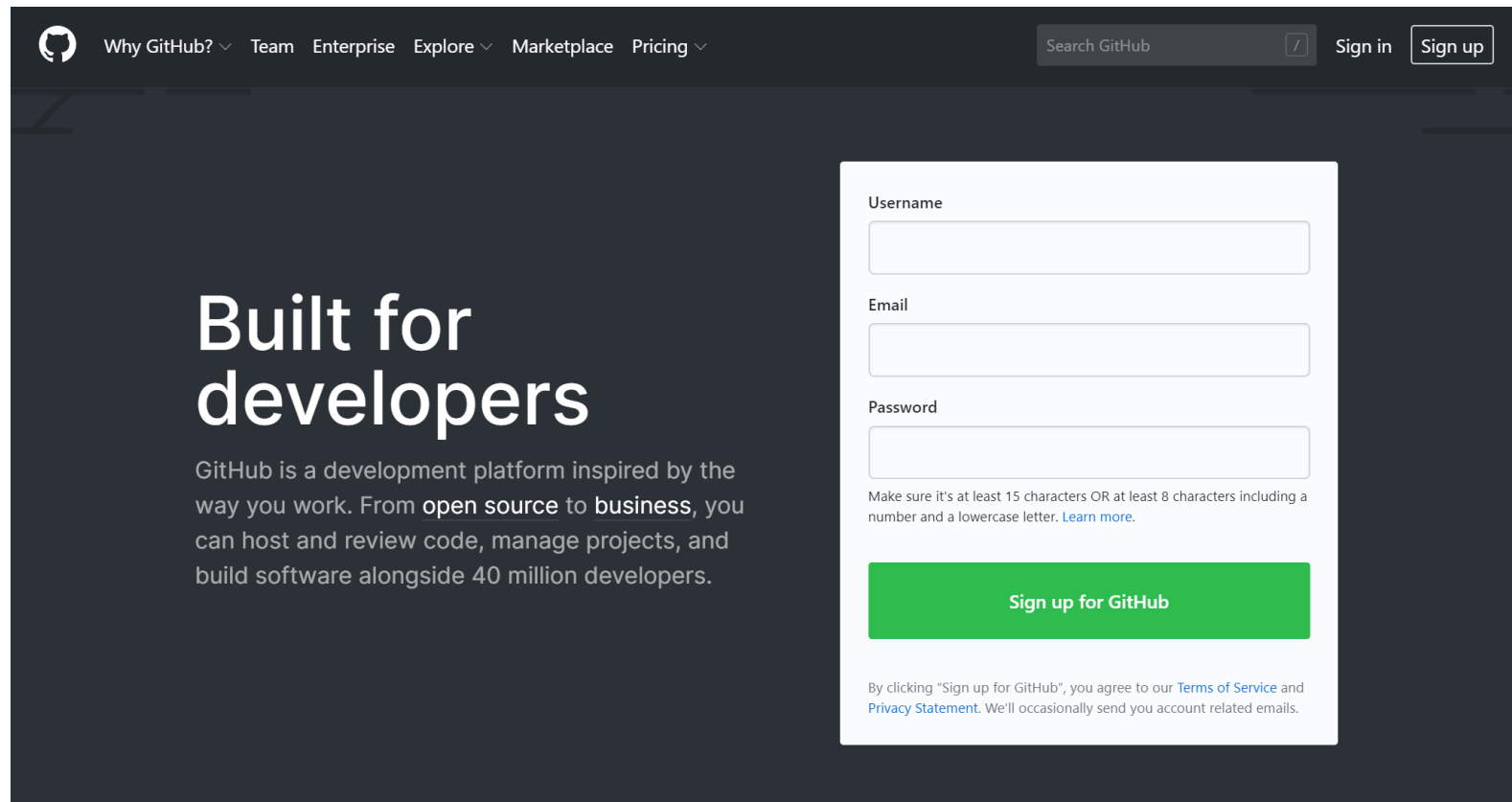
# git 설치

- <https://git-scm.com/> 에서 설치파일 다운로드
- 설치 후 'Git bash'에서 git 명령어 사용 가능



# Github 회원가입

- 설치 없이 Github 웹사이트에서 회원가입 후 서비스 이용 가능
  - <https://github.com/>



The screenshot shows the GitHub website's sign-up interface. The top navigation bar includes links for 'Why GitHub?', 'Team', 'Enterprise', 'Explore', 'Marketplace', and 'Pricing'. A search bar and 'Sign in'/'Sign up' buttons are on the right. The main content area features the text 'Built for developers' and a description of GitHub as a development platform. On the right, a sign-up form is displayed with fields for 'Username', 'Email', and 'Password'. Below the password field, there is a note about password requirements and a 'Sign up for GitHub' button. At the bottom of the form, a disclaimer states that clicking 'Sign up for GitHub' agrees to the 'Terms of Service' and 'Privacy Statement'.

Why GitHub? Team Enterprise Explore Marketplace Pricing

Search GitHub Sign in Sign up

## Built for developers

GitHub is a development platform inspired by the way you work. From open source to business, you can host and review code, manage projects, and build software alongside 40 million developers.

Username

Email

Password

Make sure it's at least 15 characters OR at least 8 characters including a number and a lowercase letter. [Learn more.](#)

Sign up for GitHub

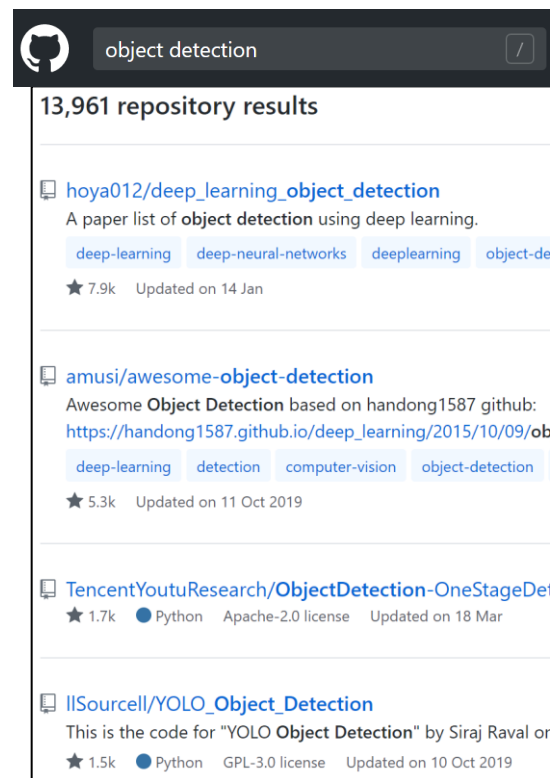
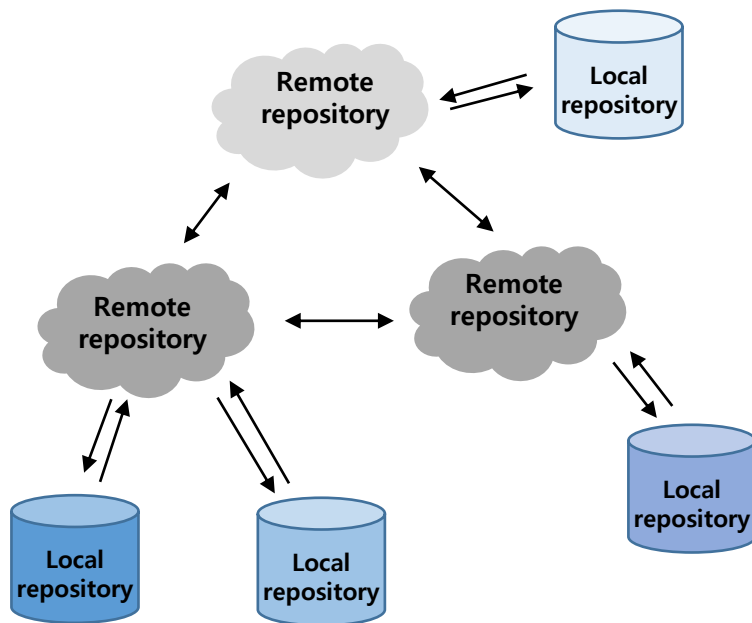
By clicking "Sign up for GitHub", you agree to our [Terms of Service](#) and [Privacy Statement](#). We'll occasionally send you account related emails.

# **LOCAL/REMOTE REPOSITORY**

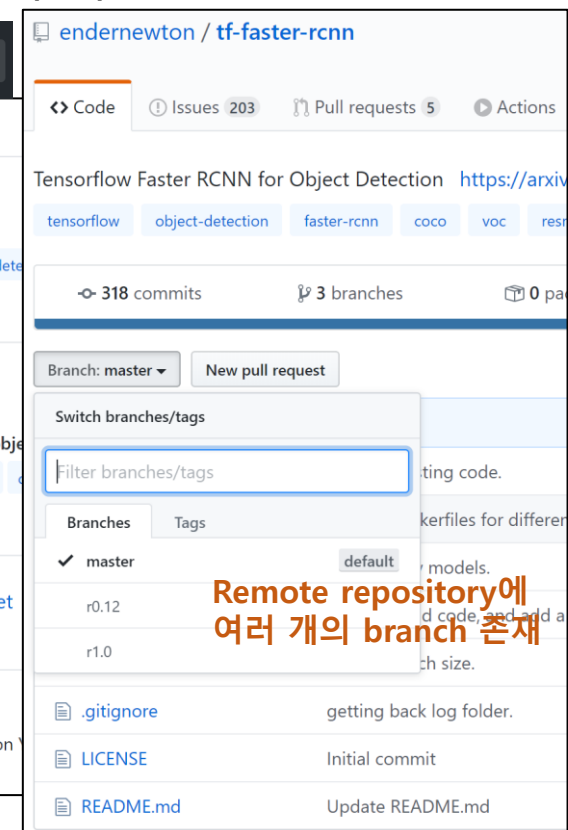
---

# Remote repository

- Remote repository는 git 원격 저장소이며 다른 사람과 공유할 수 있는 저장소
- 소스 파일, 이미지 파일, 텍스트 파일 등 저장 가능
- 하나의 remote repository에 여러 개 버전의 프로젝트(branch) 저장 가능
- Remote ↔ Remote, Remote ↔ Local 간 데이터 주고받기 가능



Remote repositories



Remote repository에  
여러 개의 branch 존재



# Remote Repository

- Remote repository 생성
  - 'New repository' 클릭 -> 옵션 선택

Create a new repository

A repository contains all project files, including the revision history. Already have a project repository elsewhere?  
[Import a repository.](#)

Owner: iloveori / Repository name:  ✓

Great repository name: VIPtest is available. [jorable](#). Need inspiration? How about [refactored-meme](#)?

Description (optional):

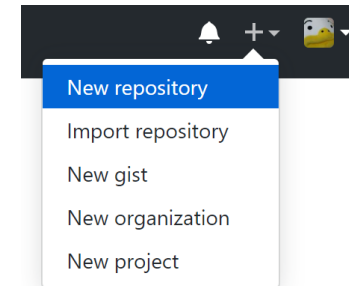
Public ☒ Anyone can see this repository. You choose who can commit.

Private ☐ You choose who can see and commit to this repository.

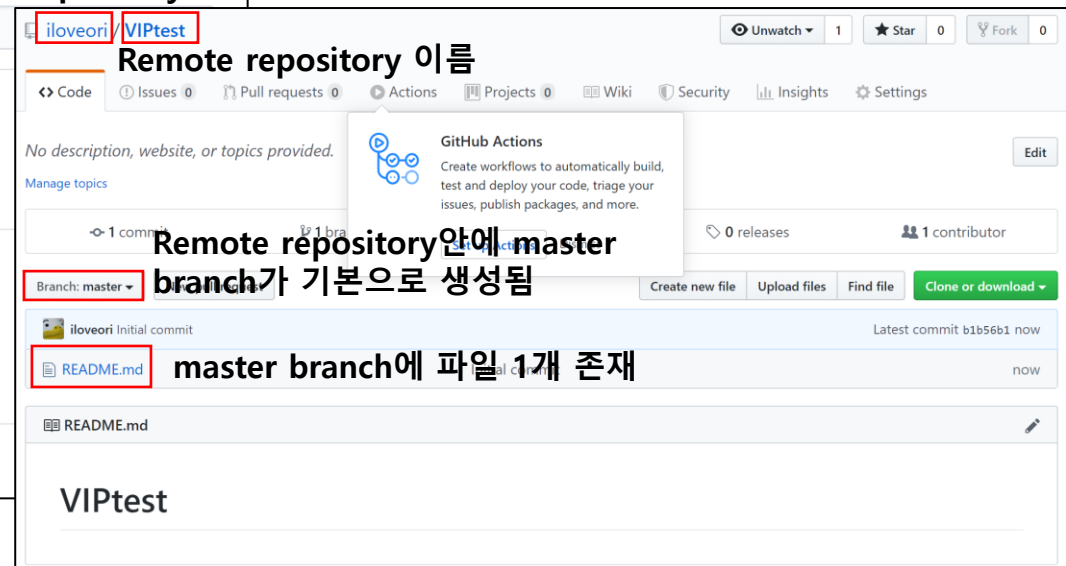
Skip this step if you're importing an existing repository.

☒ Initialize this repository with a README  
 This will let you immediately clone the repository to your computer.

Add .gitignore:  Add a license:  ⓘ



## Remote repository 소유자



# Local Repository

- 개발자의 pc의 저장소, .git 폴더 내부에 존재
- Remote repository와 마찬가지로 하나의 repository에 여러 버전의 프로젝트(branch) 생성 가능
- 생성 방법
  - 방법1: 디렉토리에 새 local repository 생성 (git init)
  - 방법2: Remote repository를 복사해옴(git clone 'remote repository 주소')

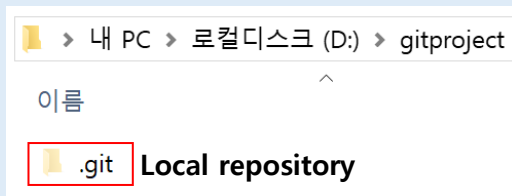
## 방법1

```

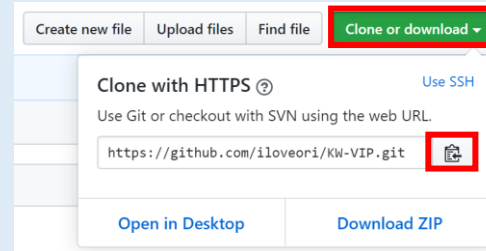
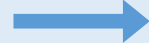
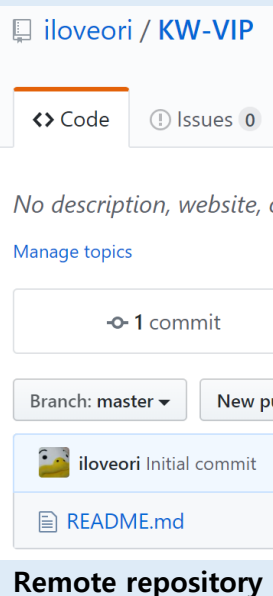
MINGW64/d/gitproject
$ cd D:\gitproject

MINGW64 /d/gitproject
$ git init
Initialized empty Git repository in D:/gitproject/.git/

MINGW64 /d/gitproject (master)
$ |
  
```



## 방법2



Remote repository 주소 복사



```

MINGW64 /d/LocalRepository
$ git clone https://github.com/iloveori/KW-VIP.git
Cloning into 'KW-VIP'...
remote: Enumerating objects: 3, done.
remote: Counting objects: 100% (3/3), done.
remote: Total 3 (delta 0), reused 0 (delta 0), pack-reused 0
Receiving objects: 100% (3/3), done.
  
```



# Git의 local 영역

## Local repository, working directory, staging area로 구분됨

### – Local repository

- .git 디렉토리 내 존재하며 프로젝트의 메타데이터와 객체 데이터베이스를 저장
- 여러 버전의 프로젝트 저장 가능

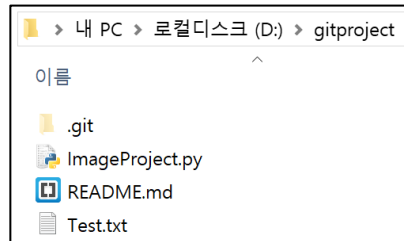
### – Working directory

- 실제 코드의 추가, 수정, 삭제 작업이 이루어지는 영역, .git directory가 존재하는 디렉토리
- 프로젝트의 특정 버전을 checkout하여 modify하는 영역

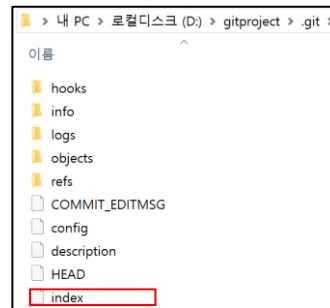
### – Staging area

- Git directory안에 있는 파일(index)이며 곧 커밋할 파일에 대한 정보를 저장

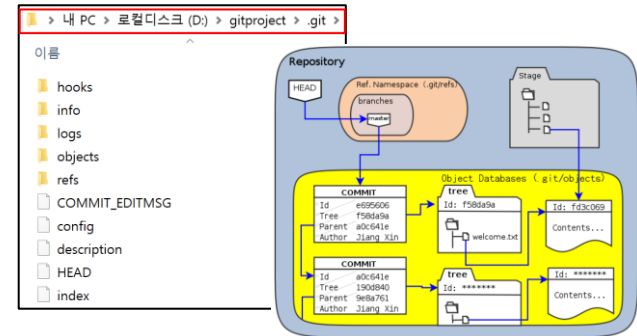
#### <Working directory>



#### <Staging area>



#### <Local Repository>



WD에서 파일 수정/  
생성/삭제

#### Working directory

- README.md
- Test.txt
- ImageProject.py

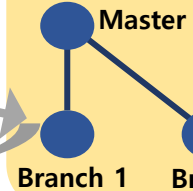
Local repository에  
반영하고 싶은 수정사  
항들을 staging area  
에 추가(add)

#### Staging area

- README.md
- Test.txt

수정사항을 local repo에  
반영(commit)

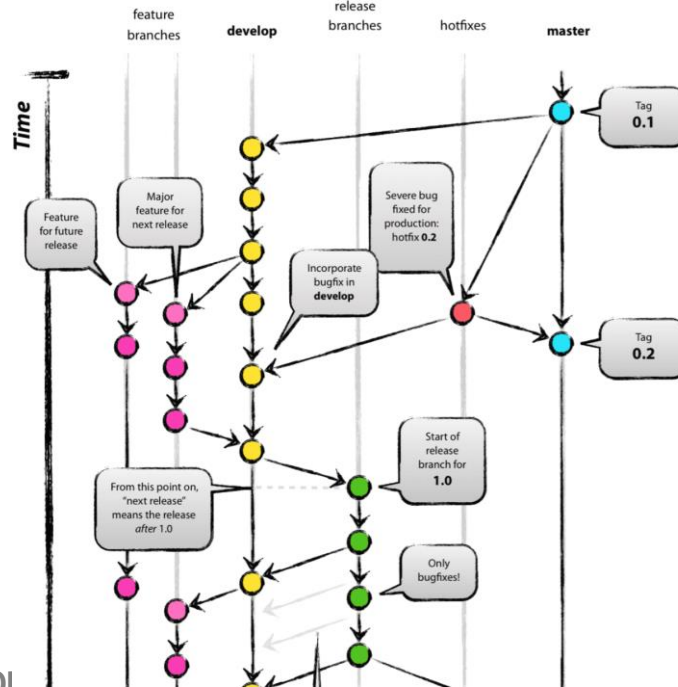
#### Local repository



WD로 특정 branch의 데이  
터 가져오기 (checkout)

# Branch

- Repository 내 일종의 독립적인 작업 공간
  - Branch를 통째로 복사하여 복사된 branch와 독립적으로 개발 진행
- 최초 Git 초기화시 기본적으로 "master"라는 브랜치가 생성됨
- 여러 갈래로 분기 또는 병합 가능
- Branch 종류
  - 항상 유지되는 메인 브랜치들(master, develop)
  - 일정 기간 동안만 유지되는 보조 브랜치들(feature, release, hotfix)

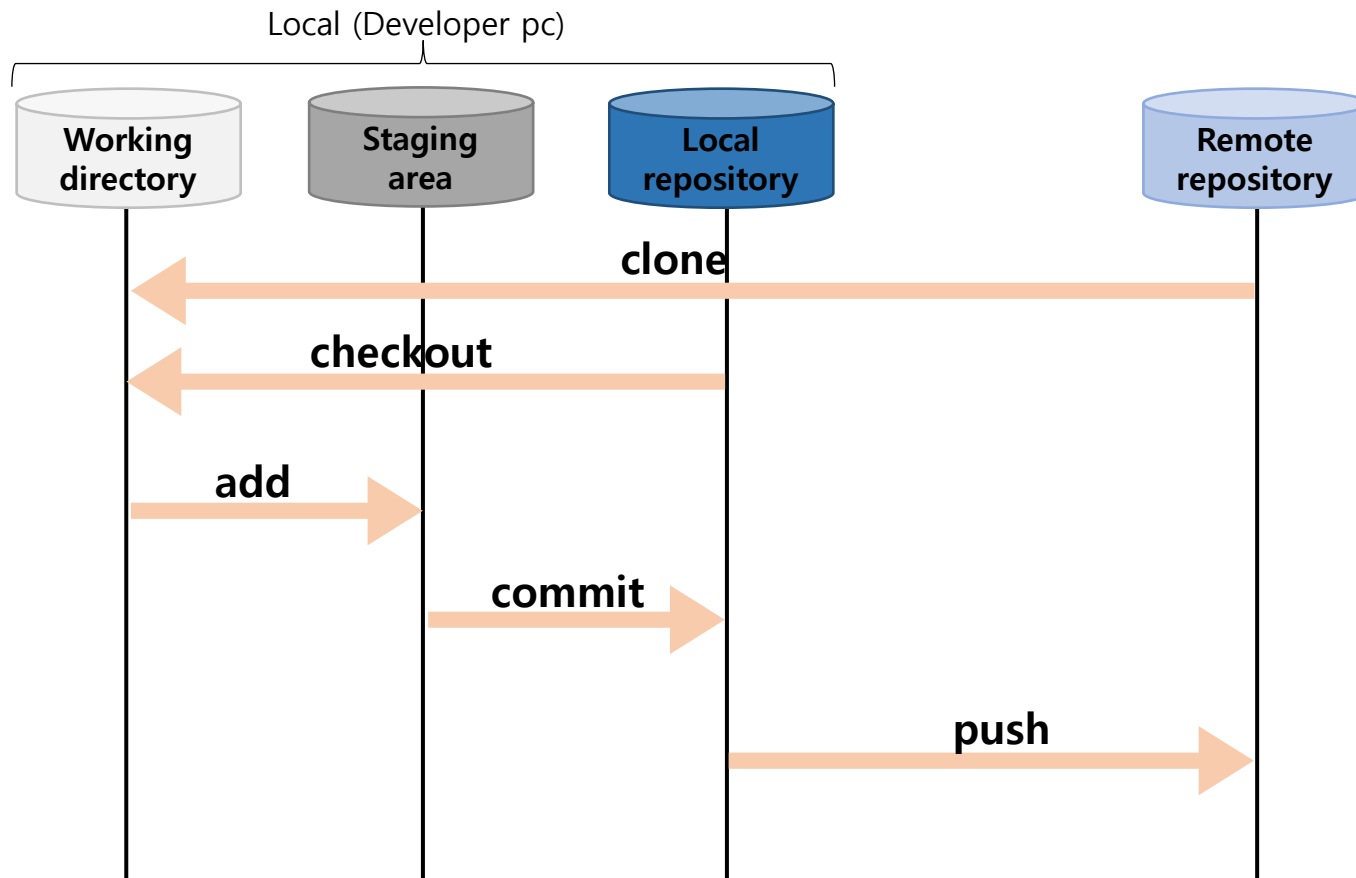


# 단독 개발을 위한 GIT WORKING FLOW 예제

---

# git working flow 예제

- 본인의 Remote repository의 파일을 로컬에서 수정하여 수정된 버전을 remote repository의 새 branch로 추가



# git Working flow

## ■ 본인의 Remote repository

The screenshot shows the GitHub interface for the repository 'iloveori / EdgeDetection'. The repository has 1 watch, 0 stars, and 0 forks. The 'Code' tab is selected, showing the repository's metadata and commit history. The '1 branch' link is highlighted with a red box. The commit history shows two commits: 'Initial commit' (6 minutes ago) and 'add main.py' (1 minute ago). The 'main.py' file is highlighted with a red box. The repository name 'EdgeDetection' is displayed at the bottom.

iloveori / EdgeDetection

Unwatch 1 Star 0 Fork 0

Code Issues 0 Pull requests 0 Actions Projects 0 Wiki Security Insights Settings

No description, website, or topics provided. Edit

Manage topics

2 commits 1 branch 0 packages 0 releases 1 contributor

Branch: master New pull request Create new file Upload files Find file Clone or download

iloveori add main.py Latest commit e268b11 1 minute ago

README.md	Initial commit	6 minutes ago
main.py	add main.py	1 minute ago

README.md

## EdgeDetection

# git Working flow

- Remote repository를 로컬로 clone
  - Local repository 생성됨
  - 현재 branch가 main branch로 설정되고, main branch의 파일들이 working directory에 셋팅 됨

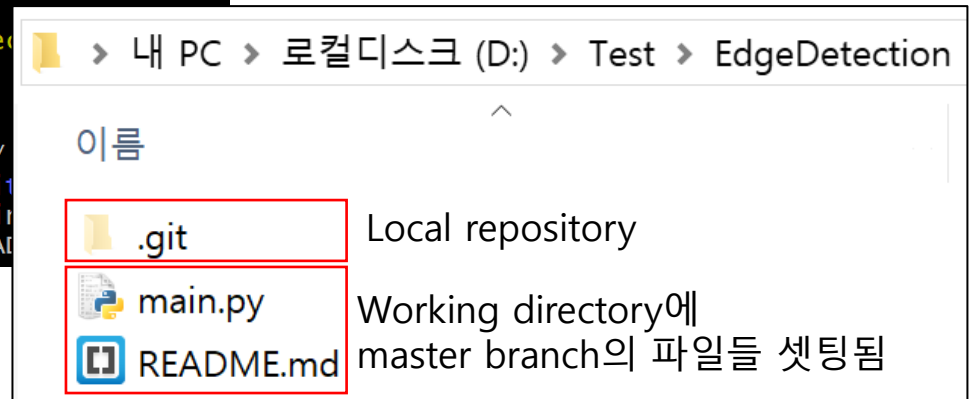
```

동글동글오리@DESKTOP-JVI18I3 MINGW64 /d/Test
$ git clone https://github.com/iloveori/EdgeDetection.git
Cloning into 'EdgeDetection'...
remote: Enumerating objects: 6, done.
remote: Counting objects: 100% (6/6), done.
remote: Compressing objects: 100% (3/3), done.
remote: Total 6 (delta 0), reused 3 (delta 0), pack-reused 0
Receiving objects: 100% (6/6), done.

동글동글오리@DESKTOP-JVI18I3 MINGW64 /d/Test
$ cd EdgeDetection/

동글동글오리@DESKTOP-JVI18I3 MINGW64 /d/Test/EdgeDetection
$ ls -al
total 6
drwxr-xr-x 1 동글동글오리 197121 0 4월 18 15:04 ./
drwxr-xr-x 1 동글동글오리 197121 0 4월 18 15:04 ../
drwxr-xr-x 1 동글동글오리 197121 0 4월 18 15:04 .git
-rw-r--r-- 1 동글동글오리 197121 20 4월 18 15:04 main.py
-rw-r--r-- 1 동글동글오리 197121 15 4월 18 15:04 README.md

```





# git Working flow

- Branch를 생성 후 해당 branch로 이동
  - 이후 새 branch에서 파일 수정 예정

```
동 글 동 글 오 리 @DESKTOP-JVI18I3 M
$ git branch
* master

동 글 동 글 오 리 @DESKTOP-JVI18I3 M
$ git branch version2

동 글 동 글 오 리 @DESKTOP-JVI18I3 M
$ git branch
* master
  version2

동 글 동 글 오 리 @DESKTOP-JVI18I3 M
$ git checkout version2
Switched to branch 'version2'

동 글 동 글 오 리 @DESKTOP-JVI18I3 M
$ git branch
master
* version2
```

- Branch 목록과 현재 branch 출력
- **git branch**

## Working directory

- master branch의 파일들

## Local repository

● master

- branch 생성
- **git branch '브랜치 이름'**

## Working directory

- master branch의 파일들

## Local repository

● master  
● version2

- 새 branch로 checkout
- **git checkout '브랜치 이름'**

## Working directory

- version2 branch의 파일들

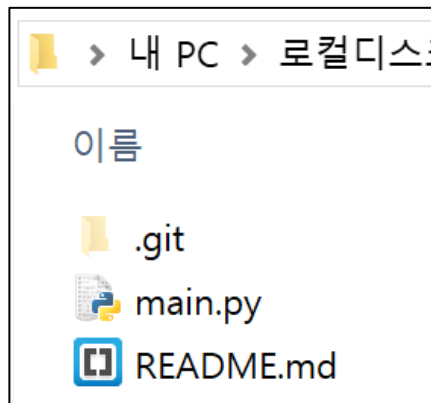
## Local repository

● master  
● version2

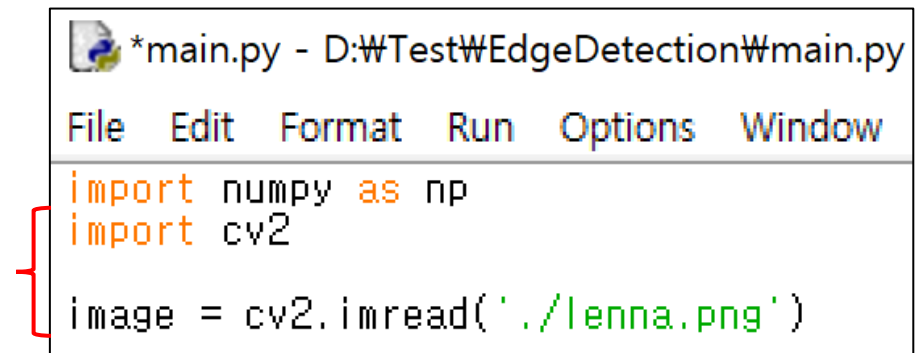
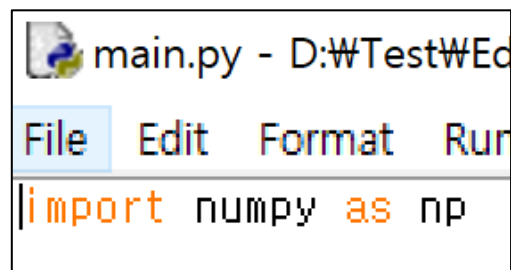
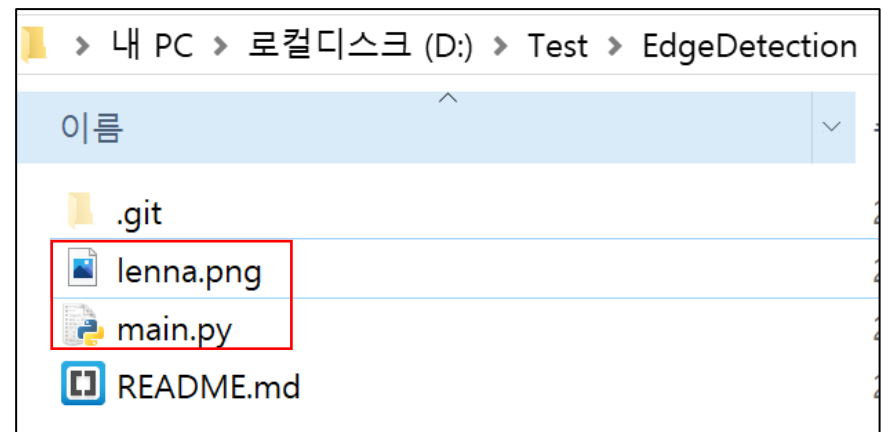
# git Working flow

- Working directory로 checkout된 branch의 파일 수정

## ➤ 파일 수정 전 working directory



## ➤ 파일 수정 후 working directory



# git Working flow

- Local repository에 반영하고 싶은 수정사항을 staging area에 추가

```
동글동글오리 @DESKTOP-JVI18I3 MINGW64 /d/Test/EdgeDet
$ git add main.py

동글동글오리 @DESKTOP-JVI18I3 MINGW64 /d/Test/EdgeDet
$ git status
On branch version2
Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
    modified:   main.py

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

동글동글오리 @DESKTOP-JVI18I3 MINGW64 /d/Test/EdgeDet
$ git add --all

동글동글오리 @DESKTOP-JVI18I3 MINGW64 /d/Test/EdgeDet
$ git status
On branch version2
Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
    new file:   lenna.png
    modified:   main.py
```

- 특정 파일을 staging area에 추가
- **git add '파일 이름'**

- 파일 상태 확인
- **git status**

## Working directory

- version2 branch의 파일들

추가

## Staging area

main.py

- 모든 변경사항을 staging area에 추가
- **git add --all**

## Working directory

- version2 branch의 파일들

추가

## Staging area

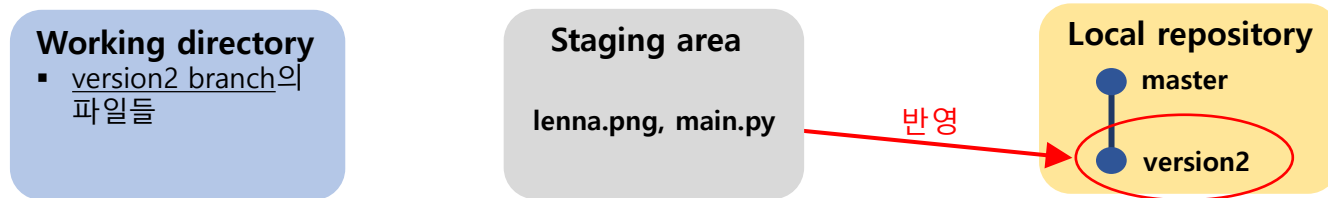
lenna.png, main.py

# git Working flow

- Staging area의 내용을 local repository에 반영

➤ `git commit -m 'commit 메시지'`

```
동글동글오리@DESKTOP-JVI18I3 MINGW64 /d/Test/EdgeDetection (version2)
$ git commit -m 'project version2 : modify code, add image'
[version2 ca18b48] project version2 : modify code, add image
2 files changed, 5 insertions(+)
create mode 100644 lenna.png
```



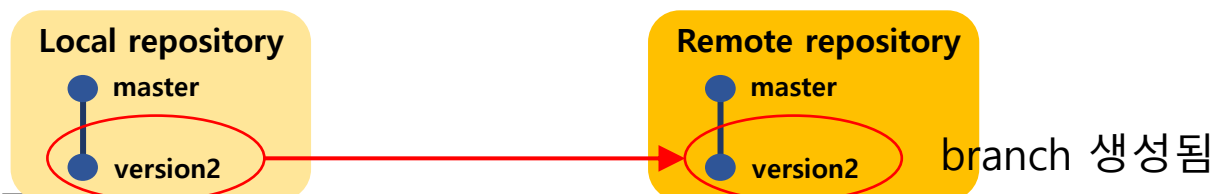
# git Working flow

## ▪ Remote repository로 local branch push

- 등록된 리모트 저장소 출력 `git remote`
- 로컬 브랜치를 리모트로 전송 `git push 'remote' 'local branch'`

```
동글동글오리 @DESKTOP-JVI18I3 MINGW64 /d/Test/EdgeDetection (version2)
$ git remote
origin

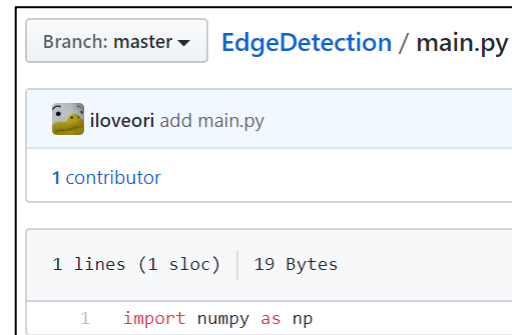
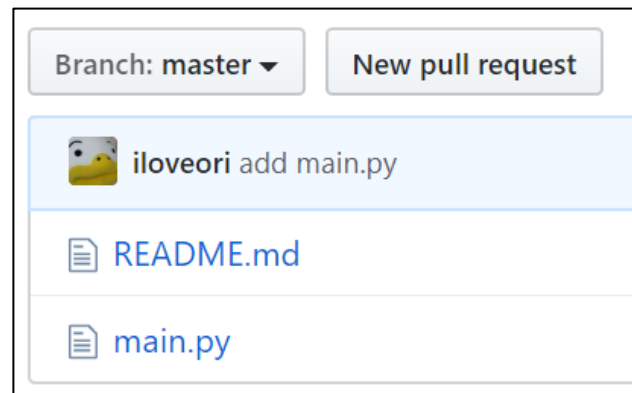
동글동글오리 @DESKTOP-JVI18I3 MINGW64 /d/Test/EdgeDetection (version2)
$ git push origin version2
Enumerating objects: 6, done.
Counting objects: 100% (6/6), done.
Delta compression using up to 8 threads
Compressing objects: 100% (4/4), done.
Writing objects: 100% (4/4), 463.30 KiB | 24.38 MiB/s, done.
Total 4 (delta 0), reused 0 (delta 0), pack-reused 0
remote:
remote: Create a pull request for 'version2' on GitHub by visiting:
remote:   https://github.com/iloveori/EdgeDetection/pull/new/version2
remote:
To https://github.com/iloveori/EdgeDetection.git
 * [new branch]      version2 -> version2
```



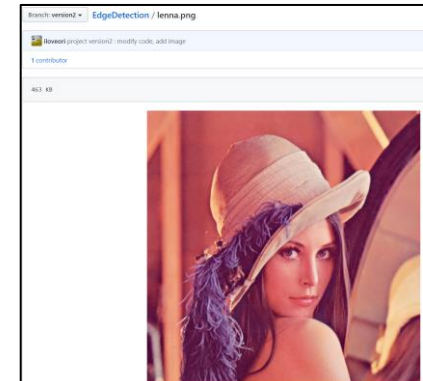
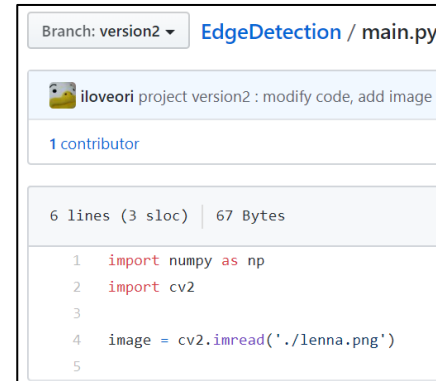
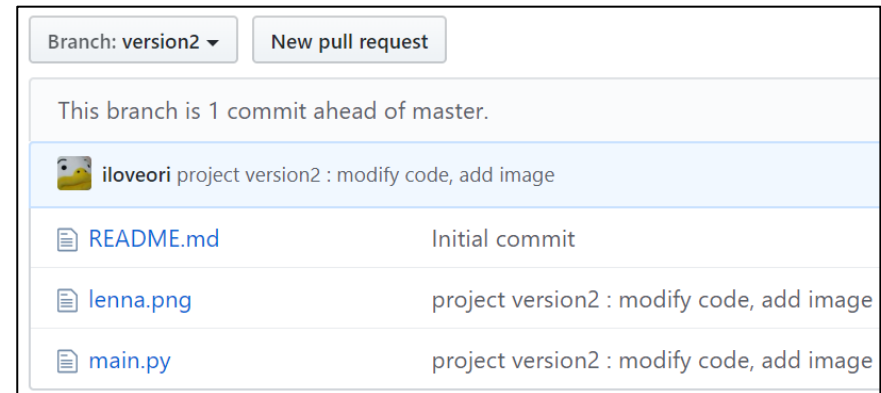
# git Working flow

## ▪ Remote repository의 최종 결과

➤ master branch는 내용 변화 없음



➤ version2 branch 생성됨

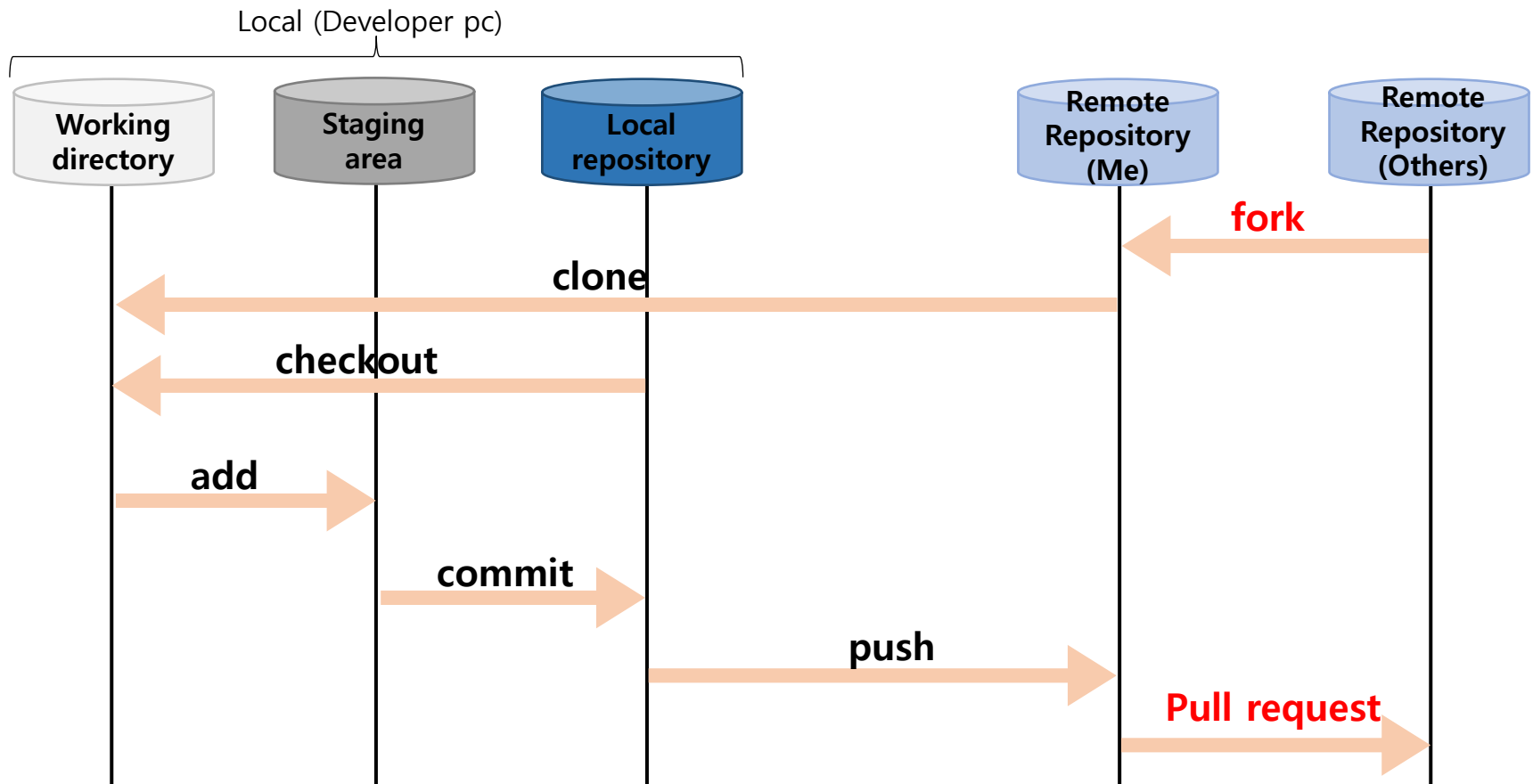


# 공동 개발을 위한 GIT WORKING FLOW

---

# git working flow 예제

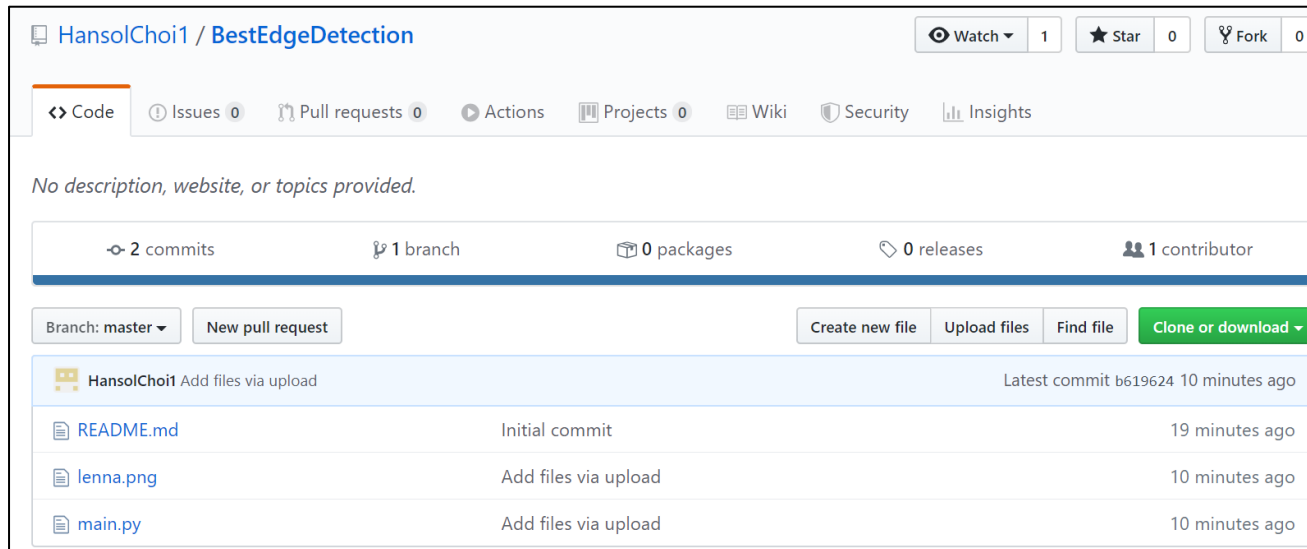
- 다른 개발자의 remote repository를 복사하여 로컬에서 내용을 수정 -> 수정 사항을 반영해 줄 것을 개발자에게 요청





# git Working flow

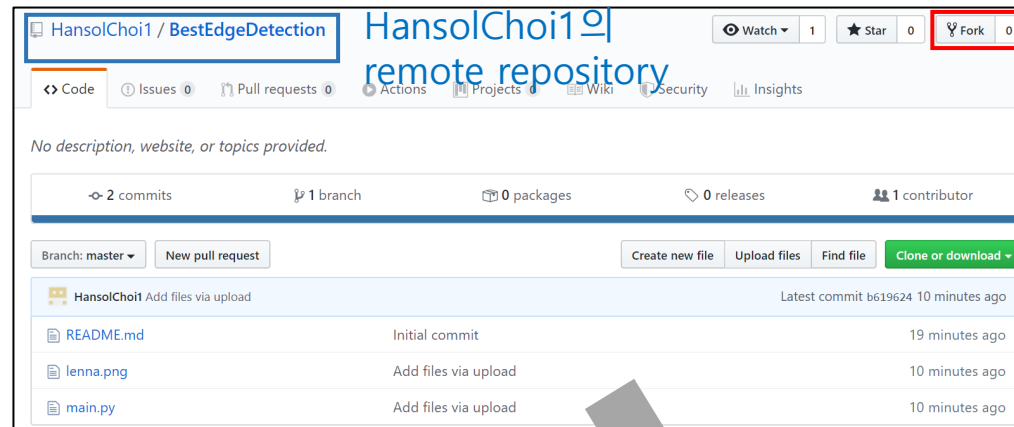
- 다른 개발자의 remote repository



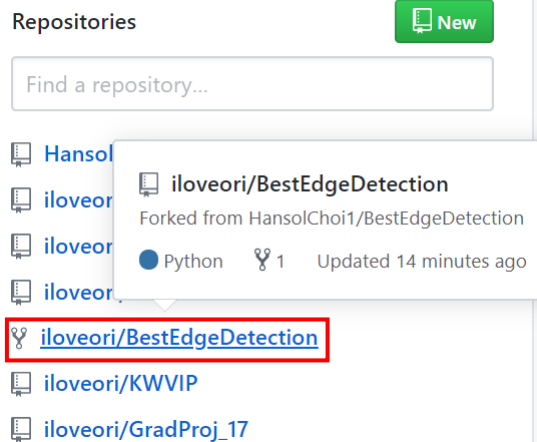
```
1 import cv2
2 import numpy as np
3 from matplotlib import pyplot as plt
4
5 img = cv2.imread('lenna.png', cv2.IMREAD_COLOR)
6
7 edges = cv2.Canny(img,100,200)
8
9 plt.subplot(121),plt.imshow(img,cmap = 'gray')
10 plt.title('Original Image'), plt.xticks([], plt.yticks([]))
11 plt.subplot(122),plt.imshow(edges,cmap = 'gray')
12 plt.title('Edge Image'), plt.xticks([], plt.yticks([]))
13
14 plt.show()
```

# git Working flow

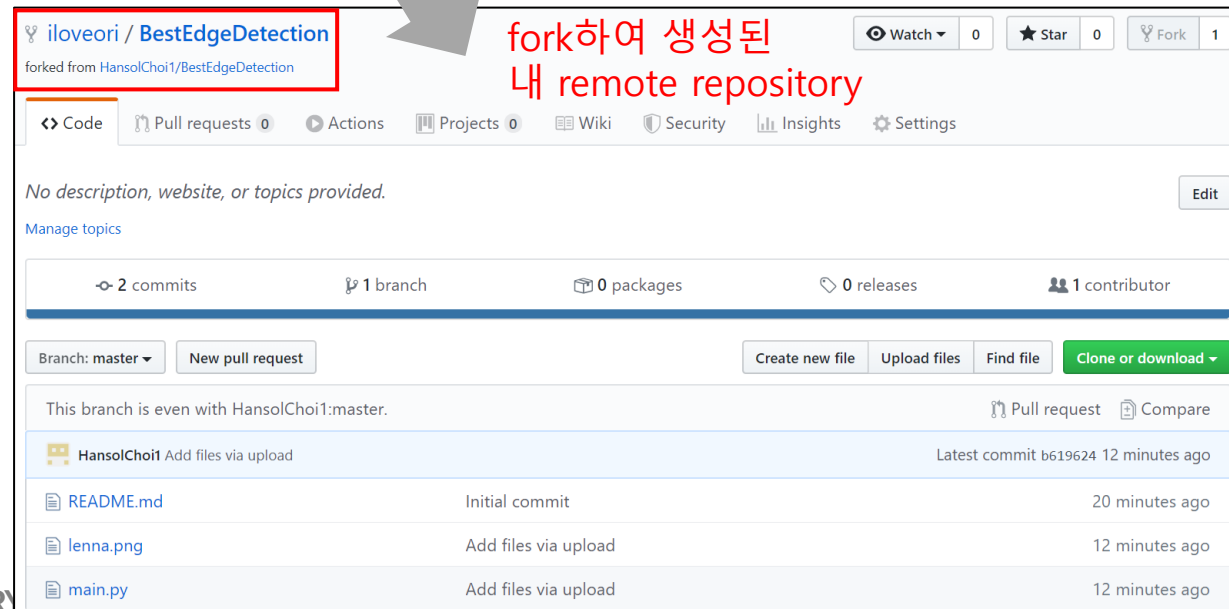
- 다른 개발자의 remote repository를 내 remote repository로 복사  
=> fork



내 remote repository 목록

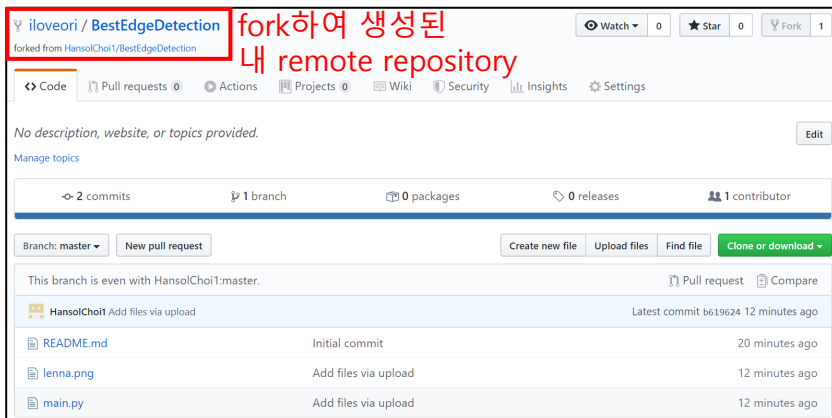


fork하여 생성된  
내 remote repository

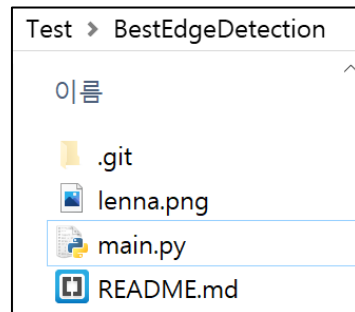


# git Working flow

- Fork하여 생성된 내 프로젝트의 코드를 수정하고 내 remote repository의 새 branch로 push
  - Clone->코드수정->add->commit->push

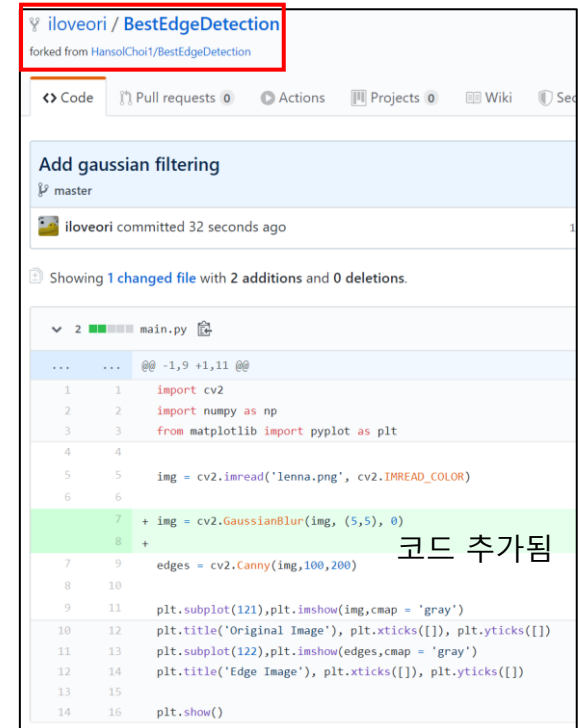


clone



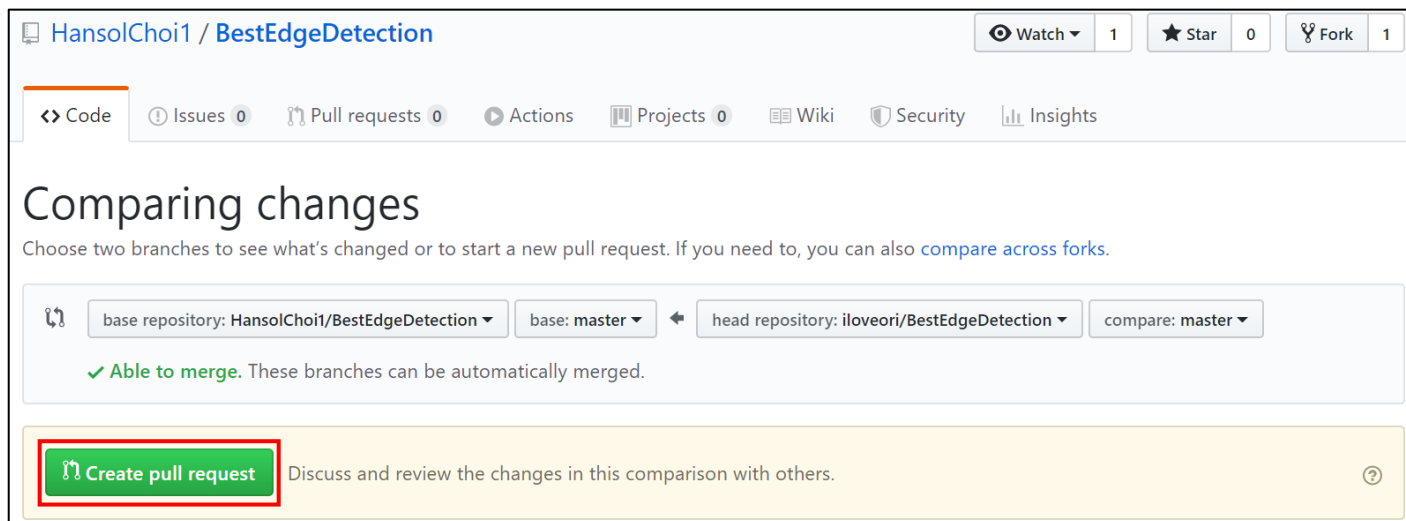
로컬에서 main.py  
코드 수정

push



# git Working flow

- 본인이 fork했던 프로젝트의 개발자에게 pull request를 전송하여 나의 수정사항을 반영해 달라고 요청



# git Working flow

- Pull request를 받은 개발자는 수락 또는 거부 가능

The screenshot displays the GitHub interface for a repository named 'HansolChoi1 / BestEdgeDetection'. The 'Pull requests' tab is active, showing a list of pull requests. One pull request, titled 'Add gaussian filtering', is highlighted with a red box. It was opened 27 seconds ago by the user 'iloveori'. The pull request details are shown on the right, including the title 'Add gaussian filtering #1', the description 'iloveori wants to merge 1 commit into HansolChoi1:master from iloveori:master', and the commit hash '2c65ecd'. The pull request is currently open. At the bottom of the pull request details, there is a green button labeled 'Merge pull request' with a dropdown arrow, which is also highlighted with a red box. The button text indicates that you can also open this in GitHub Desktop or view command line instructions.

# git Working flow

- Pull request 수락 -> 내가 요청한 수정사항이 반영됨



HansolChoi1 / BestEdgeDetection

Code Issues 0 Pull requests 0 Actions Projects

Branch: master BestEdgeDetection / main.py / <> Jump to

iloveori Add gaussian filtering

2 contributors

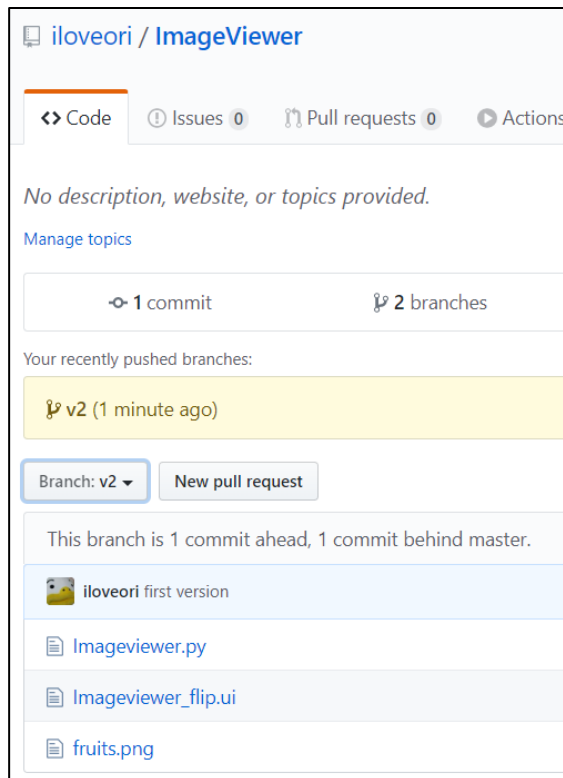
16 lines (11 sloc) 428 Bytes

```
1 import cv2
2 import numpy as np
3 from matplotlib import pyplot as plt
4
5 img = cv2.imread('lenna.png', cv2.IMREAD_COLOR)
6
7 img = cv2.GaussianBlur(img, (5,5), 0)
8
9 edges = cv2.Canny(img,100,200)
10
11 plt.subplot(121),plt.imshow(img,cmap = 'gray')
12 plt.title('Original Image'), plt.xticks([], plt.yticks([]))
13 plt.subplot(122),plt.imshow(edges,cmap = 'gray')
14 plt.title('Edge Image'), plt.xticks([], plt.yticks([]))
15
16 plt.show()
```

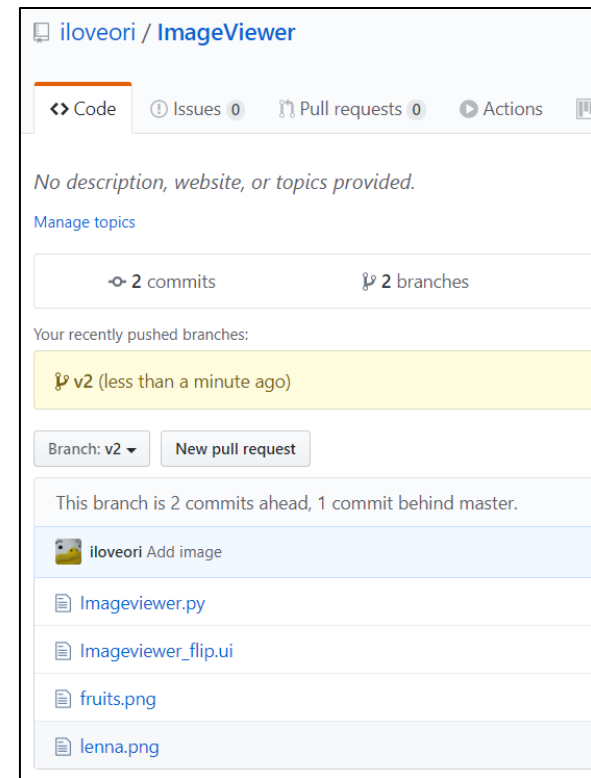
# 3차 과제

## ▪ 3-1) 본인의 Github repository에 2차 과제 파일들을 업로드한 뒤 파일을 수정/추가하기

- 강의자료에 설명된 과정들(clone, checkout, add, commit, push)과 그 이외의 방법 사용 가능
- 파일 내용 수정/업로드 시 local, remote repository의 branch 생성은 자유롭게 수행 가능

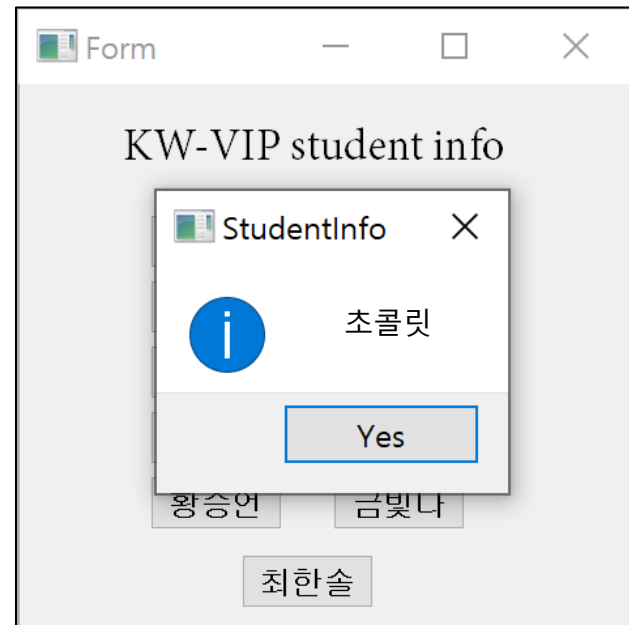
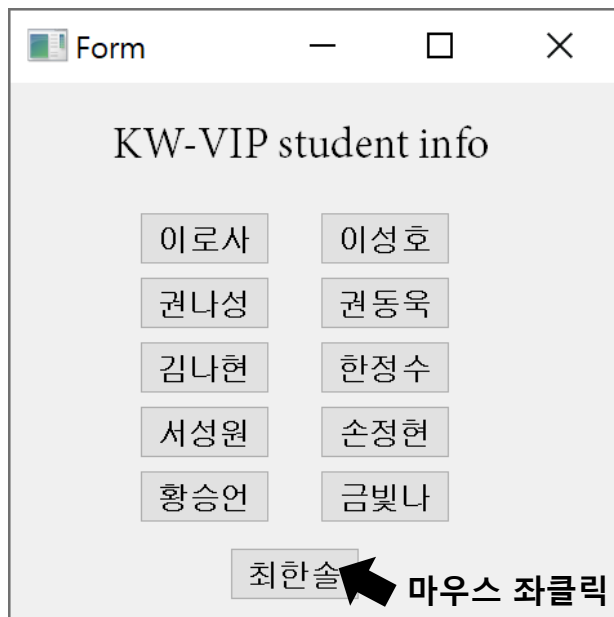


lenna.png 파일 추가



## 3차 과제

- 3-2) 조교의 Github repository에서 본인에게 할당된 코드 부분을 수정하고 pull request 수행
  - <https://github.com/iloveori/KW-VIP-StudentID>
  - Github repository의 프로젝트는 아래와 같이 동작함

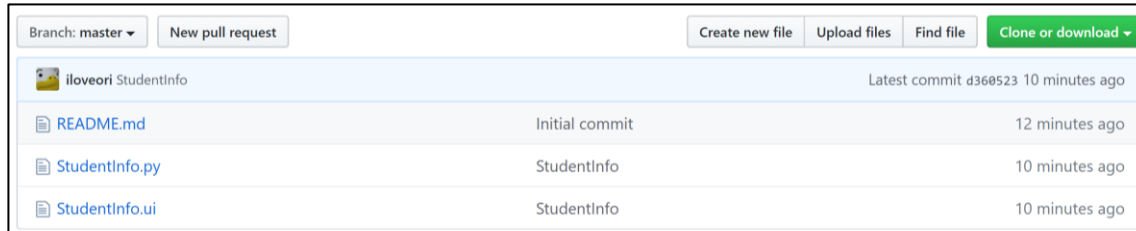


- 학생의 이름으로 된 버튼을 클릭할 시 해당 학생의 학번이 Message box의 내용으로 출력되는 프로그램



# 3차 과제

- Github repository에는 master branch만 존재하고, 따라서 pull request도 master branch로 수행해야 됨



```
def ShowInfo(self, num):
    studentid = ''

    if num == 1: #이로사
        studentid = ""

    elif num == 2: #권나성
        studentid = ""

    elif num == 3: #김나현
        studentid = ""

    elif num == 4: #서성원
        studentid = ""

    elif num == 5: #황승언
        studentid = ""
```

```
    elif num == 6: #이성호
        studentid = ""

    elif num == 7: #권동욱
        studentid = ""

    elif num == 8: #한정수
        studentid = ""

    elif num == 9: #손정현
        studentid = ""

    else :
        #금빛나
        studentid = ""
```

<StudentInfo.py의 ShowInfo함수>

- '#본인이름' 주석에 해당되는 부분에 본인의 별명/취미/좋아하는음식 등 짧은 텍스트를 작성 후 pull request 수행
- ex) studentid = "초콜릿"

# 3차 과제

---

- 제출물
  - 과제 수행 과정에 대한 설명과 캡처를 보고서의 형태로 작성한 것
    - 3-1)의 학생 본인의 Github repository 주소 또한 보고서에 포함
- 제출 장소
  - Klas 과제 제출란 '3차 과제'
- 제출 기한
  - 4월 28일 23:59까지

# END OF PRESENTATION

---

Q&A