

# Git with PyCharm



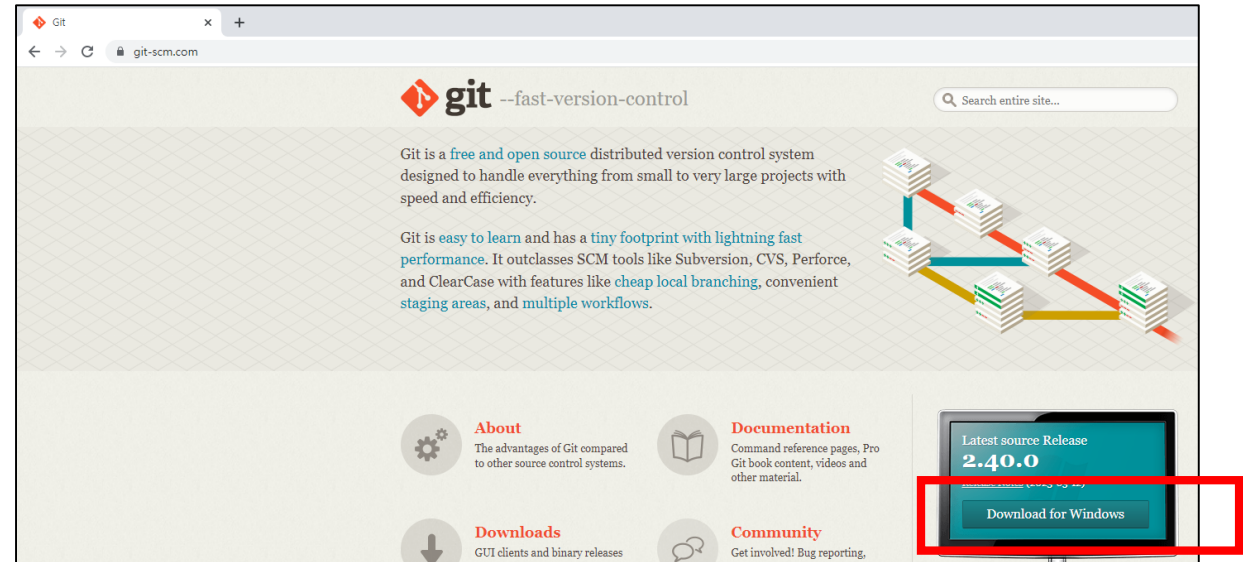
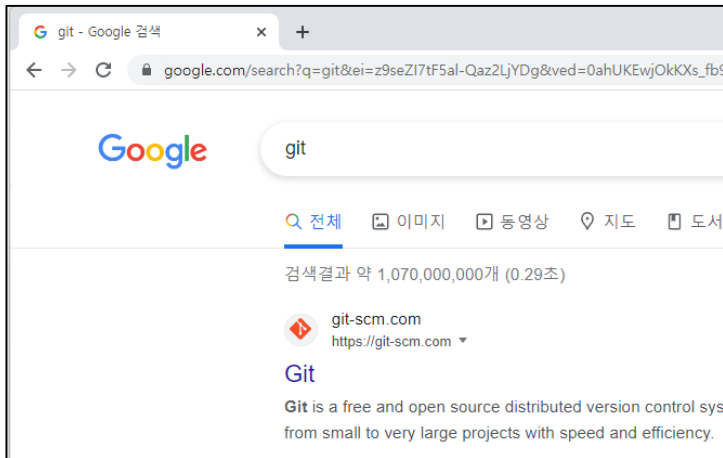
# Git Clone

## 목표 : 소스코드를 모두 Clone 받는다.

- PyCharm 와 Github 을 연동하여,  
Github에 있는 소스코드를 PyCharm 로 전체 복제한다.
- git 명령어 "Clone" 은 다음 내용을 한방에 처리한다.
  1. remote repo. 내용을 전체 복사해온다.
  2. local repo를 생성한다.
  3. local repo와 github을 연결 한다.

# Git 설치 부터 시작하자.

- Local PC에 Git 을 설치한다.
  1. 구글링 : git, 공식 사이트 접속하기
  2. Download 받고, 별 다른 설정없이 Next만 눌러 설치를 완료한다.
  3. 사용중인 PyCharm를 종료했다 다시 켜다.



# Clone 을 위한 Github 로그인

File > Project from VC > Github 에서 로그인

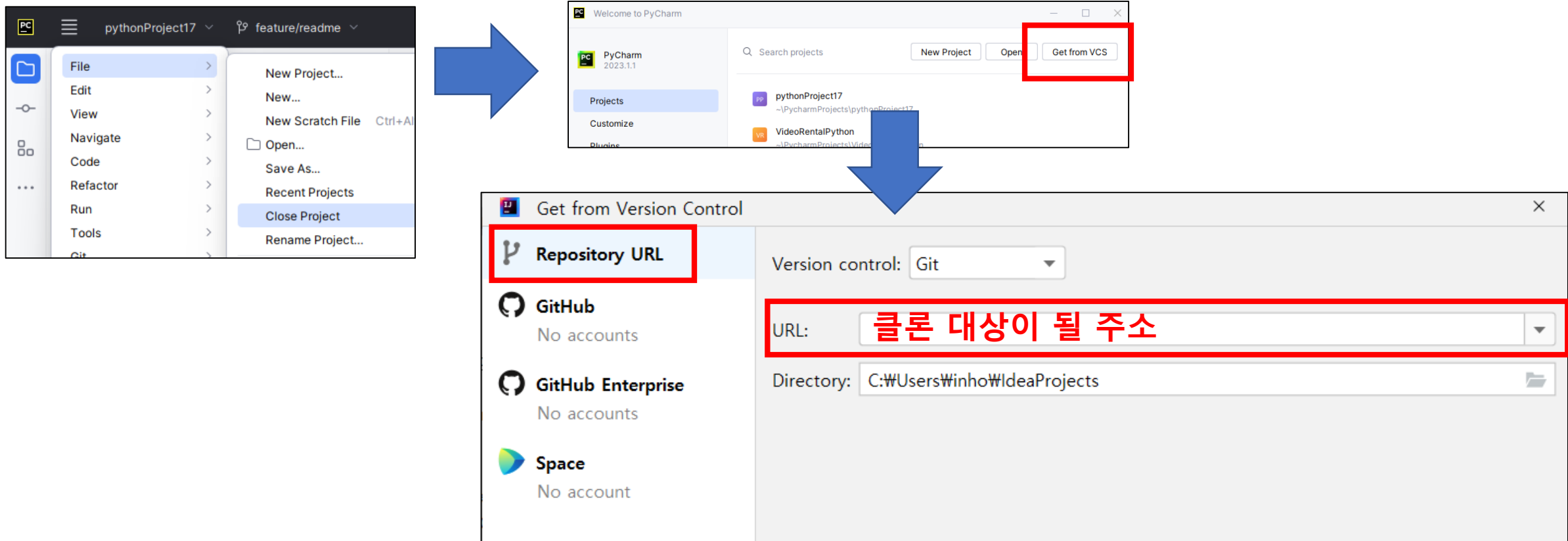
• OAuth 인증방식 or PAT 토큰 인증 방식으로 로그인을 해야한다.

화면 하단에 Generate Token 클릭

다음과 같이 Token이 생성된다.  
이 Token을 복사 붙여넣기로 넣고 로그인

# Git Clone 받기

- 앞으로 Git Clone 시,  
다음 URL에 Git 주소를 입력한 후, Clone 누르면 된다.

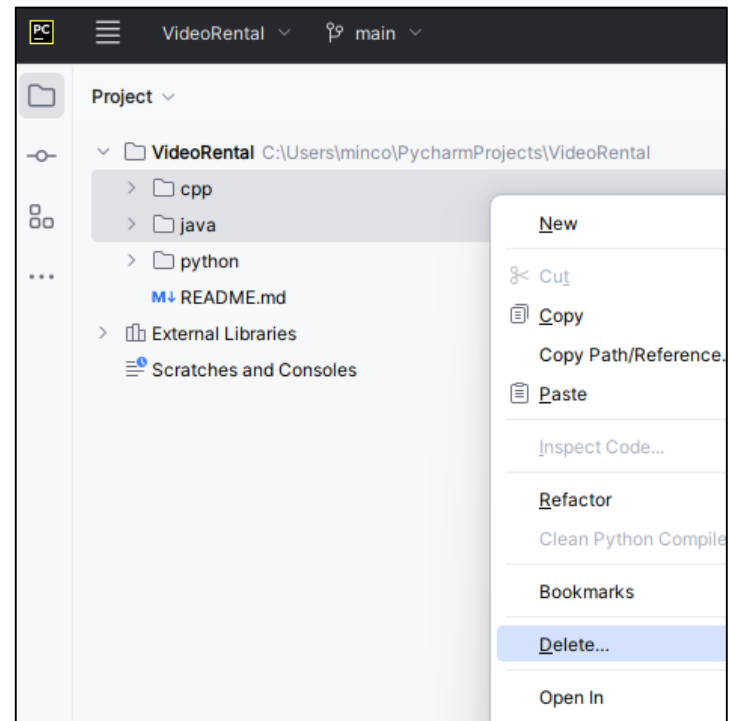


## 세팅 가이드를 위한 샘플 Clone

- 실습 카타 다운로드 (VideoRental KATA 로 실습)  
<https://github.com/mincoding1/VideoRental/>

# 불필요한 자료는 제거

- cpp, java 폴더 제거

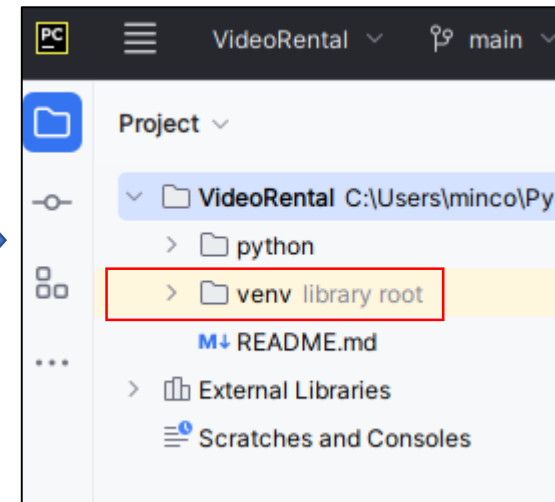
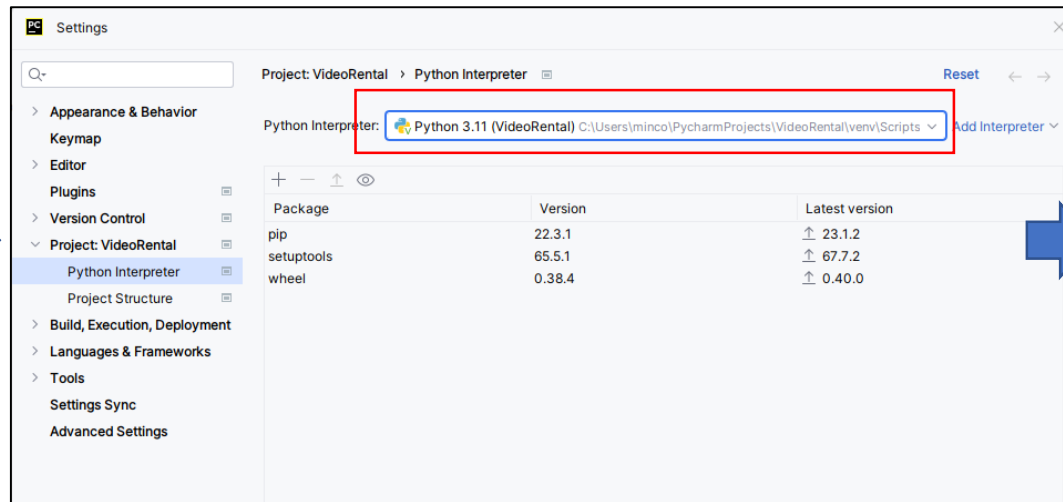
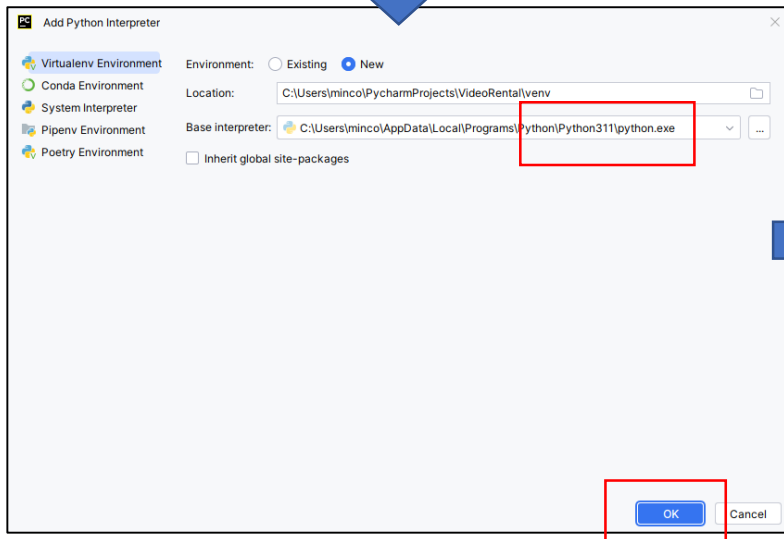




# venv 추가 / Interpreter 버전 맞추기 (3.11버전)



File > Setting >  
Python Interpreter >  
Add Interpreter



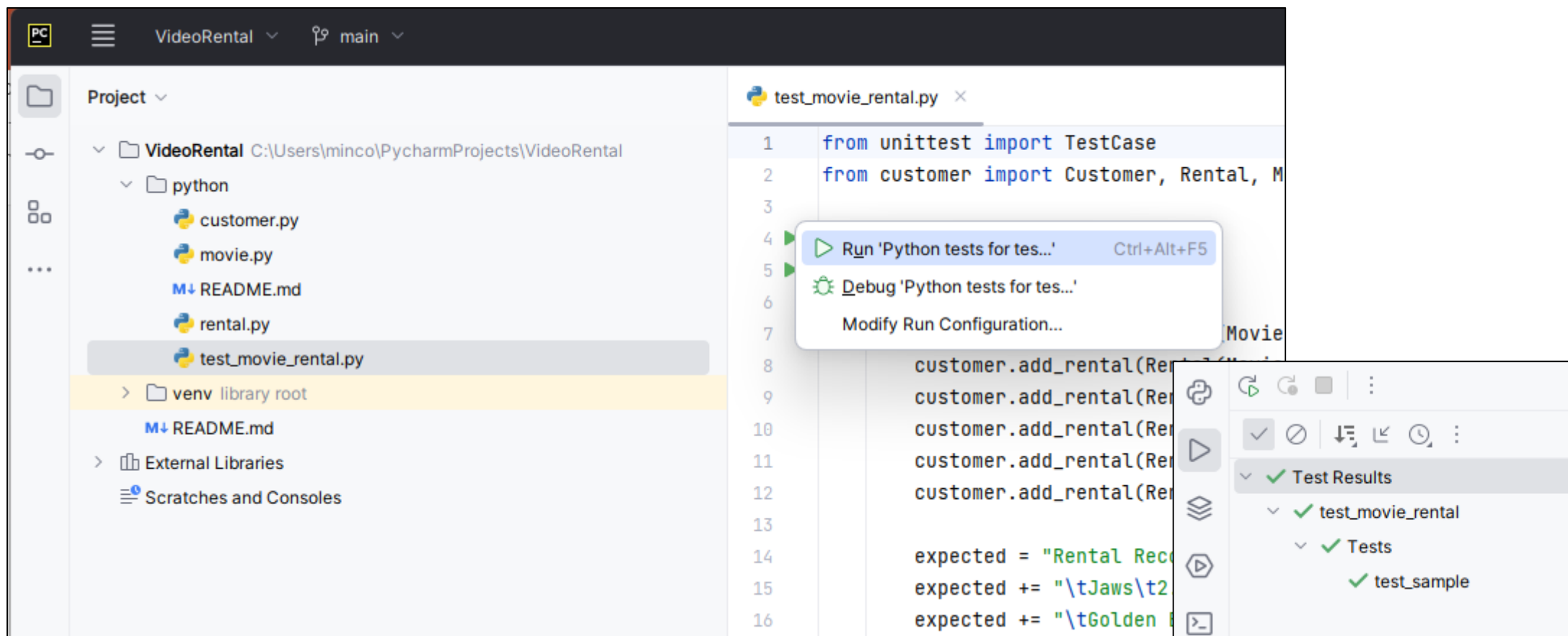
python 3.11 + venv 를 새로 하나 생성한다.

잠시 기다리면, venv가 추가되고,  
인터프리터가 선택된다.

venv 가 추가됨을  
확인할 수 있음

# 결과 확인

- unittest가 잘 동작되는지 확인한다.

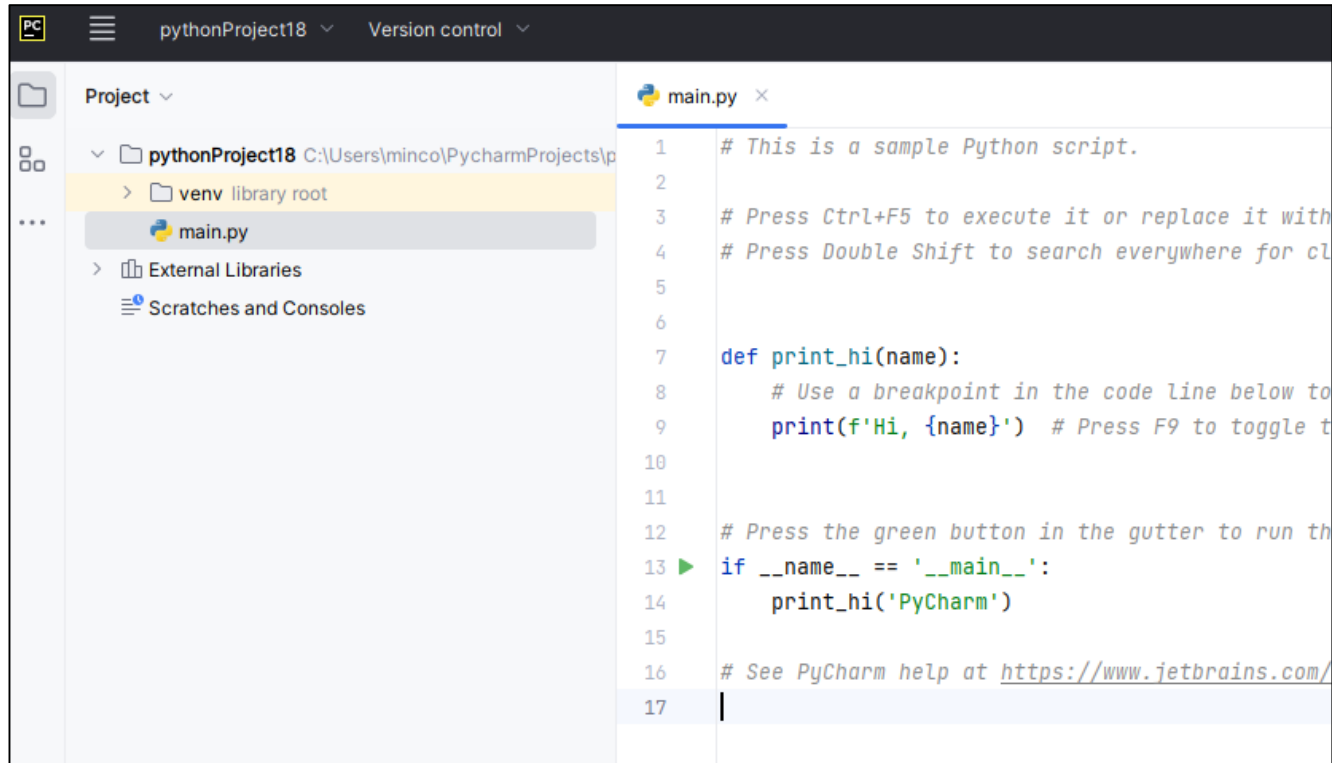




# PyCharm 에서 Git GUI 사용

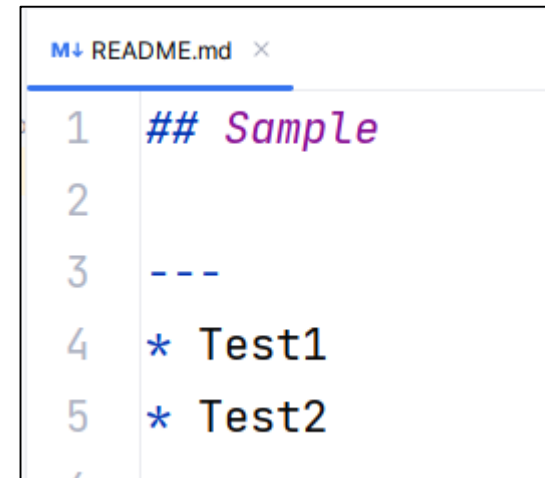
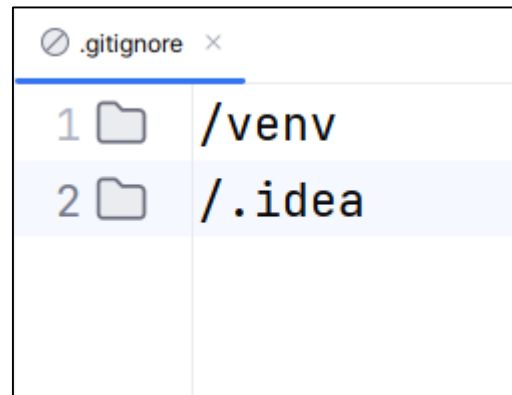
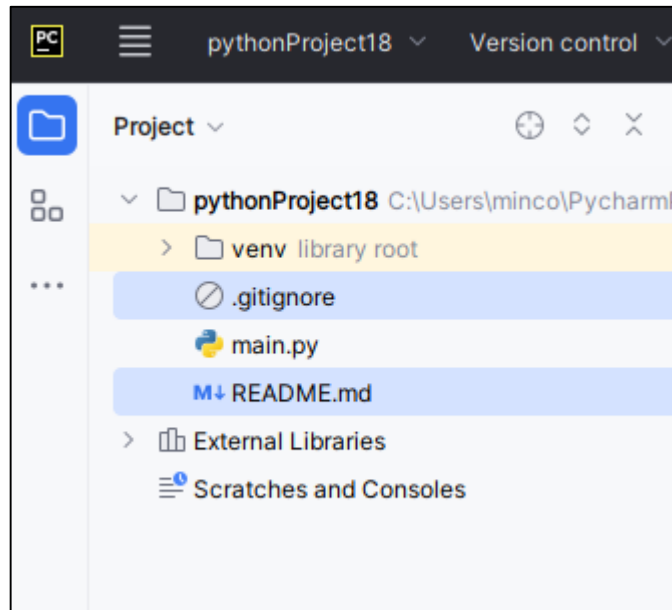
# 신규 프로젝트 생성

- Git Push 테스트를 위한 신규 프로젝트



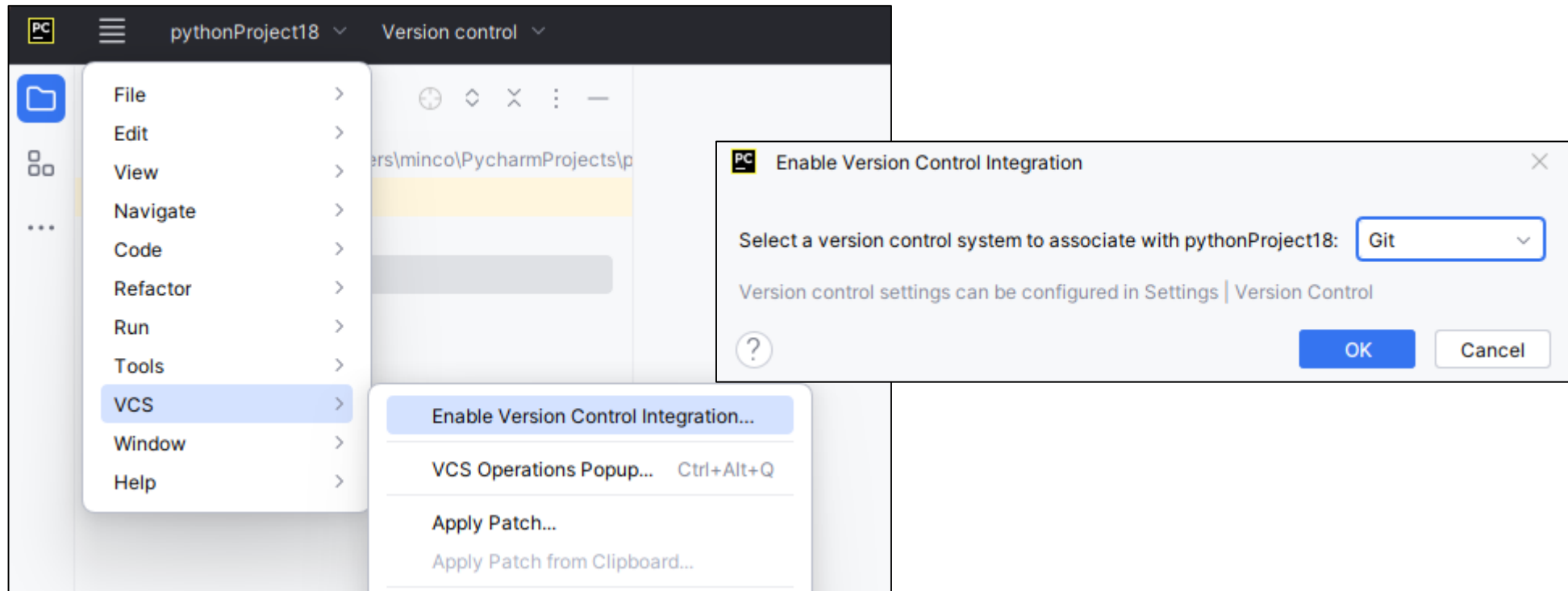
# README, .gitignore 추가

- 두 개의 파일을 추가 후, 내용을 기입한다.



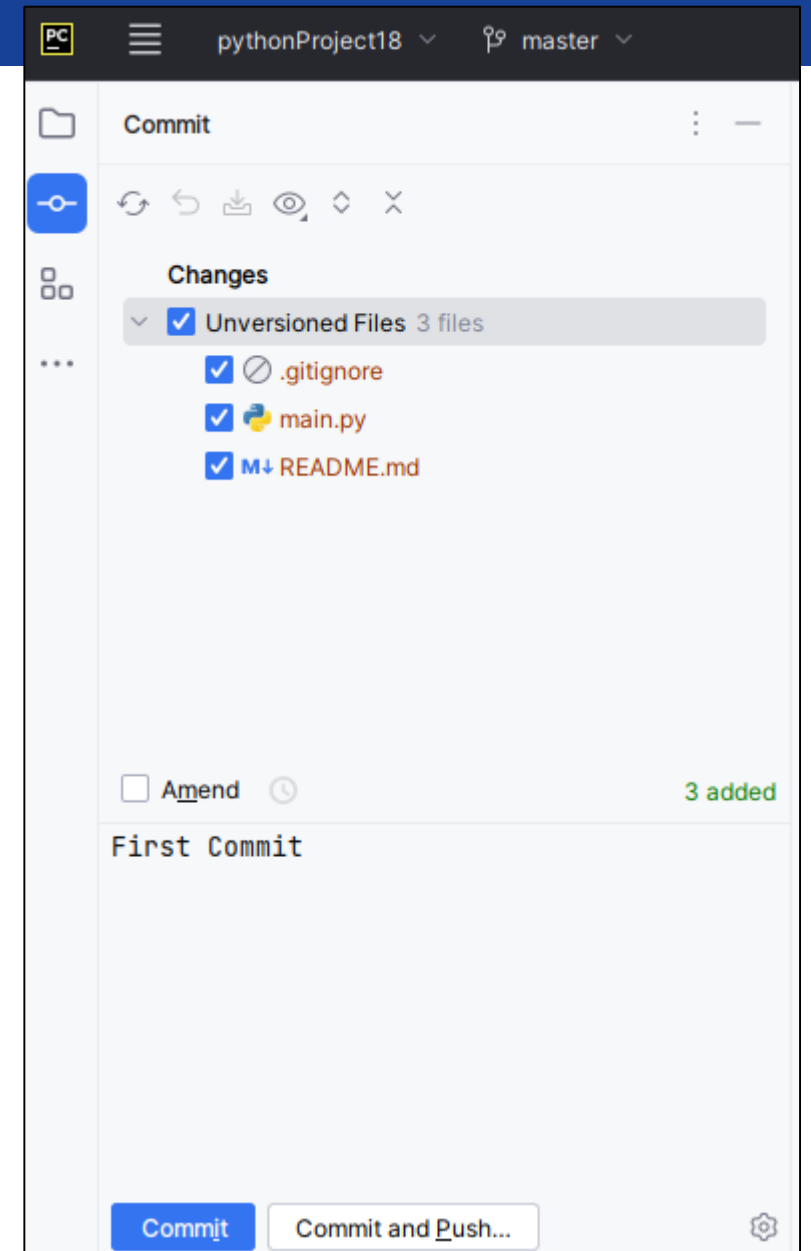
# PyCharm 에서 Git 저장소 생성

다음 작업을 해야, git init 이 수행된다.



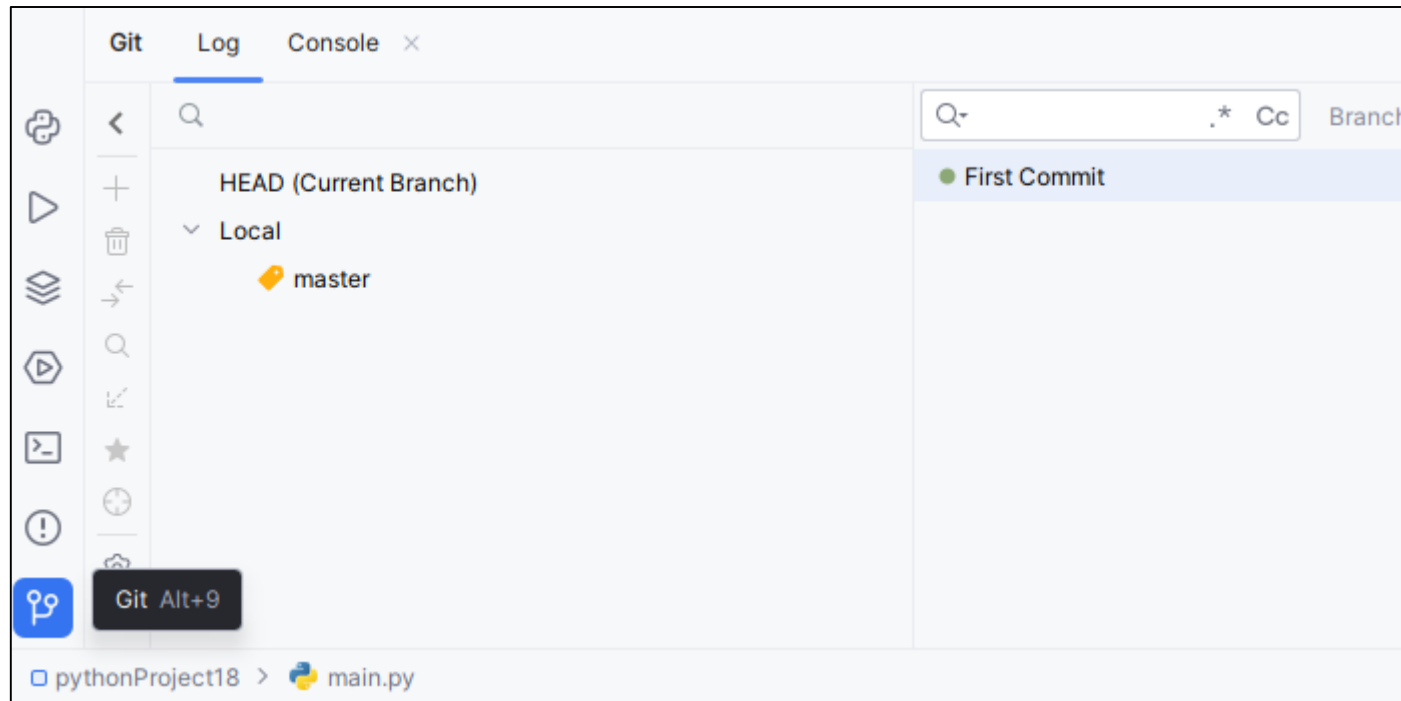
# 첫 Commit

- 왼쪽 세로탭에 표시된 커밋(Commit) 창을 클릭  
모두 체크 후, Commit Message 작성  
Commit을 1회 해야만, Push 가능하다.



# Commit Log 확인 창

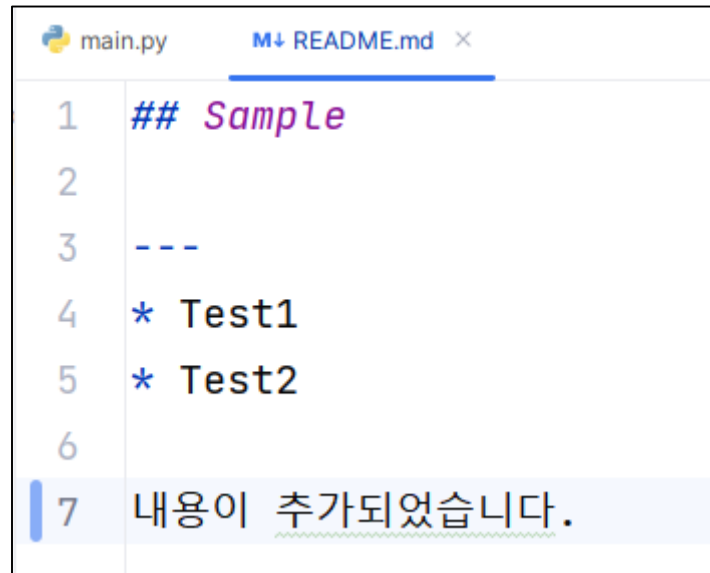
깃 로그 후 현재까지 Commit 이력 확인





# 내용 추가 후, 한번 더 Commit

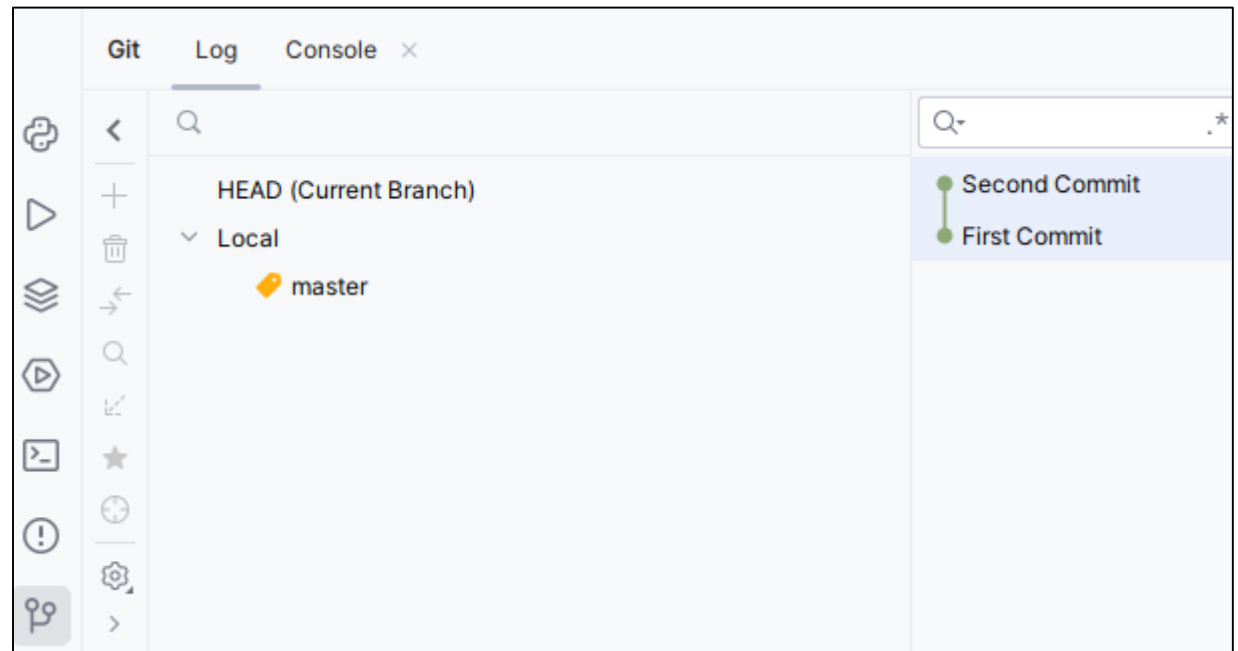
- checkout 실습을 위해, commit 1회 더 하기



The screenshot shows a code editor with two tabs: 'main.py' and 'M+ README.md'. The 'README.md' tab is active, displaying the following content:

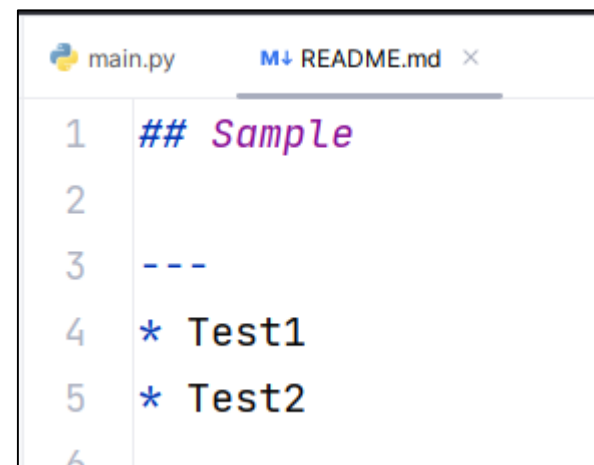
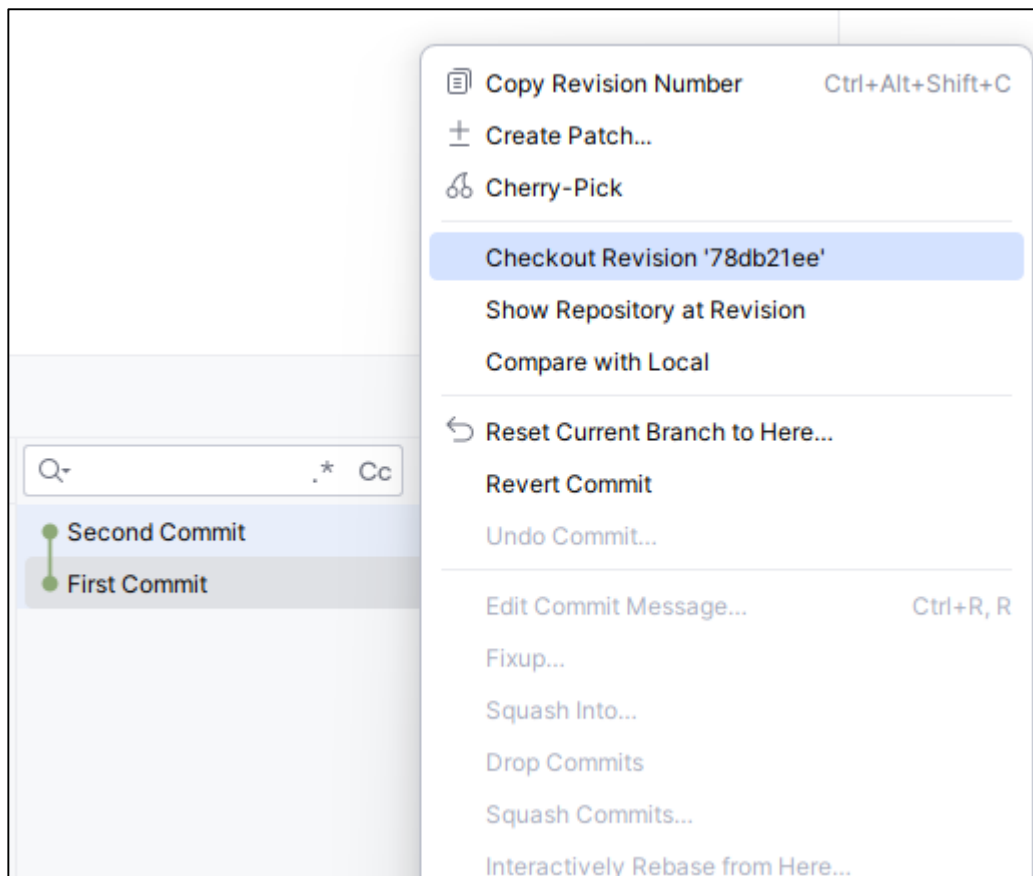
```
1  ## Sample
2
3  ---
4  * Test1
5  * Test2
6
7  내용이 추가되었습니다.
```

Line 7 is highlighted with a blue selection bar.



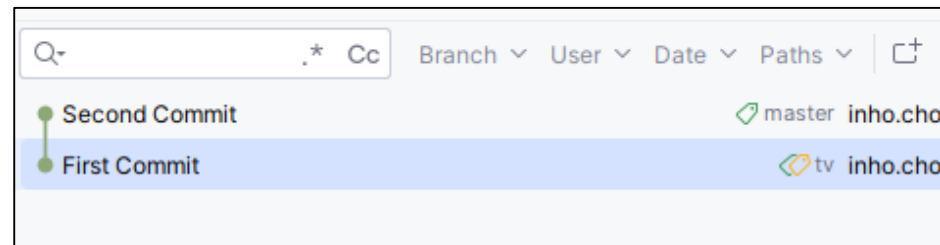
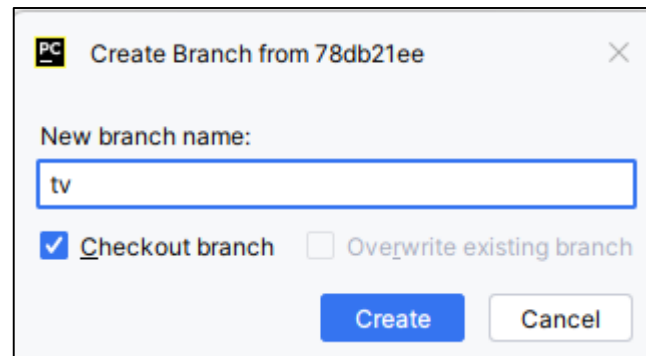
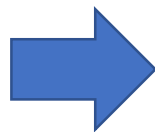
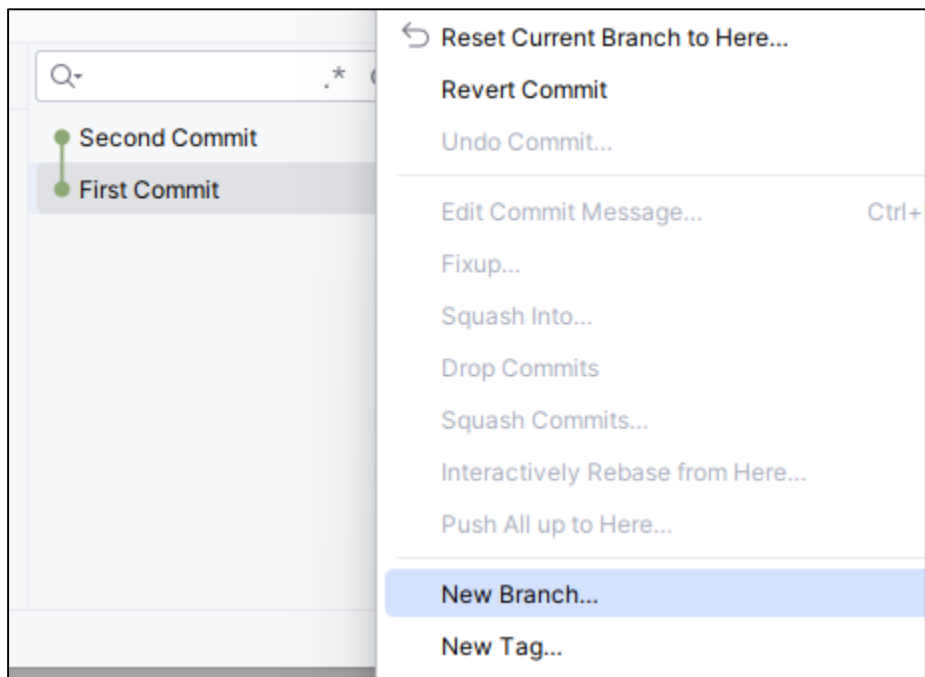
# 체크아웃

## 선택한 Commit으로 Checkout 하기



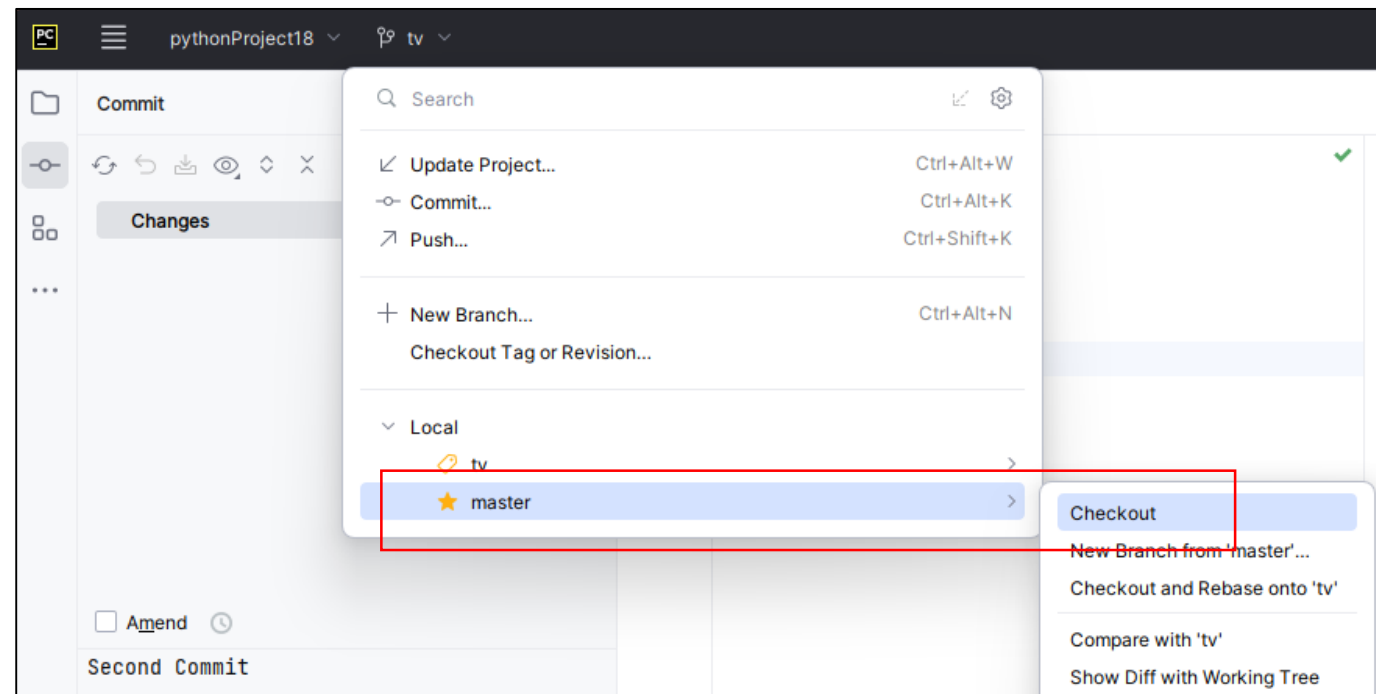
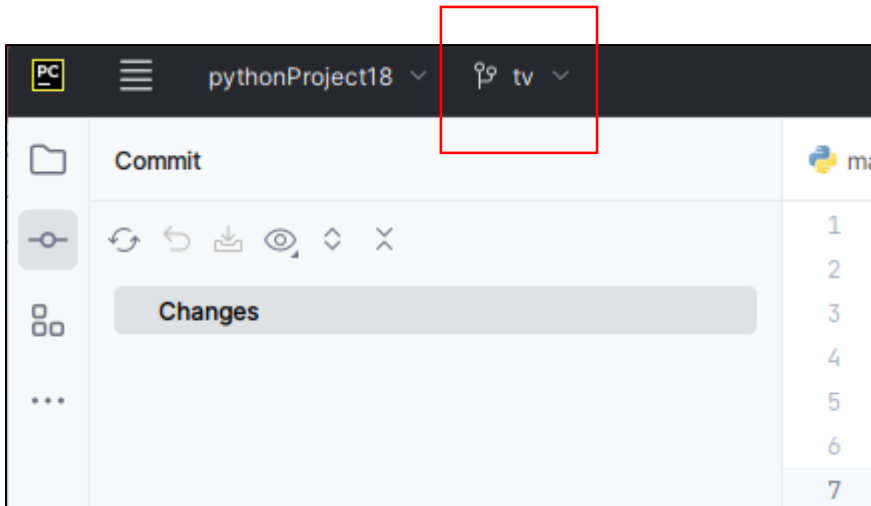
# 브랜치하기

- 이전 (선택한) Commit에서 새 Branch 생성하기



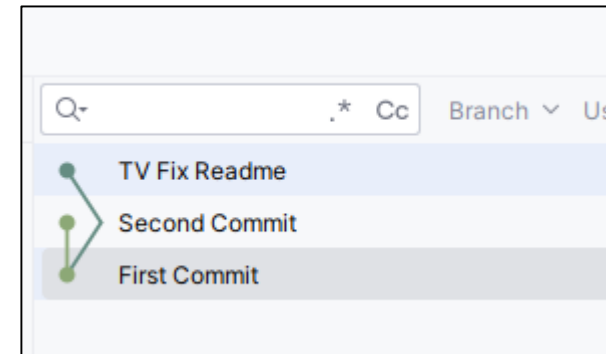
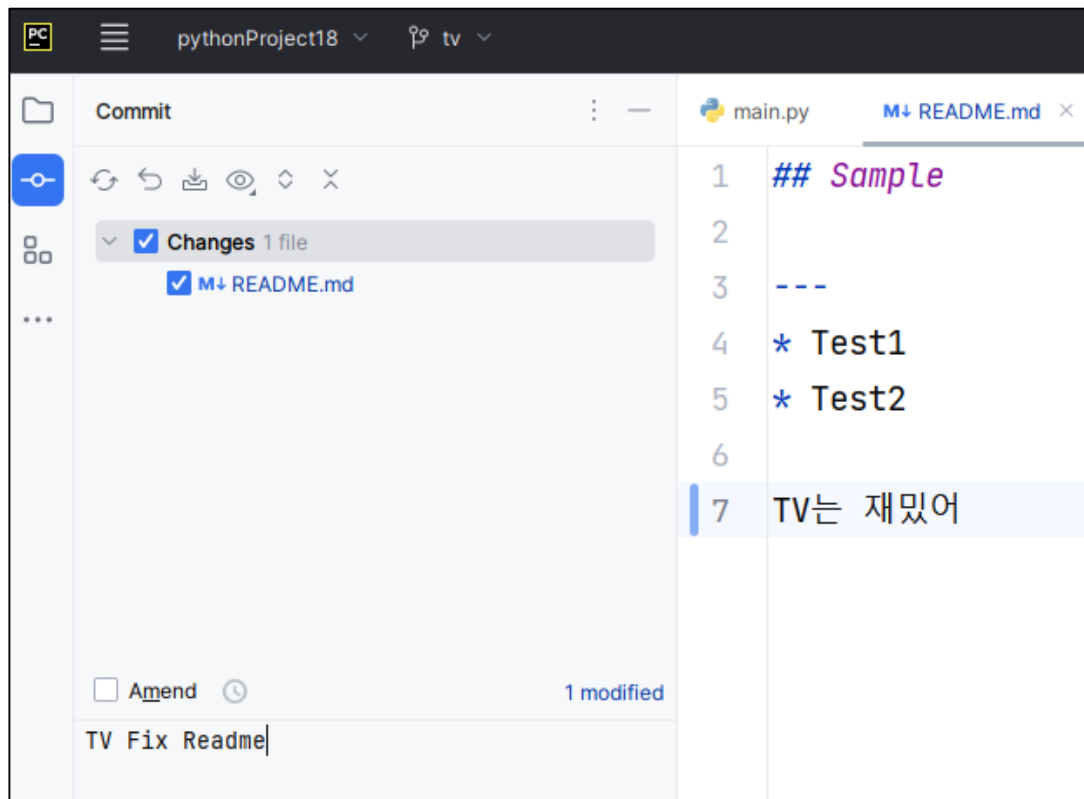
# 현재 Branch 확인 방법

- 상단에서 확인 가능  
상단에서도 checkout 가능



# Commit 한번 더 하기

- tv 브랜치에서, README 파일 수정 후, Commit 진행



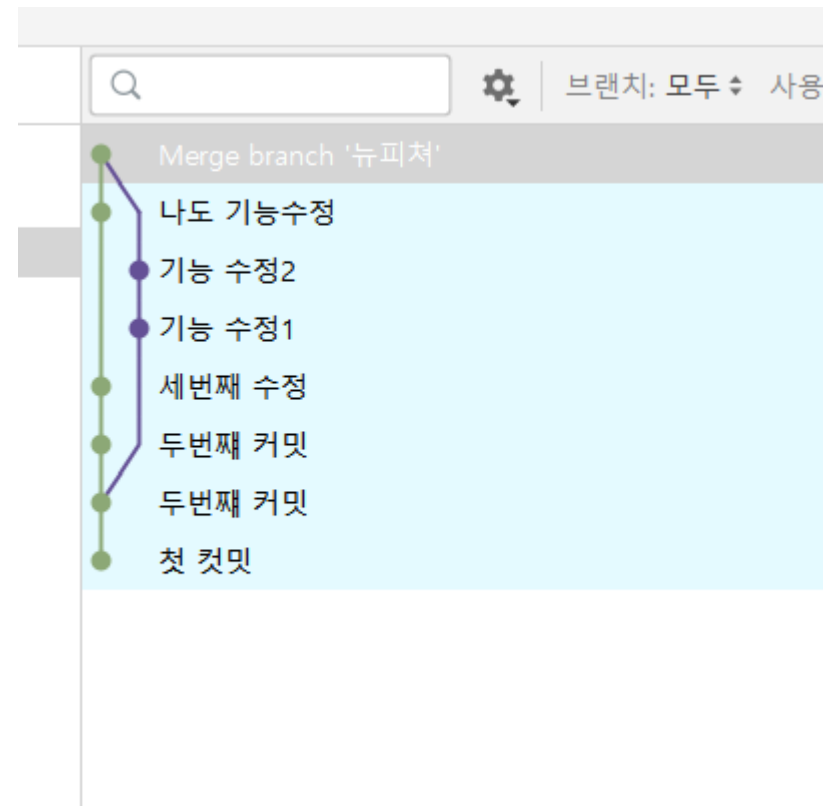
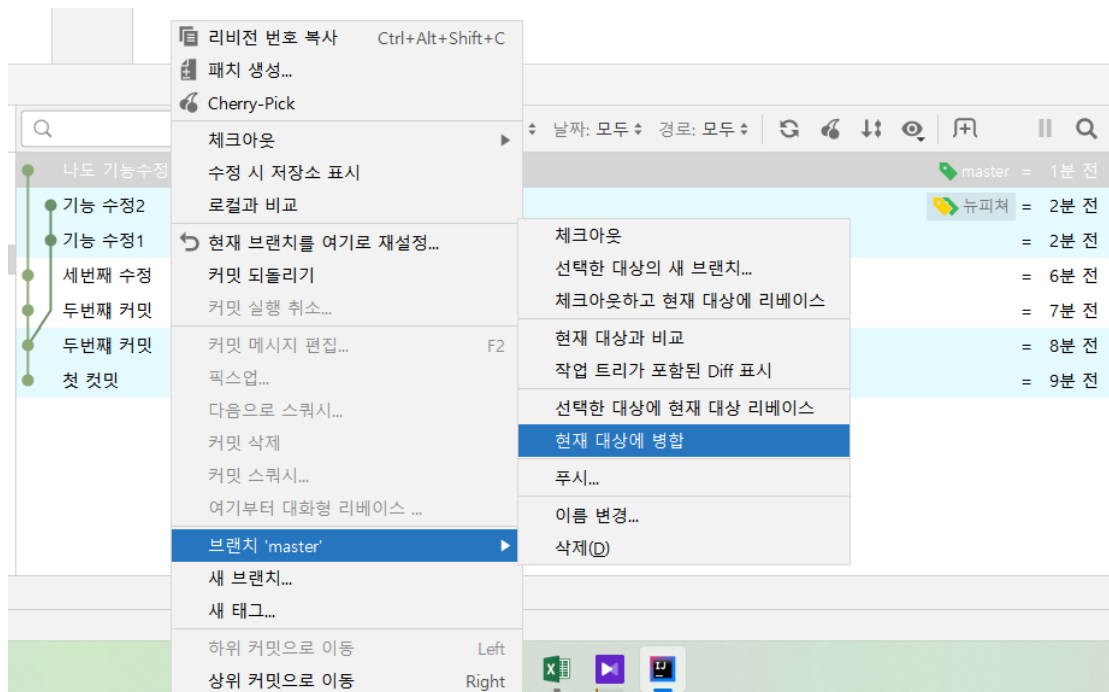
## 한번 더 Commit

- Master Branch (Second Commit) 으로 체크아웃 후, 한번 더 Commit

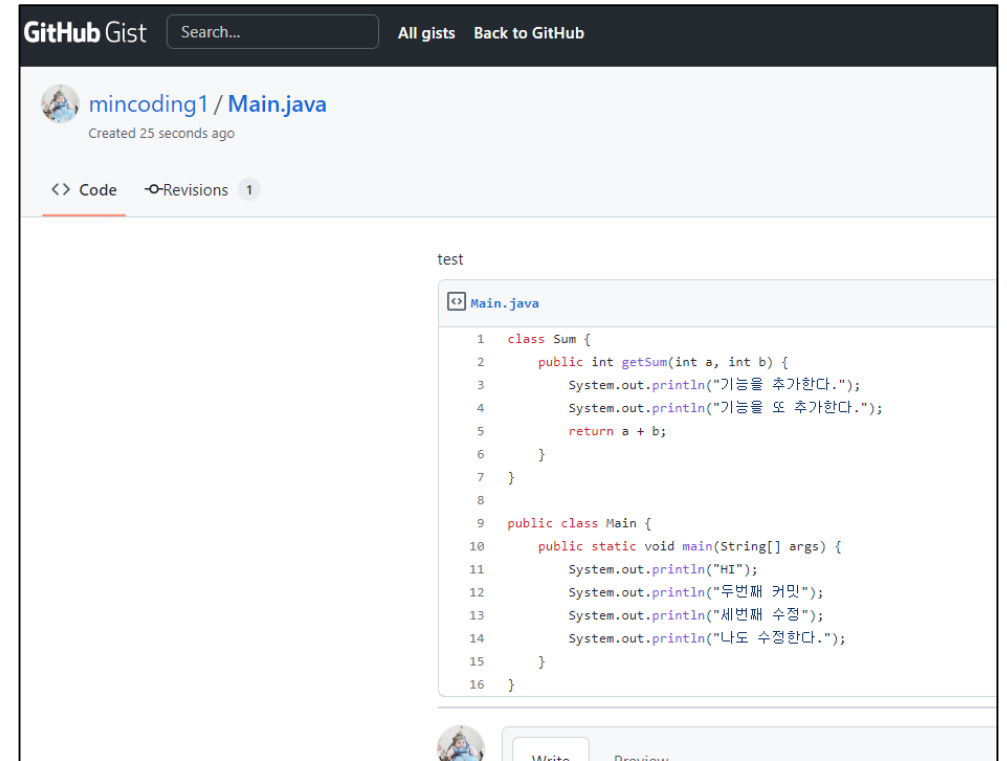
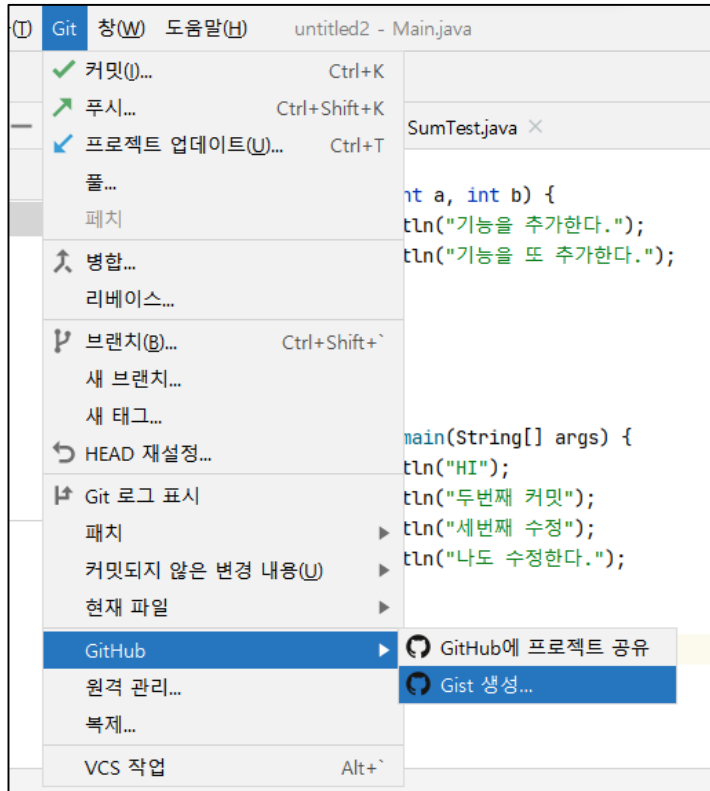
# Merge

## 병합하기

한 줄당 하나의 Commit 으로 표기 됨

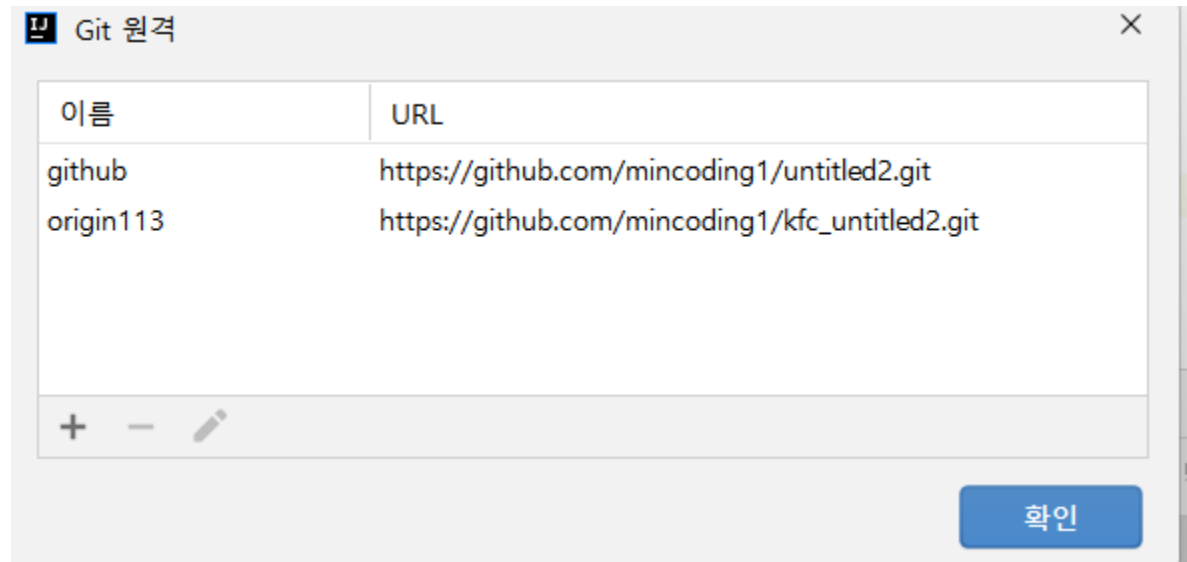


# gist 생성





# 원격관리





# PyCharm 에서 Git CLI 사용

# PyCharm 에서 Git CLI 을 쓰는 이유

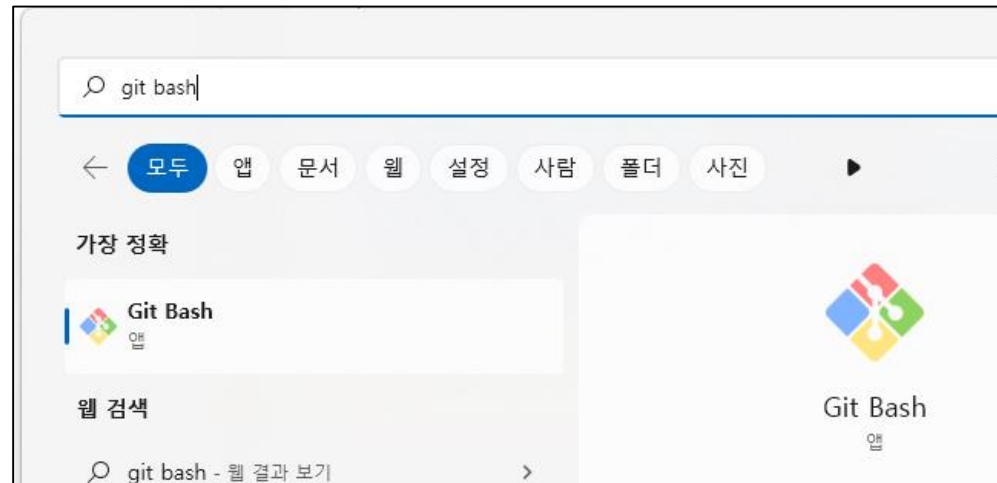
- **GUI 장점 : 가독성**

Git History 을 GUI로 가독성 있게 확인 가능.  
두 개의 파일 내용 비교 (Diff)가 가독성

- **GUI 단점 : 느린 동작속도**

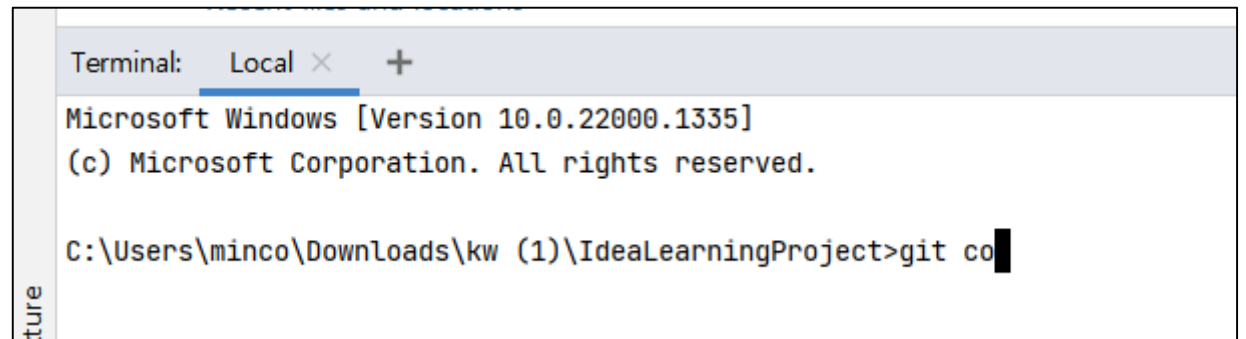
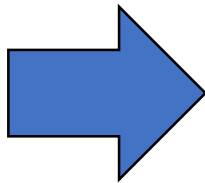
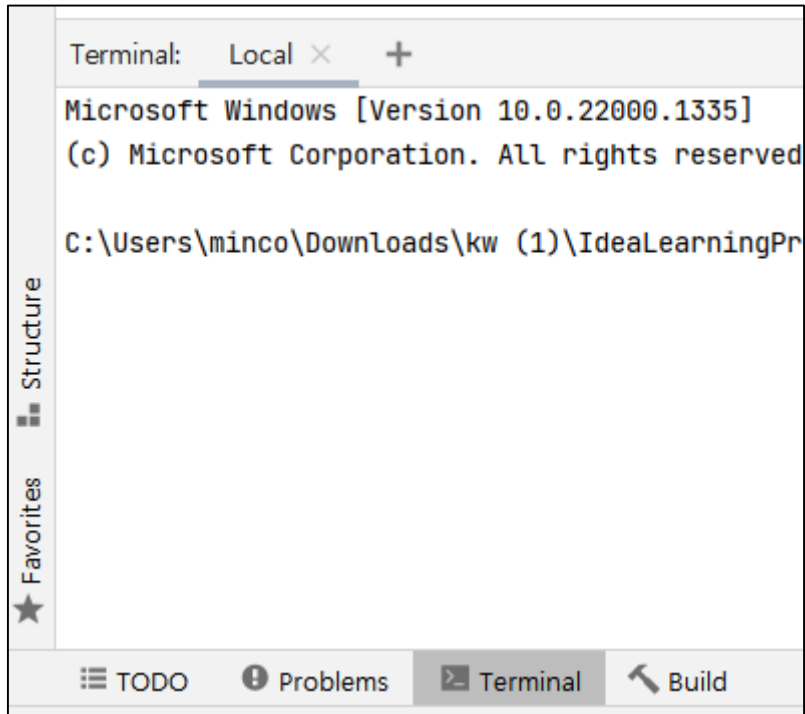
Commit & Push 를 할 때마다, 마우스 클릭 해줘야 한다.  
속도가 CLI보다 느리다.

로컬 또는 개발 PC에 git bash가 설치 필수



# PyCharm 하단 Terminal 사용하기

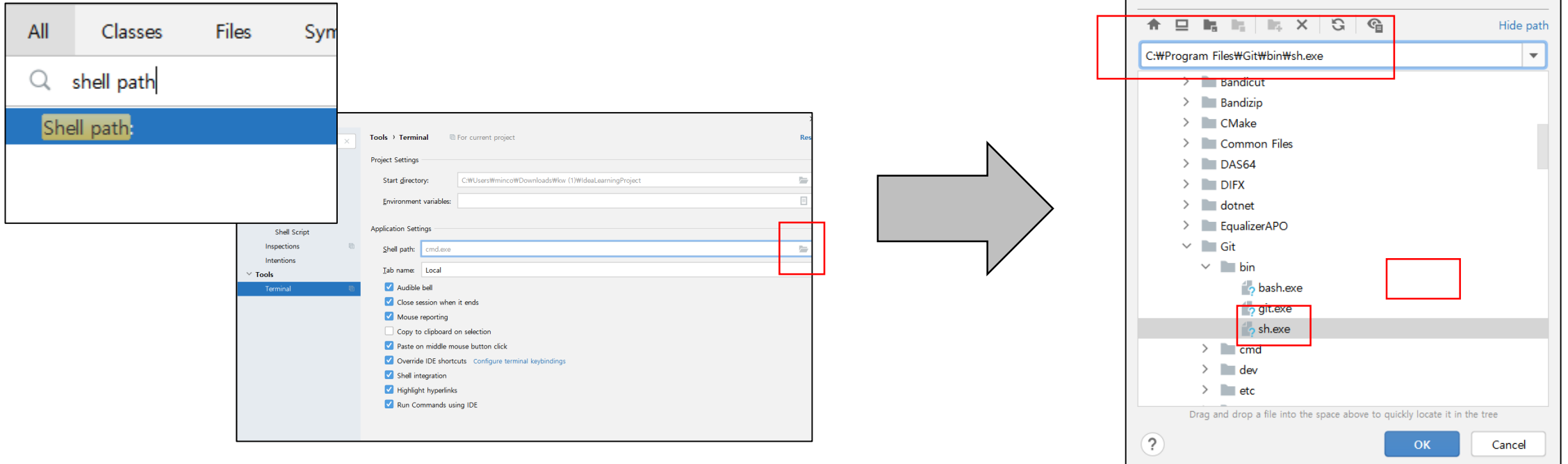
## CLI 사용 환경 제공



단점 : 자동완성이 Tab 키로 불가능하다.

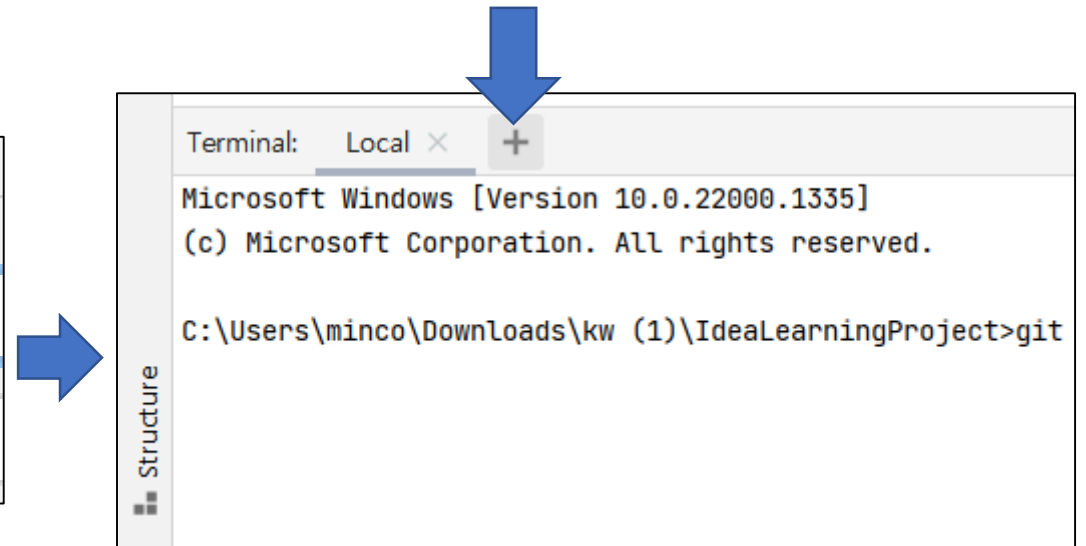
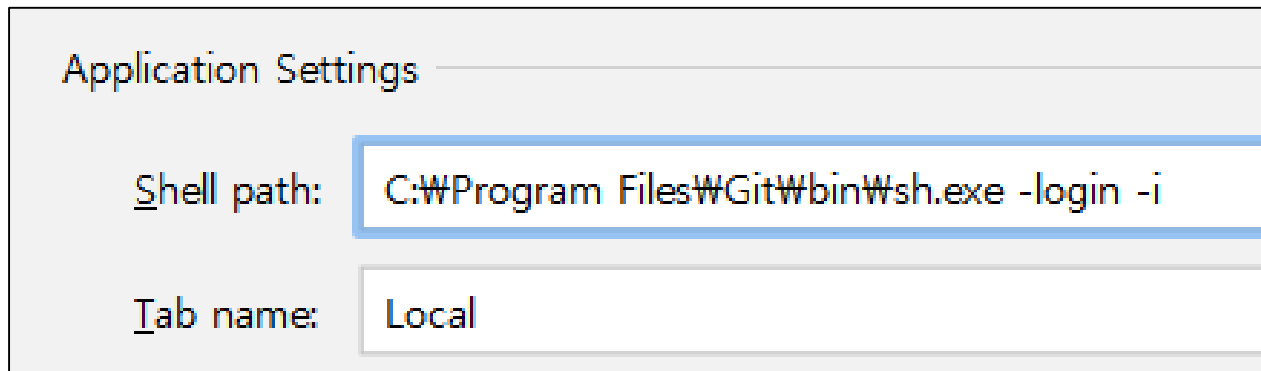
# 환경 세팅

- Shift x 2회 : shell path 입력
- cmd.exe 부분을 sh.exe 파일 경로로 변경한다.



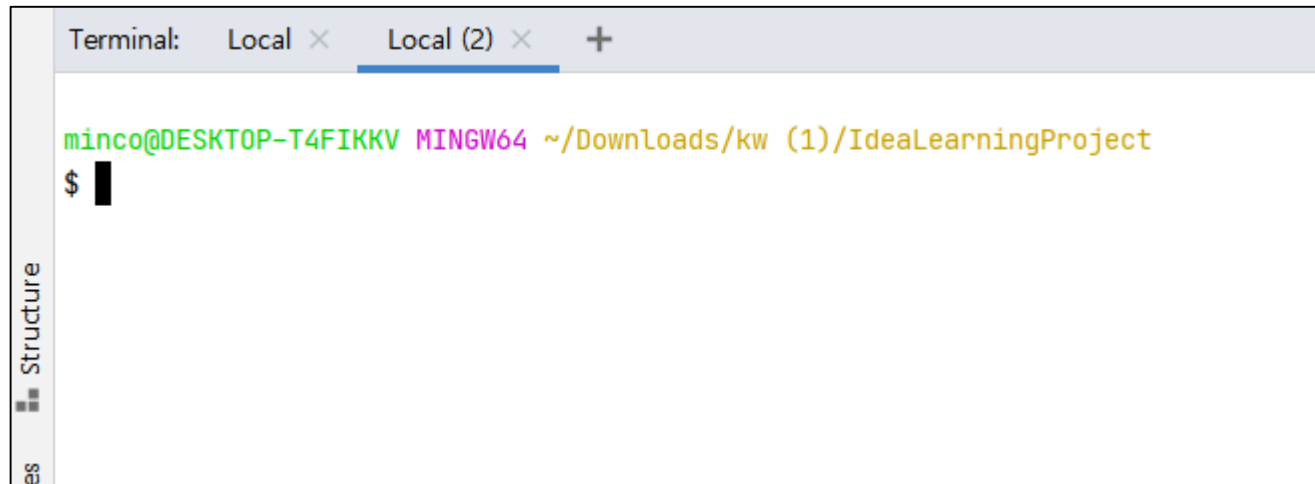
## 터미널 창 다시 열기

Shell path 옵션 추가 기입 후,  
확인버튼 누르기. 그리고 추가 Terminal 창 열기



## 익숙한 git bash 화면이 뜬다.

cmd.exe 와 달리,  
Tab 키를 이용한 키워드 자동완성 기능 사용 가능



The screenshot shows a terminal window with a tab labeled 'Local (2)'. The prompt is 'minco@DESKTOP-T4FIKKV MINGW64 ~/Downloads/kw (1)/IdeaLearningProject' followed by a '\$' and a cursor. On the left side of the terminal, there is a vertical sidebar with the word 'Structure' and a small icon.

```
Terminal: Local × Local (2) × +
minco@DESKTOP-T4FIKKV MINGW64 ~/Downloads/kw (1)/IdeaLearningProject
$
```





# 계산기 구현과 코드리뷰 실습

소스코드 Base, PR

## [도전] 소스코드 base로 PR하기 (with 충돌해결) - 1

**각 팀원들은 아래 기능중 하나를 선택 후 개발한다.**

메신저로 어떤것을 개발할지 미리 정해주세요.

팀장님, 팀원분들 모두 아래 기능 중 하나를 선택해주세요.

UnitTest - TestCase도 추가해야합니다.

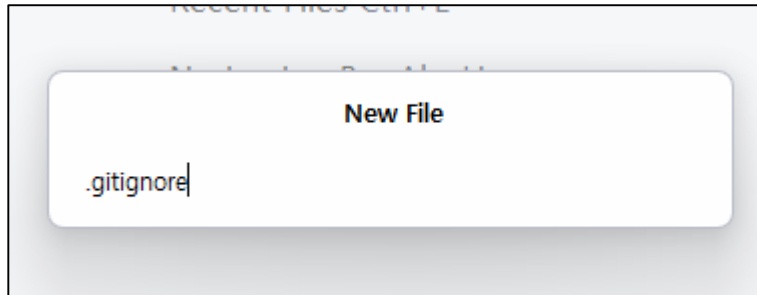
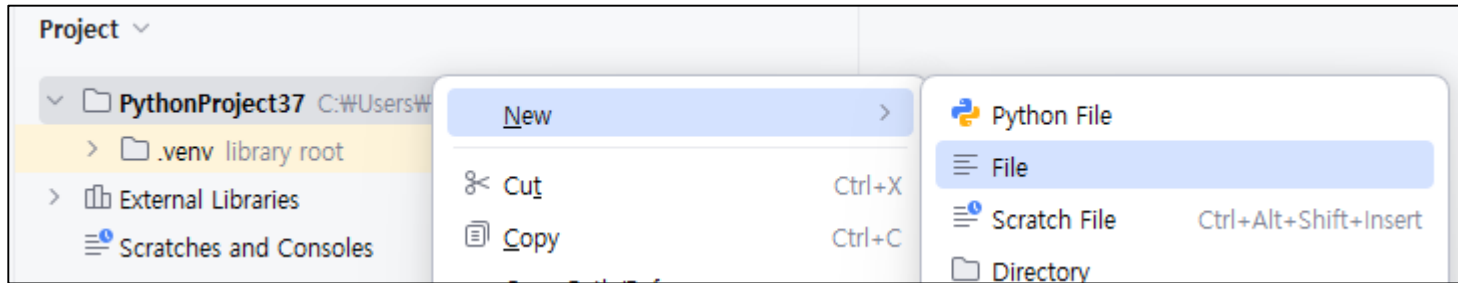
### 제작할 기능

1. getSum(a, b) :  $a + b$  반환
2. getGop(a, b) :  $a * b$  반환
3. getZegop(a) :  $a * a$  반환
4. getMinus(a, b) :  $a - b$  반환
5. getDivide(a, b) :  $a / b$  반환
6. getSumSum(a, b, c) :  $a + b + c$  반환

**팀장님은 README.md 파일 없이, 비어있는 Repository를 생성한다.**

## 스켈레톤 코드 제작

1. 프로젝트 > .gitignore 파일 추가



### .gitignore 파일 작성하기

pycharm 용 gitignore 템플릿 내용을 모두 복사 붙여넣기

```
.gitignore x
1 # Byte-compiled / optimized / DLL files
2 __pycache__/
3 *.py[cod]
4 *$py.class
5
6 # C extensions
7 *.so
8
9 # Virtual environment
10 venv/
11 env/
12 .venv/
13 .env/
14
15 # PyCharm project files
16 .idea/
17
18 # Distribution / packaging
19 .Python
20 build/
```

<https://gist.github.com/jeonghwan-seo/73ca1003296ee0a170ef722ab2a19c66>

### main 코드 작성

충돌이 자주 발생 되도록 하는 목적  
팀장님 / 팀원분들은  
모두 주석이 달린 부분에 구현을 해야 한다.

```
import pytest

class Calc:
    # 이곳에 코드를 작성
    pass

# 테스트 케이스 작성
def test_sample():
    assert 1 == 1
    pytest.fail()
```

### 터미널 열고 git push하기

git 명령어를 이용하여 지금까지 코드를 repository에 push한다.

## [도전] 소스코드 base로 PR하기 (with 충돌해결) - 2

### 하나의 소스파일과 하나의 테스트파일로만 개발을 한다.

- 충돌이 자주 발생할 예정이며, 충돌이 발생할 경우, 충돌을 해결한다. (수동 or 자동)

### 팀장님 역할

- 스켈레톤 코드 작성 (Unit Test 파일 포함)
- push 진행 후, github repo 공유
- 기능 하나 맡고, 개발 Branch 생성 후 개발 시작

### 팀원 역할

- clone
- 기능 하나 맡고, 개발 Branch 생성 후 개발 시작