# 형상관리

2019 – ㈜텔코코리아아이에스

2022 – ㈜리모샷

2023 – 넥스트IT

**강사: 서지민**

**연락처**: 010-3376-1886
**메일**: jmseokm29a1@naver.com

https://github.com/RadeonHD6750

## 일상생활

원격 저장소

시간과 공간에 제약이 없다.

넥스트 IT 교육센터

여기서 작업하던 것을

학원에서도 그대로 이어서 가능

네이버, 구글 클라우드와 무슨 차이??

집에서도 그대로 이어서 가능

집에서 주말에 했던 것을

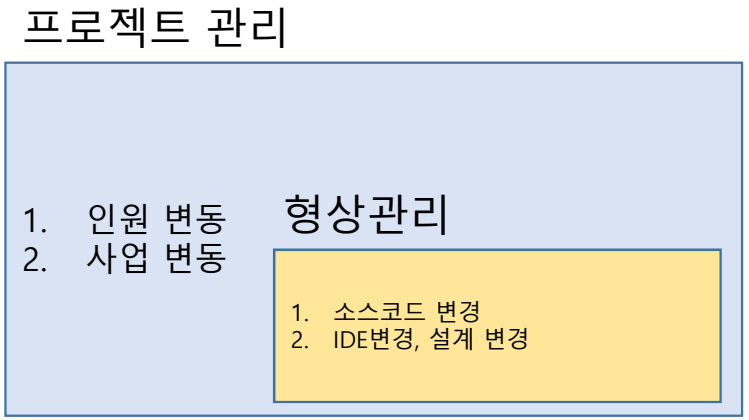
# 프로젝트 관리 개념

## ◈ 정의

프로젝트의 진행상황 및 변경사항 등을 체계적으로 추적하고 통제하는 것

## ◈ 프로젝트의 변경사항 종류

- 소스코드 변경사항 -> 깃으로 관리
- IDE, 빌드구조, 설계 변동 등등
- 참여인원에 대한 변동 (근태, 인원수 변동, 인력 보충 등)
- 사업에 대한 변동 (요구사항 변동, 일정 변동 등)

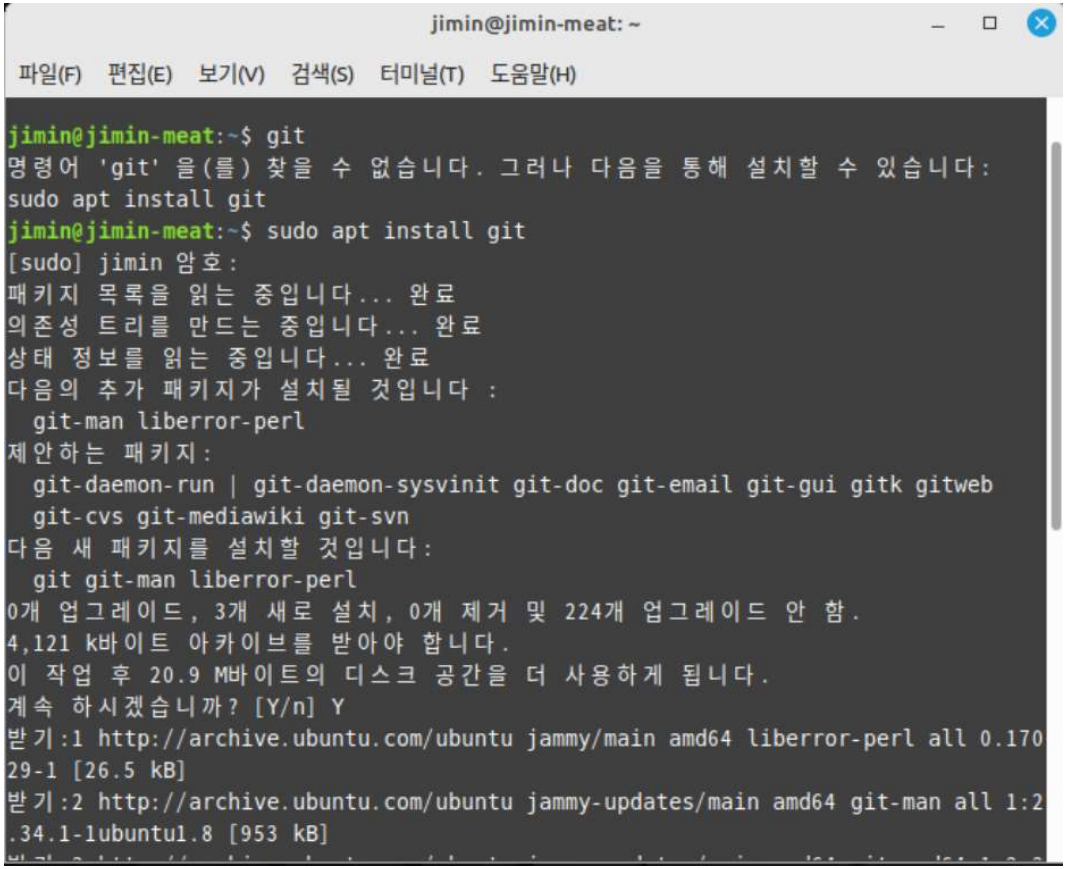



## ◈ 용어 정리

- 변경관리 : 소스코드 변경사항에 대한 관리
- 버전관리 : 변경사항을 '버전'이란 개념을 통하여 관리
- 형상관리 : 위의 개념들을 포함하여 프로젝트와 관련된 모든 변경사항을 관리

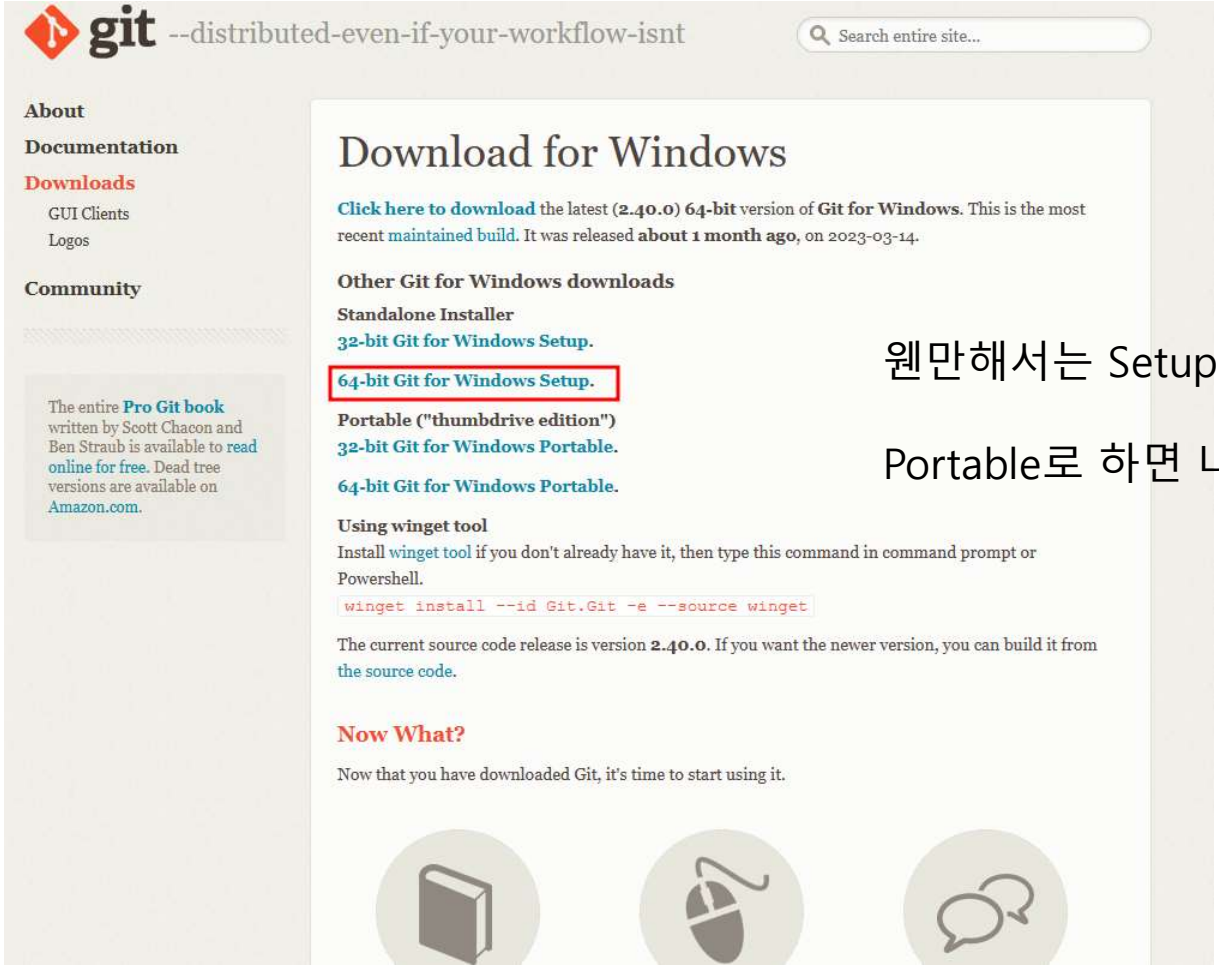
# Git Client 설치하기

# Git 설치 - 리눅스




1. Git이 있는지 터미널에 입력하여 존재여부 확인
2. 없다면 sudo apt install git으로 설치

# Git 설치 - 윈도우



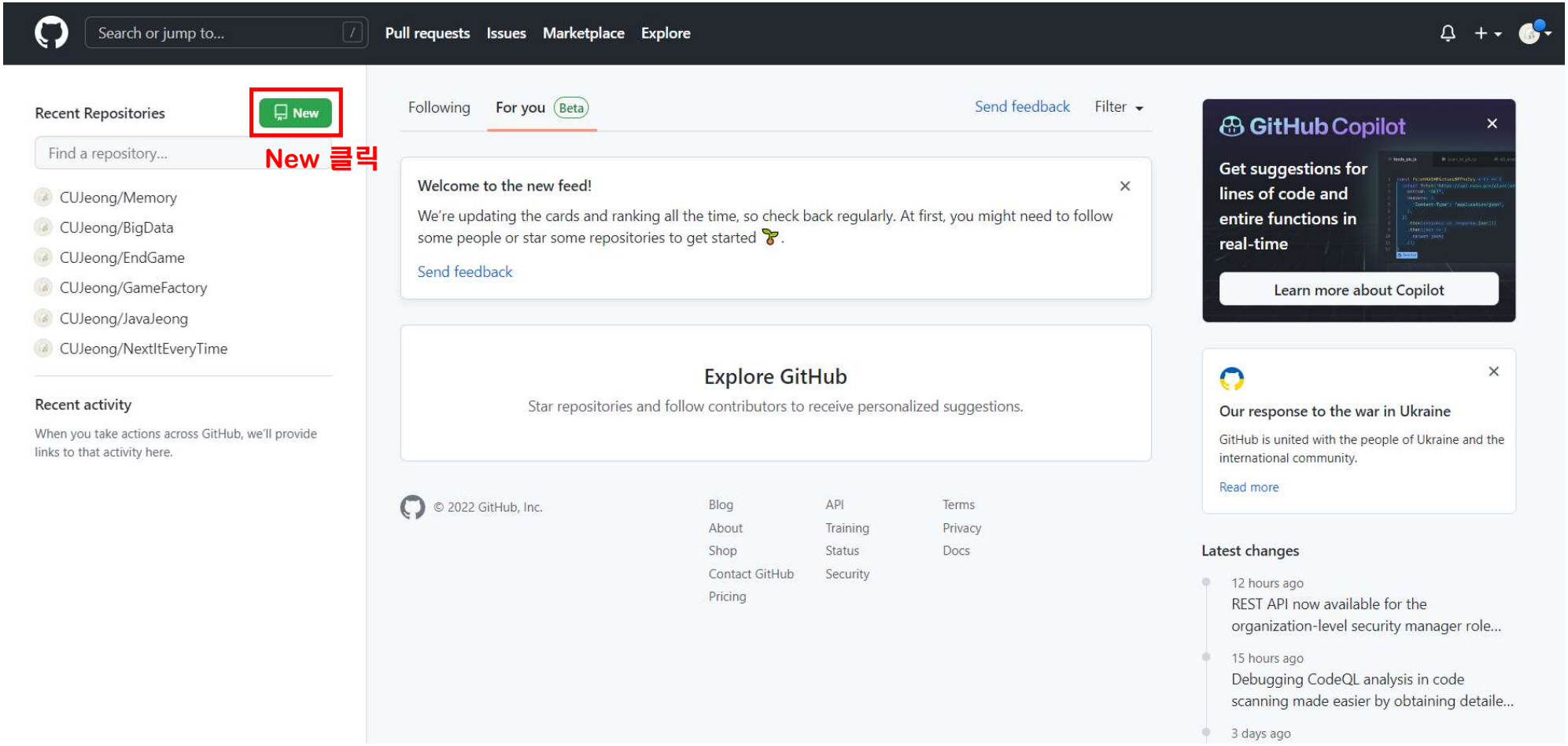

웬만해서는 Setup버전을 하자

Portable로 하면 나중에 못 찾음

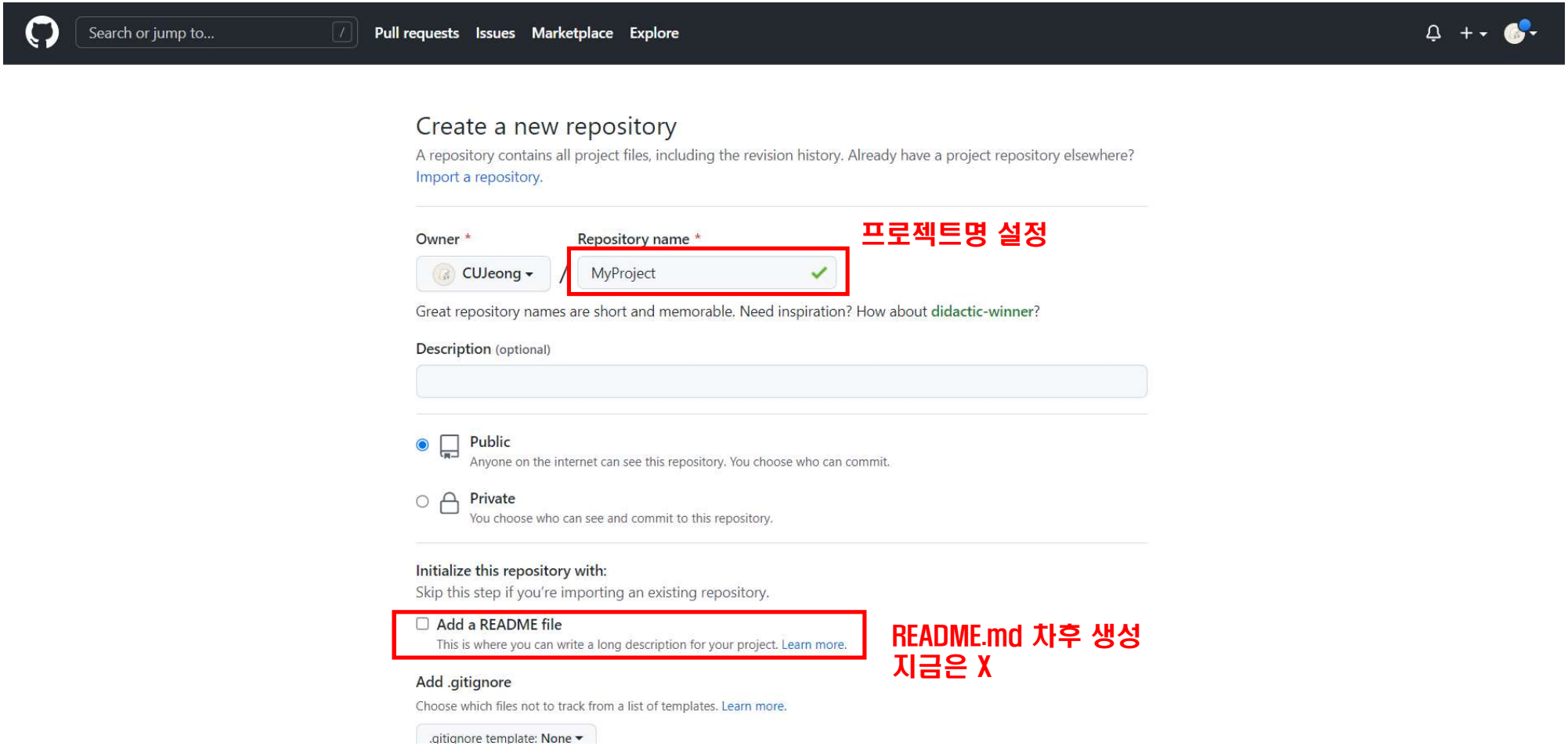
https://git-scm.com/download/win

# GitHub 저장소 생성

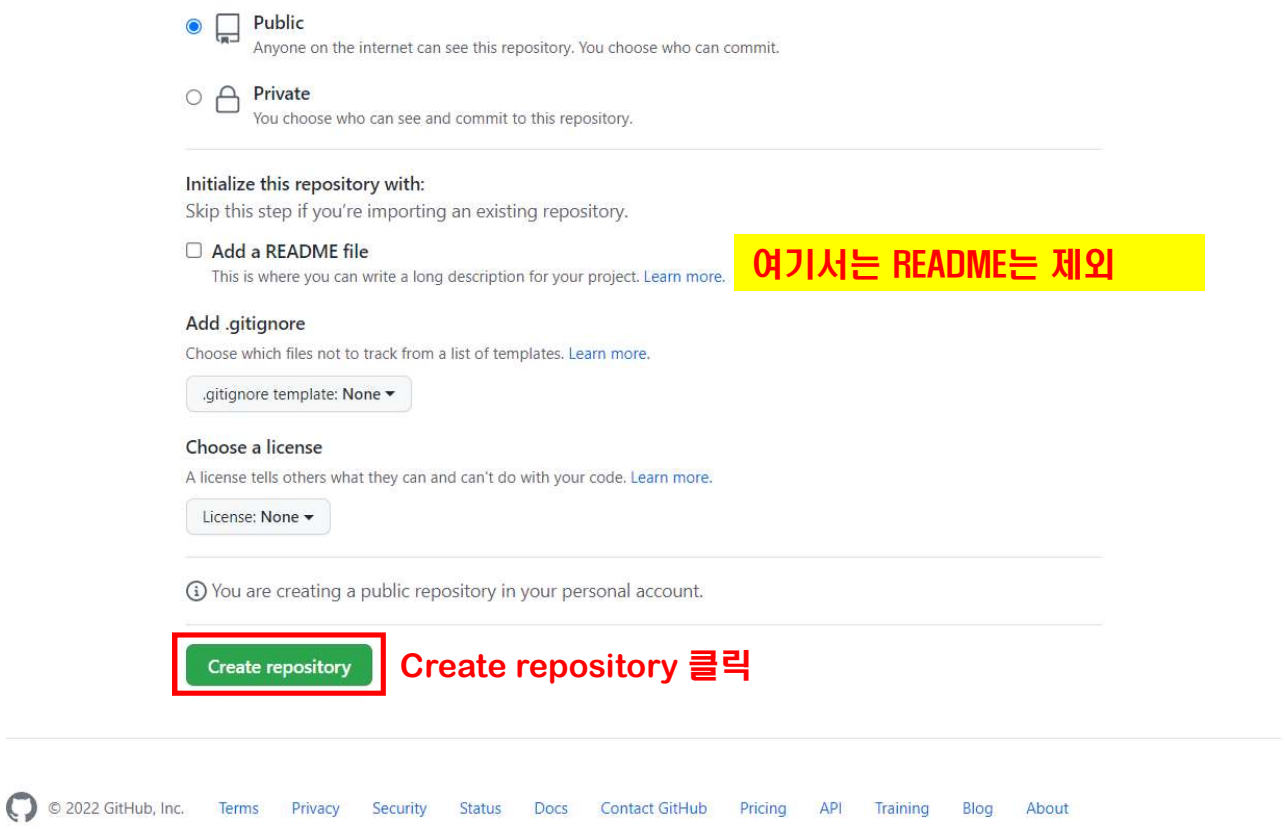
# GitHub 저장소 생성



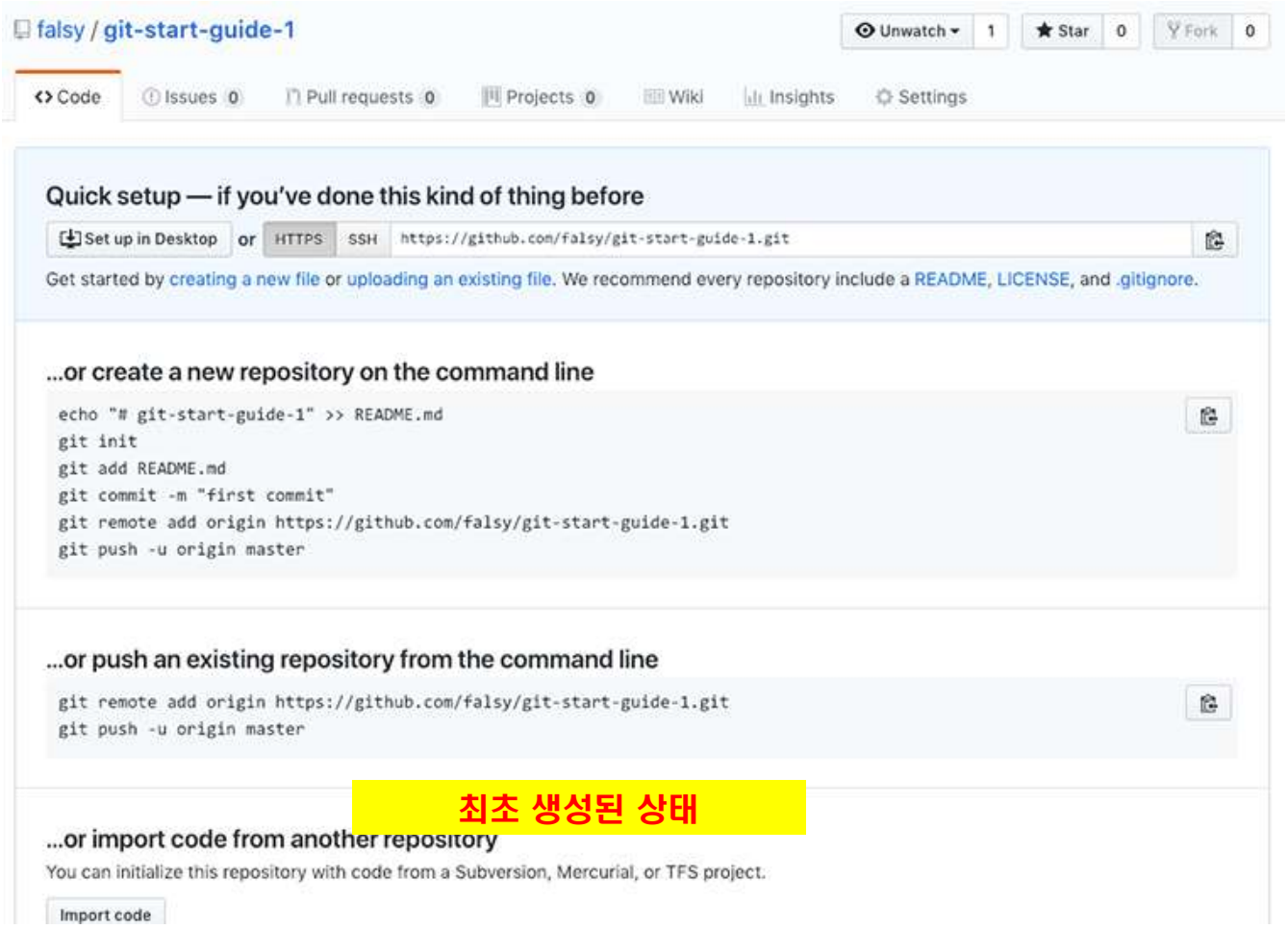
# GitHub 저장소 생성



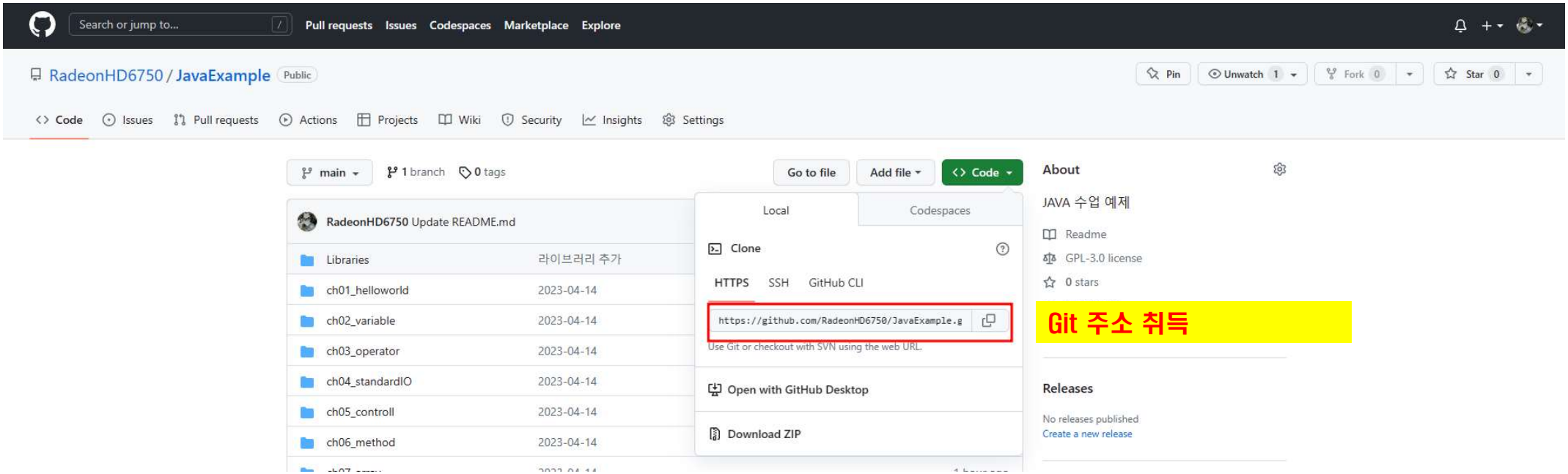
# GitHub 저장소 생성



# GitHub 저장소 생성



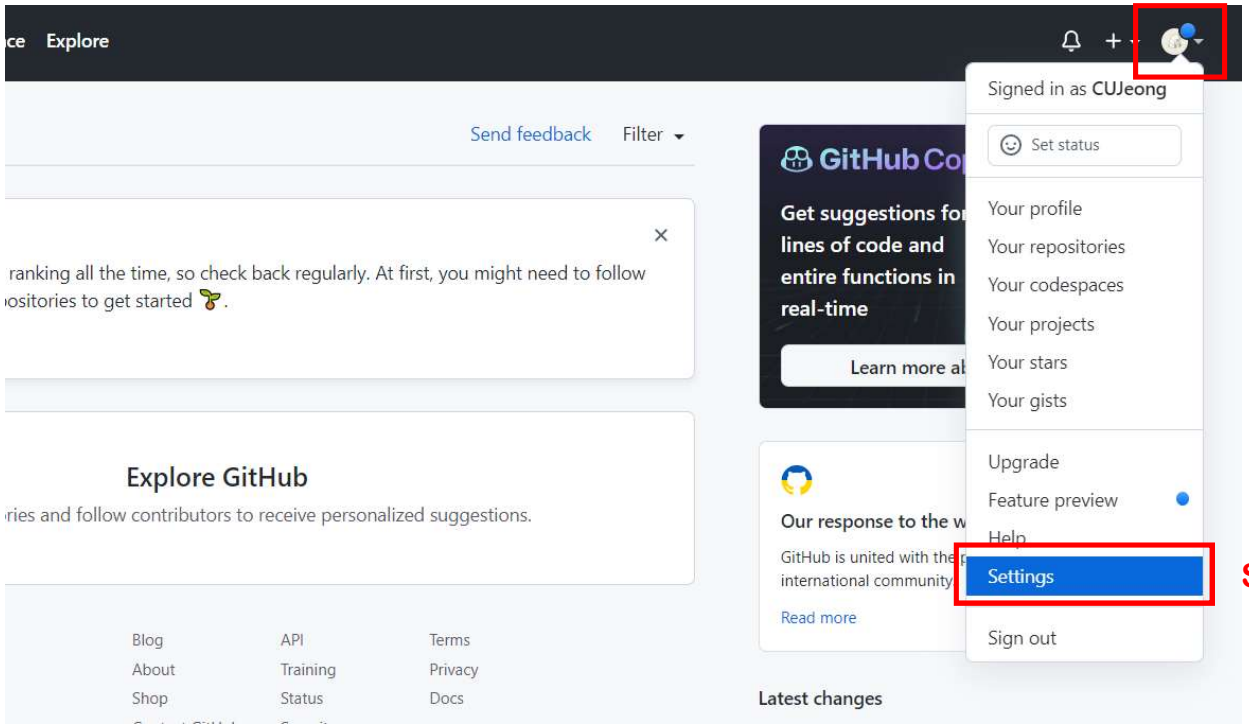
# GitHub 저장소 생성



# GitHub 토큰 생성

# GitHub 토큰 생성

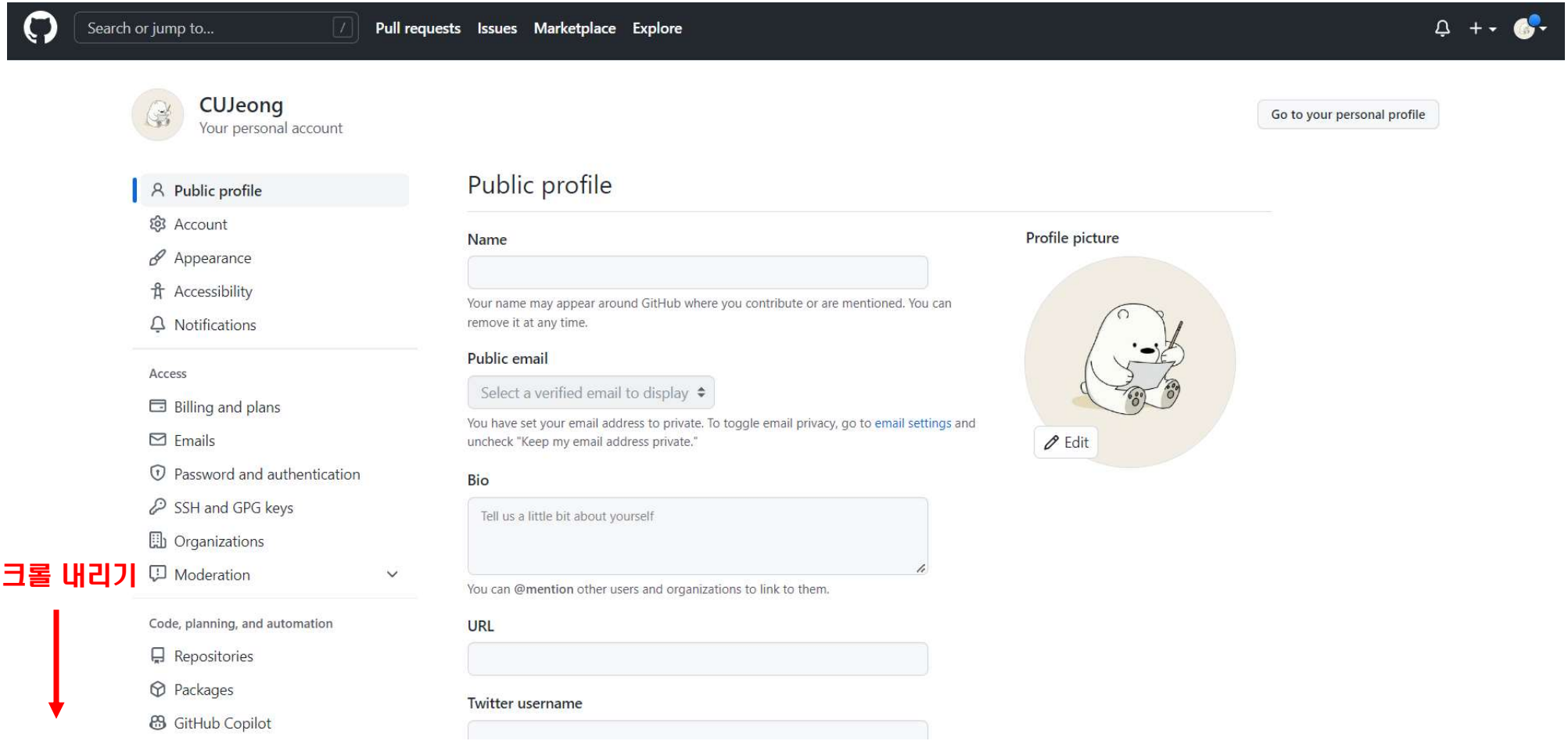
◈ GitHub 토큰은 VS Code나 Eclipse에서 GitHub에 로그인 시 사용되는 비밀번호라고 보면 된다.




프로필 이미지 클릭

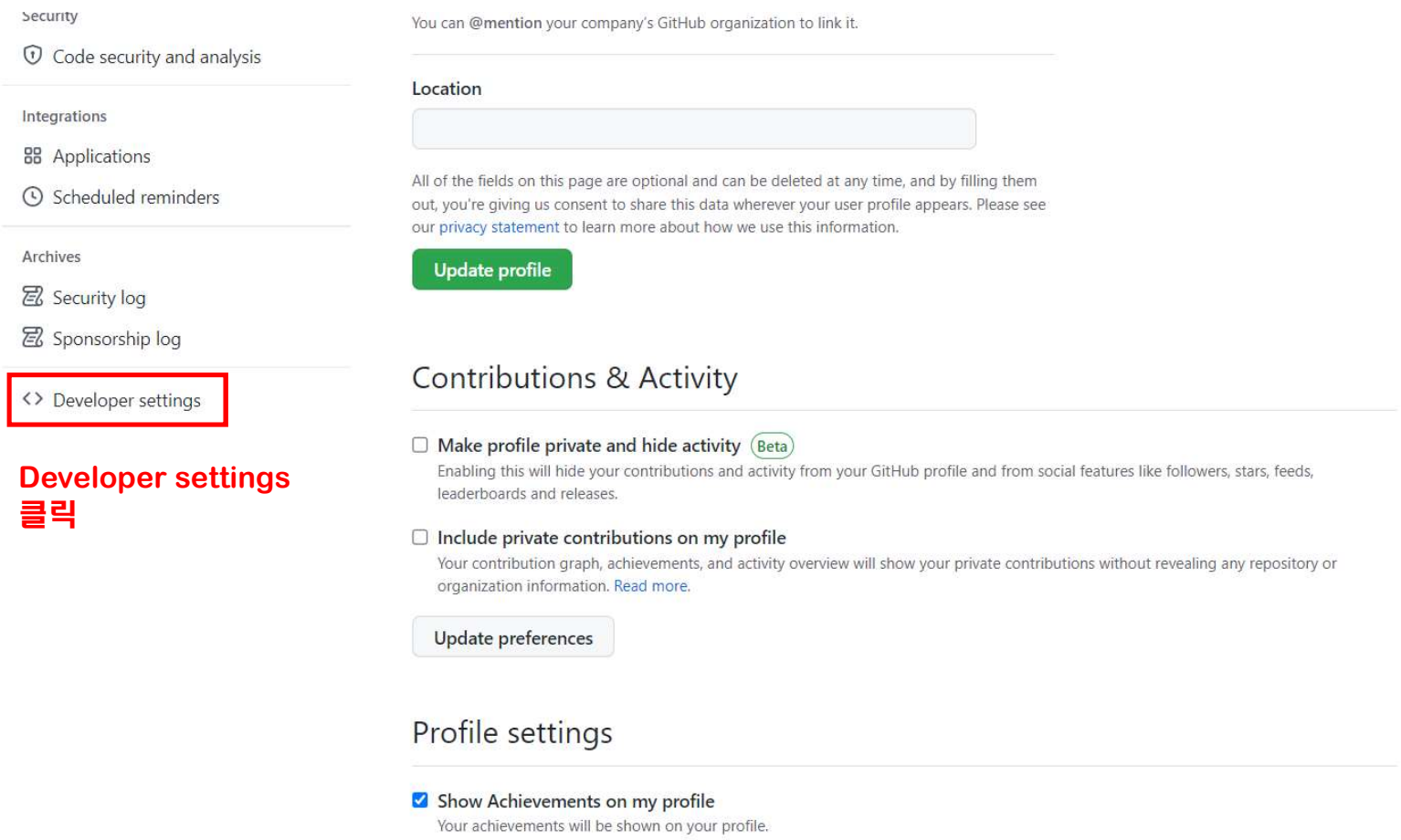
Settings 클릭

# GitHub 토큰 생성



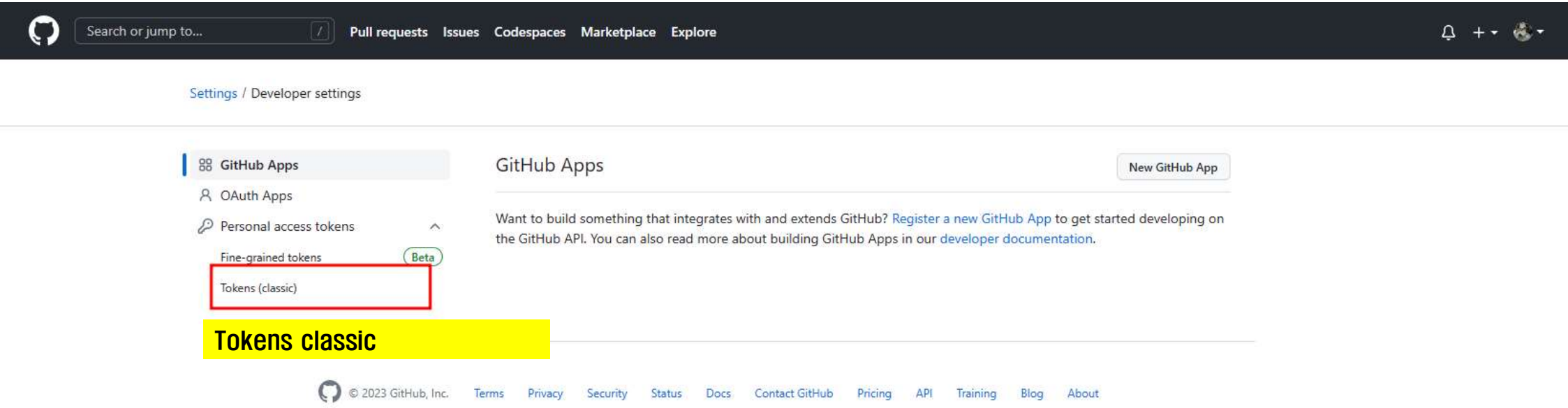

스크롤 내리기

# GitHub 토큰 생성

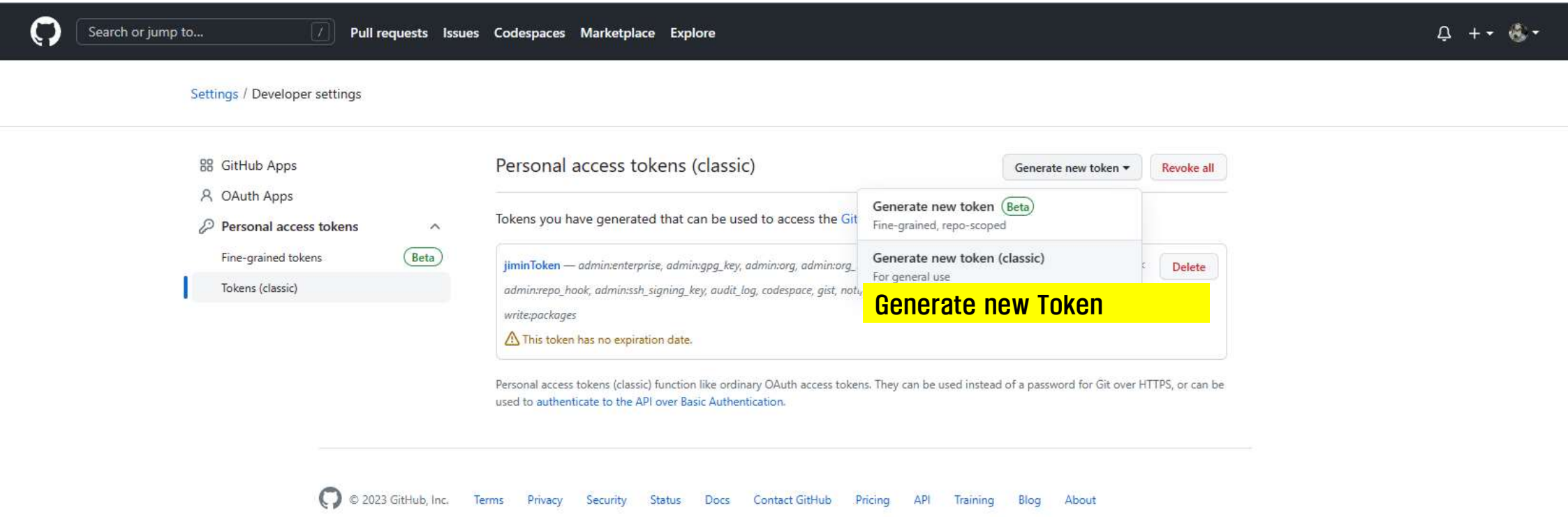



**Developer settings 클릭**

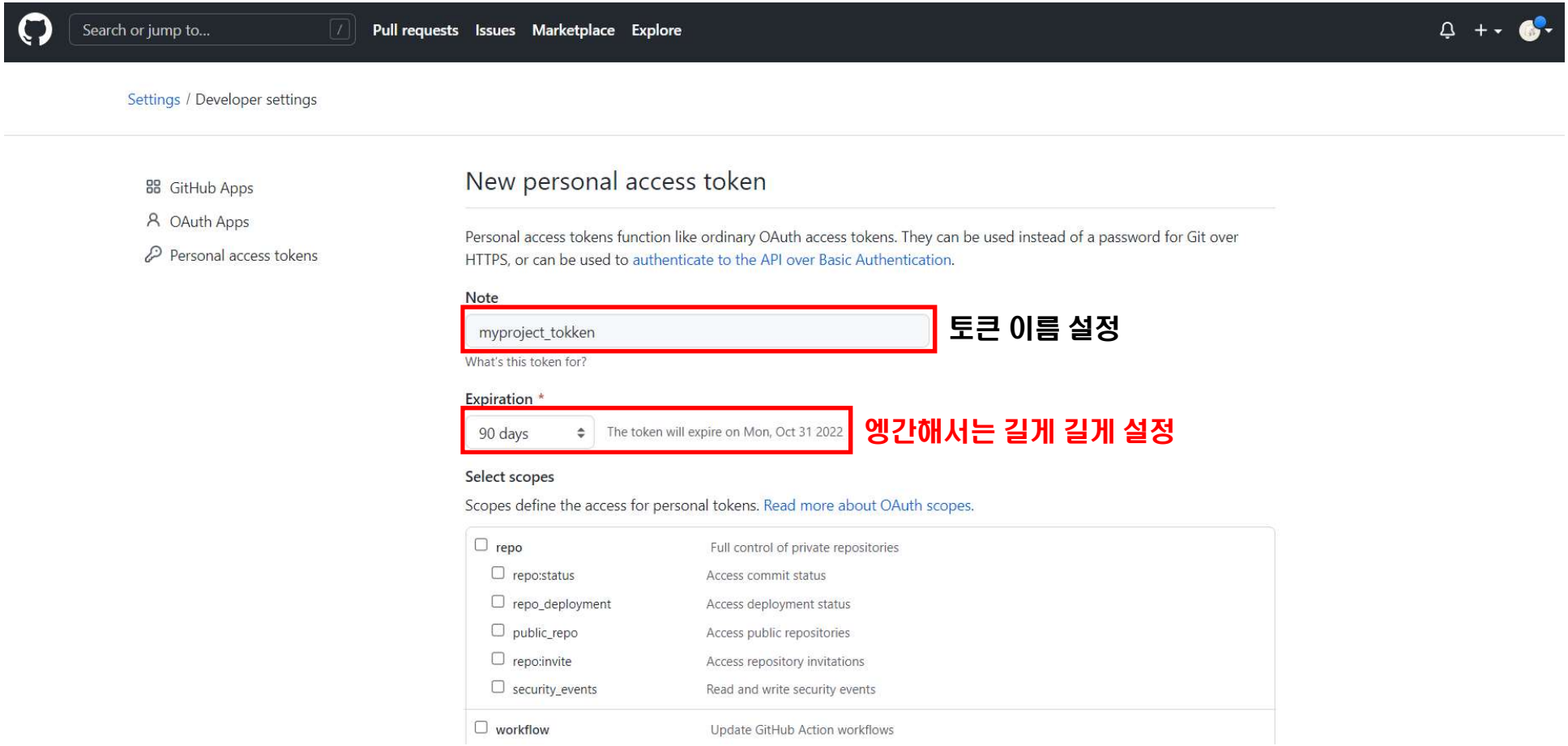
# GitHub 토큰 생성


Search or jump to...
Pull requests
Issues
Codespaces
Marketplace
Explore
Settings / Developer settings
GitHub Apps
OAuth Apps
Personal access tokens
Fine-grained tokens
Beta
Tokens (classic)
GitHub Apps
New GitHub App
Want to build something that integrates with and extends GitHub? Register a new GitHub App to get started developing on the GitHub API. You can also read more about building GitHub Apps in our developer documentation.
Tokens classic
© 2023 GitHub, Inc.
Terms
Privacy
Security
Status
Docs
Contact GitHub
Pricing
API
Training
Blog
About

# GitHub 토큰 생성

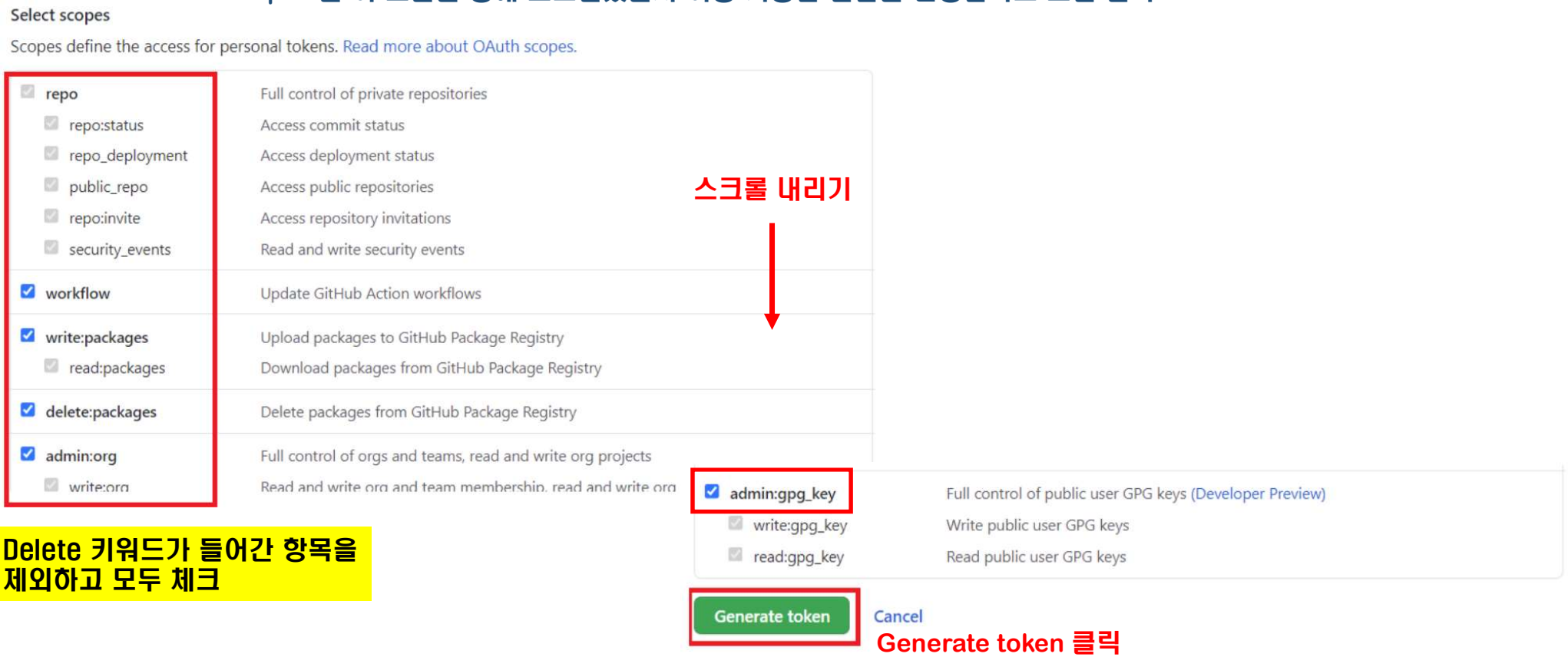

Search or jump to...
Pull requests
Issues
Codespaces
Marketplace
Explore
Settings / Developer settings
GitHub Apps
OAuth Apps
Personal access tokens
Fine-grained tokens
Beta
Tokens (classic)
Personal access tokens (classic)
Generate new token
Revoke all
Tokens you have generated that can be used to access the Git
Generate new token Beta
Fine-grained, repo-scoped
Generate new token (classic)
For general use
Delete
jiminToken — admin:enterprise, admin:gpg_key, admin:org, admin:org_
admin:repo_hook, admin:ssh_signing_key, audit_log, codespace, gist,
write:packages
This token has no expiration date.
Generate new Token
Personal access tokens (classic) function like ordinary OAuth access tokens. They can be used instead of a password for Git over HTTPS, or can be used to authenticate to the API over Basic Authentication.
© 2023 GitHub, Inc.
Terms
Privacy
Security
Status
Docs
Contact GitHub
Pricing
API
Training
Blog
About

# GitHub 토큰 생성


Search or jump to...
Pull requests
Issues
Marketplace
Explore
Settings / Developer settings
GitHub Apps
OAuth Apps
Personal access tokens
New personal access token
Personal access tokens function like ordinary OAuth access tokens. They can be used instead of a password for Git over HTTPS, or can be used to authenticate to the API over Basic Authentication.
Note
myproject_tokken
토큰 이름 설정
What's this token for?
Expiration *
90 days
The token will expire on Mon, Oct 31 2022
엥간해서는 길게 길게 설정
Select scopes
Scopes define the access for personal tokens. Read more about OAuth scopes.
repo
Full control of private repositories
repo:status
Access commit status
repo_deployment
Access deployment status
public_repo
Access public repositories
repo:invite
Access repository invitations
security_events
Read and write security events
workflow
Update GitHub Action workflows

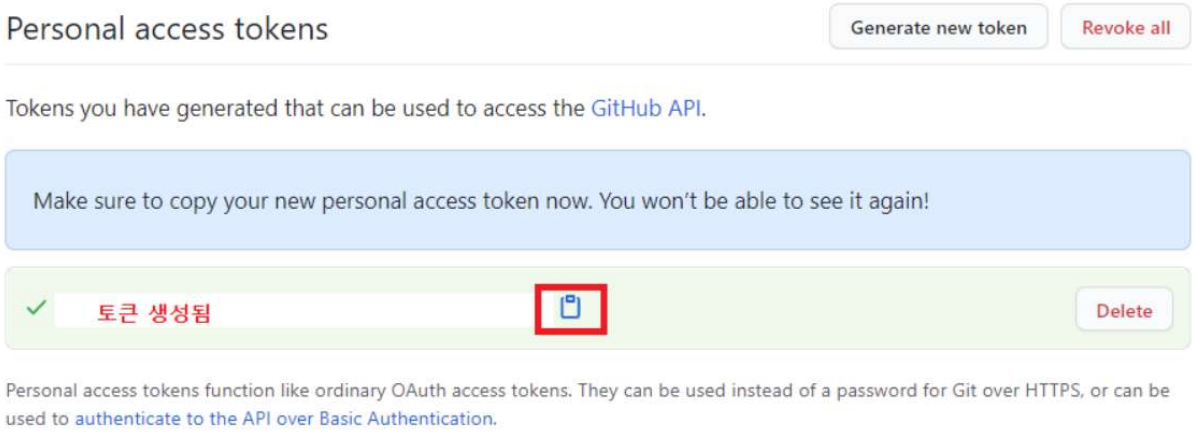
# GitHub 토큰 생성

Select scopes 는 이 토큰을 통해 로그인했을시 이용 가능한 권한을 설정한다고 보면 된다.




Delete 키워드가 들어간 항목을 제외하고 모두 체크

# GitHub 토큰 생성

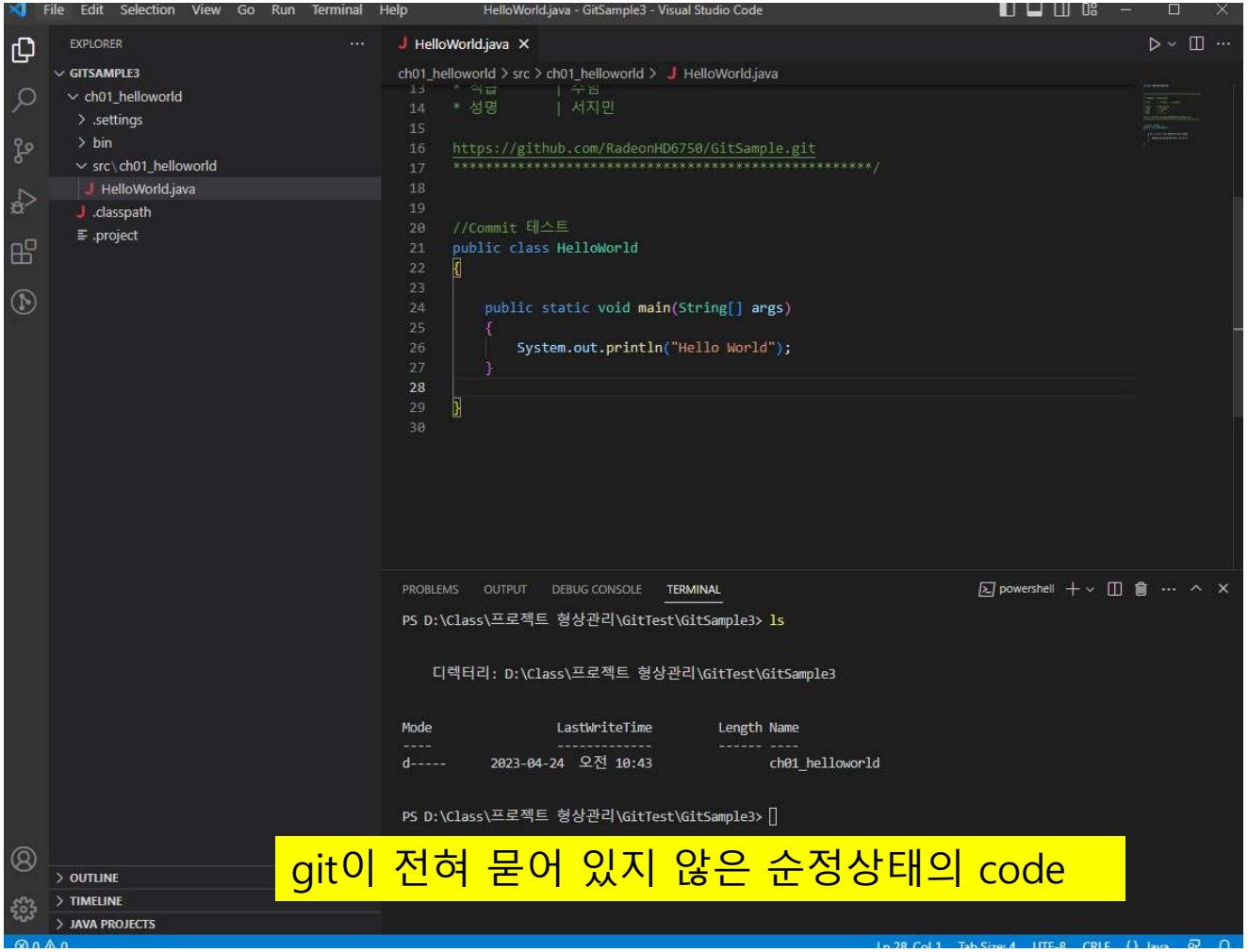



**생성된 토큰을 VS Code나 Eclipse에서 로그인할 때 언제나 사용할 수 있도록 따로 복사해서 저장해놓는다. (로컬파일 뿐 아니라 본인 메일에도 저장하는 걸 추천)**

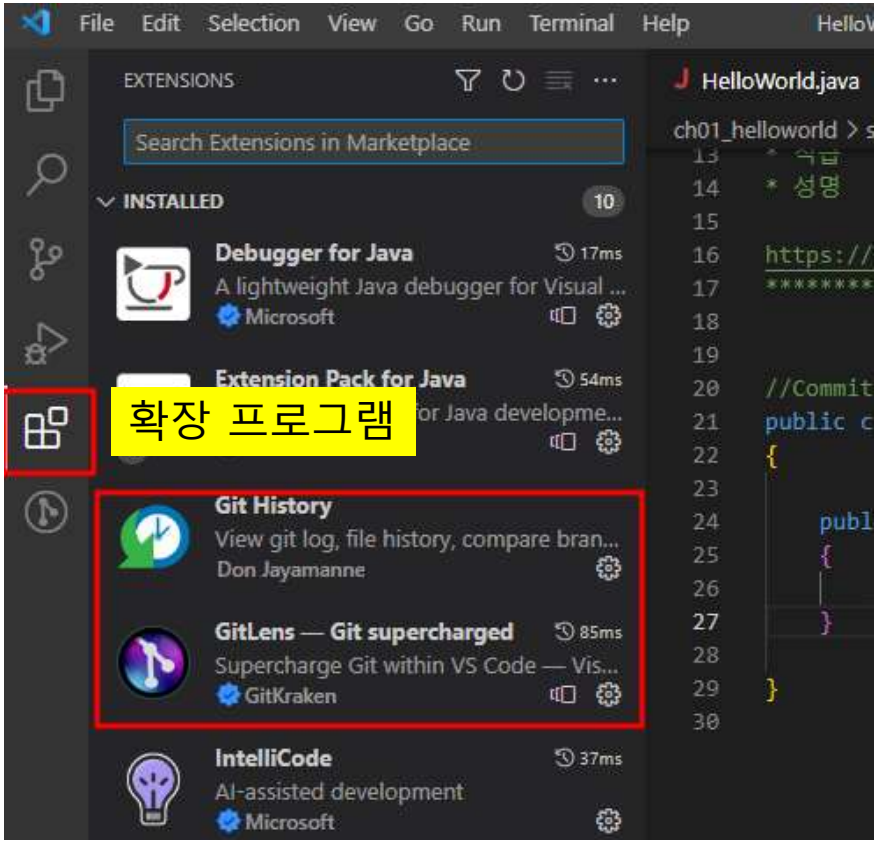
**생성했던 토큰을 잊어버렸다면 그냥 다시 발급받으면 된다.**

# Code에 Github 연동

# Github 최초 연동


File Edit Selection View Go Run Terminal Help
HelloWorld.java - GitSample3 - Visual Studio Code
EXPLORER
GITSAMPLE3
ch01_helloworld
.settings
bin
src\ch01_helloworld
HelloWorld.java
.classpath
.project
HelloWorld.java
ch01_helloworld > src > ch01_helloworld > HelloWorld.java
* 성명 | 서지민
https://github.com/RadeonHD6750/GitSample.git
//Commit 테스트
public class HelloWorld
{
public static void main(String[] args)
{
System.out.println("Hello World");
}
}
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL
powershell
PS D:\Class\프로젝트 형상관리\GitTest\GitSample3> ls
디렉터리: D:\Class\프로젝트 형상관리\GitTest\GitSample3
Mode LastWriteTime Length Name
d----- 2023-04-24 오전 10:43 ch01_helloworld
PS D:\Class\프로젝트 형상관리\GitTest\GitSample3>
OUTLINE
TIMELINE
JAVA PROJECTS
git이 전혀 묻어 있지 않은 순정상태의 code

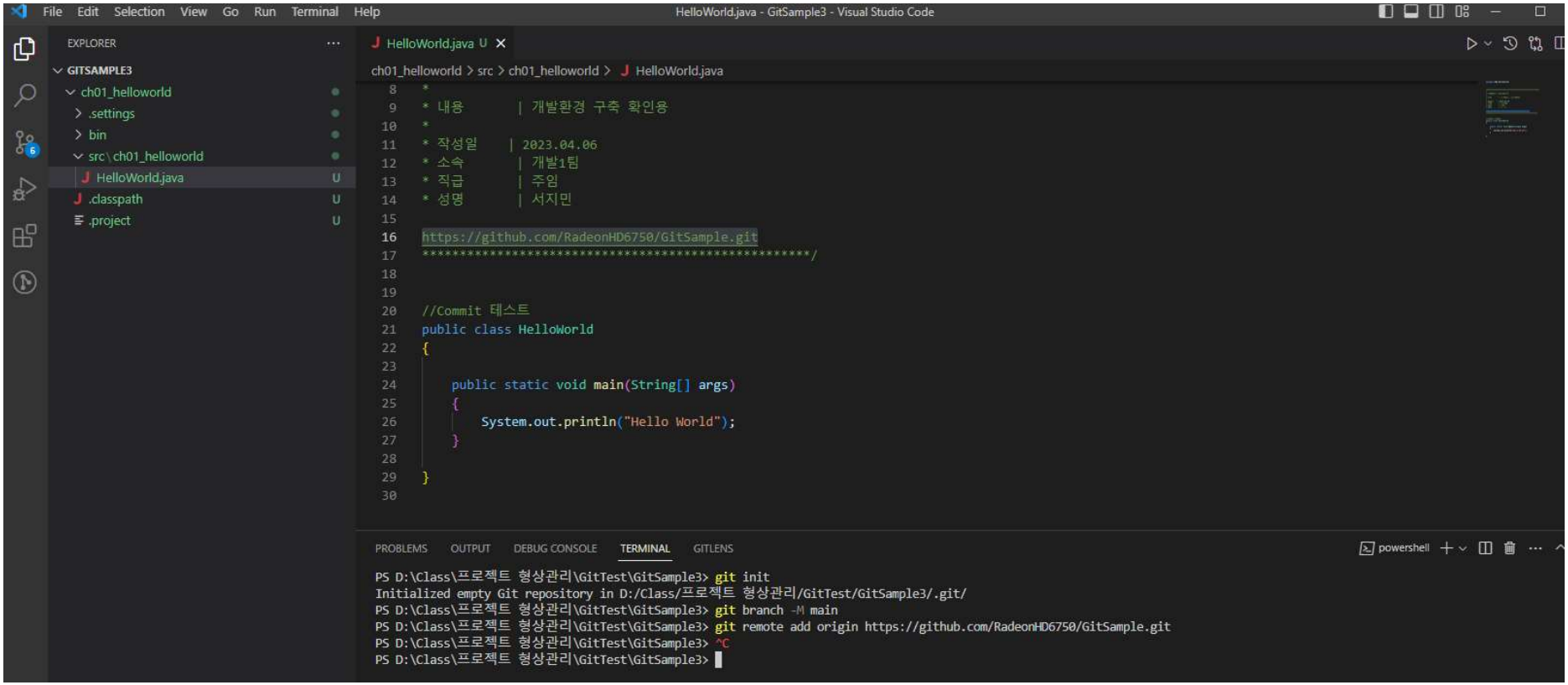
# Github 최초 연동




1. 확장 프로그램 들어가기
2. Git History, GitLens, Graph 설치

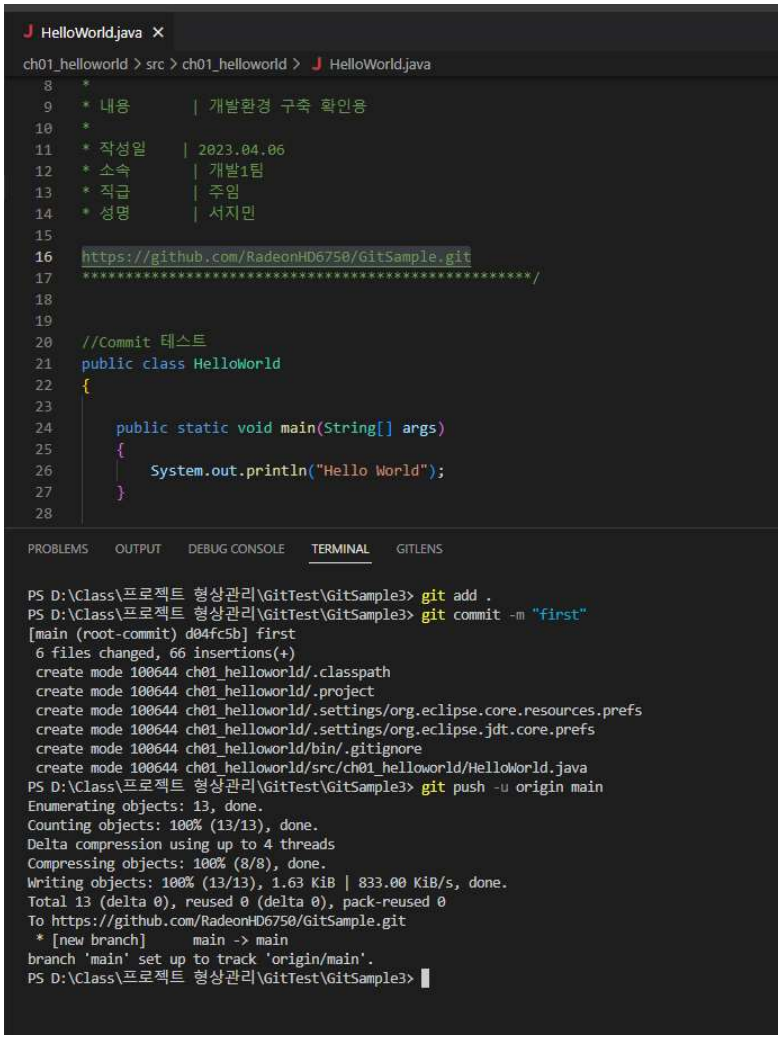
필수는 아니지만, 삶을 좀 더 풍족하게 해준다.

# Github 최초 연동



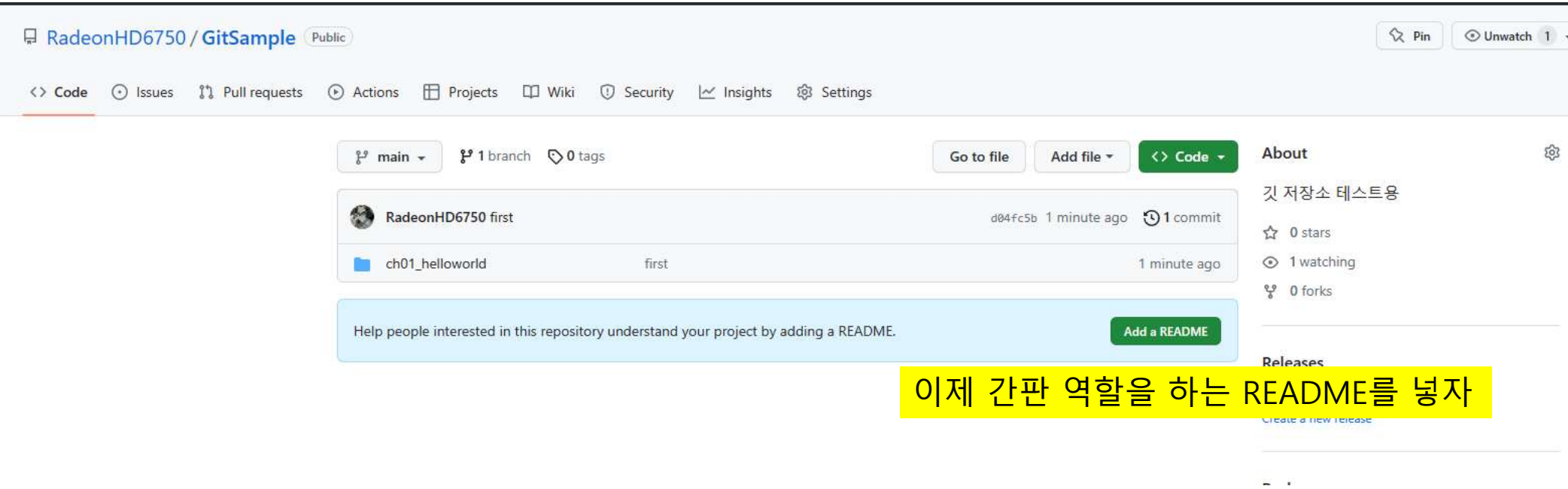

1. git init
2. git branch –M main  master가 아닌 main으로 하자
3. git remote add origin "깃허브 주소"

# Github 최초 연동




3. git add .
4. git commit –m “메세지”
5. git push –u origin main

# Github 최초 연동




비어있던 github 원격 저장소에 작업중이던 내용물 업로드 완료

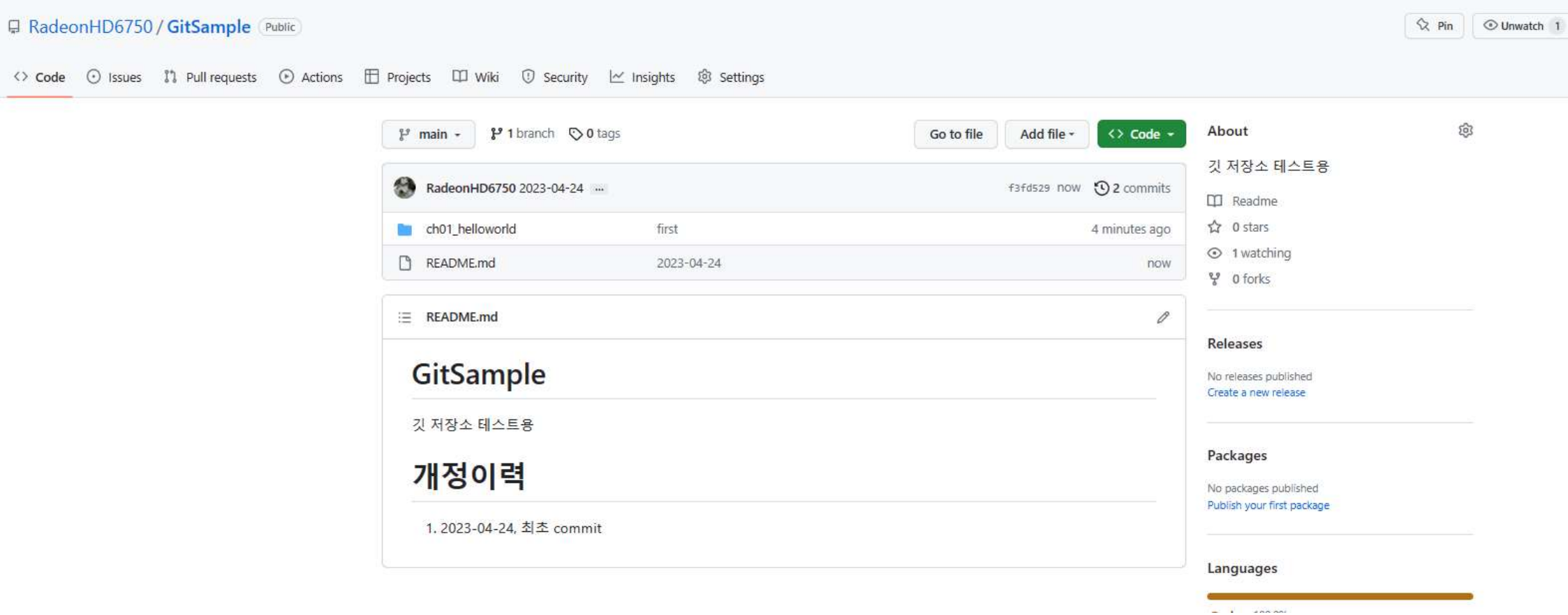
# Github 최초 연동

```
# GitSample
깃 저장소 테스트용


# 개정이력

1. 2023-04-24, 최초 commit
```

READMD.md 파일 내용 작성

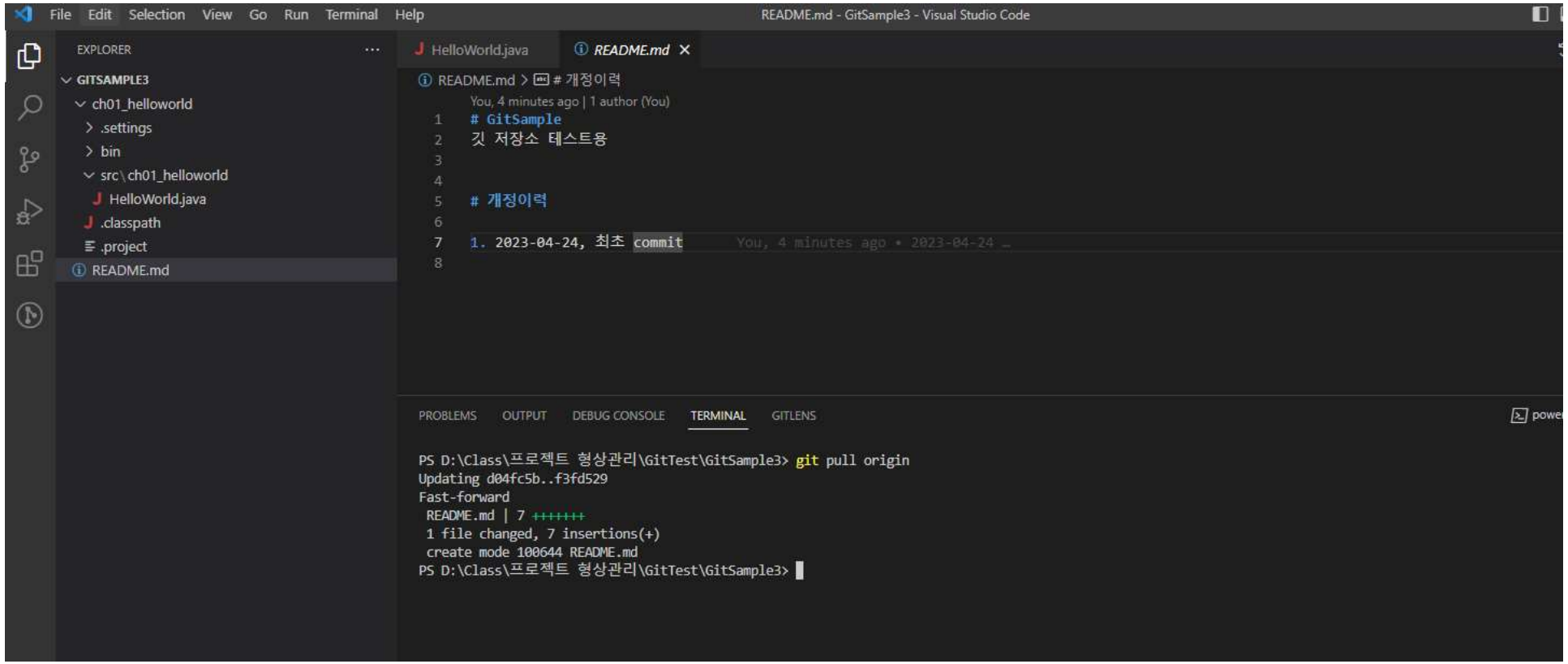

commit new file을 눌러 마저 진행

# Github 최초 연동




저장소에 간판역할을 하는 README.md 파일 생성완료

# Github 최초 연동

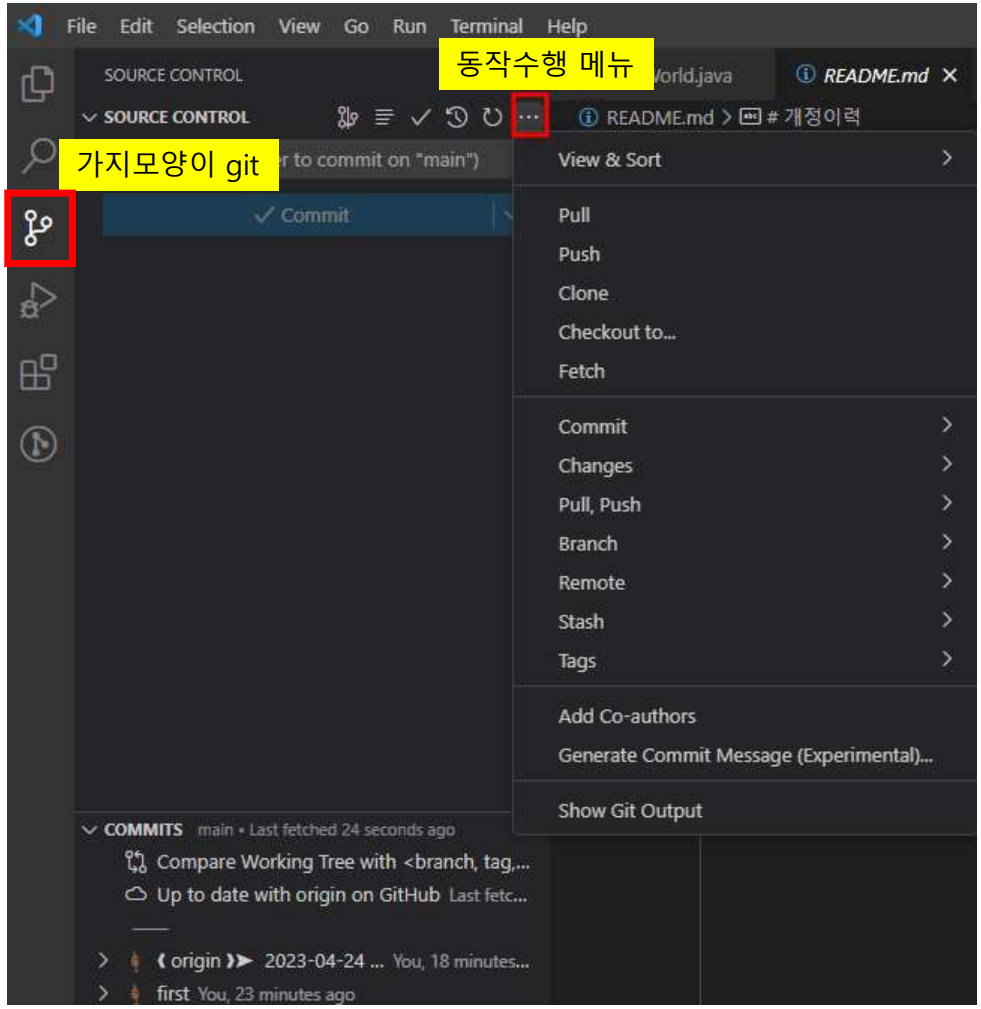



git pull origin을 하여 README.md 파일 다운로드 및 최종 동기화

이것을 안하면 나중에 reject의 원인으로 작용

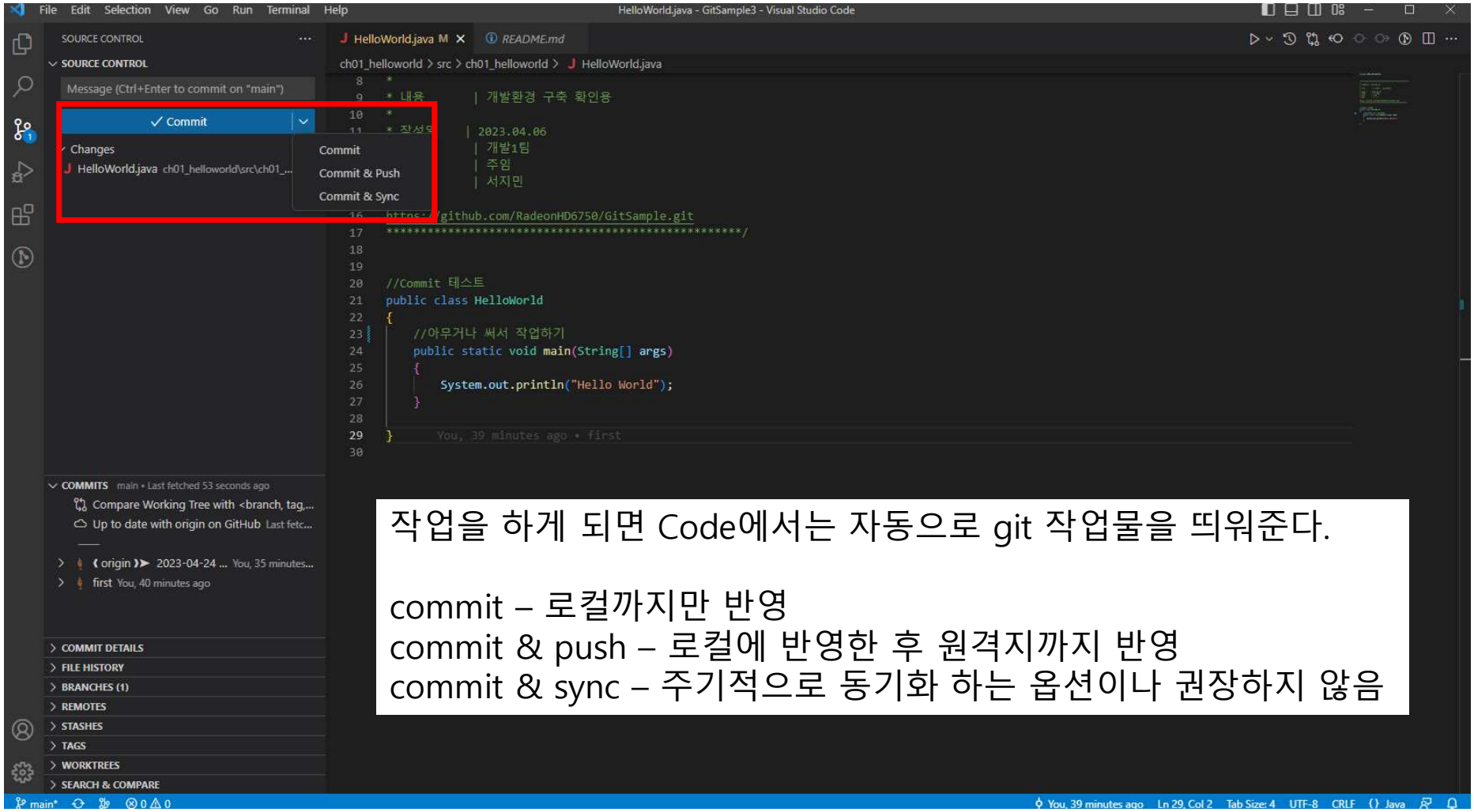
# Github으로 형상관리

# Visual Studio Code 활용




- Visual Studio Code에서는 기본적인 git GUI를 제공
- 앞서 했던 터미널 명령어는 초기설정에만 사용

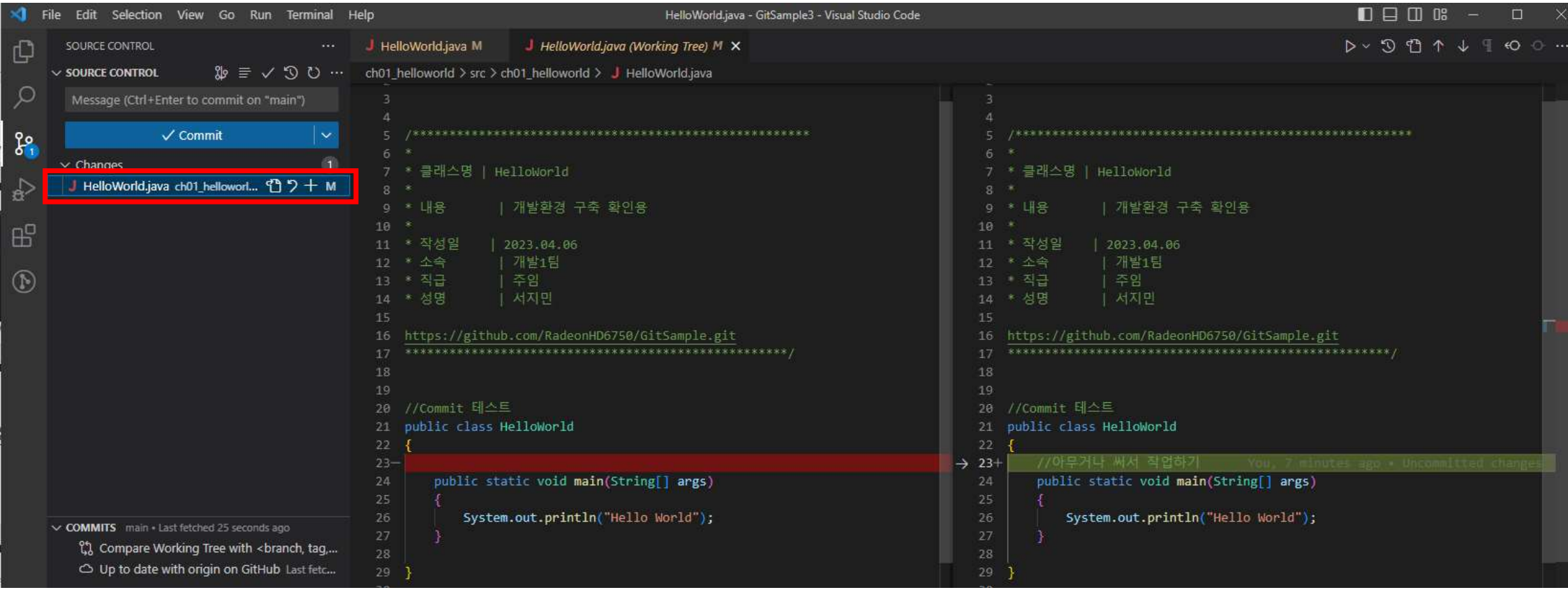
# 작업 후 업로드 하기




작업을 하게 되면 Code에서는 자동으로 git 작업물을 띄워준다.

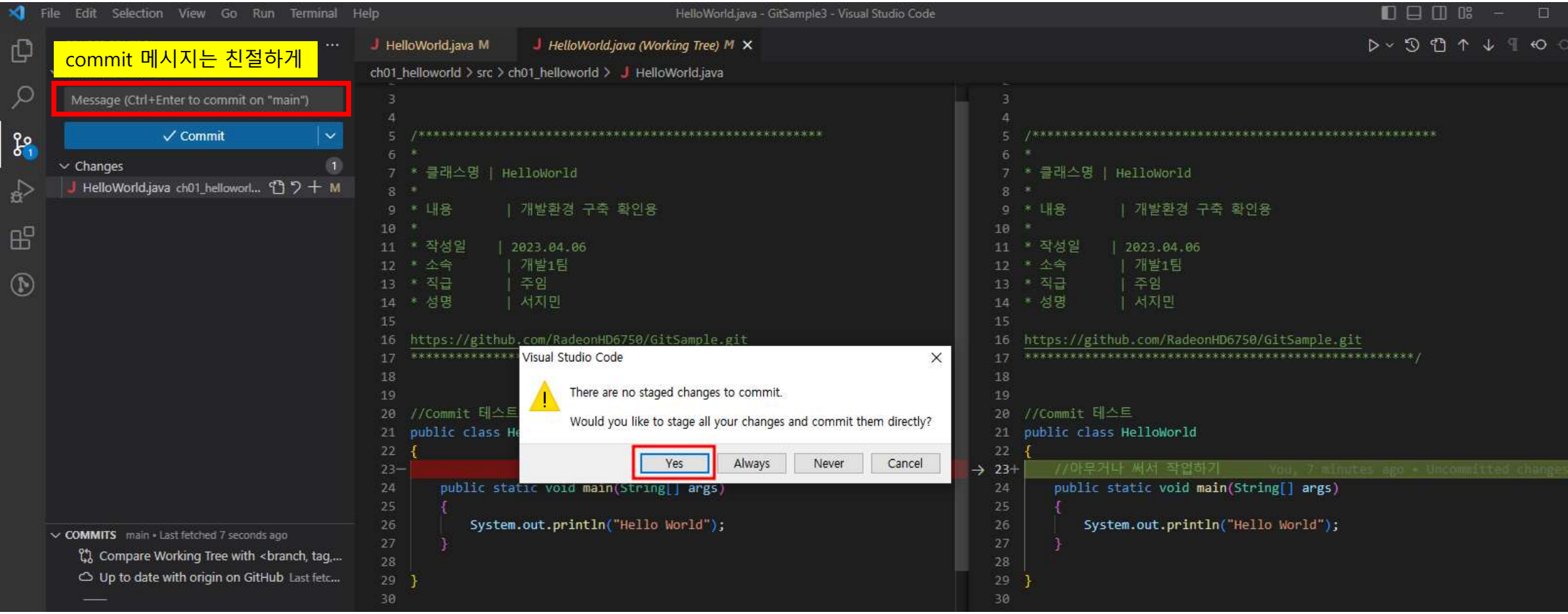
commit – 로컬까지만 반영
commit & push – 로컬에 반영한 후 원격지까지 반영
commit & sync – 주기적으로 동기화 하는 옵션이나 권장하지 않음

# 작업 후 업로드 하기



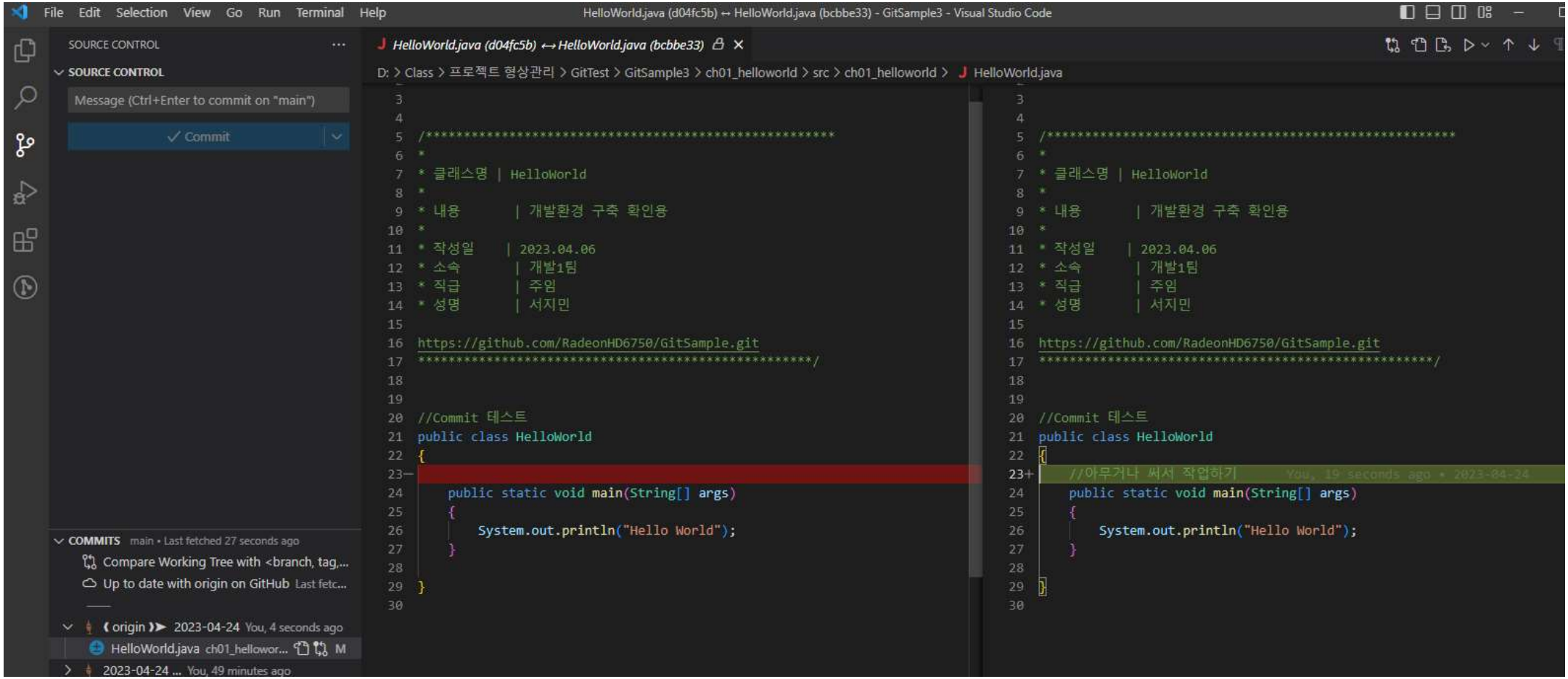

바로 commit & push하지 말고 Changes를 눌러 확인 후 진행

# 작업 후 업로드 하기



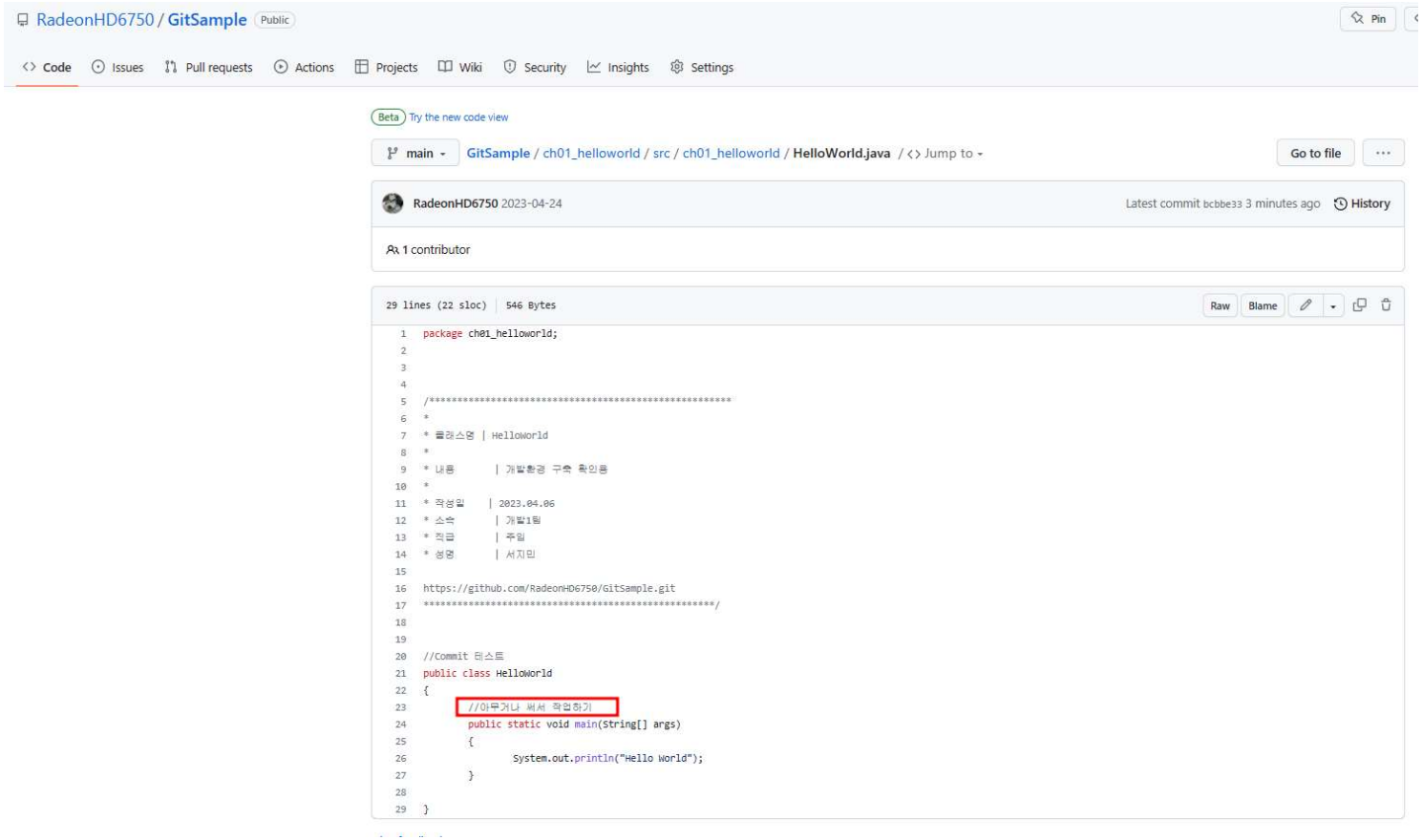

한번에 commit 하고 바로 원격지까지 가겠냐는 질문인데 여기서는 yes

# 작업 후 업로드 하기



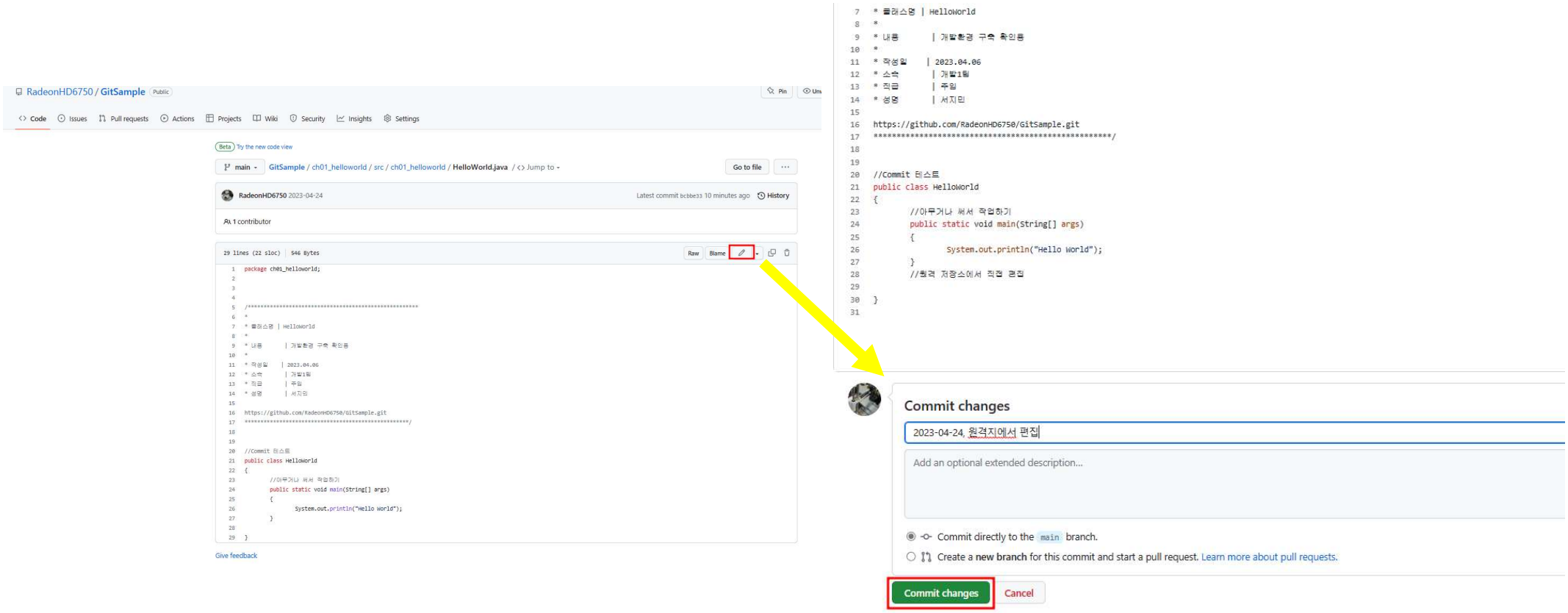

commit이 완료된 모습

# 작업 후 업로드 하기



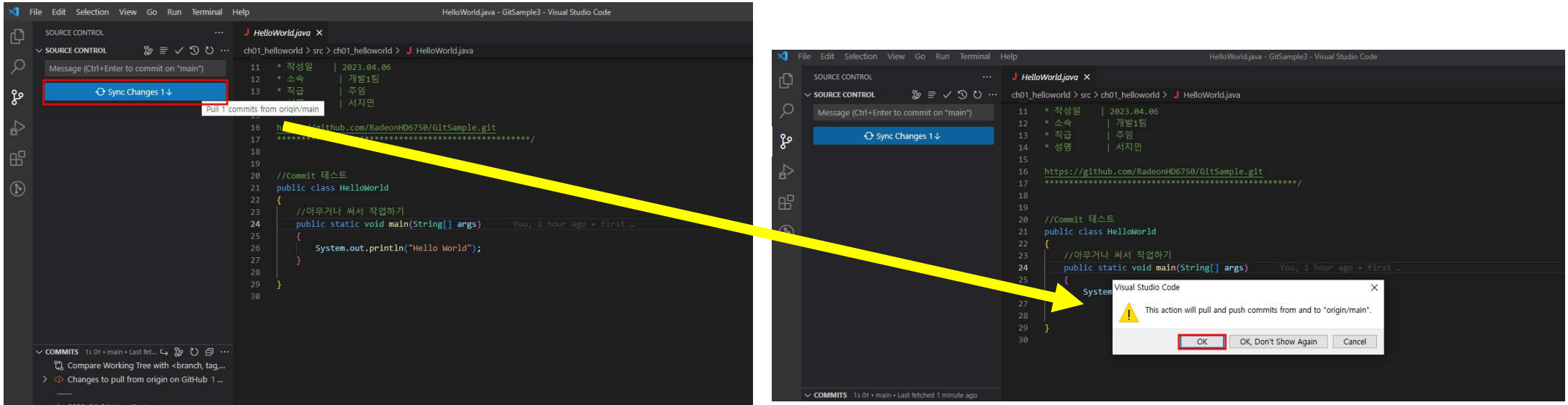

원격 저장소에도 반영된 모습

# 원격지에서 다운로드 받기



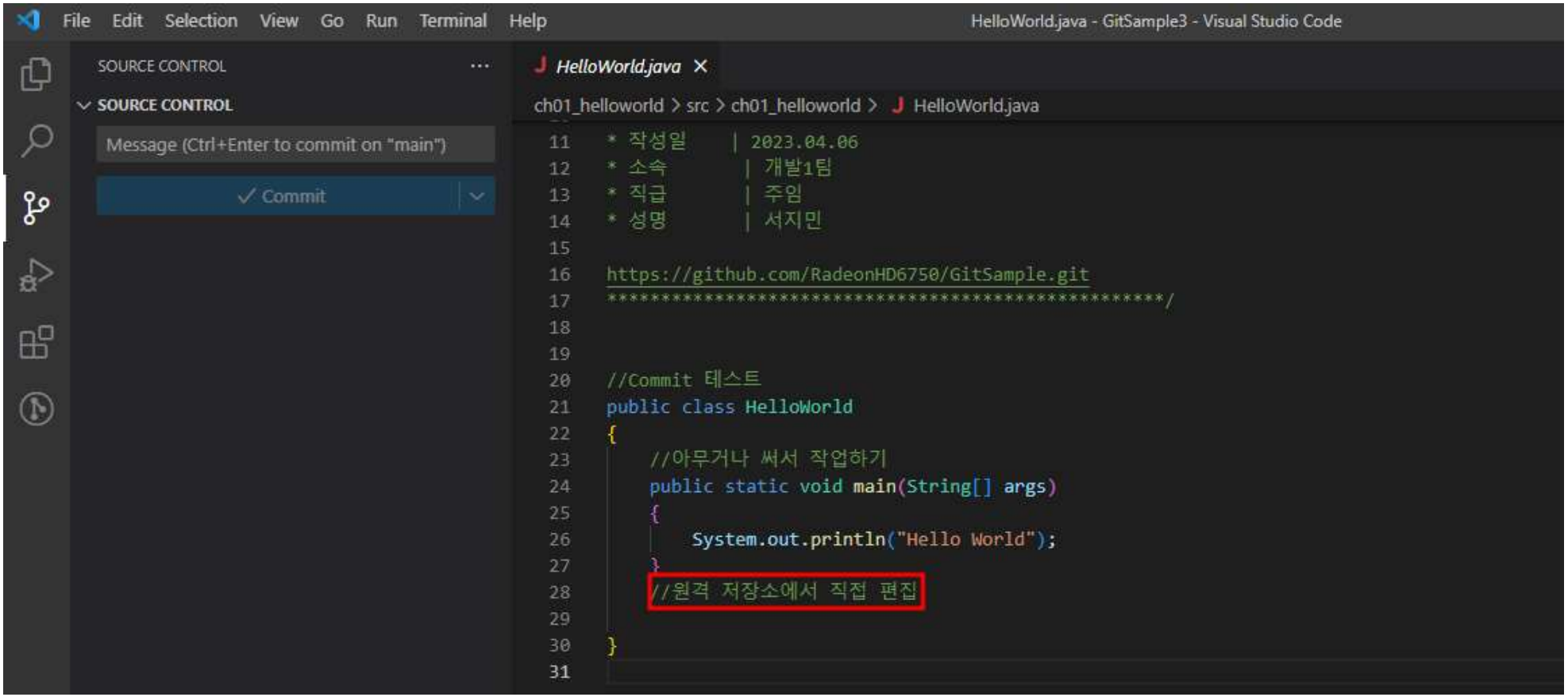

본래 다른 사람이나 다른 장소에서 commit을 했겠지만
여기서는 저장소에서 직접 편집

# 원격지에서 다운로드 받기




인터넷이 연결되어 있다면 자동으로 원격지와 비교 후 동기화 메세지

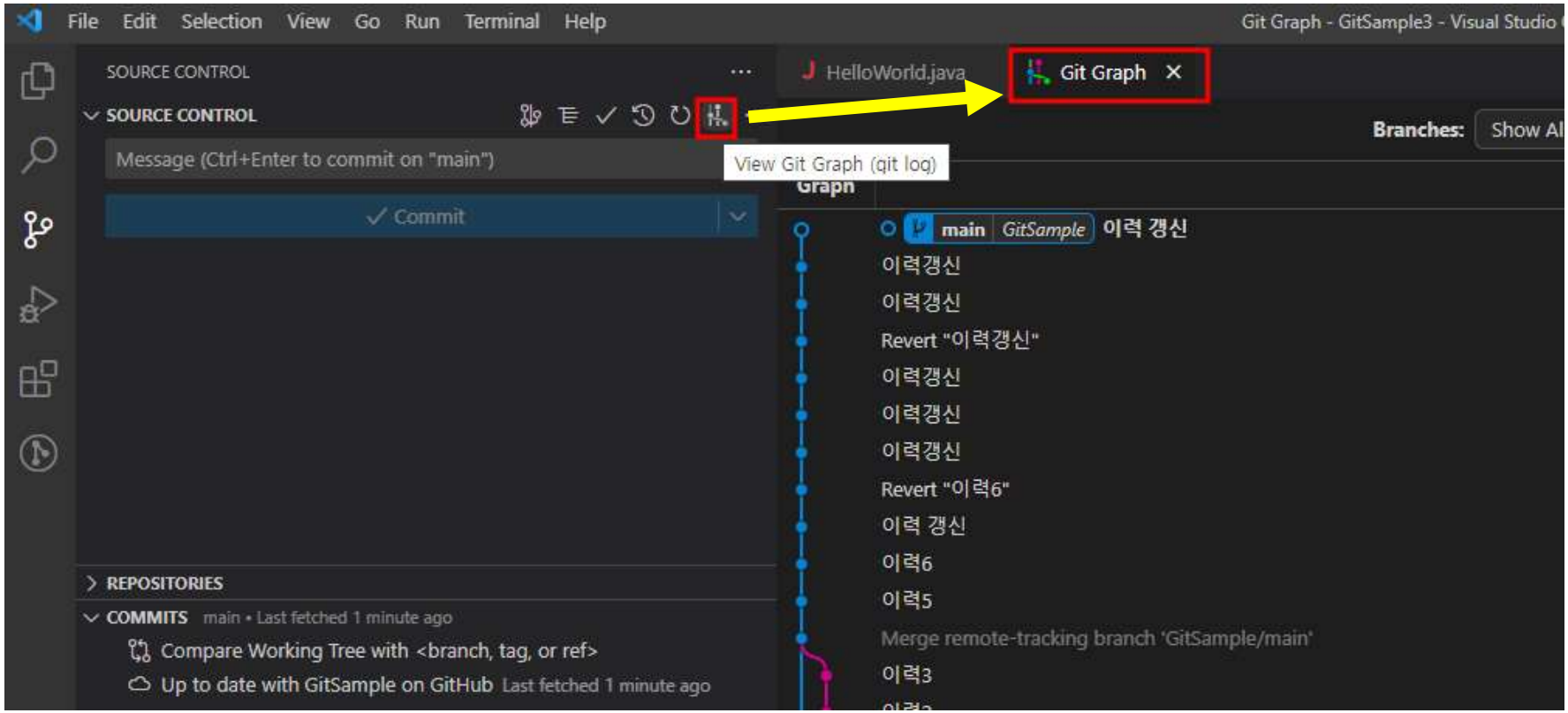
# 원격지에서 다운로드 받기




자동으로 동기화되어 다운로드 후 수정된 모습

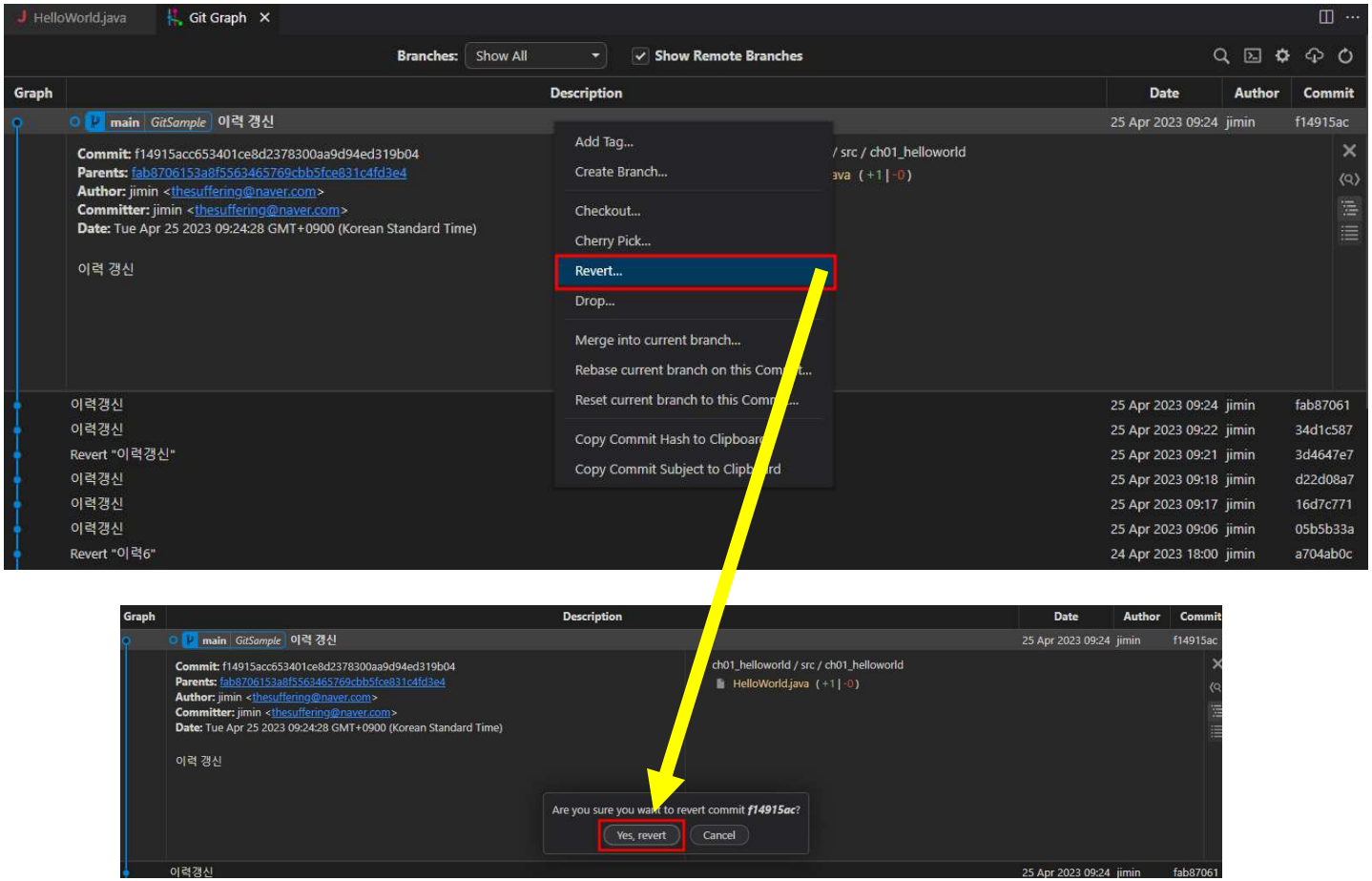
# Rollback

# 복구하기




확장 기능인 Git Graph를 사용하면 훨씬 편하게 복구가능

# 복구하기

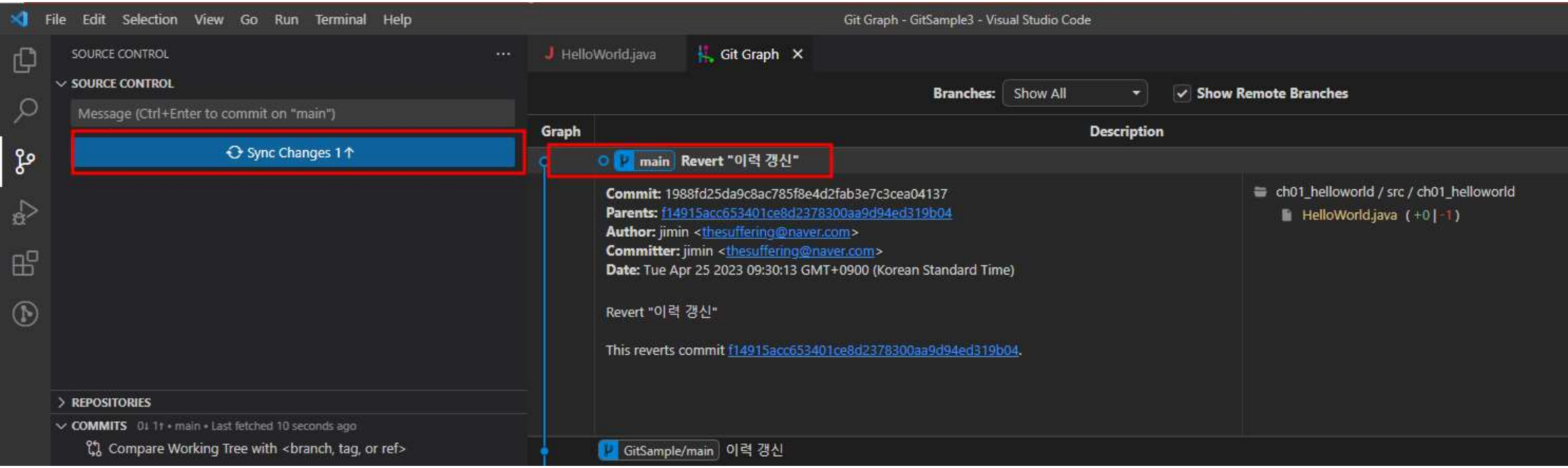






취소하고 싶은 Commit에서 우클릭 후 Revert

# 복구하기




Sync Changes 누르면 자동으로 원격 저장소까지 반영

# 복구하기

```
20  //Commit 테스트
21  public class HelloWorld
22  {
23          //아무거나 써서 작업하기
24          public static void main(String[] args)
25          {
26                  System.out.println("Hello World");
27          }
28          //원격 저장소에서 직접 편집
29
30
31          //이력 1
32          //이력 2
33          //이력 3
34          //이력 4
35          //이력 5
36          //이력 6
37          //이력 7
38          //이력 8
39          //이력 9
40          //이력 10
41  }
```

원격 저장소

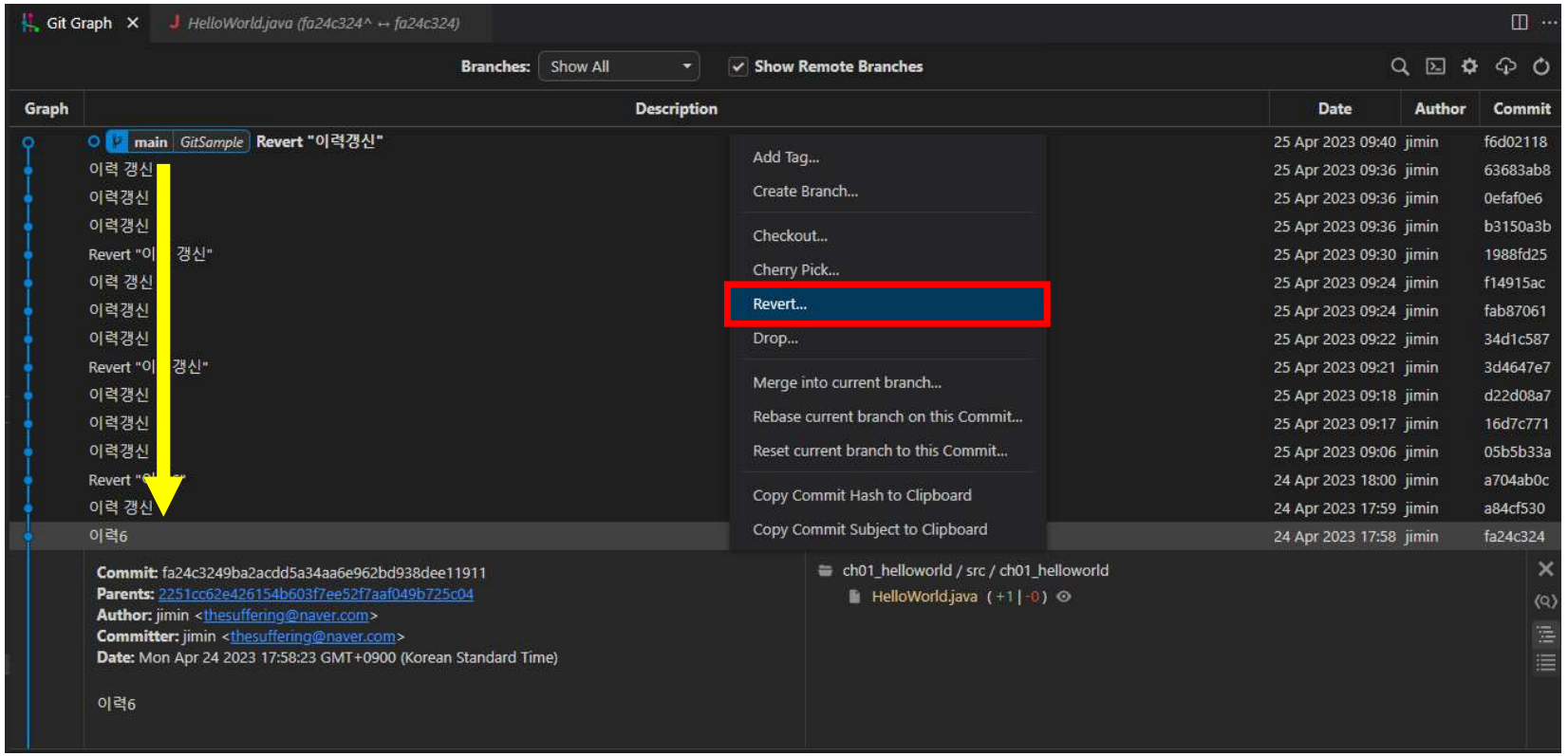
D: > Class > 프로젝트 형상관리 > GitTest > GitSample3 > ch01_helloworld > src > ch01_helloworld > J HelloWorld.java

```
26      System.out.println("Hello World");
27  }
28  //원격 저장소에서 직접 편집
29
30
31  //이력 1
32  //이력 2
33  //이력 3
34  //이력 4
35  //이력 5
36  //이력 6
37  //이력 7
38  //이력 8
39  //이력 9
40  //이력 10
41  //이력 11
42  }
43
```

로컬 저장소

```
26      System.out.println("Hello World");
27  }
28  //원격 저장소에서 직접 편집
29
30
31  //이력 1
32  //이력 2
33  //이력 3
34  //이력 4
35  //이력 5
36  //이력 6
37  //이력 7
38  //이력 8
39  //이력 9
40  //이력 10      You, 8 minutes ago • 이력갱신 …
41  }
42
```

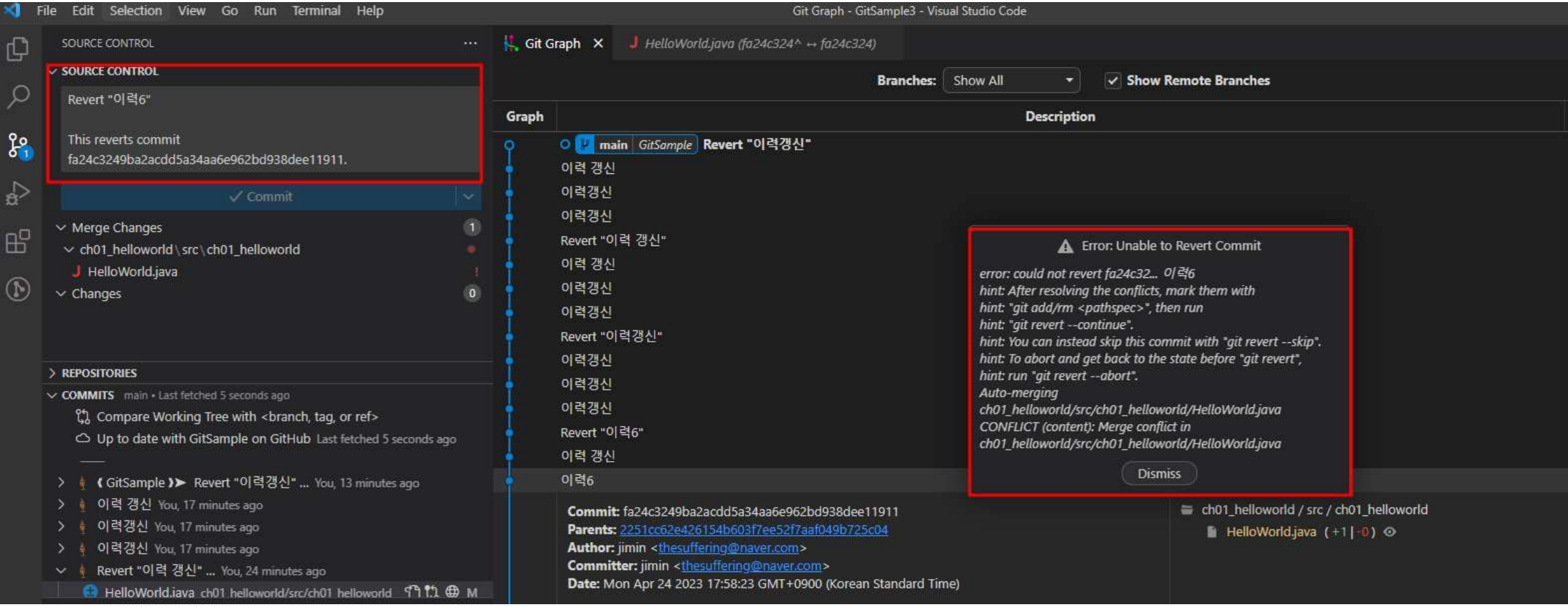
# 머나먼 과거로 돌아가기




직전 단계 뿐 아니라 아예 머나먼 과거로 까지 돌아갈 수 있다.

마찬가지로 돌아가고자 하는 commit 바로 직후의 commit을 취소 시키면 된다.

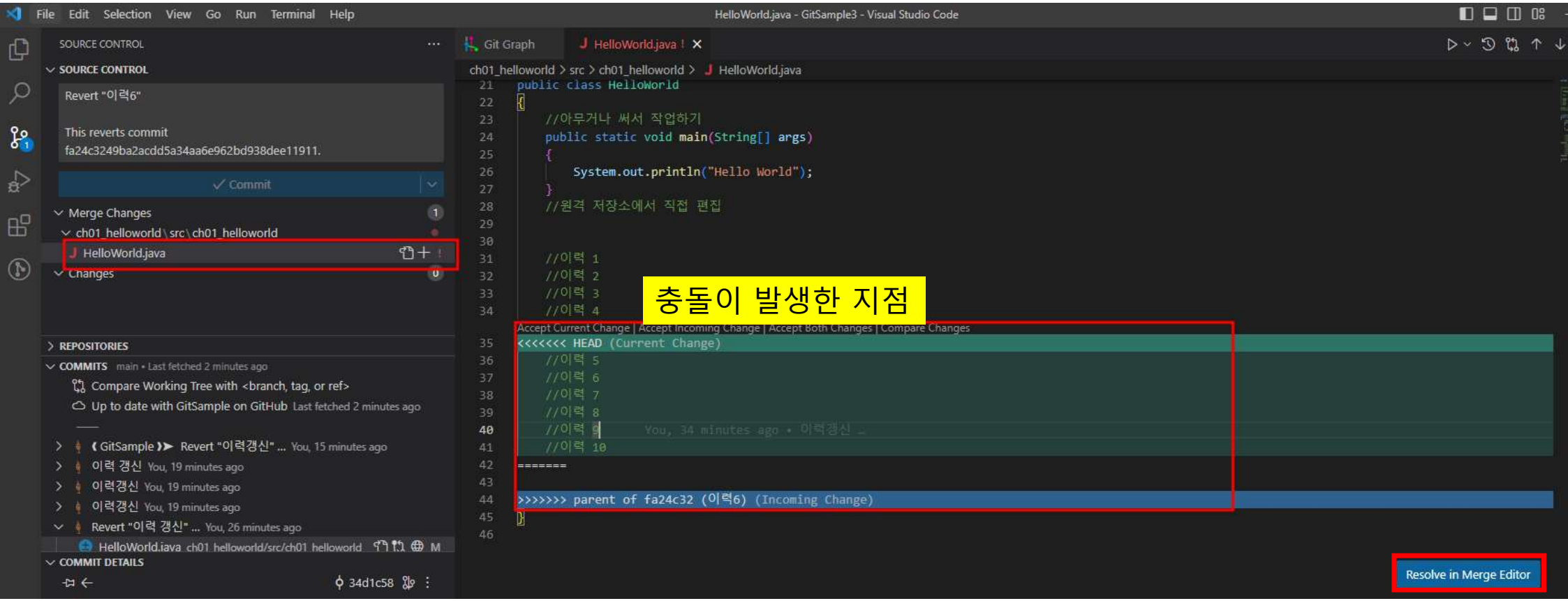
# 머나먼 과거로 돌아가기




Revert 하자마자 위와 같은 오류가 발생할 것이다.

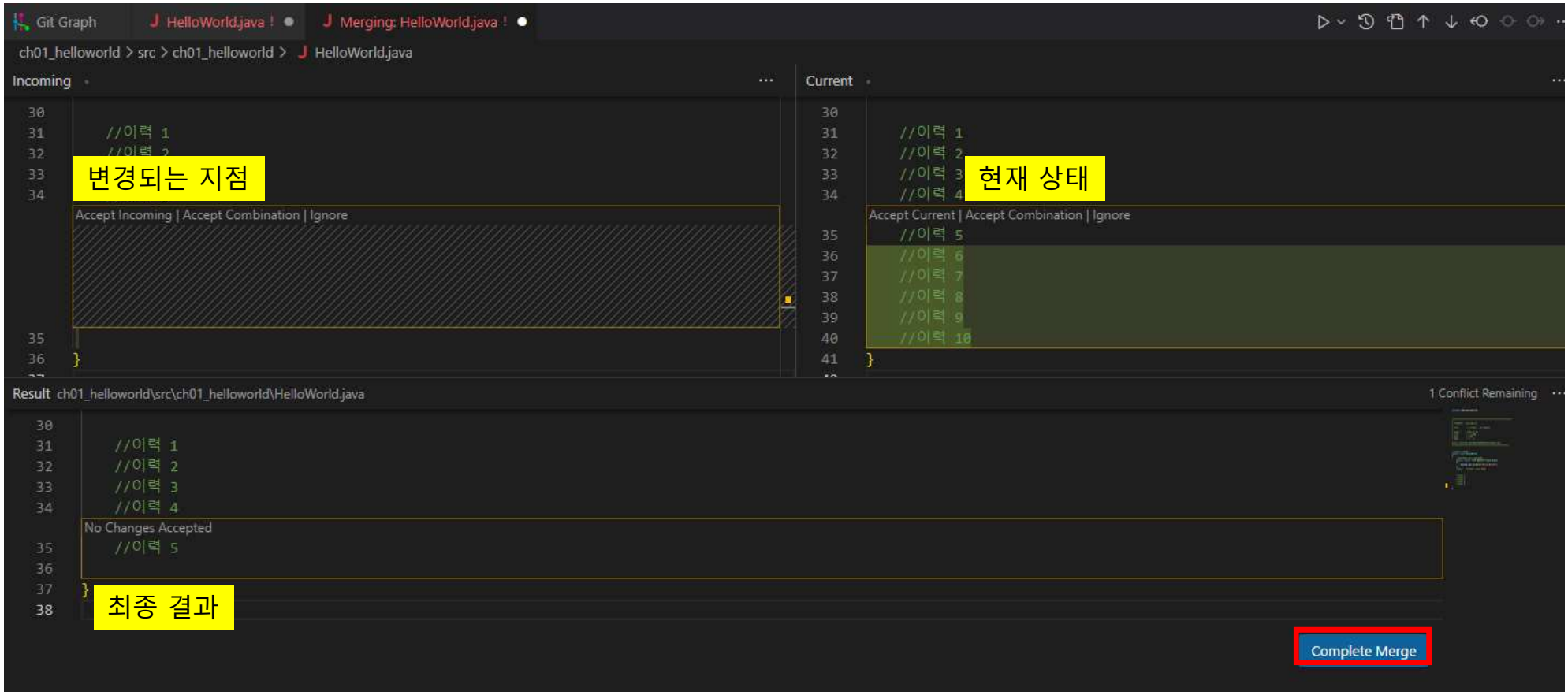
바로 너무 먼 과거 여서 현재와 충돌한 것이다.

# 머나먼 과거로 돌아가기



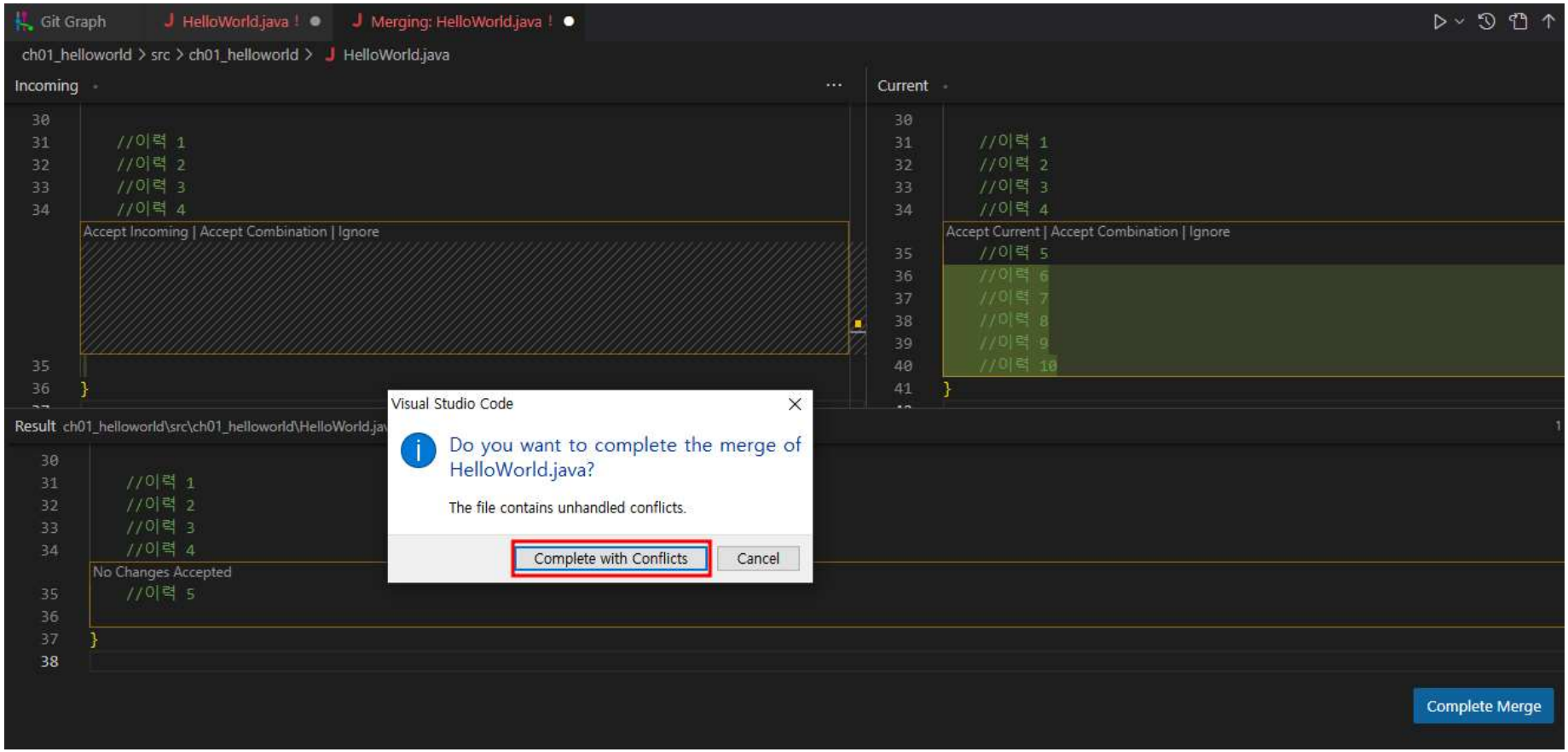

현재의 코드와 과거의 코드가 충돌한 것이니 Merge Editor를 사용하여 조정해주자

# 머나먼 과거로 돌아가기

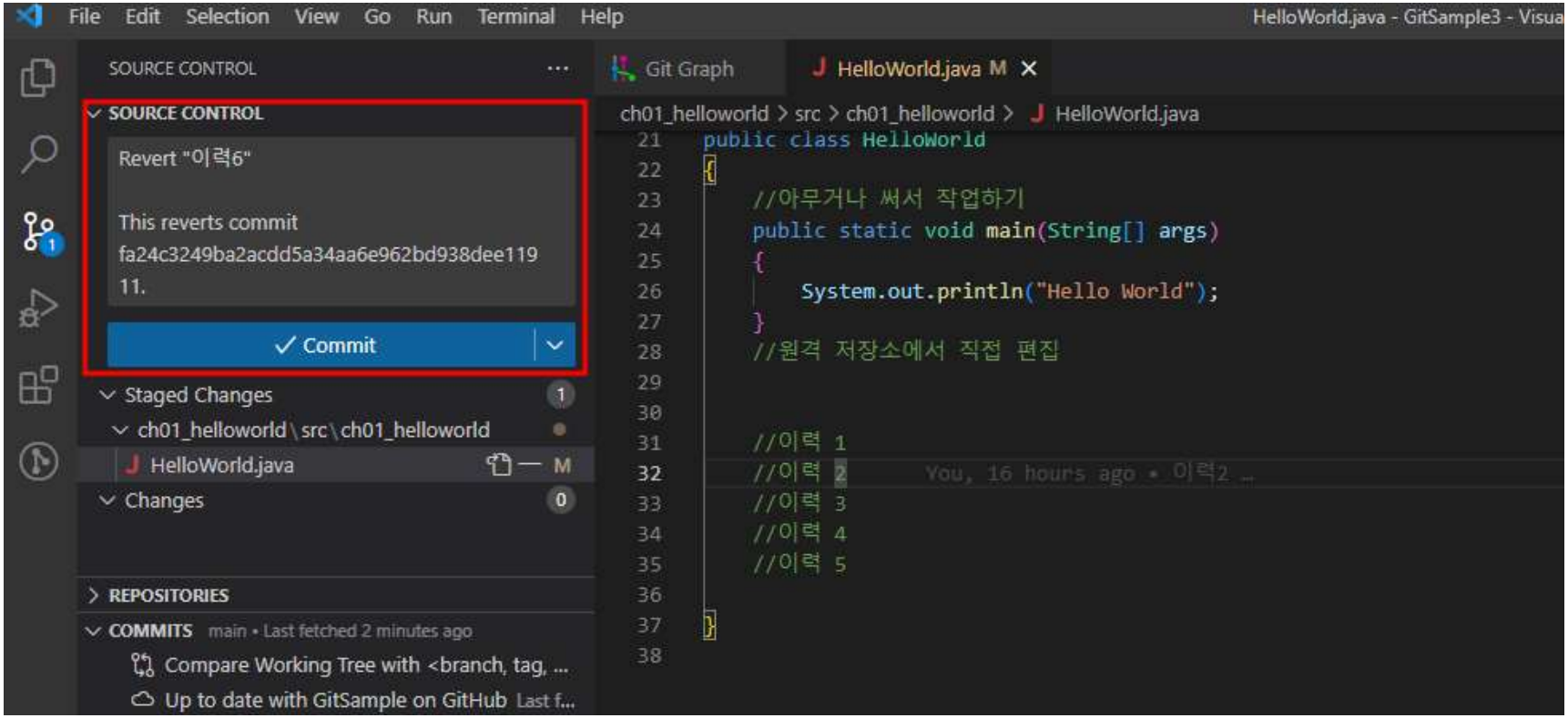



현재 상태를 편집하거나 아니면 자동으로 반영된 결과를 보고 Merge를 하자

# 머나먼 과거로 돌아가기


Git Graph
HelloWorld.java !
Merging: HelloWorld.java !
ch01_helloworld > src > ch01_helloworld > HelloWorld.java
Incoming
Current
Accept Incoming | Accept Combination | Ignore
Accept Current | Accept Combination | Ignore
Result ch01_helloworld\src\ch01_helloworld\HelloWorld.ja
No Changes Accepted
Visual Studio Code
Do you want to complete the merge of HelloWorld.java?
The file contains unhandled conflicts.
Complete with Conflicts
Cancel
Complete Merge

# 머나먼 과거로 돌아가기




merge 이후에 마저 commit & push 해주도록 하자

# 머나먼 과거로 돌아가기

```
//Commit 테스트
public class HelloWorld
{
	//아무거나 써서 작업하기
	public static void main(String[] args)
	{
		System.out.println("Hello World");
	}
	//원격 저장소에서 직접 편집


	//이력 1
	//이력 2
	//이력 3
	//이력 4
	//이력 5

}
```

이력 10까지 있었다가 훨씬 더 과거인 이력5로 돌아간 모습이다.

터미널로 직접 다루는 것 보다 어느 정도 절차가 짜여진 GUI가
실수를 방지해준다.