

Git 공부 시작 전, 필수 지식

개발자 = 프로그래머

프로그래머를 **개발자**라고 부른다.

SW 프로그램을 만드는 사람 = 개발자

프로그램?

0과 1로 된

컴퓨터 명령어가 가득한 문서

```
100101110010101010  
100101110010101010  
100101110010101010  
100101110010101010  
100101110010101010  
100101110010101010
```



프로그램 실행방법

0과 1로 가득 찬,
컴퓨터 명령어로 가득찬,
문서를 더블클릭 하면 실행됨



100101110010101010
100101110010101010
100101110010101010
100101110010101010
100101110010101010
100101110010101010



개발자들이 프로그램 만드는 방법

예전

- 0과 1을 하나씩 타이핑 했음

현재

1. 영어로 된 **소스코드**를 작성한다.
2. 번역기 돌린다.
→ 프로그램이 자동으로 만들어진다.

개발자가 하는 역할

1. 소스코드를 작성한다.
2. 번역기를 돌린다.
3. 프로그램을 완성하여, 테스트 해본다.

[참고] 소스코드 종류

C언어 문법으로 작성한 소스코드

Java 문법으로 작성한 소스코드

C++ 문법으로 작성한 소스코드

Python 문법으로 작성한 소스코드

JavaScript 문법으로 작성한 소스코드

...

엄청 많다.

5분간 정리 시간을 갖자.

[내용을 99%가 아닌, 100% 이해하자!!]

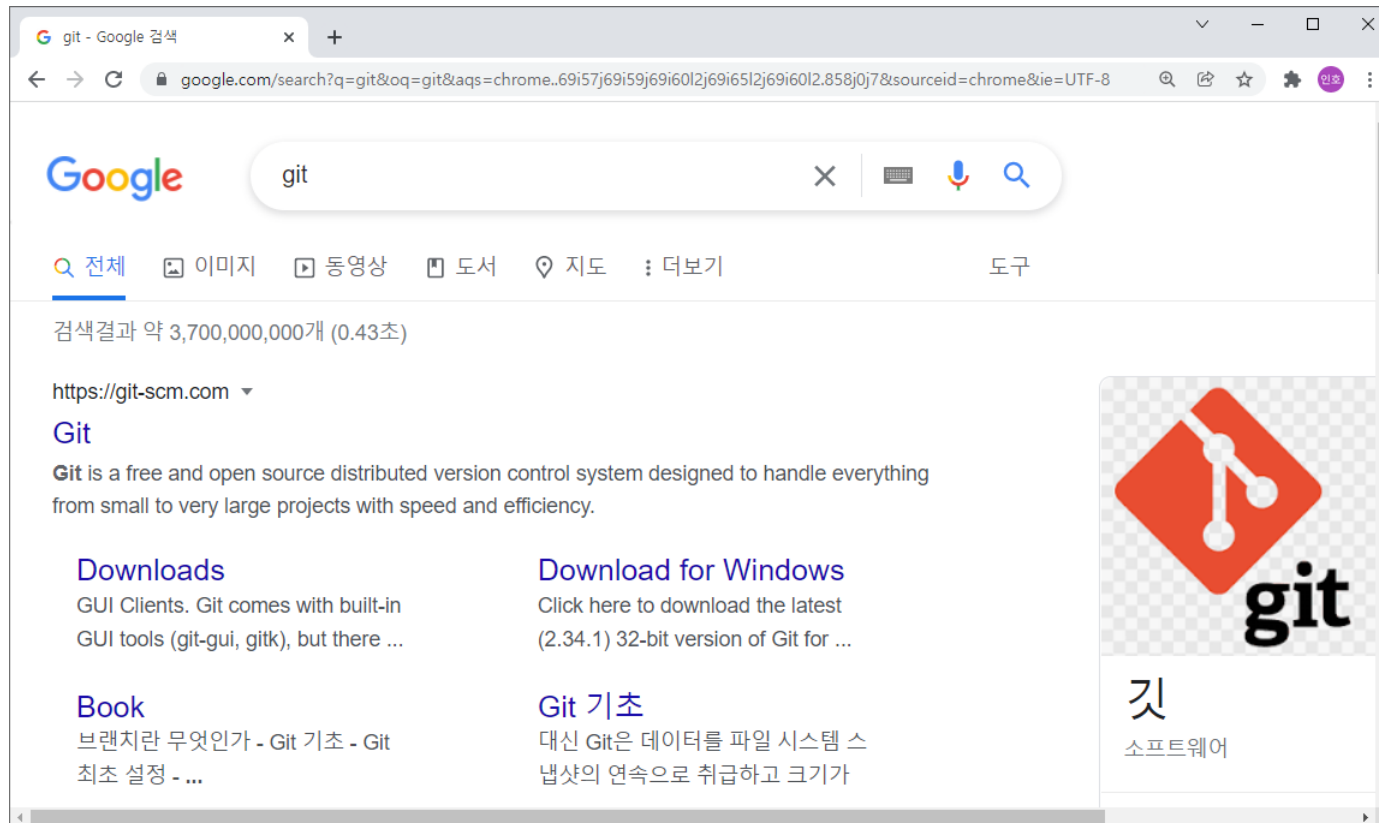
Git 개념잡기

소스코드는 재산이다.

개발자들은 git에다가 소스코드를 저장한다.

왜? **소스코드 날리는 경우 방지!!**

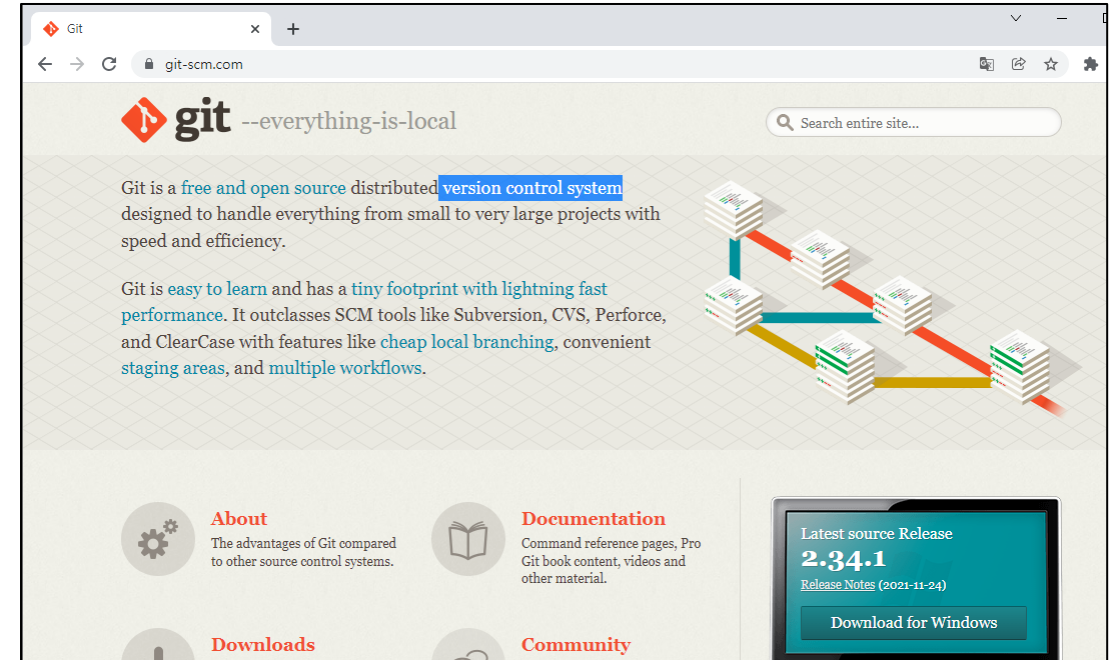
구글에 git 검색



Git = 버전 관리 시스템의 대표적인 프로그램

Version Control System

- 문서들을
게시판 처럼
저장하는 곳
- 여기에 저장(업로드)하면
뒤편어쓰기로 파일을 날리는 경우
없음!!!



"버전 관리 " 라는 의미는 안중요

용어 암기 1

내 소스코드를 "내 컴퓨터의 Git"에 저장 하는 것

Commit

내 소스코드를 "외부"에 있는 안전한 금고에 저장하는 것

Push

정리

개발자들은 본인의 소스코드를 Commit 후, 모아서 Push 한다.

용어 암기 2

문서 or 소스코드를 저장하는

두 가지 금고

1. 내 컴퓨터 저장소 : Local Repository
2. 원격지 저장소 : Remote Repository

내 컴퓨터는 망가질수 있다!



Local Repository



Remote Repository

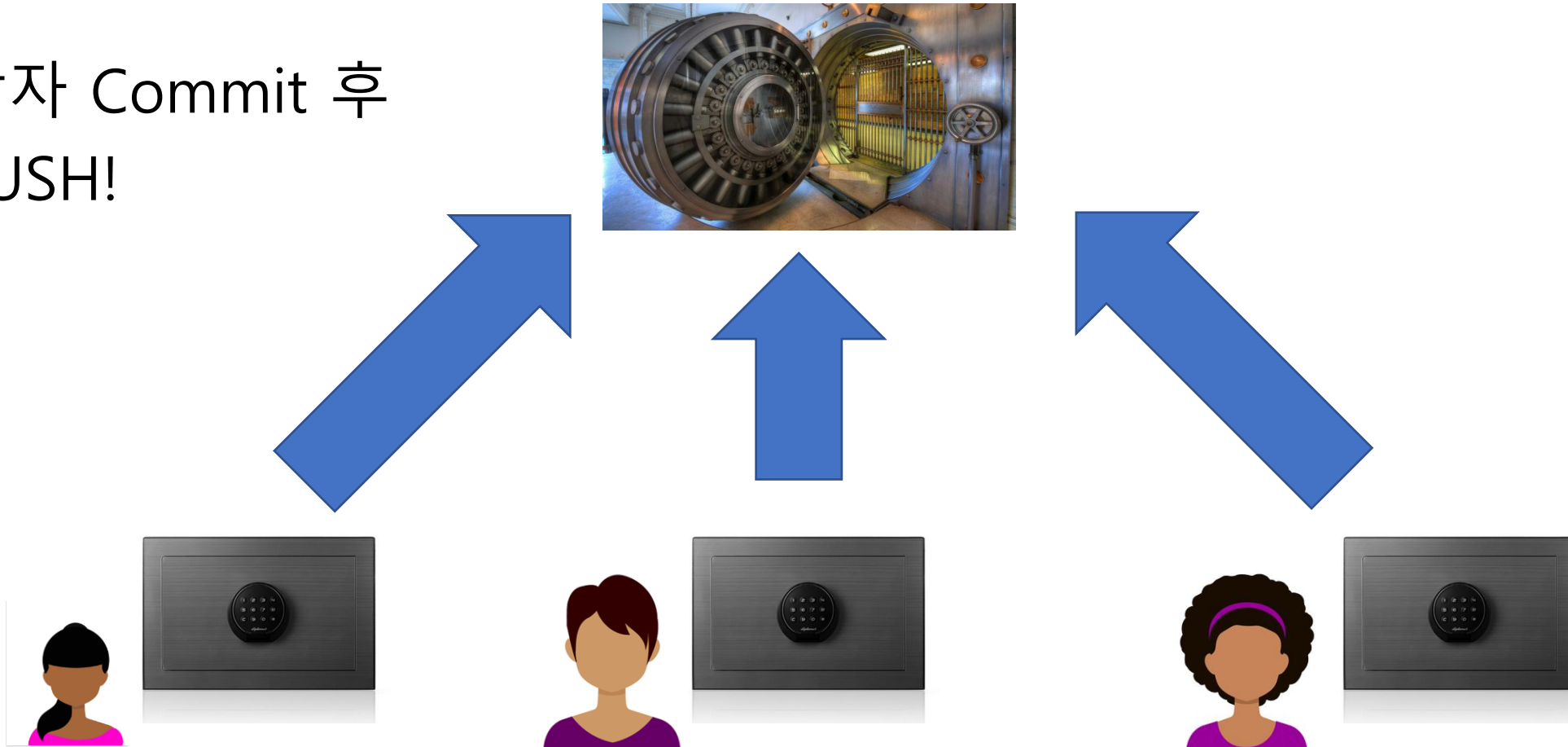
다음 용어를 이해해보자.

개발자들은
본인의 소스코드를
Local Repository에 Commit 한다.

이러한 Commit 된 Source Code들을 모아서,
Remote Repository에 Push 하여 안전하게 저장한다.

개발자들 일하는 방식

각자 Commit 후
PUSH!

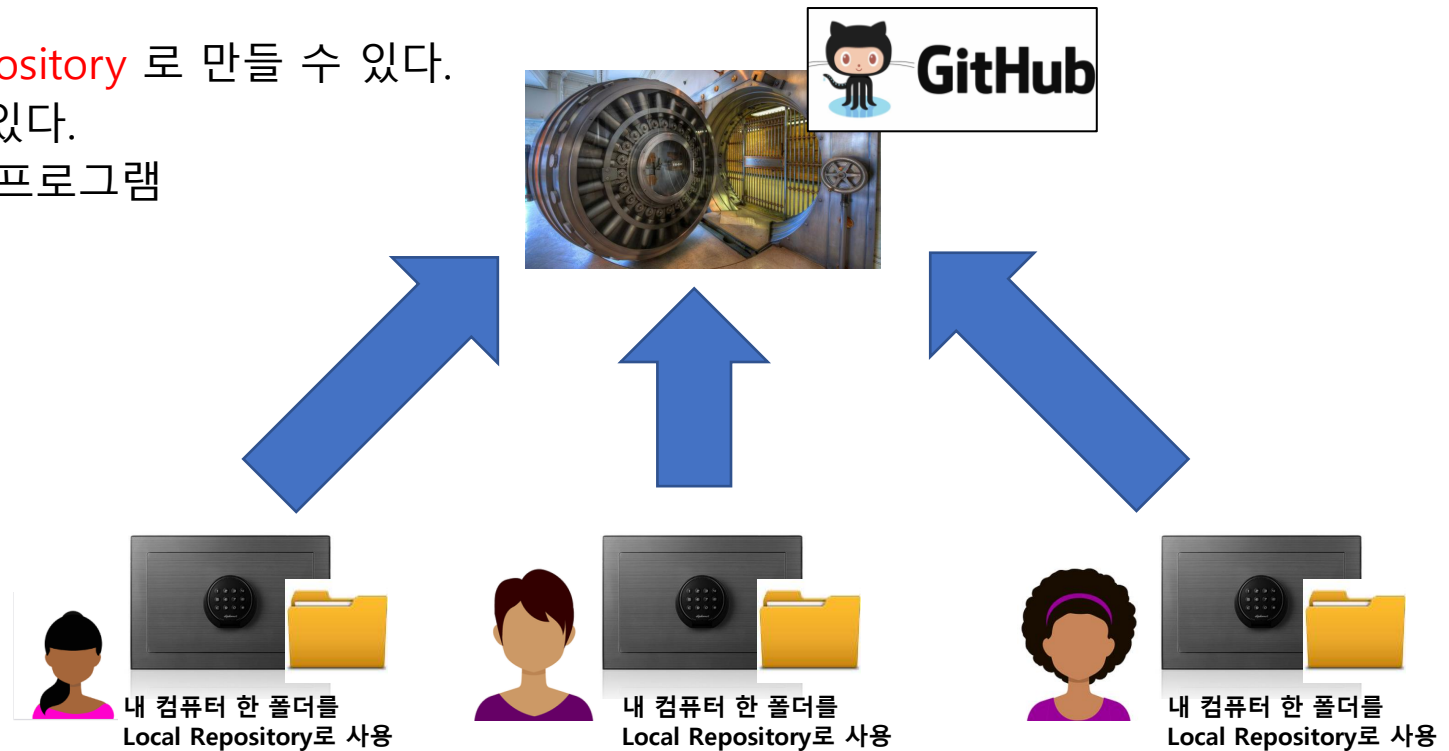


[중요] Git과 GitHub의 이해

GitHub : 단순한 원격저장소

Git

- 내 컴퓨터 내부의 한 폴더를 **Local Repository** 로 만들 수 있다.
- Local Repository 에 **Commit** 도 할수 있다.
- **GitHub**에 소스코드도 **Push**할 수 있는 프로그램



[Quiz 1] Git은 프로그램인가 저장소인가?

Git은 프로그램일까!?

[Quiz 1] Git은 프로그램이다.

Git은 저장소가 아니다!!

Git 이라는 프로그램을 사용해서,

1. 내 폴더 하나를 Local Repository 로 사용 가능
2. Commit도 가능
3. Remote Repository 에 내 코드도 Push 할 수 있음

[Quiz 1] GitHub과 Git을 쉽게 이해하는 법

GitHub은 스택래프트 / 롤 처럼 실행이 가능

Git은 스택래프트 / 롤 처럼 실행할까?

[알아두기] Git의 그 밖에 많은 기능들

Git 프로그램은 정말 많은 기능들이 있다!

- ex) 히스토리 관리
- ex) 예전 소스코드로 복원
- ex) 다른 사람 소스코드 다운로드 받기
- ex) 원격지 Repository에 코드 Push 하기
- ex) 내 컴퓨터 특정 폴더를 Local Repository 로 활용하기

[Quiz 2] GitHub은 소스코드를 보관하는 웹사이트?

GitHub은 웹사이트로 되어있으면서,
소스코드를 저장하는 금고 역할을 하는가? (Yes or No)

[Quiz 2] 그렇다.

우리가 실습 할 내용

1. Git을 설치한다. Git은 프로그램이다.
2. GitHub 에 가입을 해보자.
 - GitHub은 Remote Repository 이다.

10분간 정리 시간을 갖자.

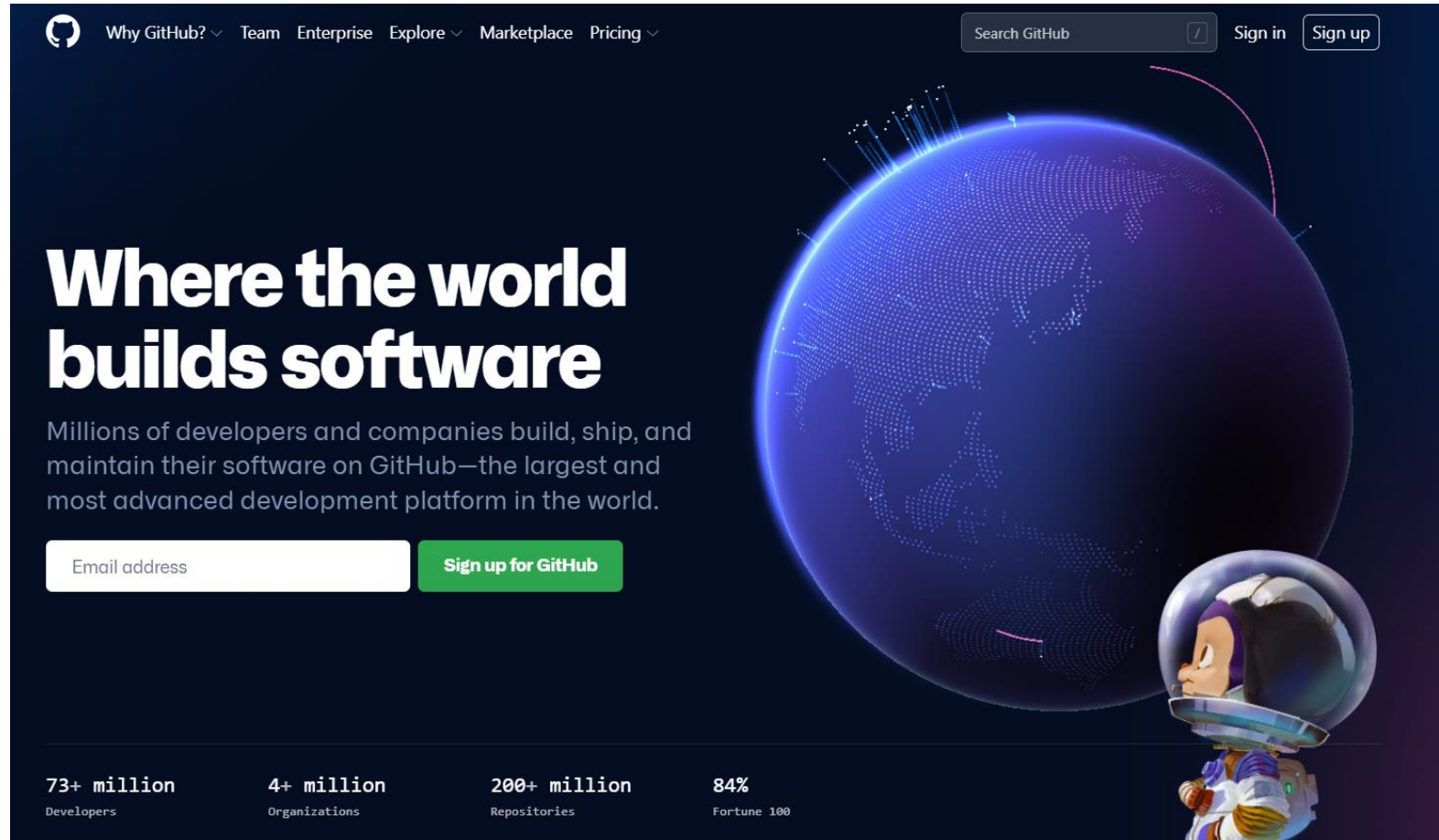
[내용을 99%가 아닌, 100% 이해하자!!]

GitHub 가입하기

Remote Repository 준비하기

회원가입 : Sign Up

github.com 접속 후
Sign Up 클릭



가입 / 이메일 인증

가입 / 인증

Welcome to GitHub!
Let's begin the adventure

Enter your email
✓ edukyle.hphk@gmail.com **1. Email 작성**

Create a password
✓ **2. 비밀번호 작성**

Enter a username
✓ edukyle **3. 닉네임 작성 (신중!!)**

Would you like to receive product updates and announcements via email?
Type "y" for yes or "n" for no
✓ n **4. 메일 수신 여부**

Verify your account

확인
이 퍼즐을 풀어서 귀하가 인간이라는 것을 알 수 있게 해주세요 **5. 로봇 여부 확인**

확인

You're almost done!
We sent a launch code to edukyle.hphk@gmail.com

→ Enter code

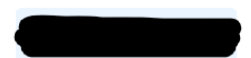
Did not get your email? Resend the code or update your email address.



Here's your GitHub launch code, @edukyle!



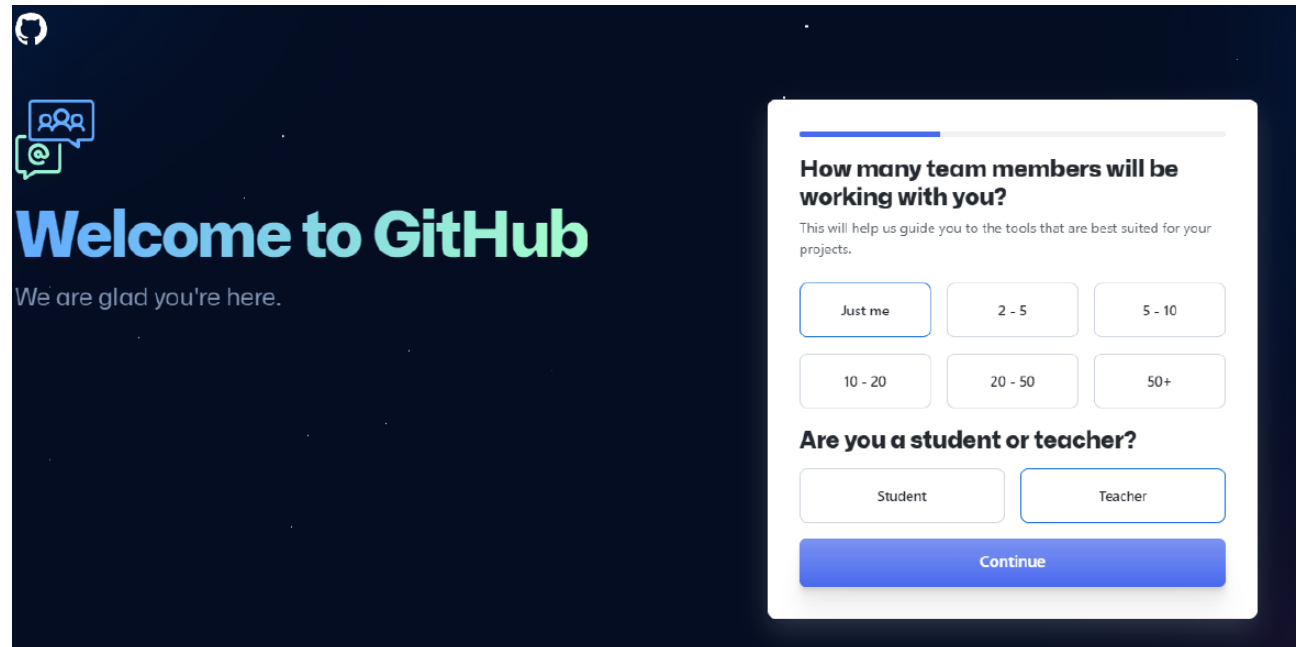
Continue signing up for GitHub by entering the code below:



Open GitHub

설문조사

설문조사
대충하고, 그냥 넘기자.



The image shows the GitHub welcome screen with a dark blue background. On the left, the GitHub logo is at the top, followed by two small icons (a group of people and a speech bubble with an @ symbol). Below these, the text "Welcome to GitHub" is displayed in a large, bold, light blue font, with "We are glad you're here." in a smaller, white font underneath. On the right side, a white survey overlay is present. It has a title "How many team members will be working with you?" and a subtitle "This will help us guide you to the tools that are best suited for your projects." Below the subtitle are six buttons arranged in two rows: "Just me", "2 - 5", "5 - 10" in the first row, and "10 - 20", "20 - 50", "50+" in the second row. Below these buttons is another section titled "Are you a student or teacher?" with two buttons: "Student" and "Teacher". At the bottom of the survey overlay is a large blue button labeled "Continue".

Welcome to GitHub
We are glad you're here.

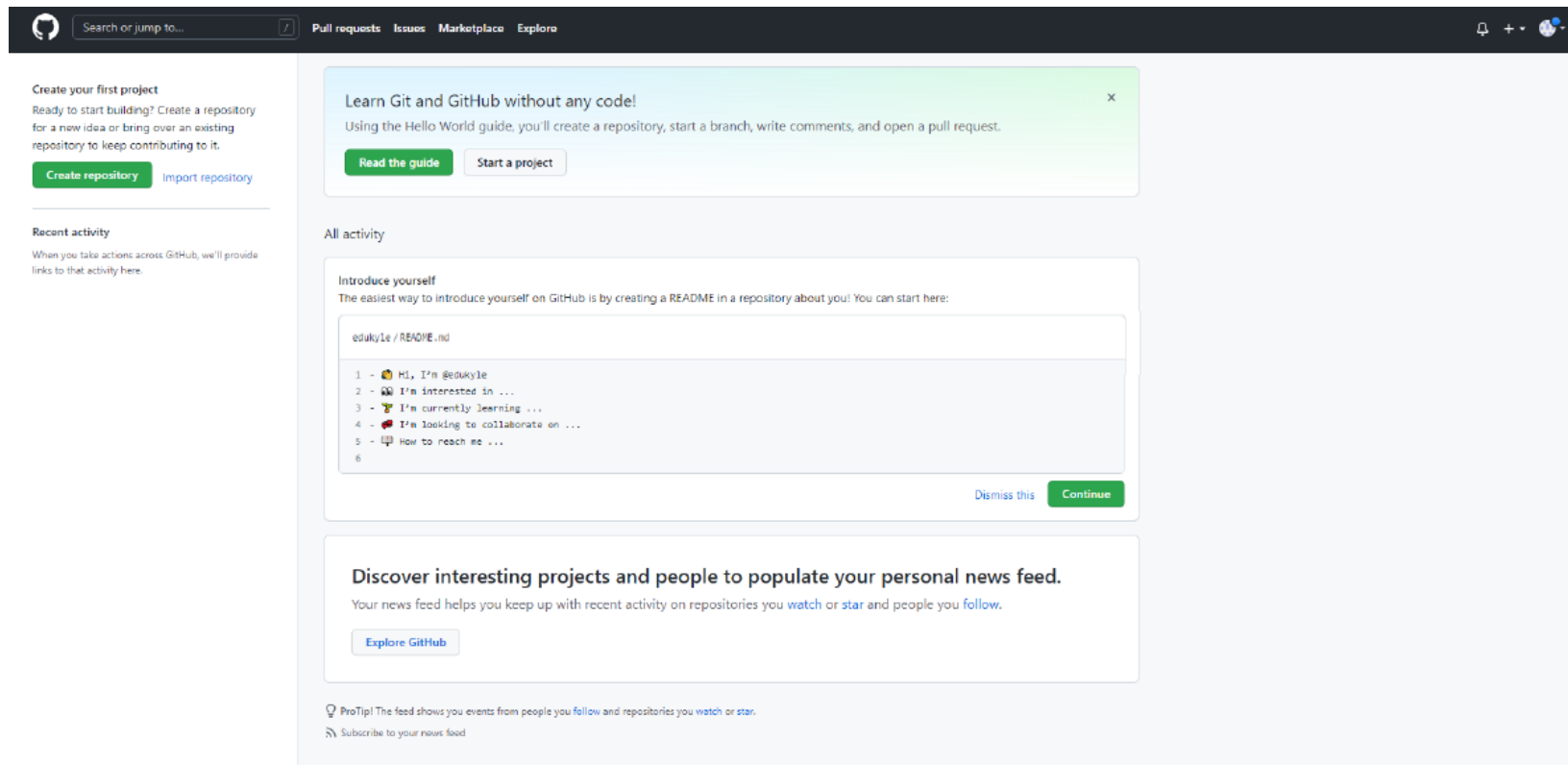
How many team members will be working with you?
This will help us guide you to the tools that are best suited for your projects.

Just me 2 - 5 5 - 10
10 - 20 20 - 50 50+

Are you a student or teacher?
Student Teacher

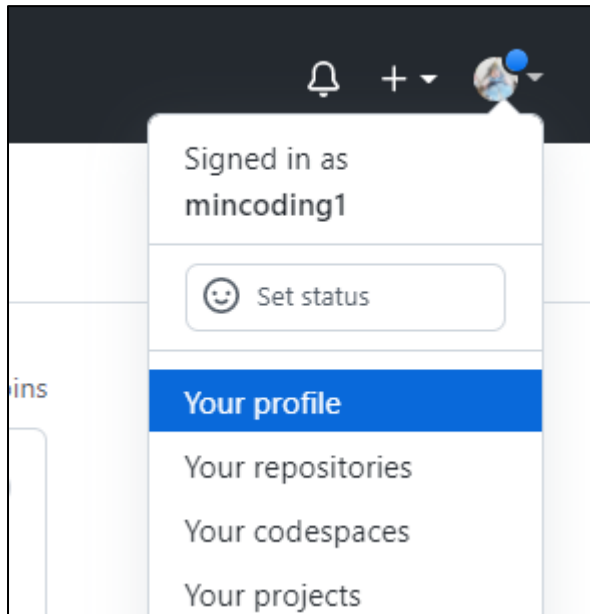
Continue

회원가입 완료 / 로그인 후 화면



Your Profile 수정

1. 회원정보 수정
2. 사진 변경해두기



GitHub Desktop 다운로드

Git을 편리하게 쓰기 위한 프로그램

Git 편하게 쓰는 GUI 프로그램

<https://desktop.github.com/>

다운로드 후 설치



GitHub Desktop 프로그램 이란?

Git + GUI 프로그램

편리하게

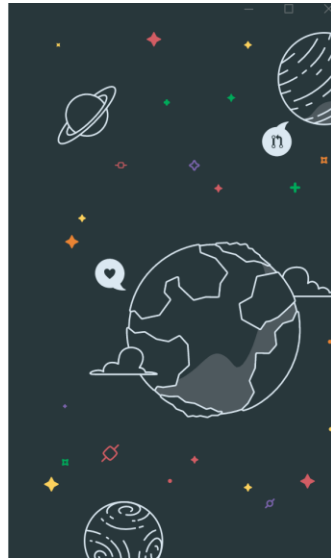
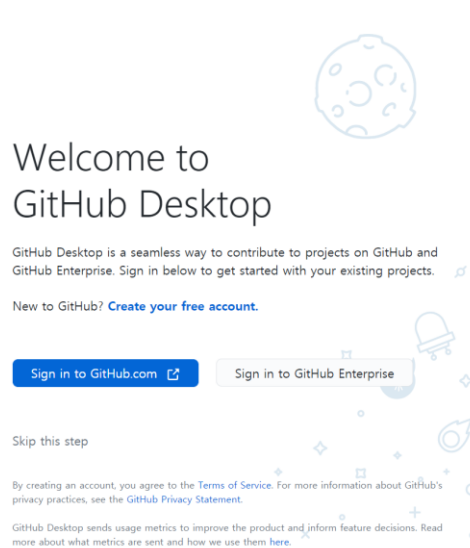
Local Repository 에 Commit 가능


편리하게


Remote Repository에 Push 가능

GitHub 로그인

GitHub Desktop 실행 후 로그인







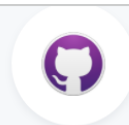


Sign in to GitHub
to continue to GitHub Desktop


Username or email address


Password [Forgot password?](#)


Sign in




Authorize GitHub Desktop

 **GitHub Desktop by desktop**
wants to access your **sfminho** account


 **Repositories**
Public and **private**


 **Personal user data**
Full access


 **Workflow**
Update GitHub Action Workflow files.

Cancel **Authorize desktop**

Authorizing will redirect to
x-github-desktop-auth://oauth

 Owned & operated
by GitHub

 Created 6 years ago

 More than 1K
GitHub users

Finish 클릭

Configure Git

This is used to identify the commits you create. Anyone will be able to see this information if you publish commits.

- ☒ Use my GitHub account name and email address
☐ Configure manually

Name

sfminho

Email


97452043+sfminho@users.noreply.github.com

Finish

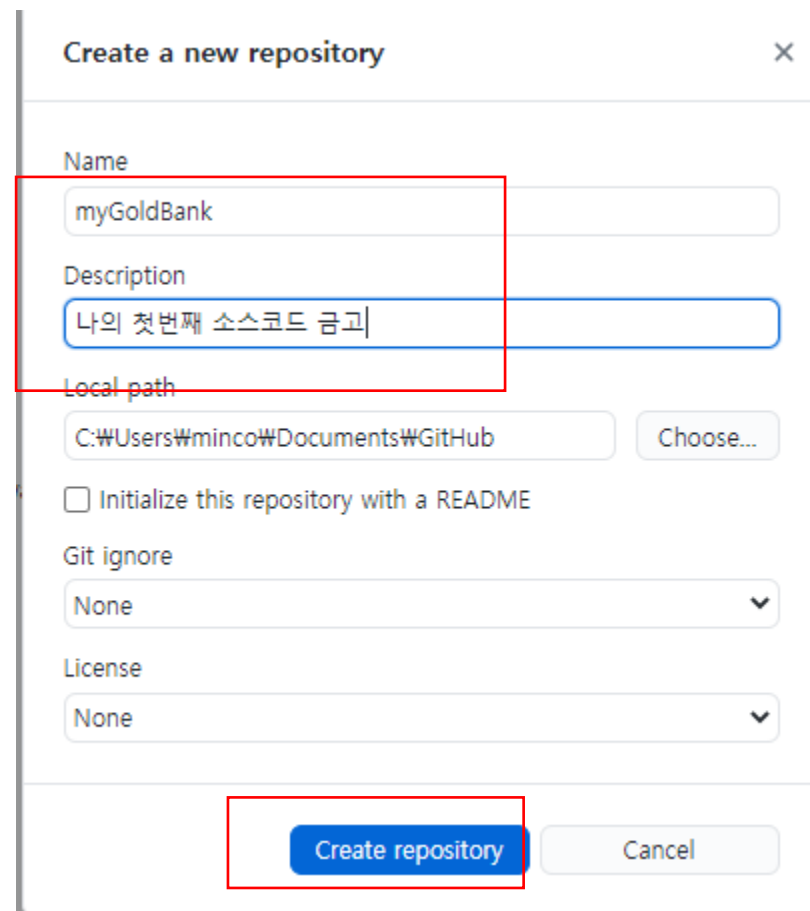
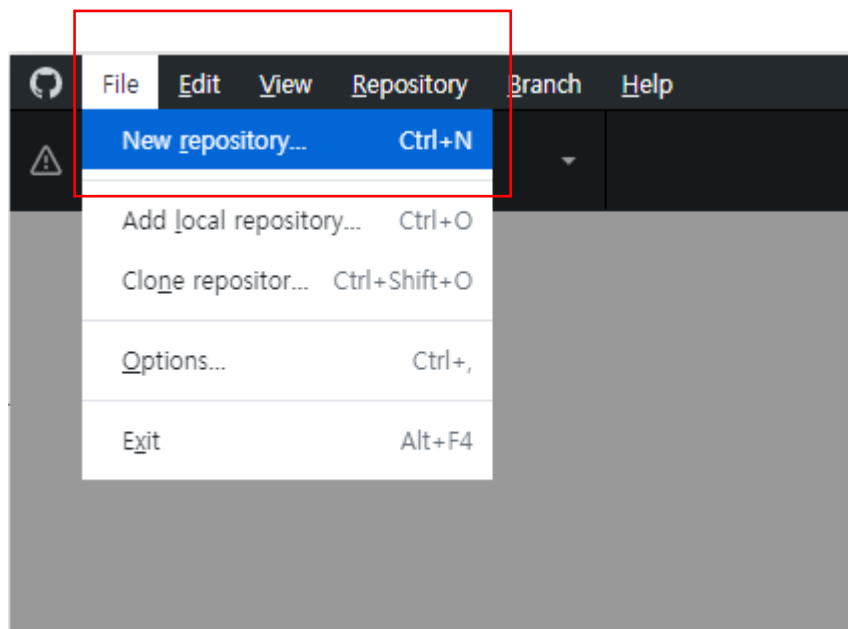
Cancel

Example commit

Fix all the things

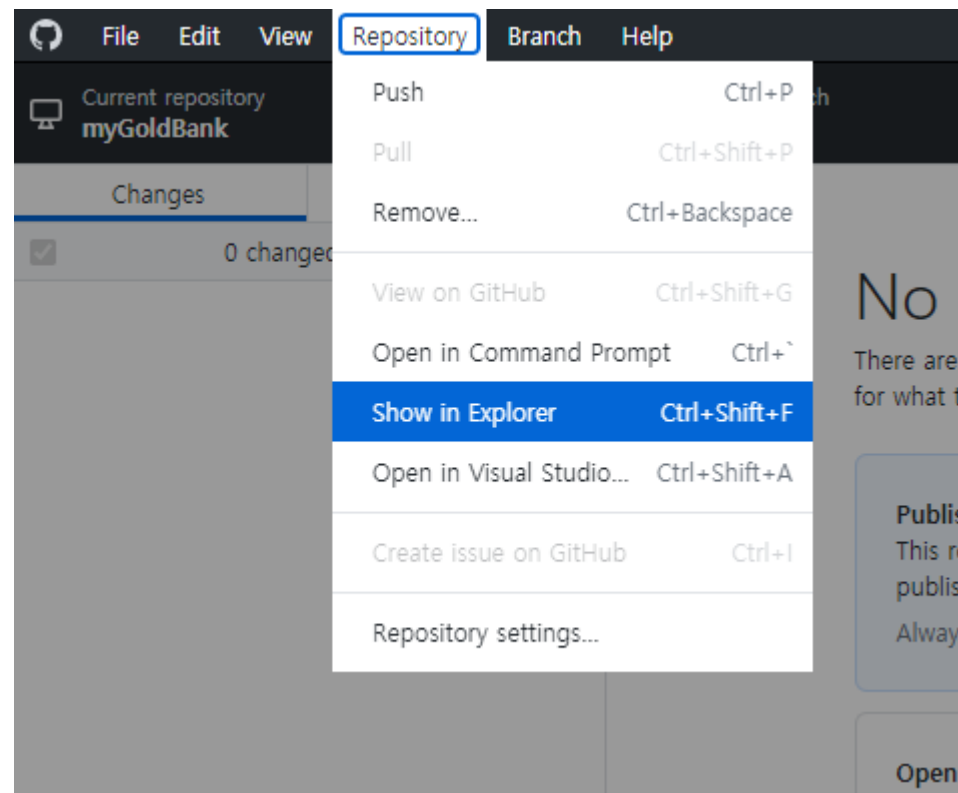
 sfminho • 30m

Local Repository 만들기



폴더 열기

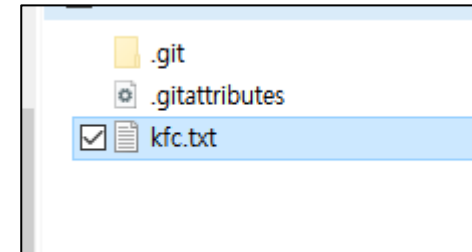
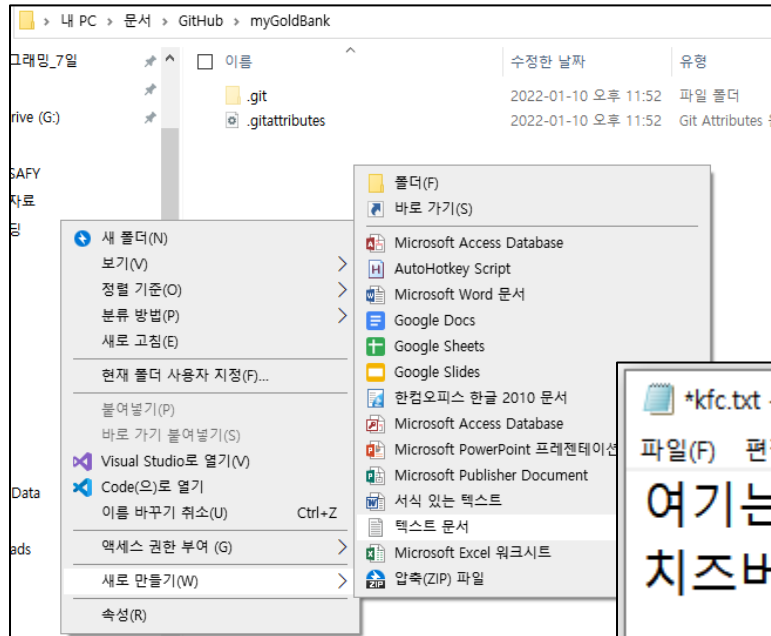
내 레파지토리 폴더 열기



이곳에 파일을 만들어보자

텍스트 파일 하나 만들기

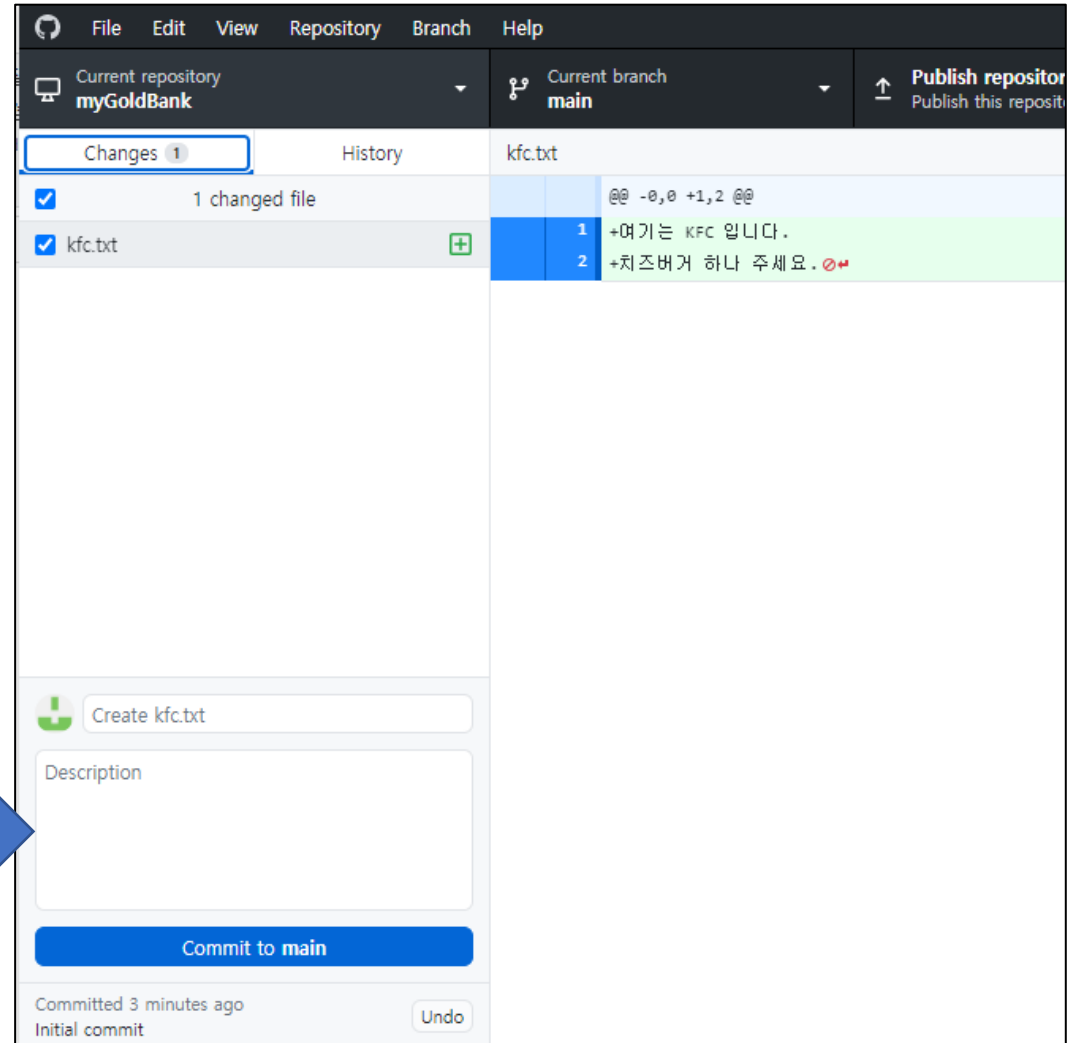
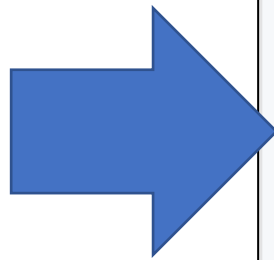
kfc.txt



Github Desktop 실행

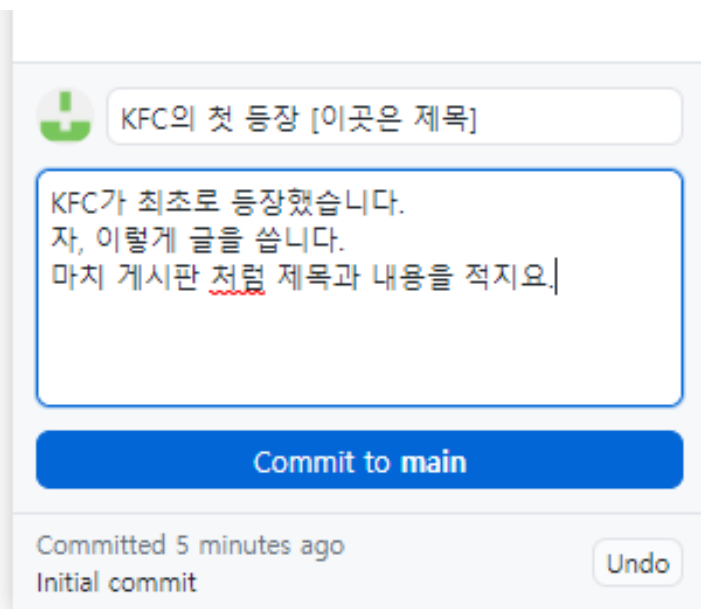
자동으로 kfc.txt가 보인다.

이제 commit을 해보자.



Commit 해보기

제목과 내용을 적고,
Commit 하기



The screenshot shows a commit interface with a green download icon on the left. The title field contains 'KFC의 첫 등장 [이곳은 제목]'. The message field contains the text: 'KFC가 최초로 등장했습니다. 자, 이렇게 글을 씁니다. 마치 게시판 처럼 제목과 내용을 적지요.' Below the message field is a blue button labeled 'Commit to main'. At the bottom, it says 'Committed 5 minutes ago' and 'Initial commit', with an 'Undo' button on the right.

KFC의 첫 등장 [이곳은 제목]

KFC가 최초로 등장했습니다.
자, 이렇게 글을 씁니다.
마치 게시판 처럼 제목과 내용을 적지요.

Commit to main

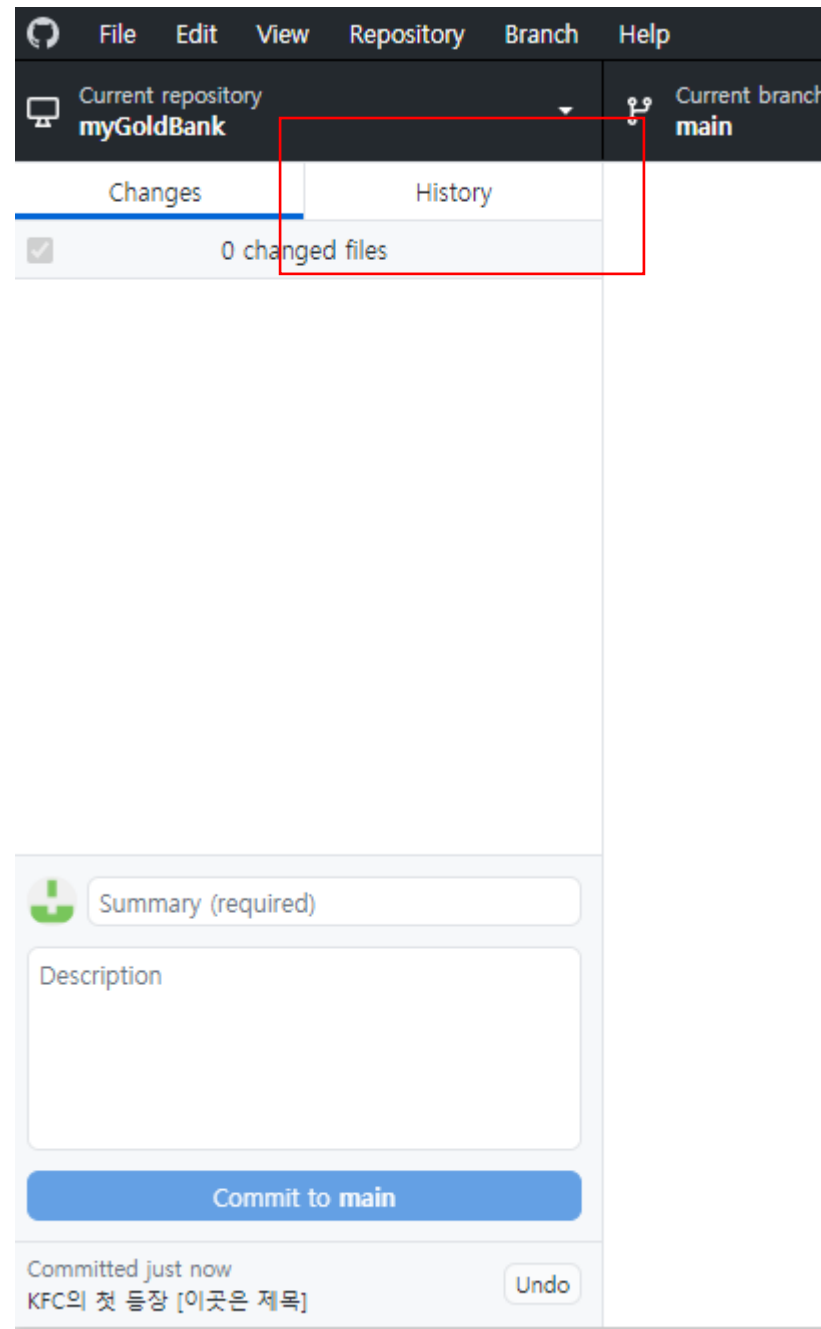
Committed 5 minutes ago
Initial commit

Undo

History 클릭

Commit 완료

History 클릭해보기



[도전] Commit 추가

1. kfc.txt 내용에 "콜라도 주세요." 라는 내용을 추가 후, Commit 하기
2. kfc.txt 내용에 "치즈버거" 단어 대신, "상하이 버거 " 단어로 수정 후 Commit 하기
3. **bts.txt 파일 하나 추가하기**
내용 : 안녕하세요. BTS 입니다.

+ 와 - 표시

-는 제거된 문장
+는 추가된 문장

자동인식 됨

The screenshot shows a code editor interface with a dark theme. The top menu bar includes File, Edit, View, Repository, Branch, and Help. Below the menu, the current repository is 'myGoldBank' and the current branch is 'main'. A 'Publish repository' button is visible. The 'History' tab is selected, showing a commit titled '콜라도 달라는 요청' by 'inho.choi' just now. The diff view shows changes to 'kfc.txt'. The diff content is as follows:

		@@ -1,2 +1,3 @@	
1	1	여기는 KFC 입니다.	
2		-치즈버거 하나 주세요. ✖	
	2	+치즈버거 하나 주세요.	
	3	+콜라도 주세요. ✖	

그동안 했던 작업들

Local Repository 에만, 정보를 저장했다.

- 그동안 작업 내역도 쉽게 볼 수 있음

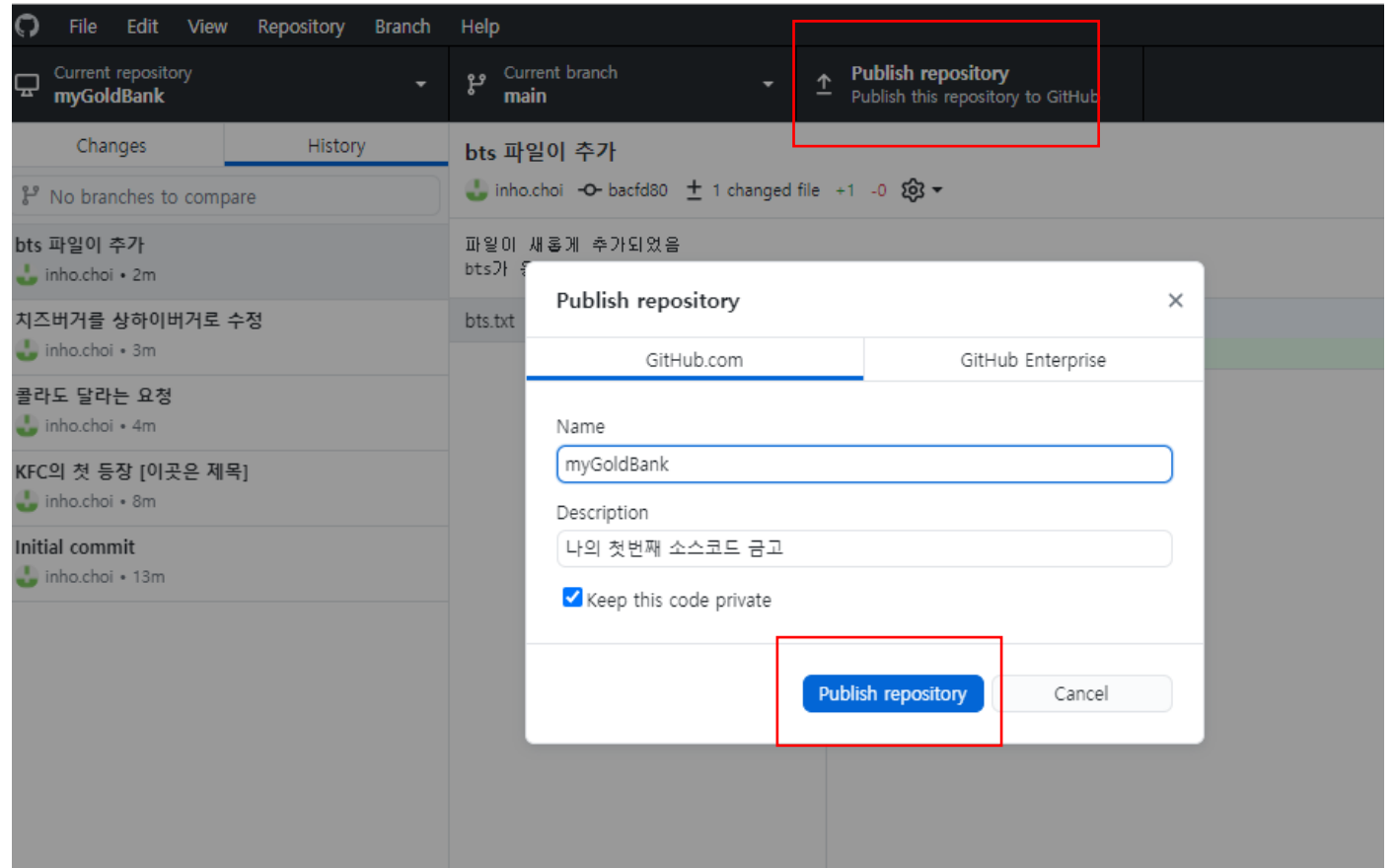
이제, Local이 아닌

GitHub 이라는 Remote Repository에 Push 해보자.

Publish 하기

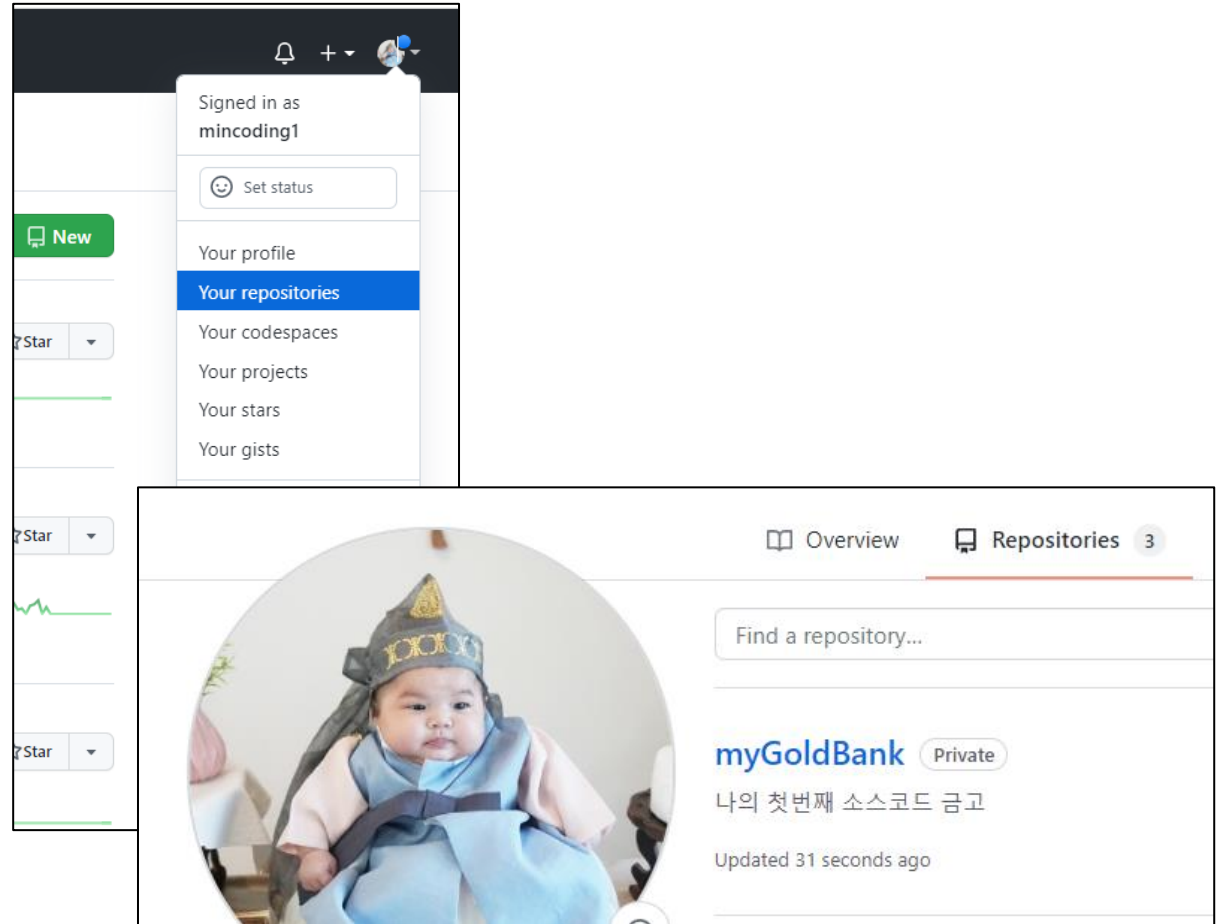
맨 처음은
Publish 하여

Remote Repository에
새 공간을 만든다.



결과 확인하기

1분 기다리고
github.com 사이트의
repository를 확인해보자.



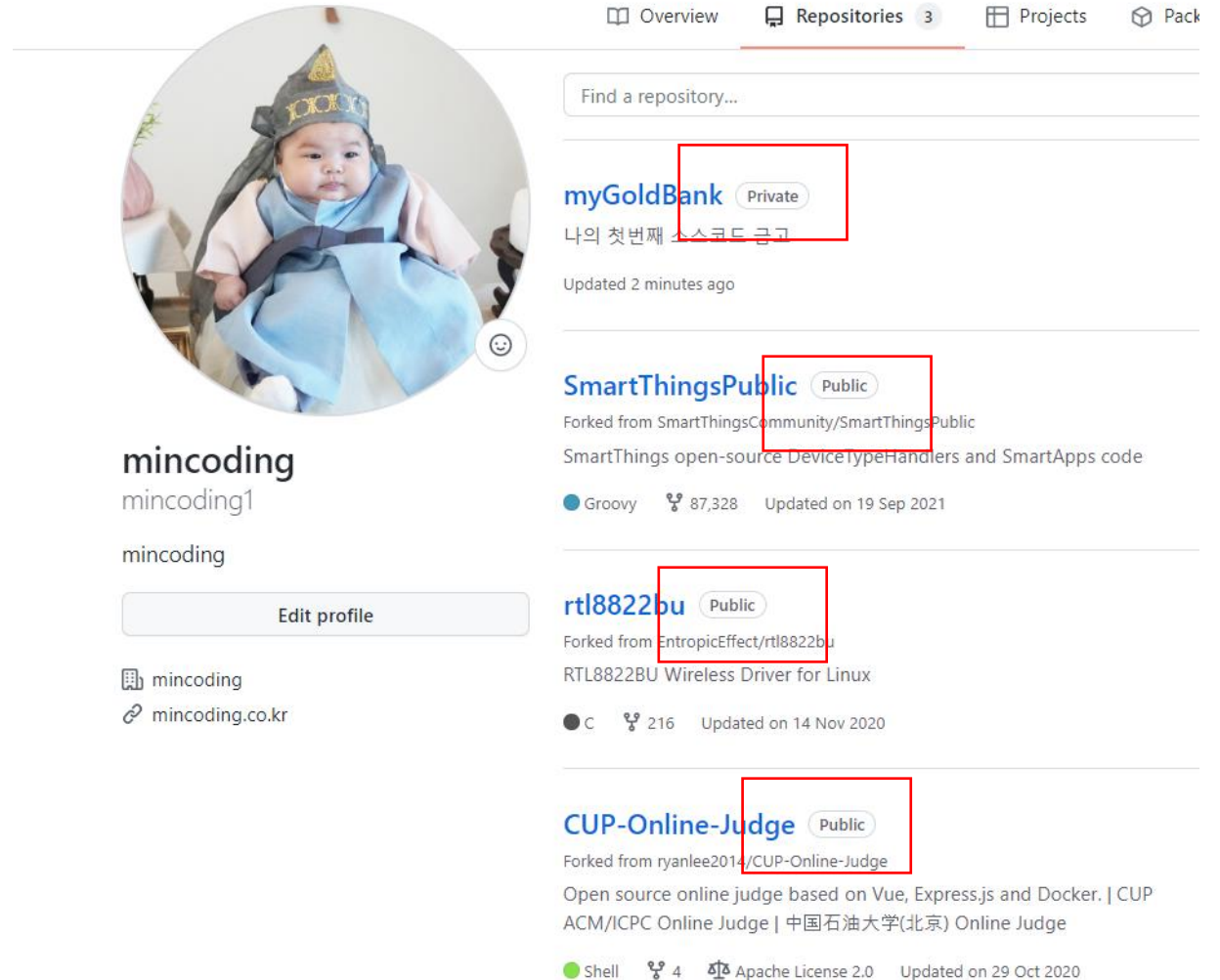
Private 과 Public

Private

- 내가 권한을 준 사람만 볼 수 있는 Repository

Public

- 누구나 볼 수 있는 Repository



The screenshot displays the GitHub profile of a user named 'mincoding'. The profile picture shows a baby wearing a traditional Korean hat. The user's bio and website are visible. The 'Repositories' tab is selected, showing a list of repositories. Three repositories are highlighted with red boxes to illustrate visibility settings:

- myGoldBank**: Labeled as **Private**. Description: 나의 첫번째 스코드 글.
- SmartThingsPublic**: Labeled as **Public**. Description: Forked from SmartThingsCommunity/SmartThingsPublic. SmartThings open-source DeviceTypeHandlers and SmartApps code.
- rtl8822bu**: Labeled as **Public**. Description: Forked from EntropicEffect/rtl8822bu. RTL8822BU Wireless Driver for Linux.

At the bottom, another repository **CUP-Online-Judge** is shown, also labeled as **Public**. The interface includes navigation tabs for Overview, Repositories (3), Projects, and Packages.

[참고] Github 유료 ? 무료 ?

예전

- 무료 : Public
- 유료 : Private

현재 (Microsoft사에서 Github을 인수한 이후)

- 무료 : Private, Public 모두 무료, 500MB 까지
- 유료 : 2GB 까지 사용 가능, 여러가지 추가 기능

[암기] Clone 기능

복제!!

[암기]

Remote Repository 통째로
나의 Local Repository 로 복제한다.

Clone을 언제 사용할까?

옛날에 내가 만들었던 소스코드 가져오기

내가 작업했던 Remote Repository 내용들을,
내 Local Repository로 가져올때

남이 만든, 훌륭한 소스코드를 가져오기

타인이 만든 소스코드를 가져와서,
내가 마음대로 수정해서 개발해본다.

Open Source 란?

-> 공개된 소스코드

유명한 소스코드인 "리눅스 커널" 을
내 컴퓨터 안, Local Repository 로 Clone 해보자.

리눅스 커널, Remote Repository

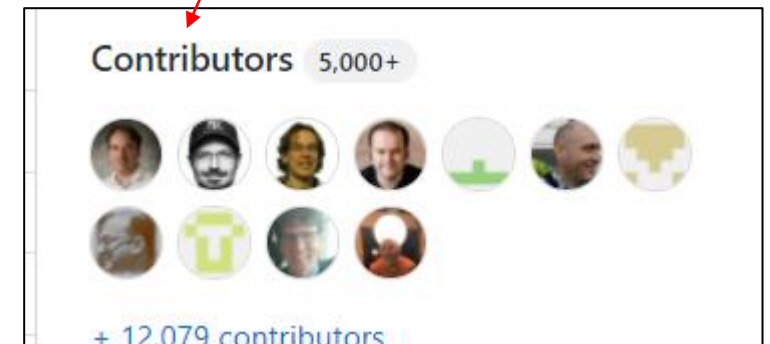
<https://github.com/torvalds/linux>

[참고]

리눅스 커널

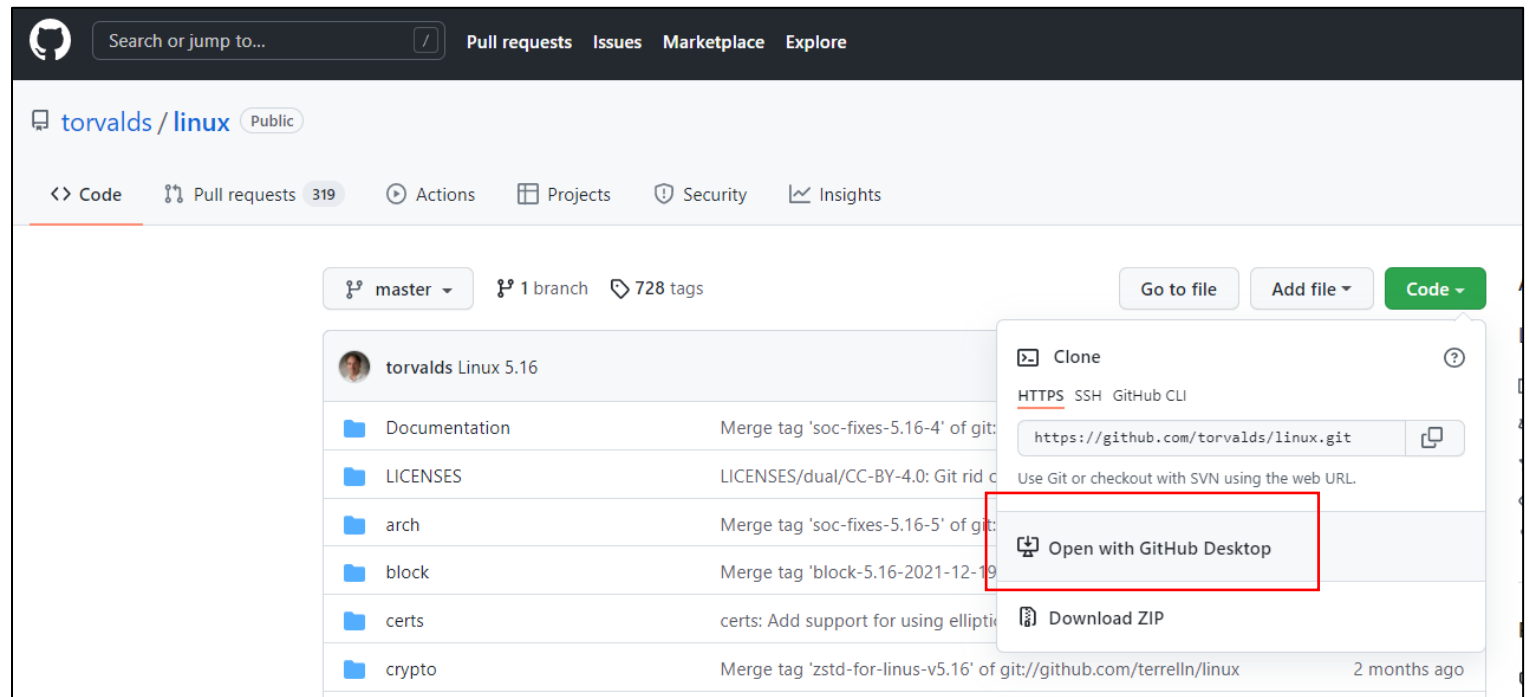
- 리눅스라는 운영체제의 핵심 소스코드를 리눅스 커널이라고 한다.
- 리누즈 토발즈 라는 사람이 처음 제작하였고, 지금은 5,000명 이상의 사람들이 함께 개발한다.

[암기할 용어] 컨트리뷰터
→ 오픈소스 기여자



Clone 하기

Code > **Open with GitHub Desktop** 클릭

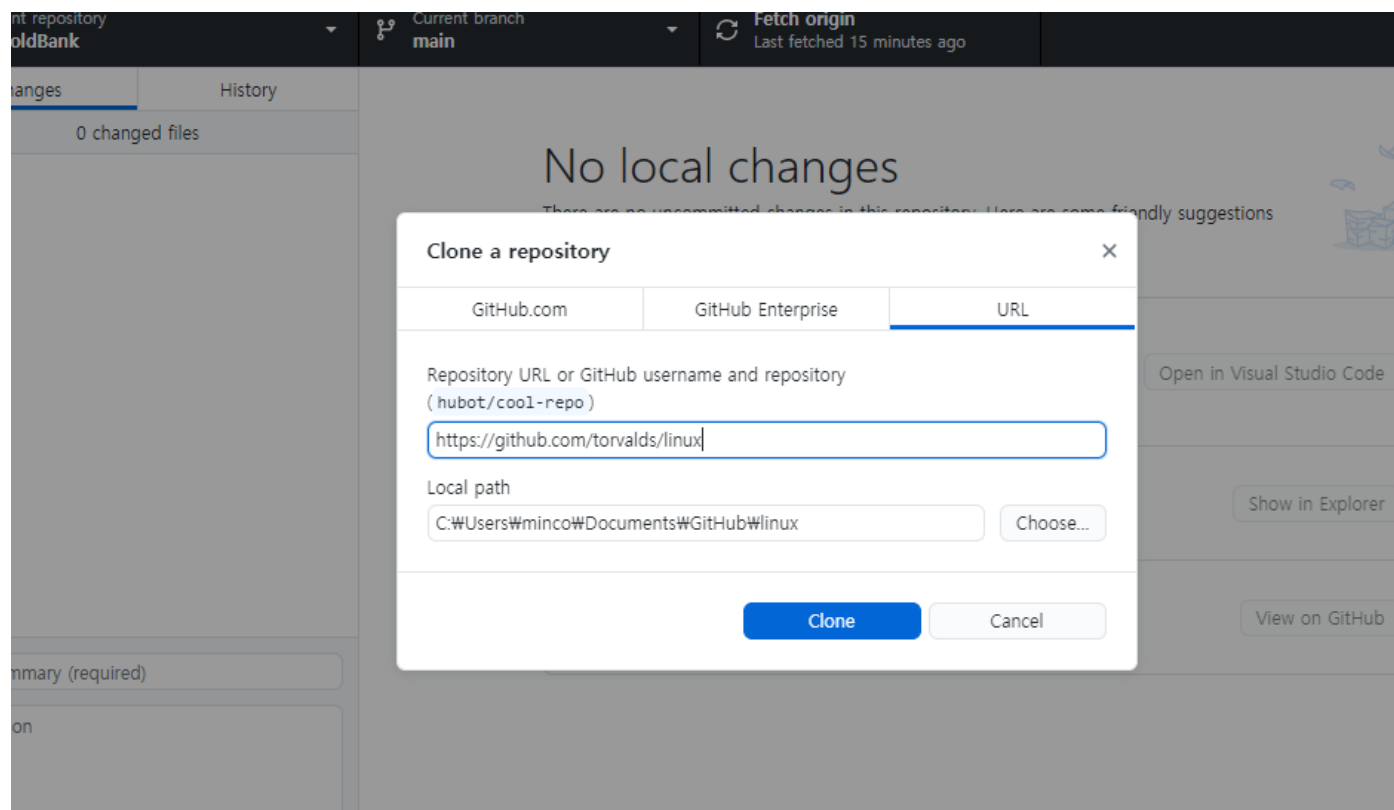


Clone a repository

Clone 버튼을

누르지말자.

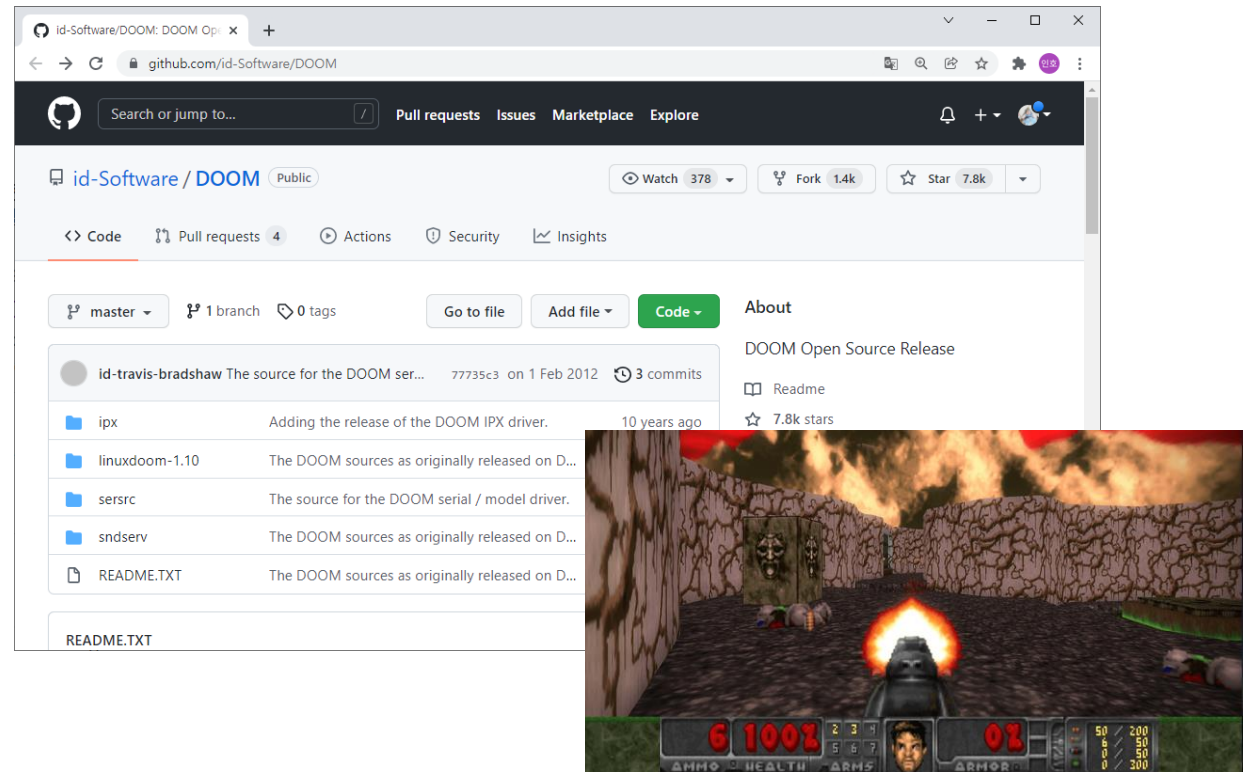
너무 오래걸린다.



[도전] DOOM Clone 하기

<https://github.com/id-Software/DOOM>

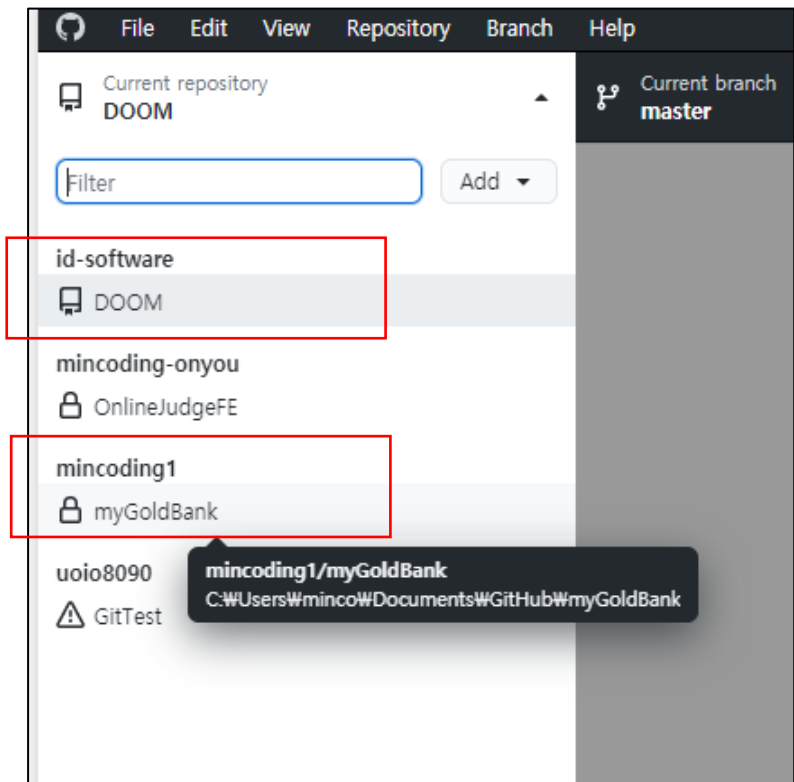
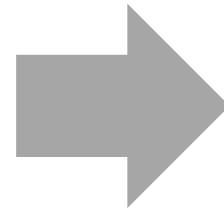
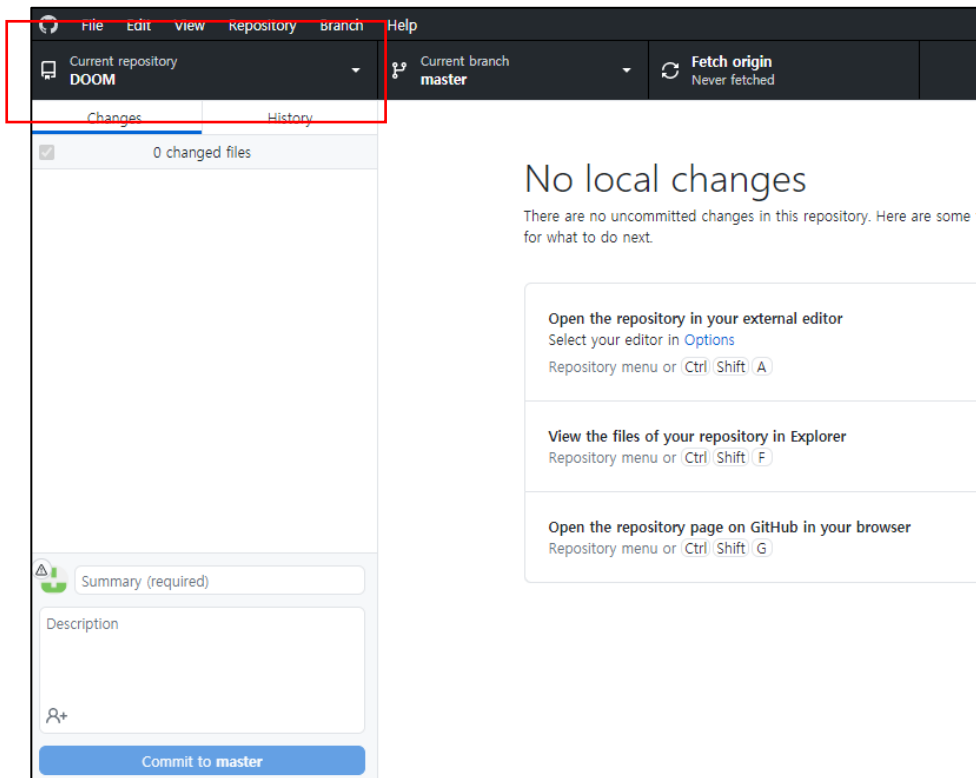
나의 Local Repository로
Open Source를 Clone 한다.



DOOM 완료

DOOM : Clone 한 Local Repo. (앞으로 줄여서 Repo. "레포"라고 하겠음)

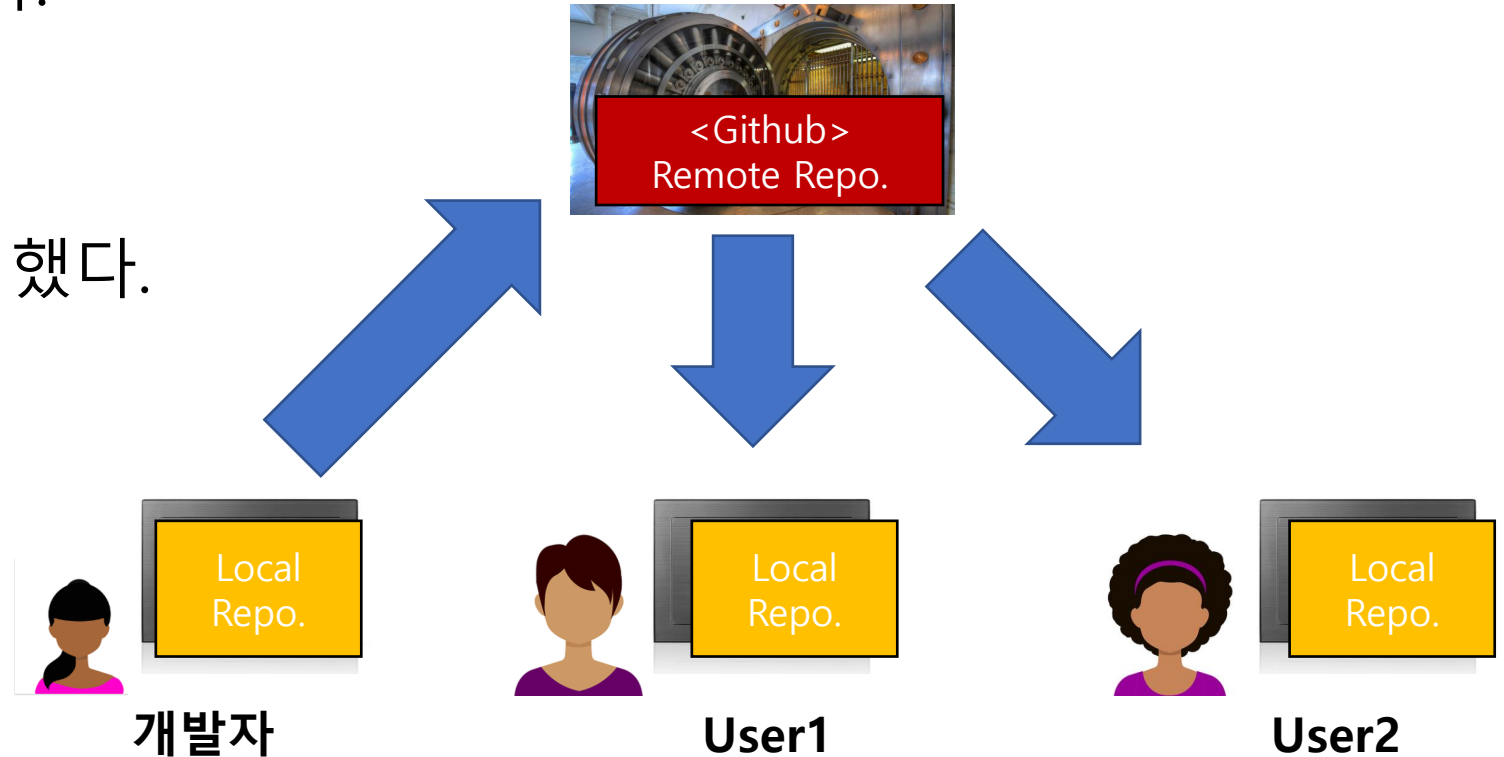
myGoldBank : 아까 실습한 레포.



코드를 가져오는 방법 - Clone

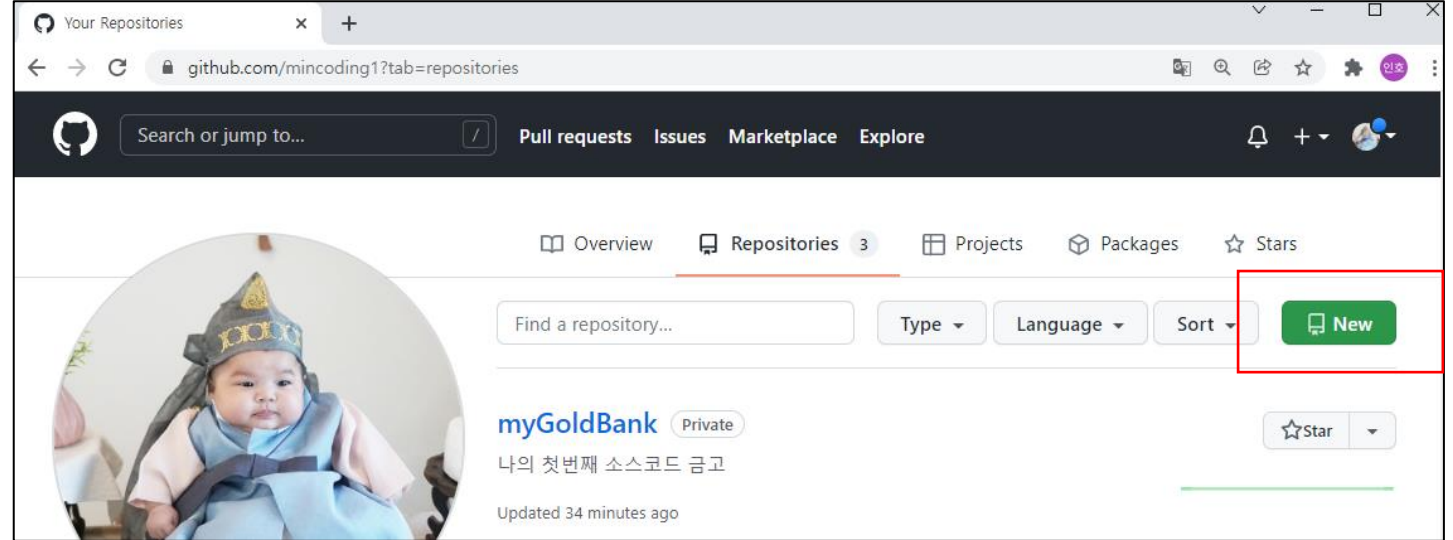
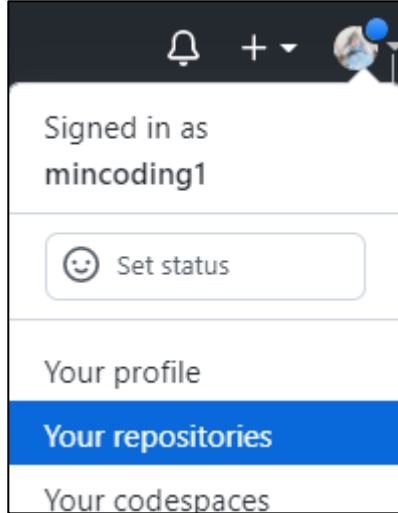
개발자가 처음으로 Publish (개설) 하고,
소스코드를 Push를 하곤한다.

다른 User들이
자신의 Repository로 Clone 했다.



[강사만 진행] 눈으로만 보세요.

신규 Remote Repo. 생성함
• 이번엔 Github에서 생성함.



[강사만 진행] 눈으로만 보세요.


Public 으로 생성

- Repo 이름 : publicStudy119
- README 파일 생성
 - Repo에 대한 소개글

Create a new repository

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

Owner *

 mincoding1 ▾

Repository name *

publicStudy119 ✓

Great repository names are short and memorable. Need inspiration? How about [jubilant-garbanzo?](#)

Description (optional)

Clone과 Pull 학습

☒  Public

Anyone on the internet can see this repository. You choose who can commit.

☐  Private

You choose who can see and commit to this repository.

Initialize this repository with:

[Skip this step if you're importing an existing repository.](#)

☒ Add a README file


This is where you can write a long description for your project. [Learn more.](#)

☐ Add .gitignore

Choose which files not to track from a list of templates. [Learn more.](#)

☐ Choose a license

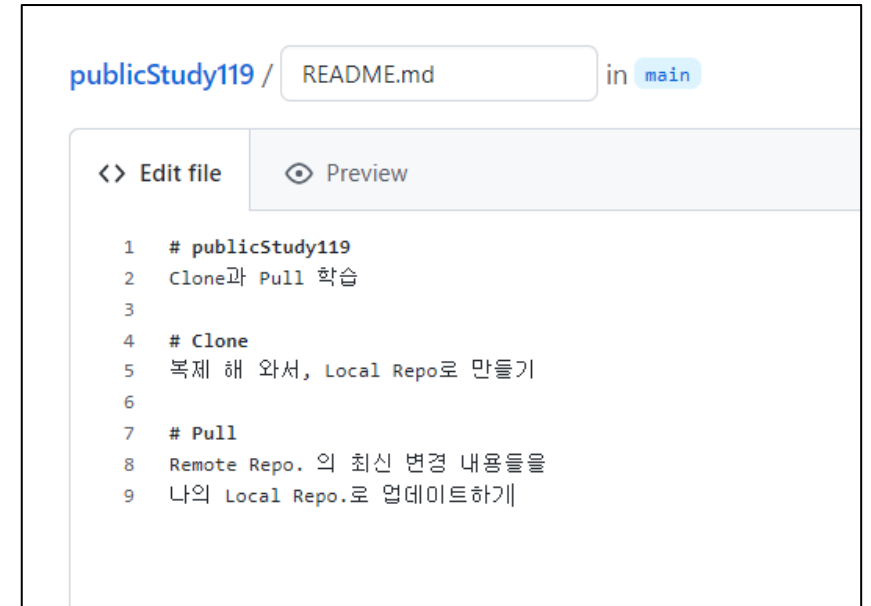
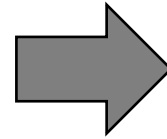
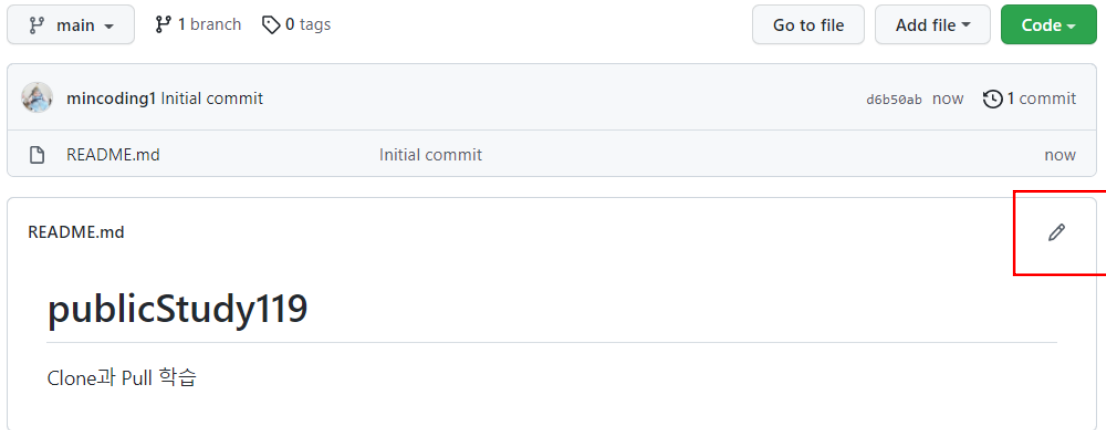
A license tells others what they can and can't do with your code. [Learn more.](#)

This will set  main as the default branch. Change the default name in your [settings](#).

Create repository

[강사만 진행] 눈으로만 보세요.

README.md 파일이 기본 생성되어있음

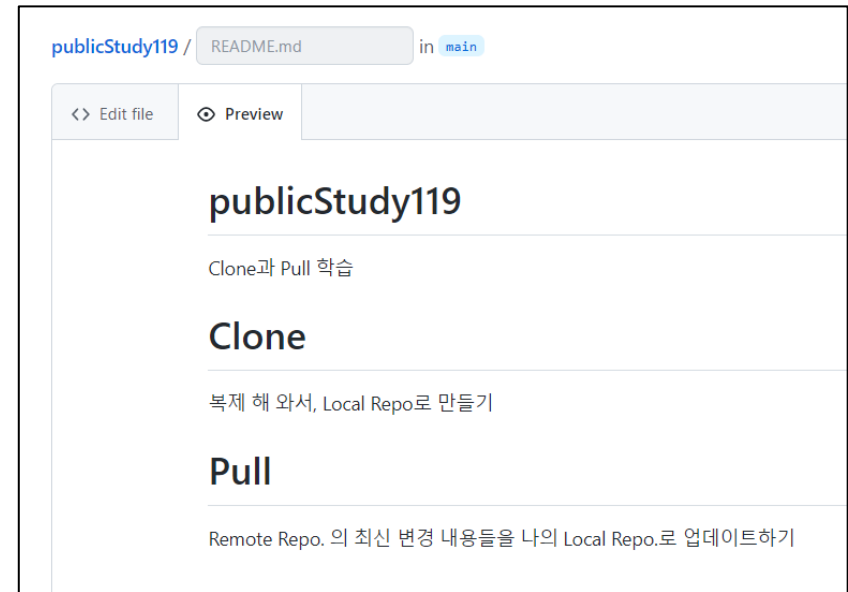
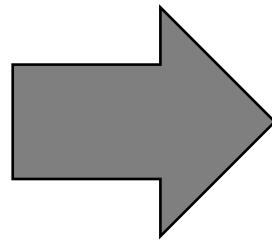
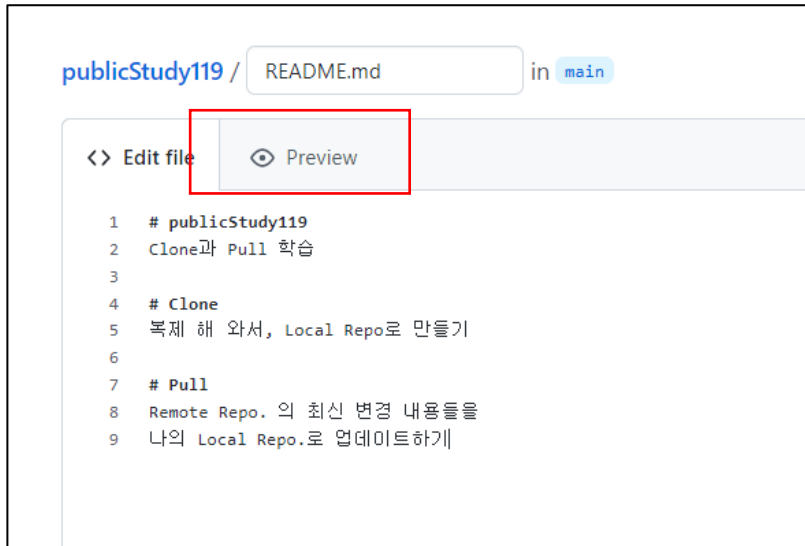


타이핑하기
는 주제서식으로 표현 됨

[강사만 진행] 눈으로만 보세요.

[참고] 마크다운

- 빠르게, 문서를 이쁘게 만들 수 있음
- 확장자 : **md 파일**
- 곧, 수업 예정



[강사만 진행] 눈으로만 보세요.

GitHub 에서의 Commit은,
원격지(Remote) Repo. 에 바로 등록됨



publicstudy112


Clone과 Pull 학습

Clone

복제 해 와서, Local Repo로 만들기

Pull

Remote Repo. 의 최신 변경 내용들을 나의 Local Repo.로 업데이트하기

 **Commit changes**

README 문서를 하나 적어보았습니다.

을 써서, 주제를 표현해봤습니다.
이것을 마크다운 문서라고 합니다.

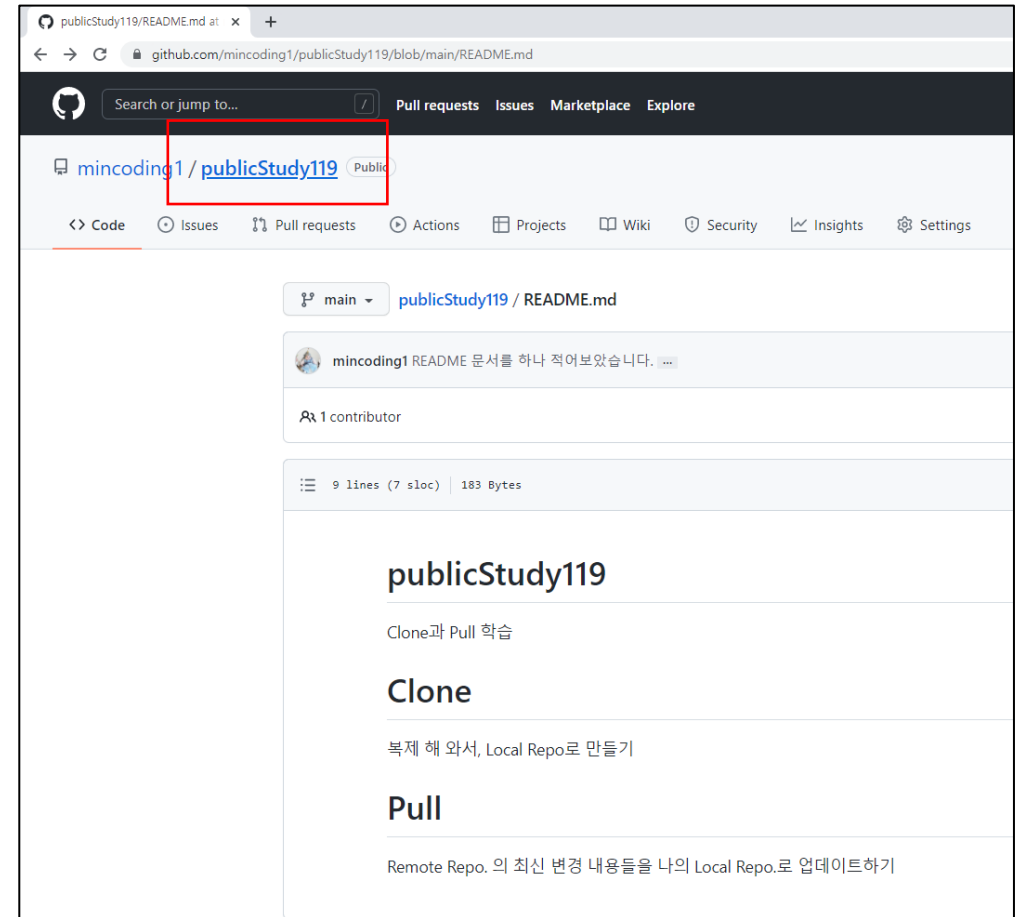
☒ Commit directly to the `main` branch.

☐ Create a new branch for this commit and start a pull request. [Learn more about pull requests.](#)

Commit changes Cancel

[강사만 진행] 눈으로만 보세요.

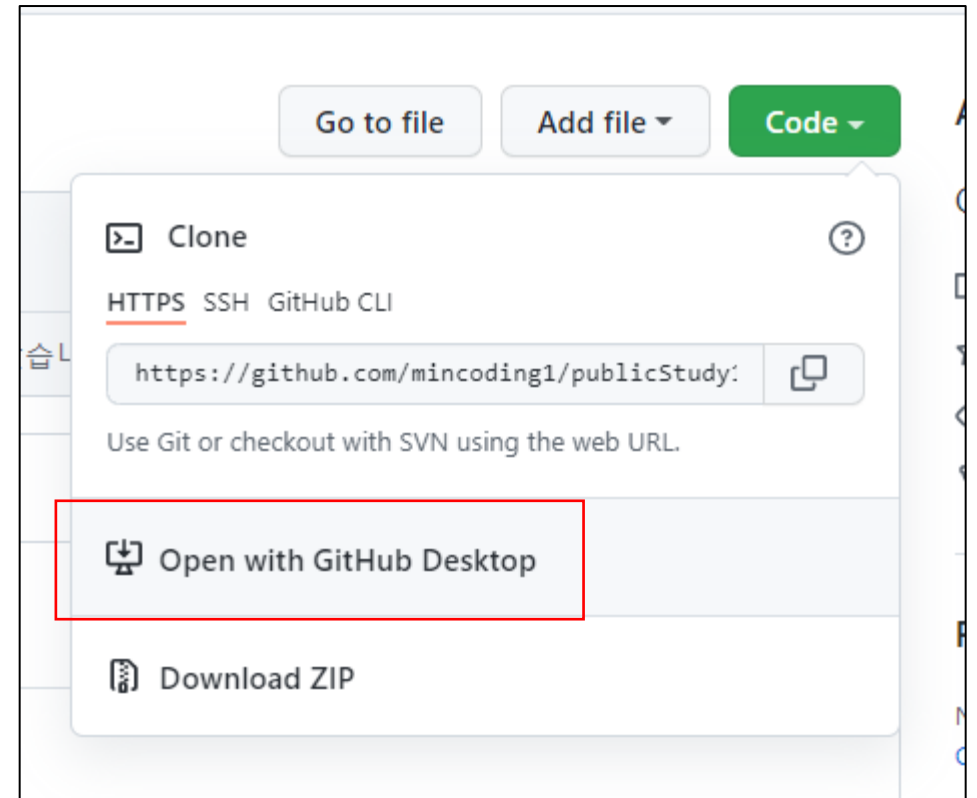
Remote Repo. 이름 눌러서
홈으로 돌아간다.



[강사만 진행] 눈으로만 보세요.

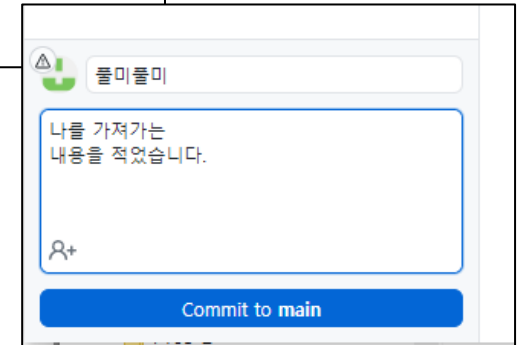
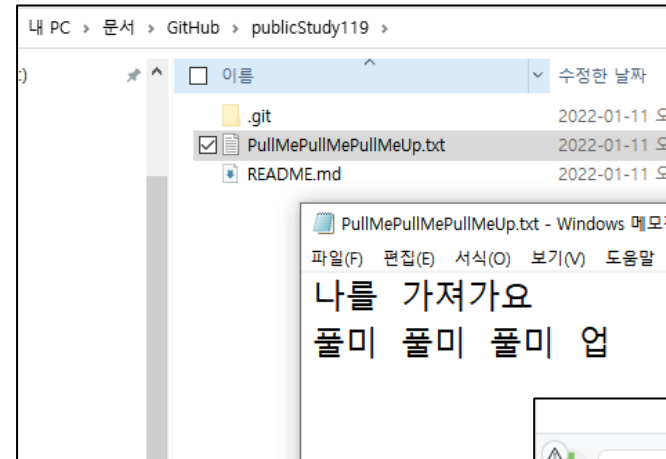
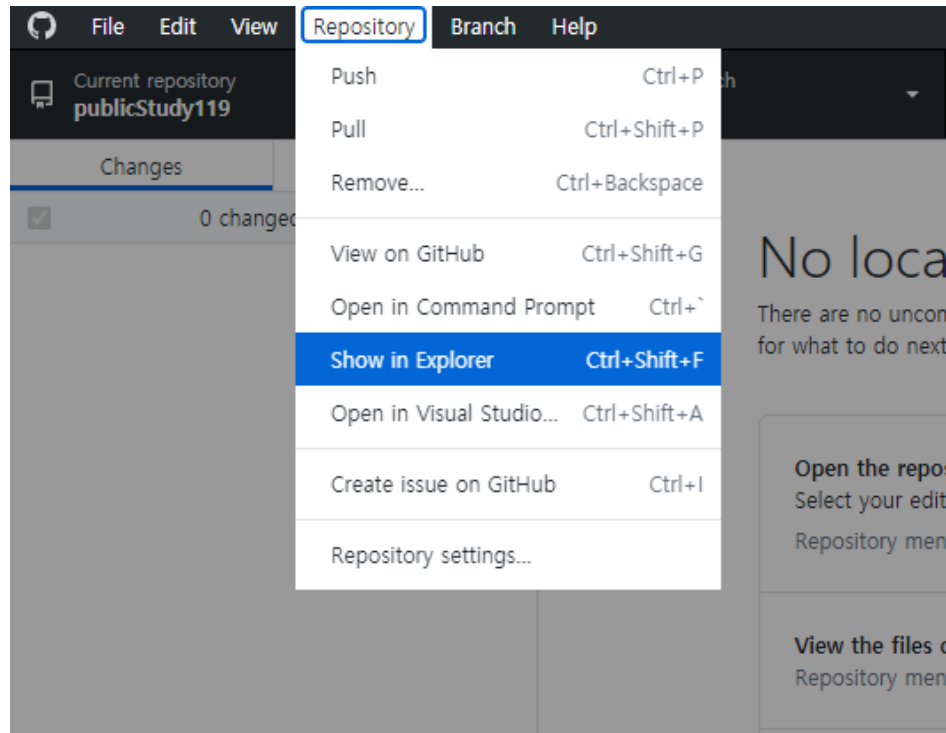
Clone 하기

1. 내 Local Repo. 를 하나 생성하면서,
2. 모든 파일을 전체 복제한다.



[강사만 진행] 눈으로만 보세요.

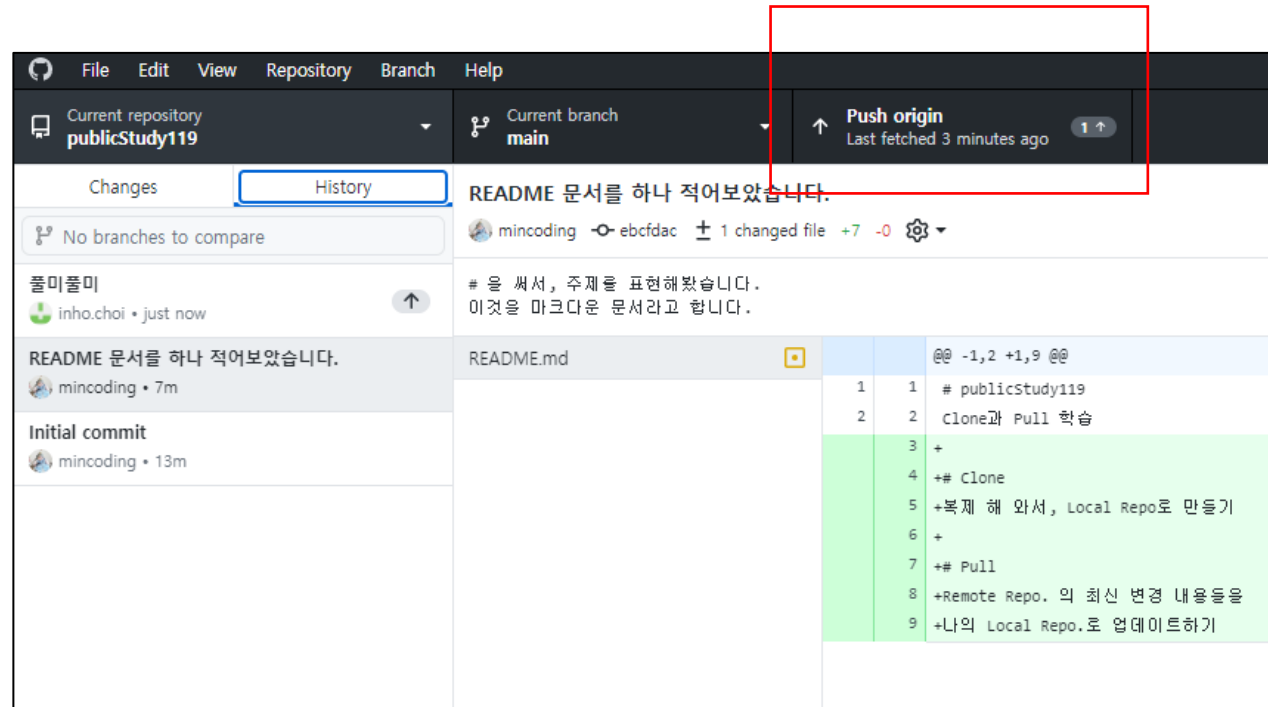
text 파일을 하나 더 생성하여, Commit 한다.



[강사만 진행] 눈으로만 보세요.

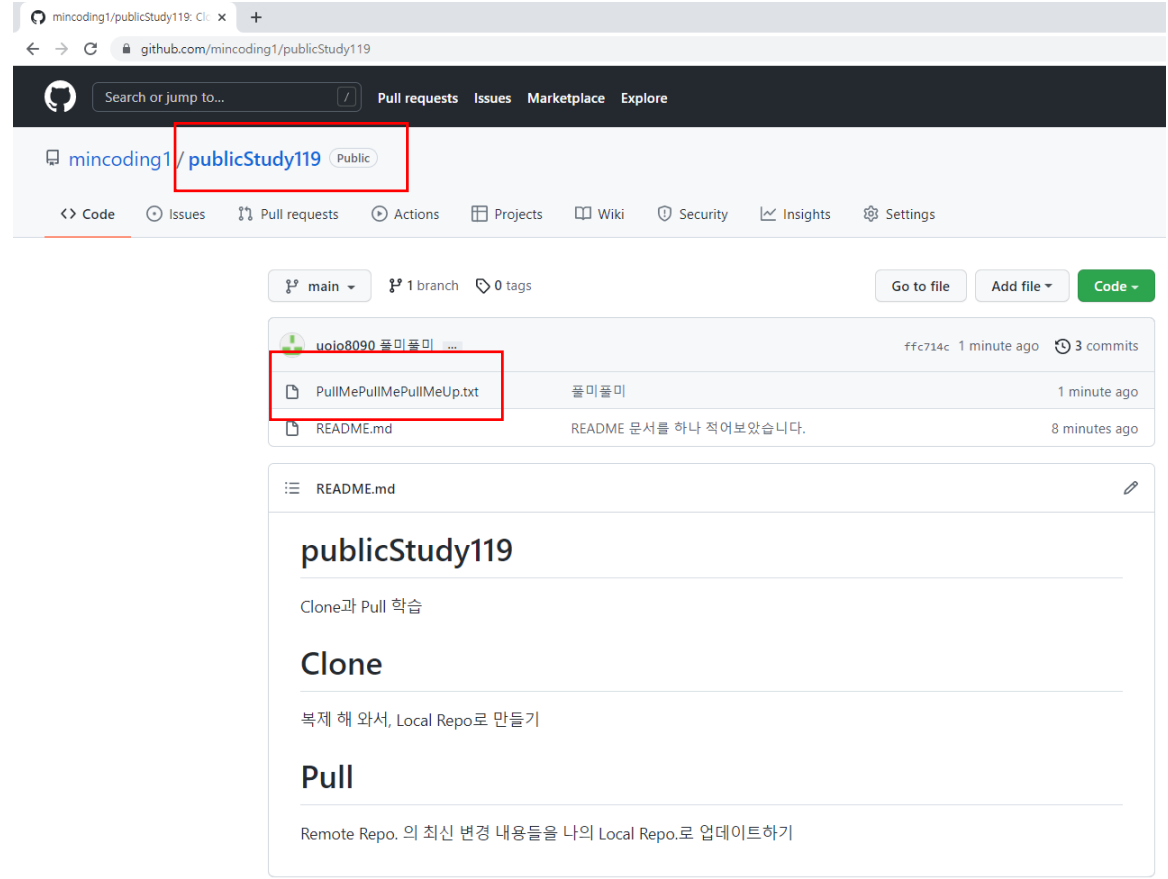
Push 하기!

- Commit 된 내용을 Remote Repo.로 Push한다.



[강사만 진행] 눈으로만 보세요.

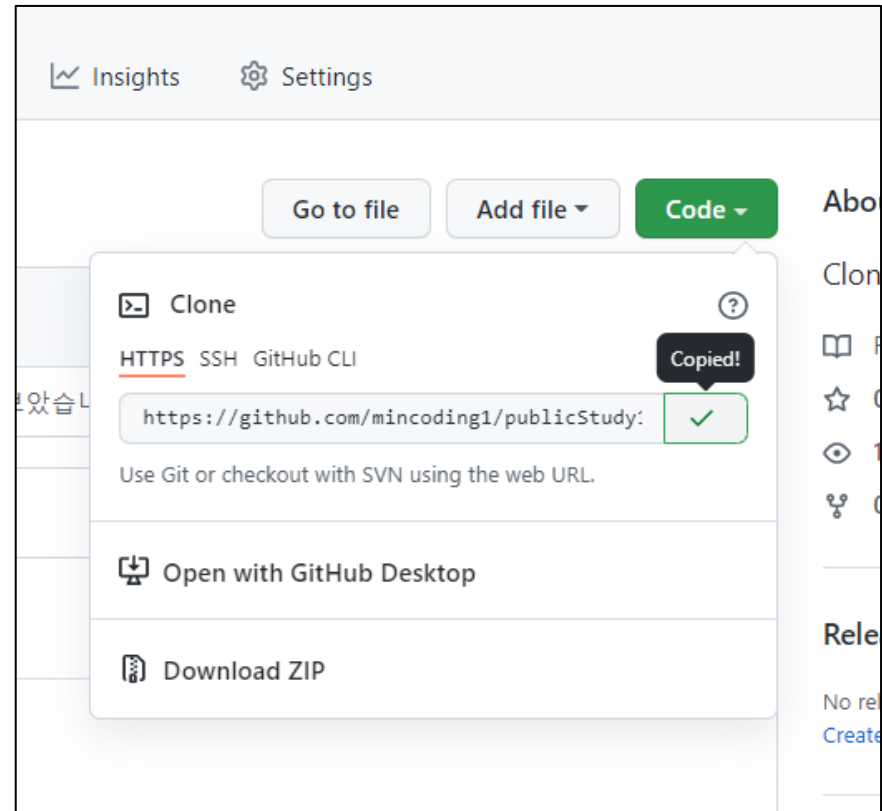
결과 확인



[도전] Clone 하기

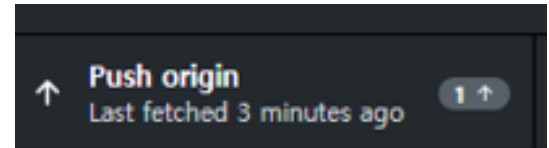
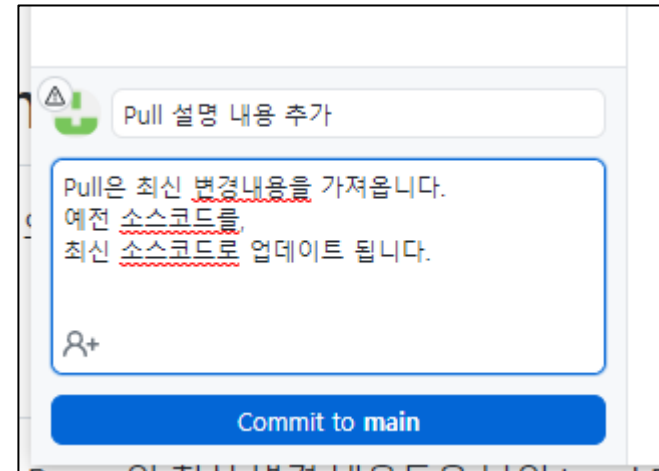
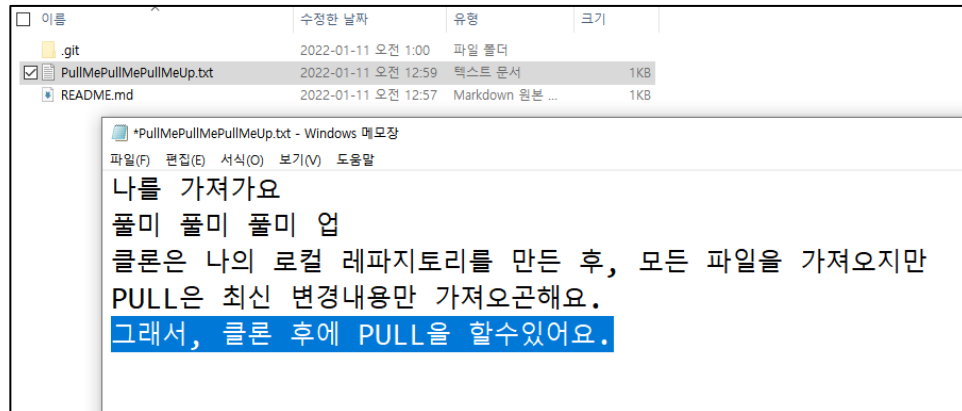
**Git URL을 복사해서
여러분께 전달합니다.**

이제 여러분은
이 Repo.를 Clone 하세요.



[강사만 진행] 눈으로만 보세요.

1. PullMePullMePullMeUp.txt 파일 수정하기
2. Commit 후, Push 하기



여러분이 하실 일

최신 버전이 업데이트 되었으니,

여러분 Local Repo. 에 업데이트 하셔야합니다.

→이때는 Pull 을 하면
업데이트가 완료됩니다.

Pull 하는 방법

Fetch를 한번 누르면,
Pull 버튼이 나타남

Branch Help

Current branch **main**

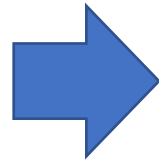
Fetch origin
Last fetched 8 minutes ago

풀미풀미

inho.choi ffc714c ± 1 changed file +2 -0 ⚙ New

나를 가져가는
내용을 적었습니다.

PullMePullMePullMeUp.txt	+		@@ -0,0 +1,2 @@
		1	+나를 가져가요
		2	+풀미 풀미 풀미 업 ↻



Branch Help

Current branch **main**

Pull origin
Last fetched just now 1 ↓

풀미풀미

inho.choi ffc714c ± 1 changed file +2 -0 ⚙ New

나를 가져가는
내용을 적었습니다.

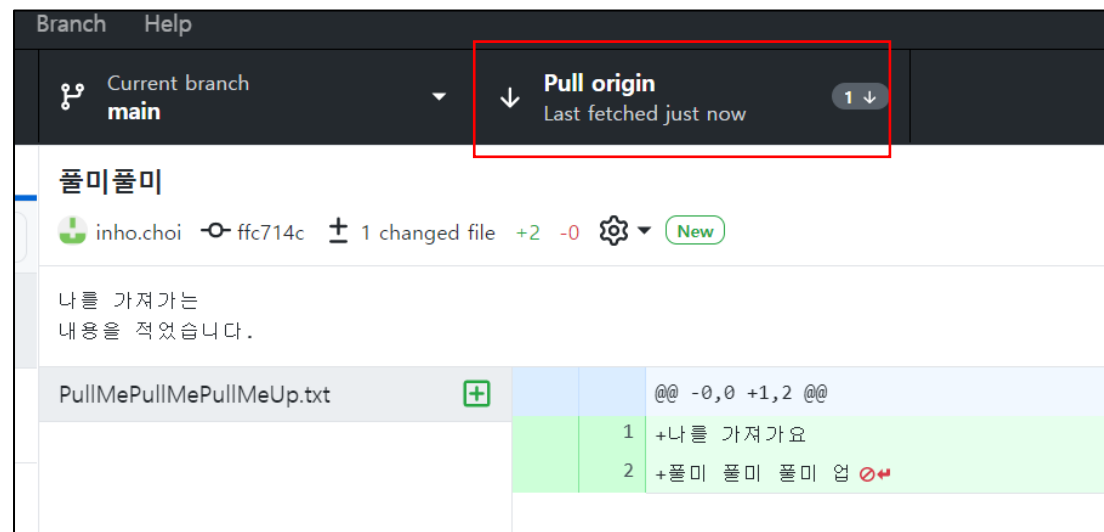
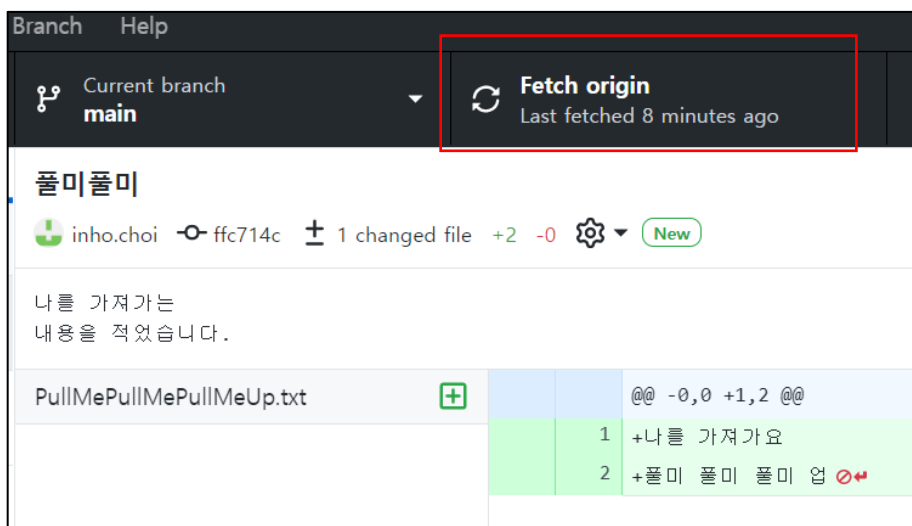
PullMePullMePullMeUp.txt	+		@@ -0,0 +1,2 @@
		1	+나를 가져가요
		2	+풀미 풀미 풀미 업 ↻

이해해보자! → 이해 성공했다면 “완료” 라고하기

Fetch : 최신 업데이트 내용이 있는지, 다운로드 후 확인만 하는 작업

Merge : Fetch로 확인한 내용을, 기존 Local Repo.에 합치기!

Pull : Fetch + Merge로 한방에 다 하기 = **한마디로 최신버전 업데이트!**



GitHub Desktop 프로그램은, Fetch 버튼을 한번 누르고, Pull 누르도록 만들어져있음

[실습] Fetch 후 Pull 결과

결과 화면

Current repository
publicStudy119

Current branch
main

Fetch origin
Last fetched just now

Changes

History

No branches to compare

Pull 설명 내용 추가

inho.choi • 3m

풀미풀미

inho.choi • 14m

README 문서를 하나 적어보았습니다.

mincoding • 21m

Initial commit

mincoding • 28m

Pull 설명 내용 추가

inho.choi f0dc007 1 changed file +4 -1 New

Pull은 최신 변경내용을 가져옵니다.
예전 소스코드를,
최신 소스코드로 업데이트 됩니다.

PullMePullMePullMeUp.txt

@@ -1,2 +1,5 @@

1

1

나를 가져가요

2

-풀미 풀미 풀미 업

2

+풀미 풀미 풀미 업

3

+클론은 나의 로컬 레파지토리를 만든 후, 모든 파일을 가져오지만

4

+PULL은 최신 변경내용만 가져오곤해요.

5

+그래서, 클론 후에 PULL을 할수있어요.

기다리면서
15분간 정리 시간을 갖자.
[내용을 99%가 아닌, 100% 이해하자!!]

Git의 활용

먼저, 디버깅이란

De : 죽이다.

Bug : 벌레

ing : ~ 하다.

Debugging : 버그 죽이는 활동

Git을 사용해서 디버깅을 할 수 있다.

예시를 들어보자.

Git History에 다음과 같다.

12/15일, 마리오 점프 기능 업데이트
12/16일, 마리오 점프 기능 버그 수정
12/19일, 와리오 캐릭터 추가
12/20일, 점프 기능 업그레이드
12/21일, 슈퍼 점프 기능 추가.

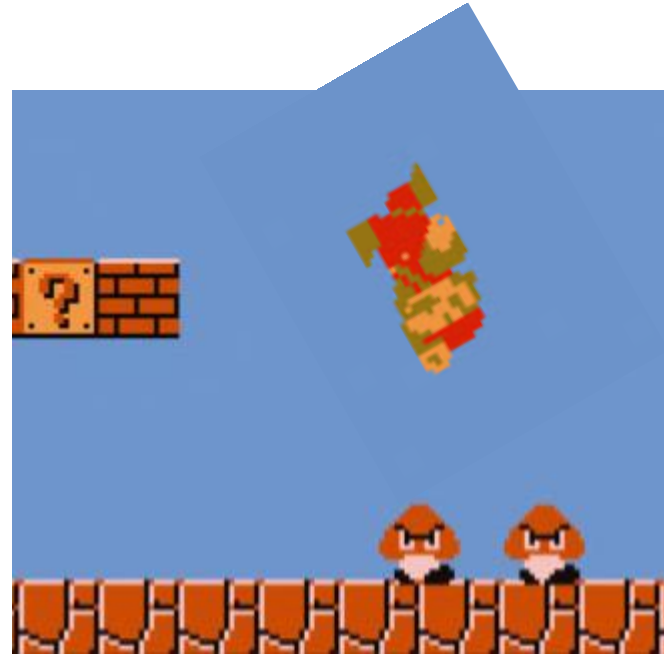


갑자기 마리오 점프가 이상하다!

버그가 갑자기 발견되었다.!

오늘 날짜는 12/22일 이다.

12/15일, 마리오 점프 기능 업데이트
12/16일, 마리오 점프 기능 버그 수정
12/19일, 와리오 캐릭터 추가
12/20일, 점프 기능 업그레이드
12/21일, 슈퍼 점프 기능 추가.



언제부터 이 버그가 발견되었을까?

소스코드를 과거로 돌려보자.

12/15일, 마리오 점프 기능 업데이트
12/16일, 마리오 점프 기능 버그 수정
12/19일, 와리오 캐릭터 추가
12/20일, 점프 기능 업그레이드
12/21일, 슈퍼 점프 기능 추가.

잘 됨

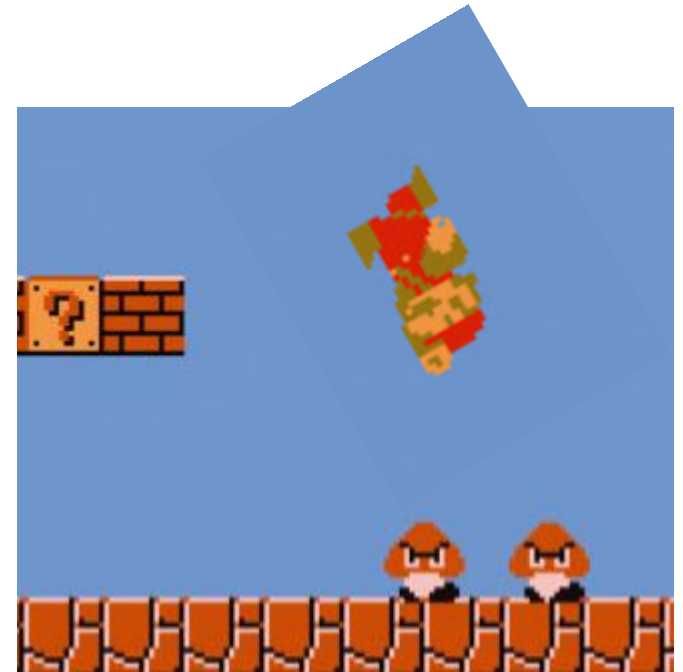


언제부터 이 버그가 발견되었을까?

소스코드를 과거로 돌려보자.

12/15일, 마리오 점프 기능 업데이트
12/16일, 마리오 점프 기능 버그 수정
12/19일, 와리오 캐릭터 추가
12/20일, 점프 기능 업그레이드
12/21일, 슈퍼 점프 기능 추가.

버그 재현



그렇다면 범인은?

와리오 캐릭터 추가할 때 생긴 버그로 추정!

→ 이렇게 디버깅을 할 수 있다.

12/15일, 마리오 점프 기능 업데이트
12/16일, 마리오 점프 기능 버그 수정
12/19일, 와리오 캐릭터 추가
12/20일, 점프 기능 업그레이드
12/21일, 슈퍼 점프 기능 추가.



Git의 History 기능

- 타임머신타고, 그 당시 소스코드로 회기 할 수 있음

Git 시험보기

[오픈북] 20분간 공부해서 제출하기

1. Local Repository과 Remote Repository의 차이
2. Git과 Github의 차이
3. GitHub Desktop과 Github의 차이
4. Fetch란?
5. Merge란?
6. Clone과 Pull의 차이
7. Git으로하는 디버깅 방법

MatterMost
강사의 개인 메시지로 보내기

* 유의하세요!! *
SSAFY 에서
사소한 부정행위 == 퇴소조치