



Github 사용

Git

- Git은 컴퓨터 파일의 변경 사항을 추적하고 여러 명의 사용자들 간에 해당 파일의 작업들을 조율하는 데에 사용되는 분산 버전 관리 시스템이다.
- 주로 소프트웨어 개발 과정에서 소스코드 관리 (버전 관리, 협업 등)을 위해서 주로 사용된다.
- 다양한 종류가 있으며 대표적으로 github가 있다.



Git을 쓰는 이유

- Git 등의 소스 관리 프로그램을 사용하지 않을 경우, 다음과 같이 버전관리를 하는 경우가 많다.

이름



project end.zip



project end_final_finish(2).zip



project end_final_finish.zip



project final.zip



project final_finish.zip

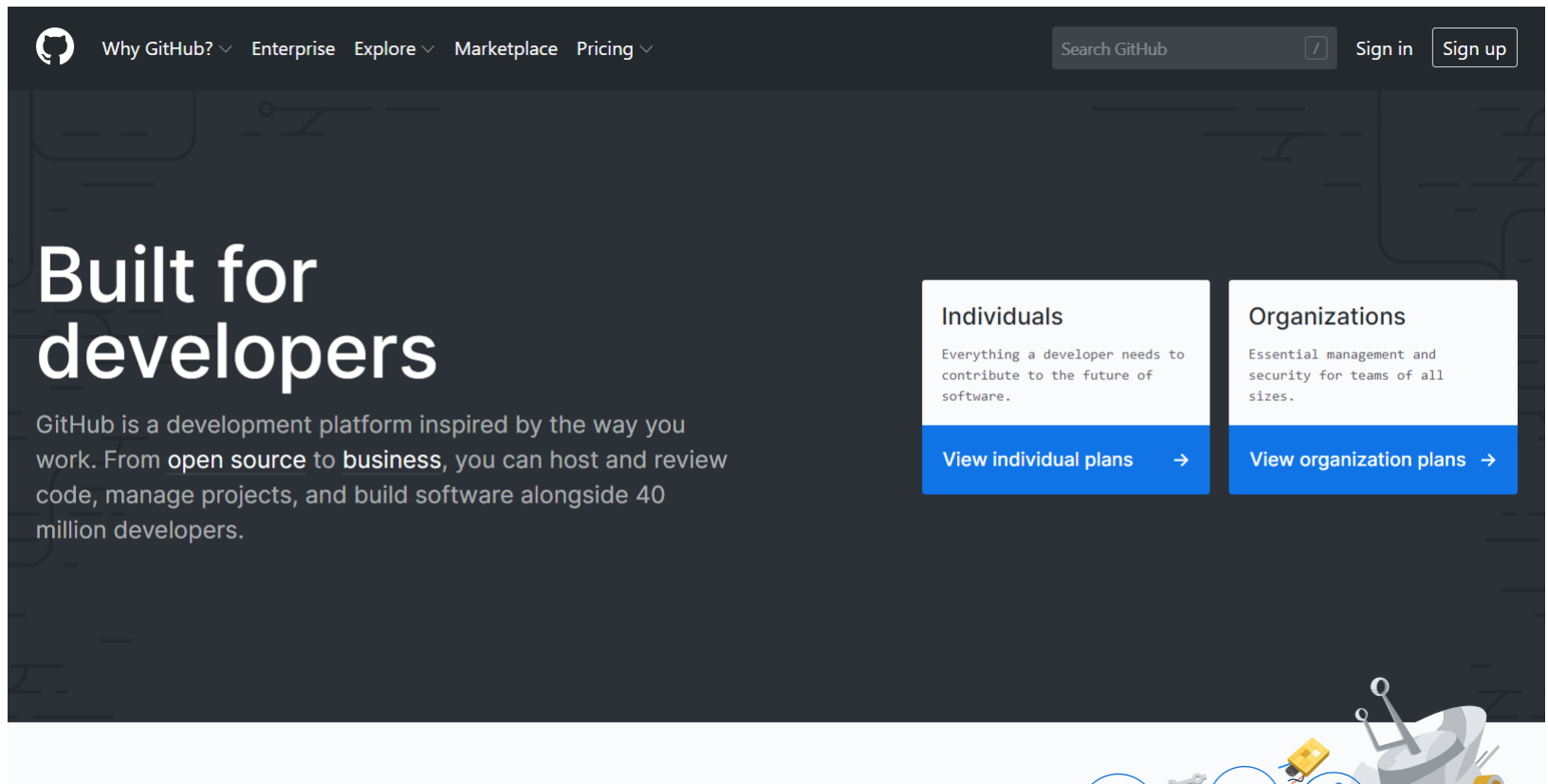


project final2.zip

- 위와 같이 사용하는 경우 조금만 시간이 지나도 어떤 파일이 정말 마지막 파일인지 알기 어렵다.

Github 가입

- <https://github.com/>
- Sign Up을 눌러 무료 회원가입할 수 있다.



Git 설치하기

- <https://git-scm.com/downloads>
- 자신의 PC에 맞는 버전을 다운로드하여 설치한다.



Git 설치하기

 Git 2.26.0 Setup

Information

Please read the following important information before continuing.



When you are ready to continue with Setup, click Next.

GNU General Public License

Version 2, June 1991

Copyright (C) 1989, 1991 Free Software Foundation, Inc.
59 Temple Place - Suite 330, Boston, MA 02111-1307, USA

Everyone is permitted to copy and distribute verbatim copies
of this license document, but changing it is not allowed.

Preamble

The licenses for most software are designed to take away your
freedom to share and change it. By contrast, the GNU General Public
License is intended to guarantee your freedom to share and change

<https://gitforwindows.org/>

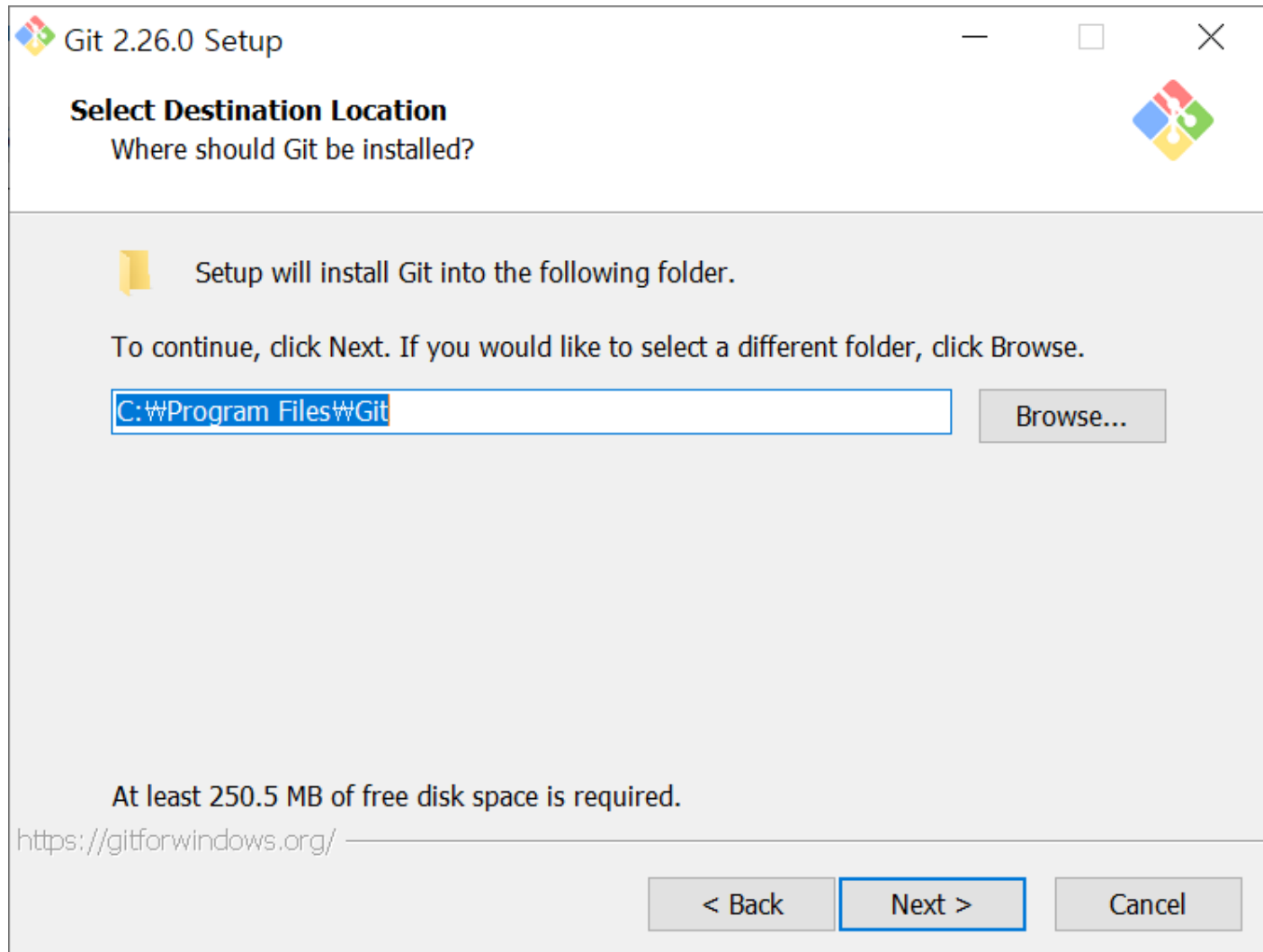
☒ Only show new options

Next >

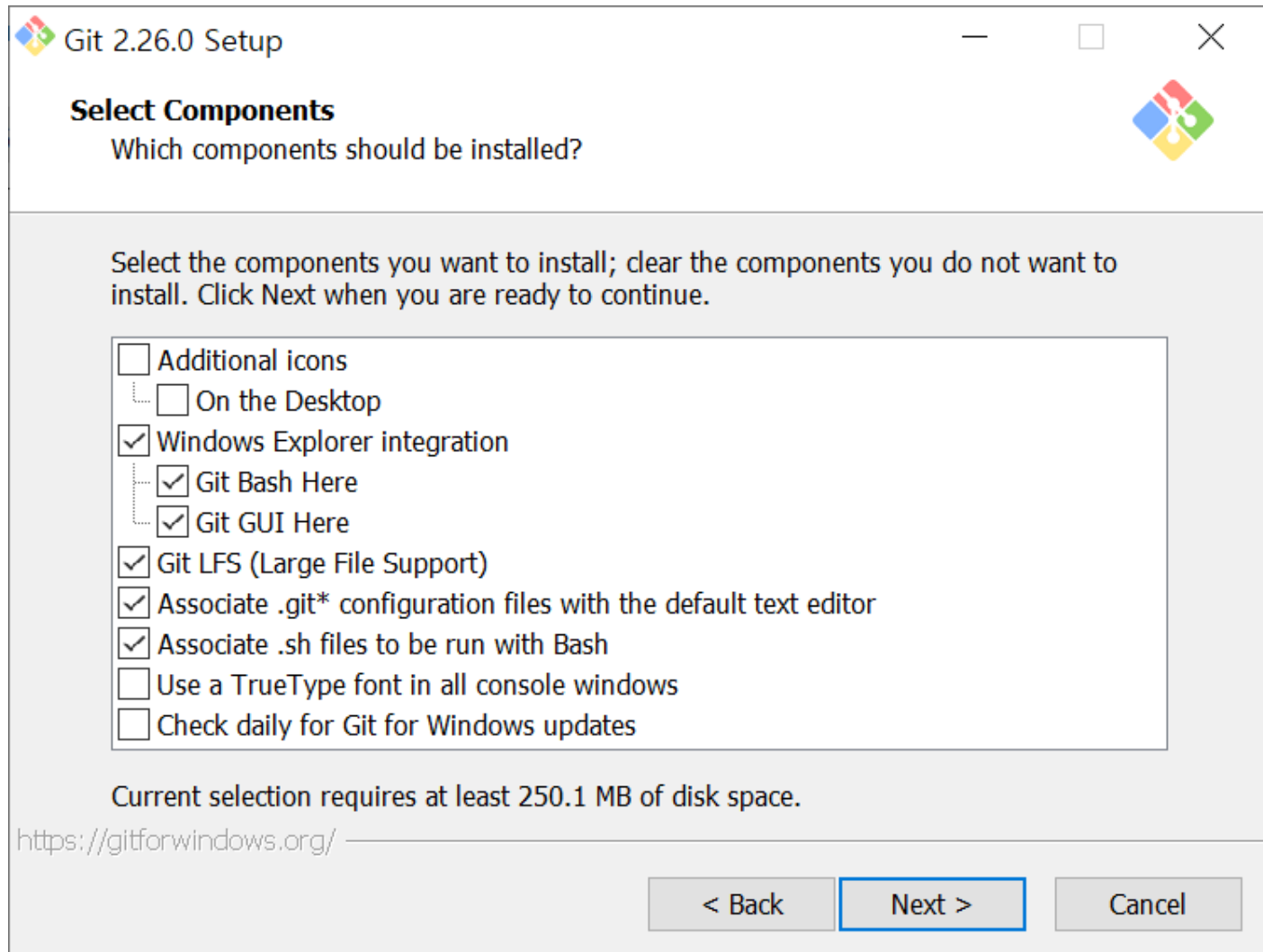
Cancel



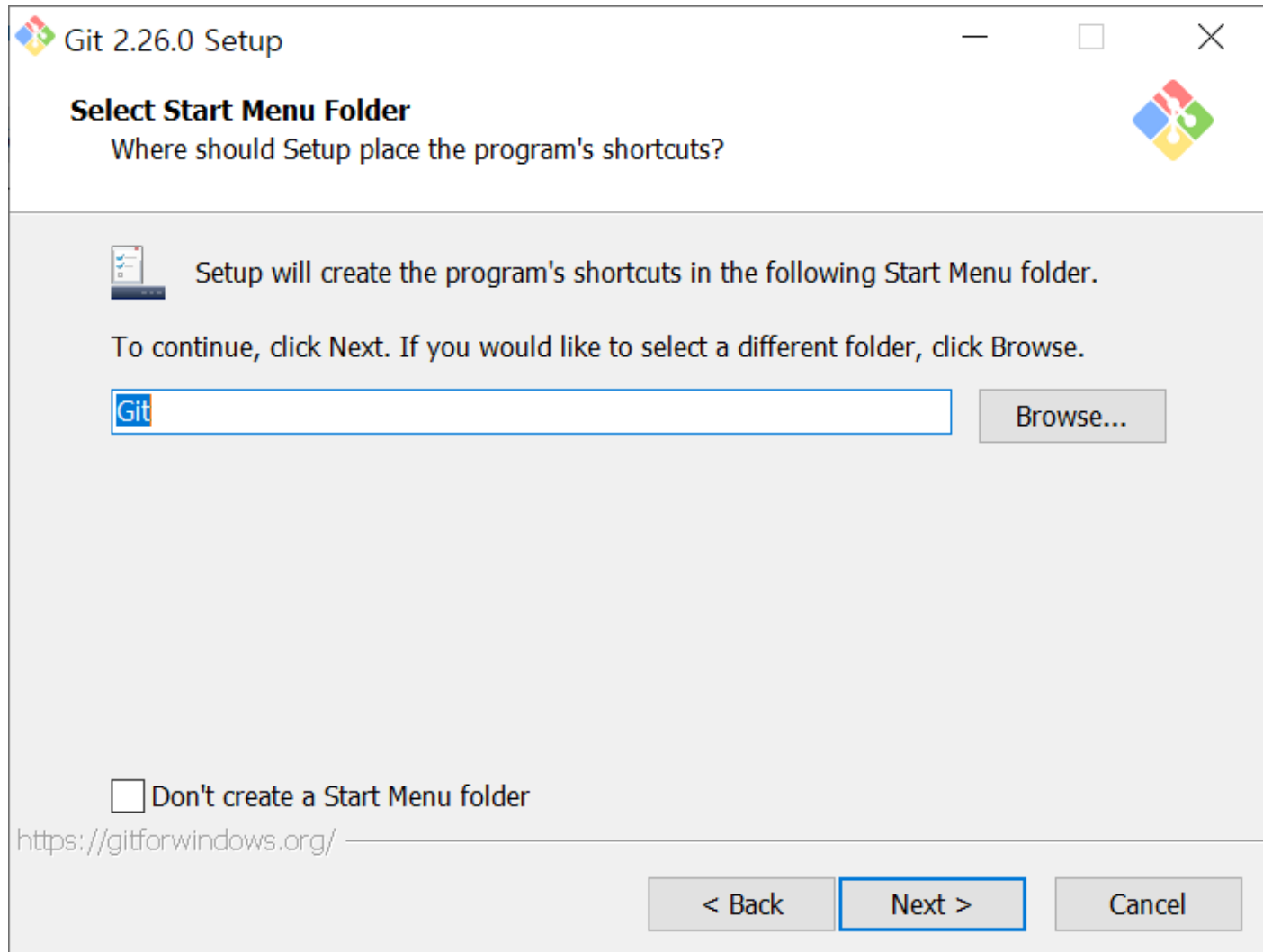
Git 설치하기



Git 설치하기



Git 설치하기



Git 설치하기

■ 원하는 편집기를 선택

Git 2.26.0 Setup

Choosing the default editor used by Git

Which editor would you like Git to use?

Use Vim (the ubiquitous text editor) as Git's default editor
Use Vim (the ubiquitous text editor) as Git's default editor
Use Notepad++ as Git's default editor
Use Visual Studio Code as Git's default editor
Use Visual Studio Code Insiders as Git's default editor
Use Sublime Text as Git's default editor
Use Atom as Git's default editor
Use VSCodium as Git's default editor
Select other editor as Git's default editor

to the EDITOR environment variable. The default editor is Vim, but you may set it to some other editor of your choice.

<https://gitforwindows.org/>

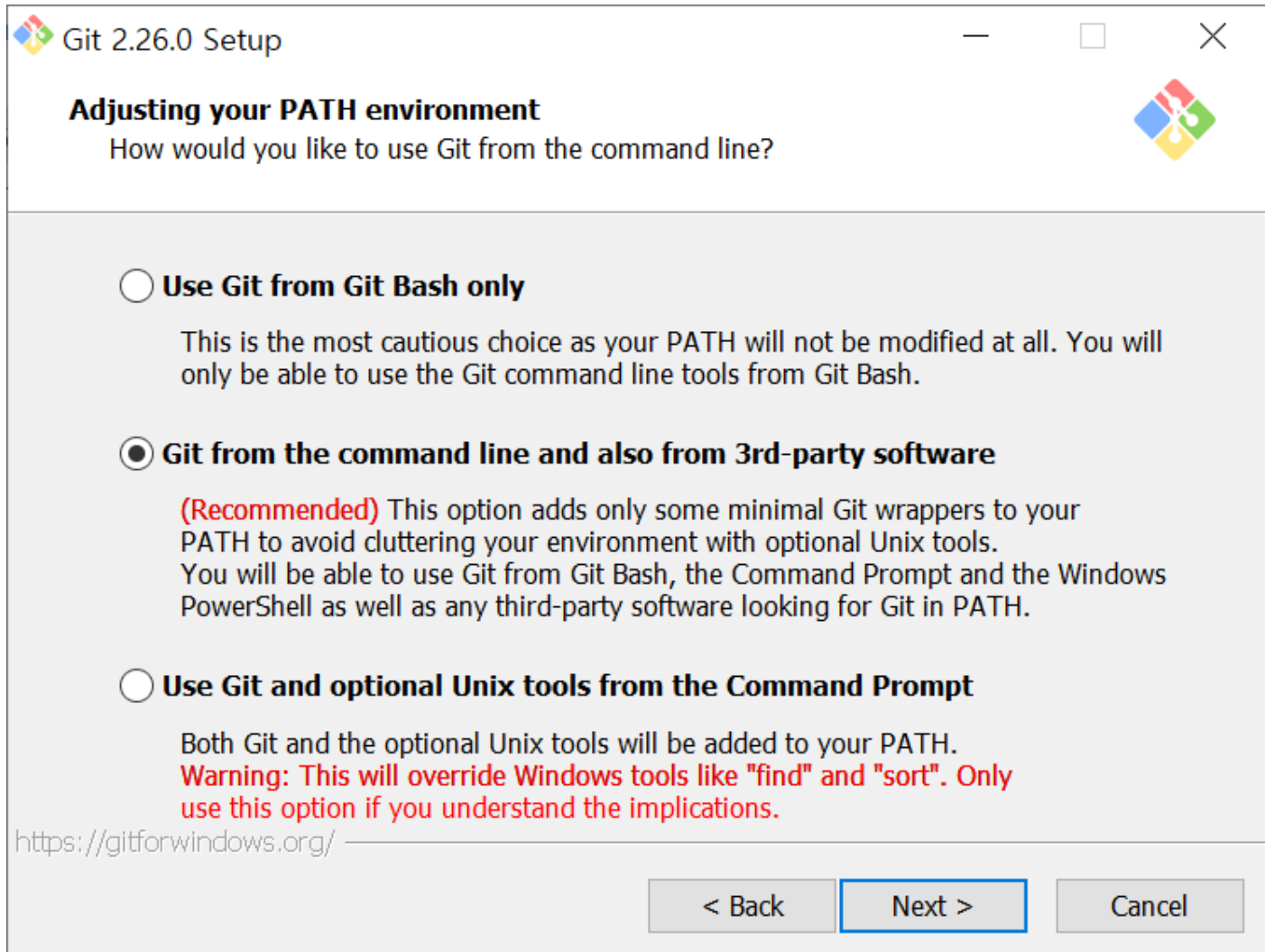
< Back

Next >

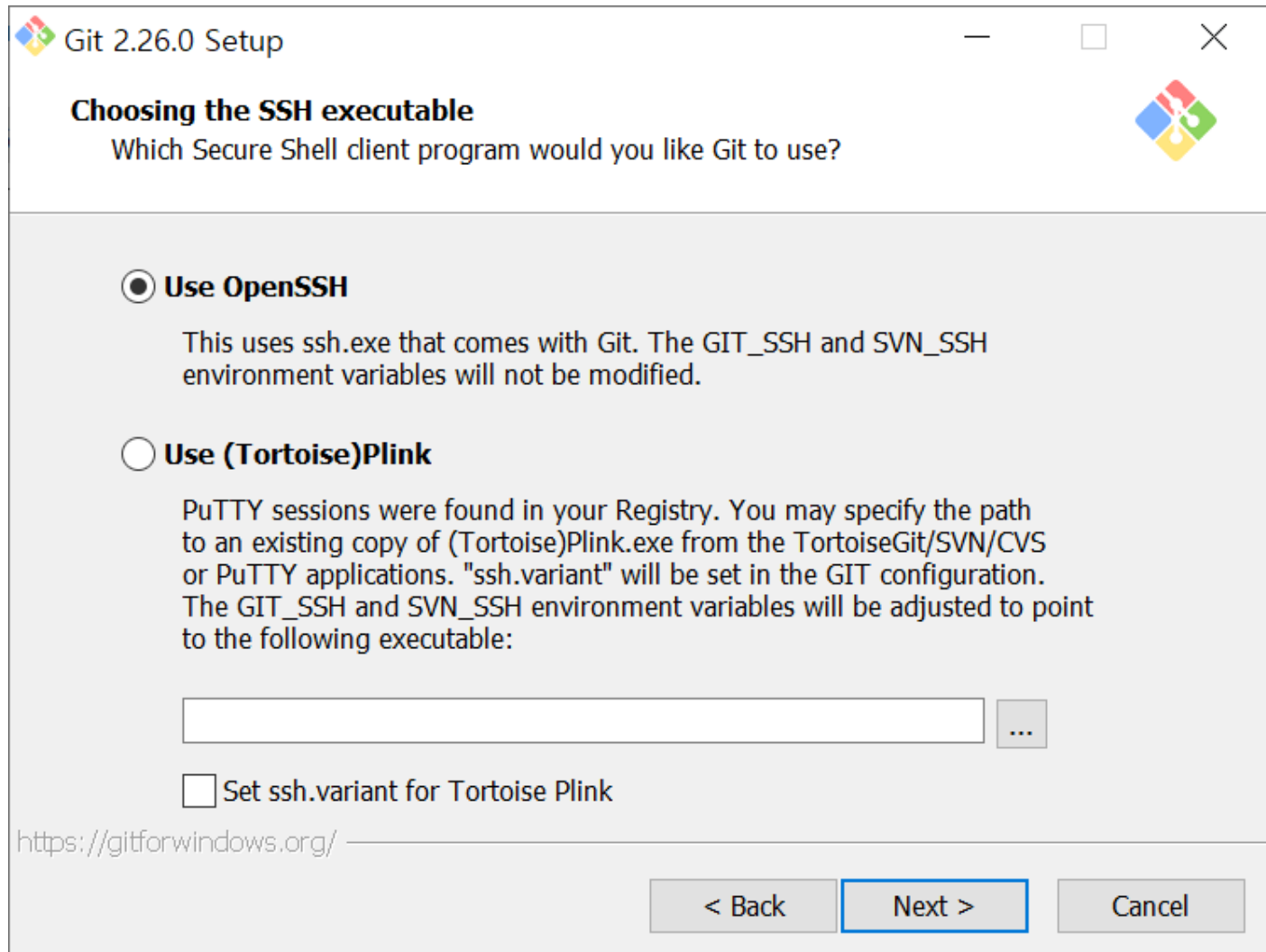
Cancel



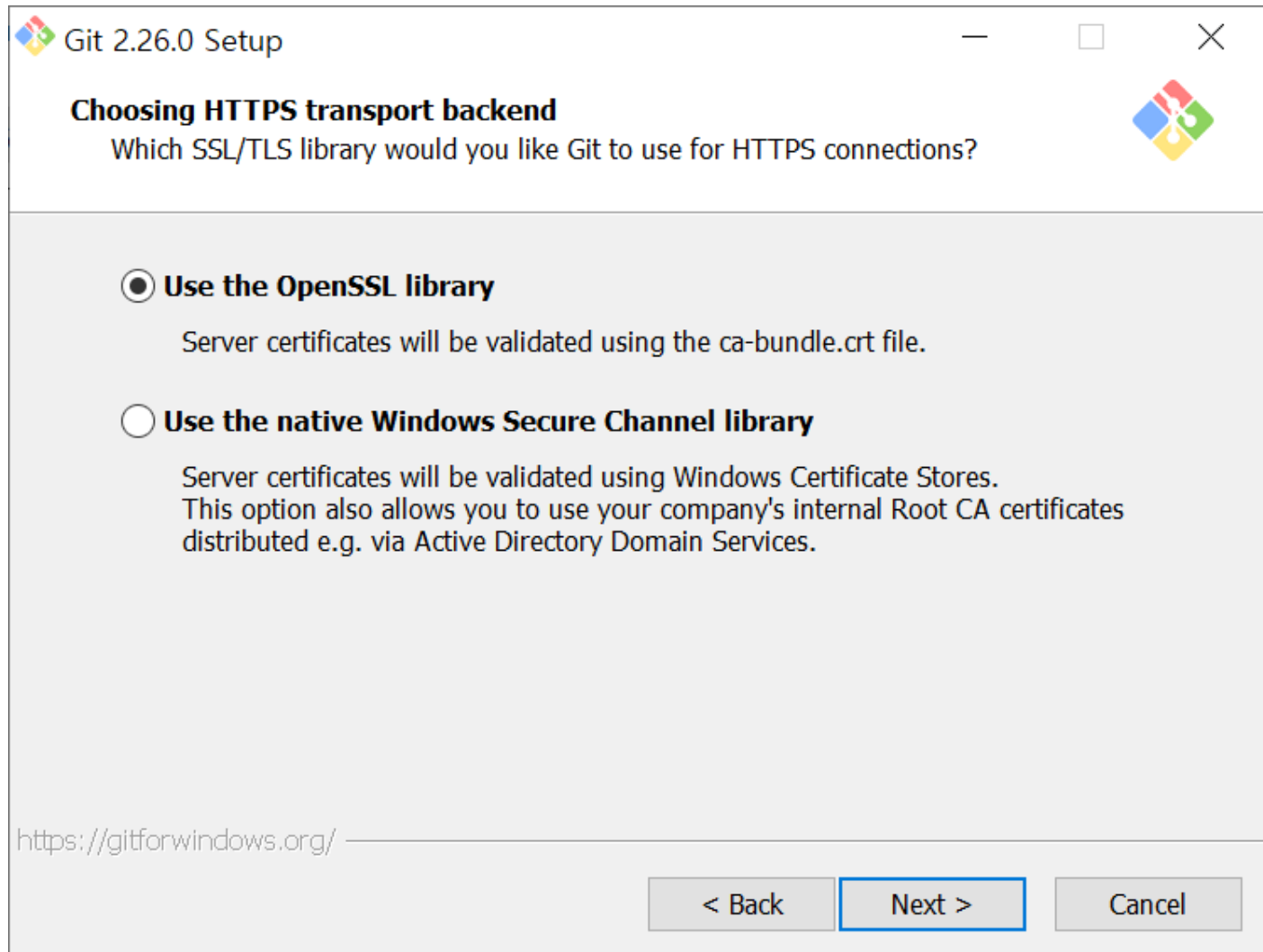
Git 설치하기



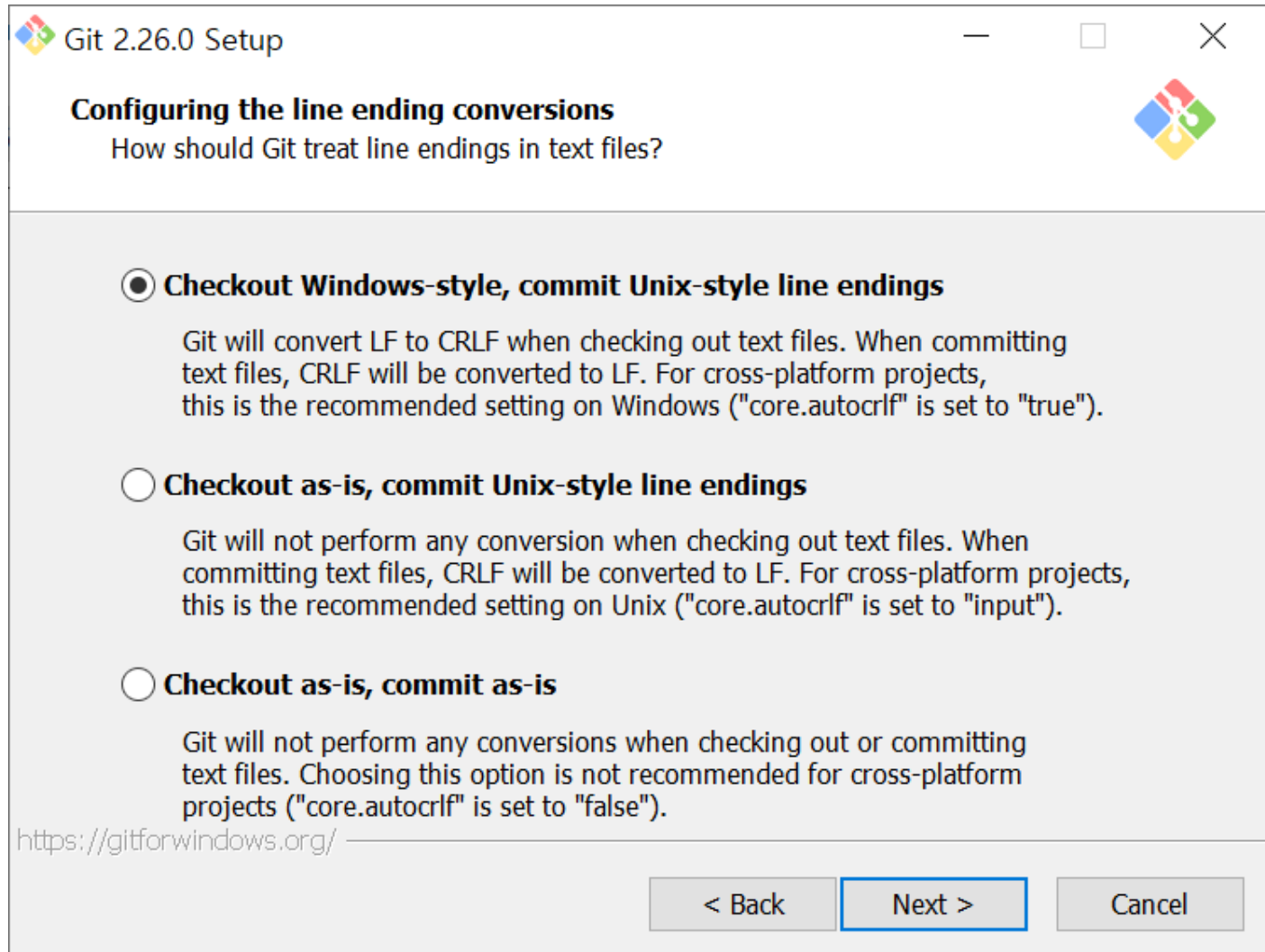
Git 설치하기



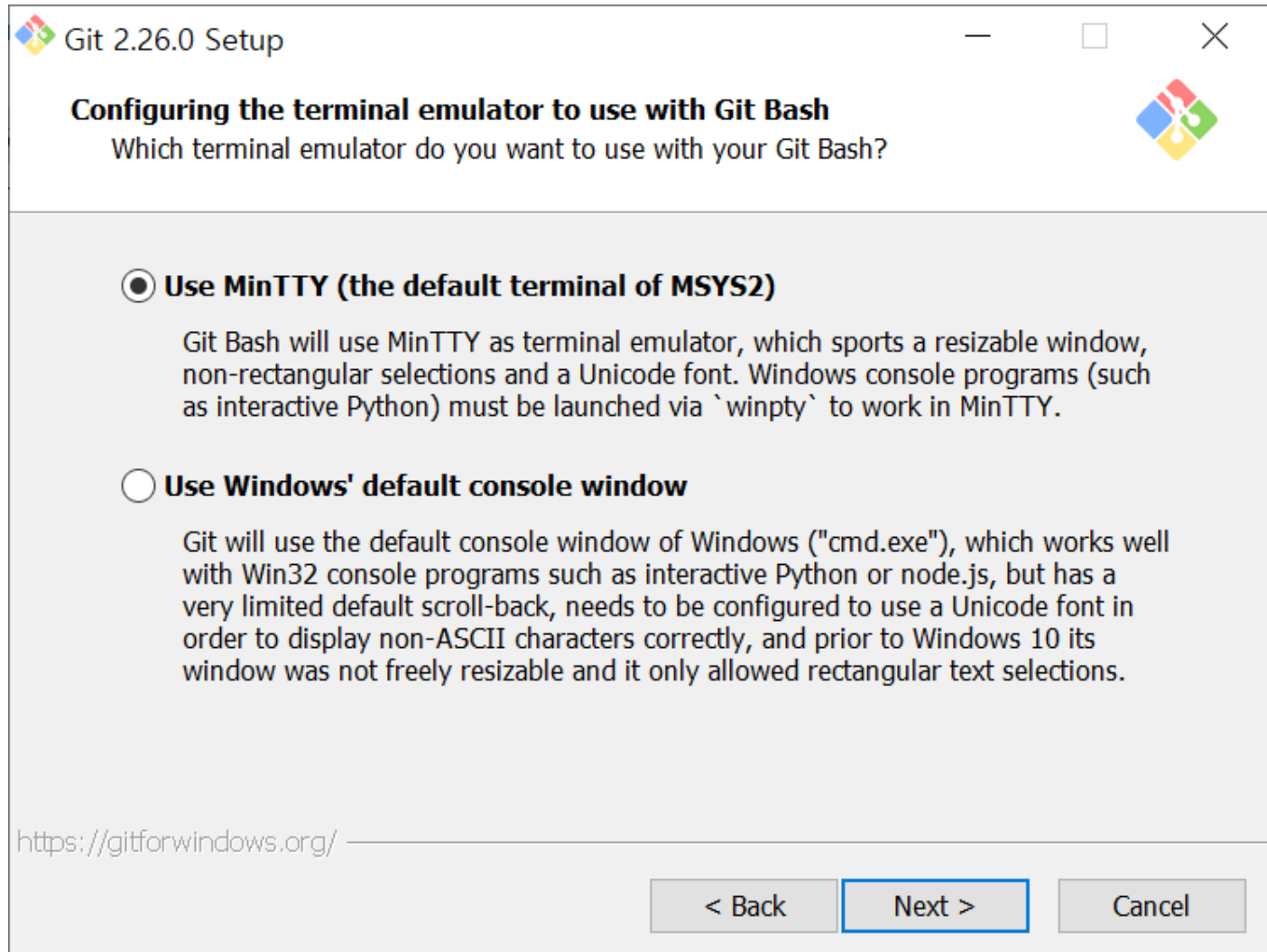
Git 설치하기



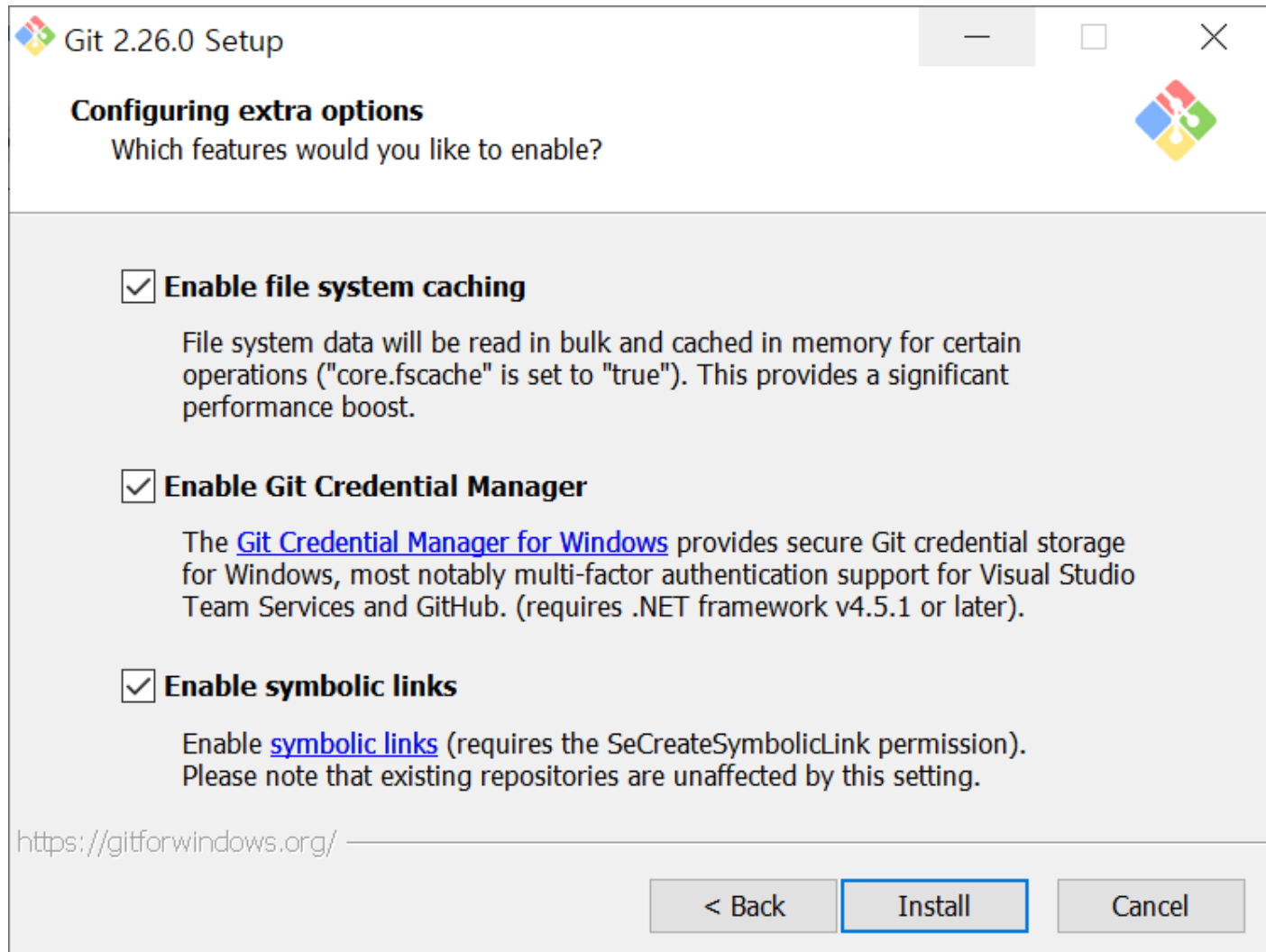
Git 설치하기



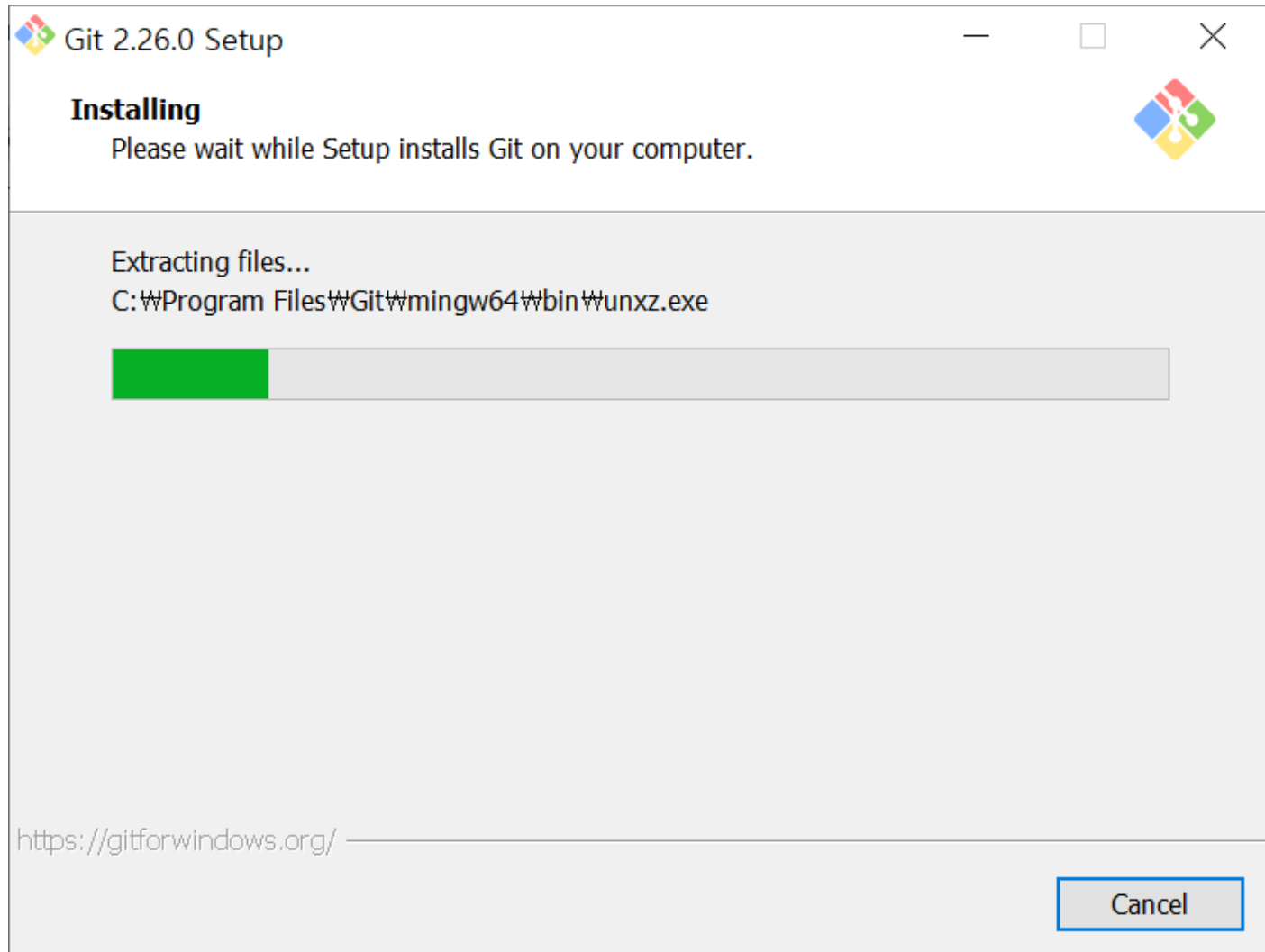
Git 설치하기



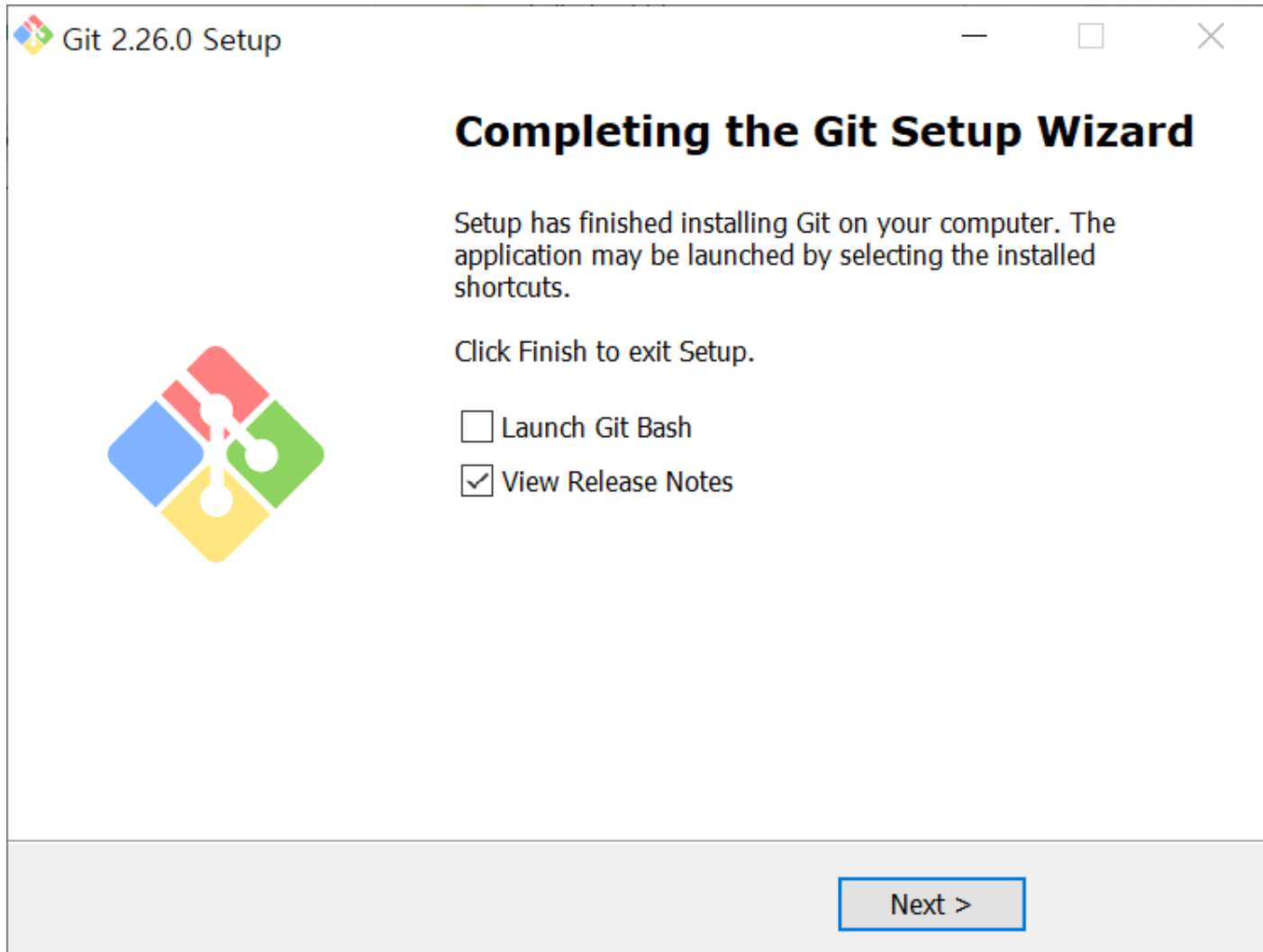
Git 설치하기



Git 설치하기



Git 설치하기



Git 설치하기

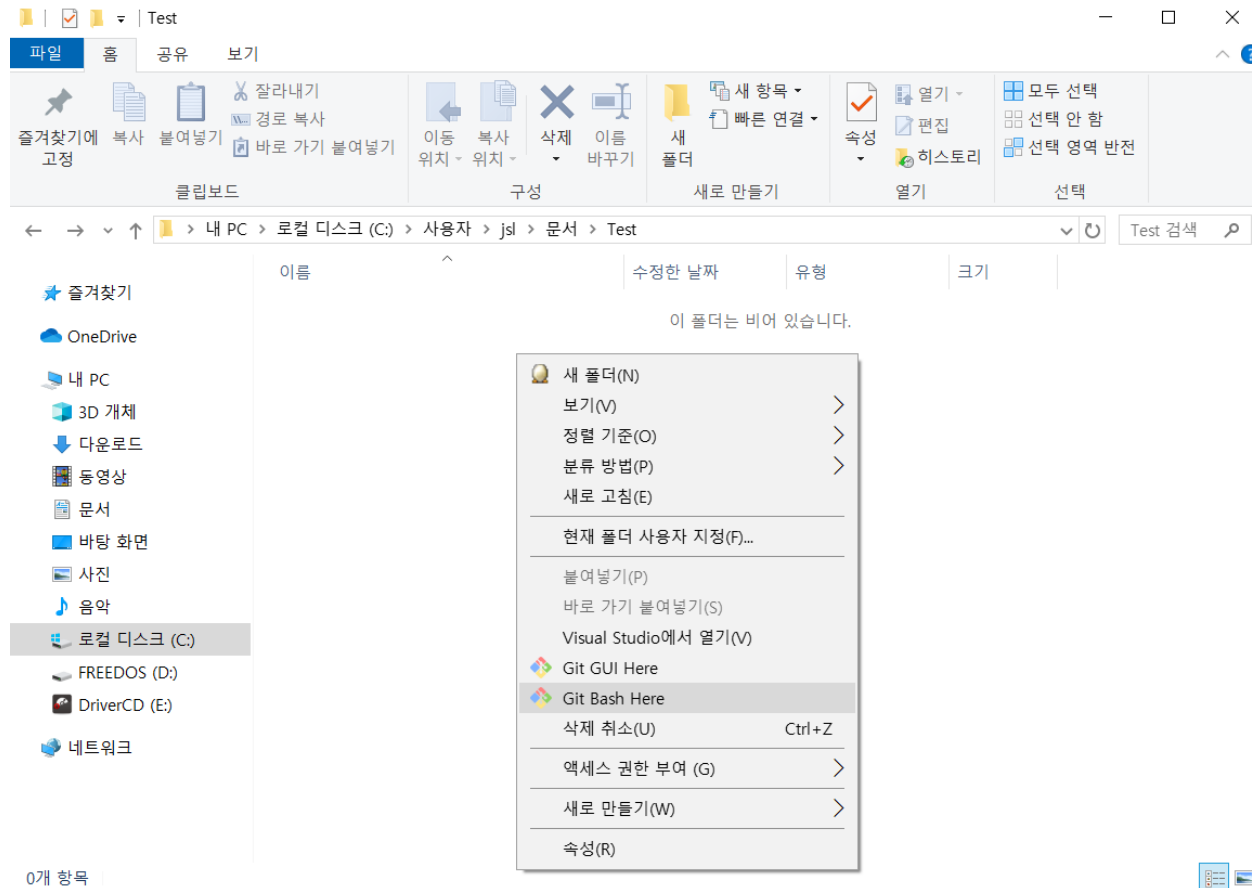
- 설치된 Git CMD를 실행 후 `git --version` 명령어 입력



```
Git CMD  
C:\Users\jsl>git --version  
git version 2.26.0.windows.1  
C:\Users\jsl>
```

Git 설치하기

- 사용하기를 원하는 폴더를 띄워놓고 우클릭하여 Git Bash Here 이 있는 것을 확인

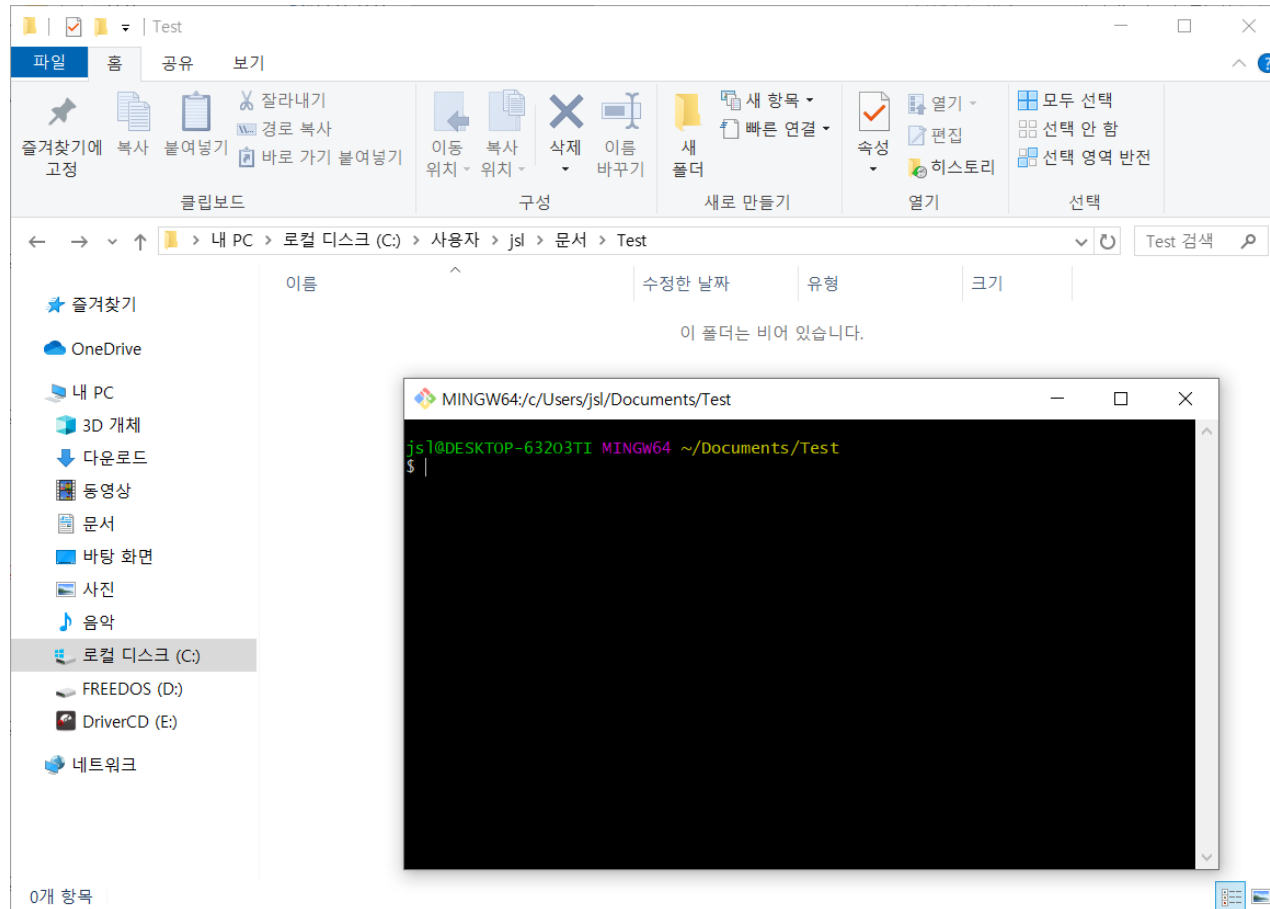


저장소

- Git의 저장소는 파일을 저장하고 버전별로 관리한다.
저장소는 로컬 저장소와 원격 저장소 두 가지가 있다.
- 로컬 저장소
 - ◆ 내 PC에 파일이 저장되는 개인 전용 저장소/
- 원격 저장소
 - ◆ 파일이 원격 저장소 서버(github 등)에서 관리된다. 여러 사람이 함께 공유하기 위한 저장소

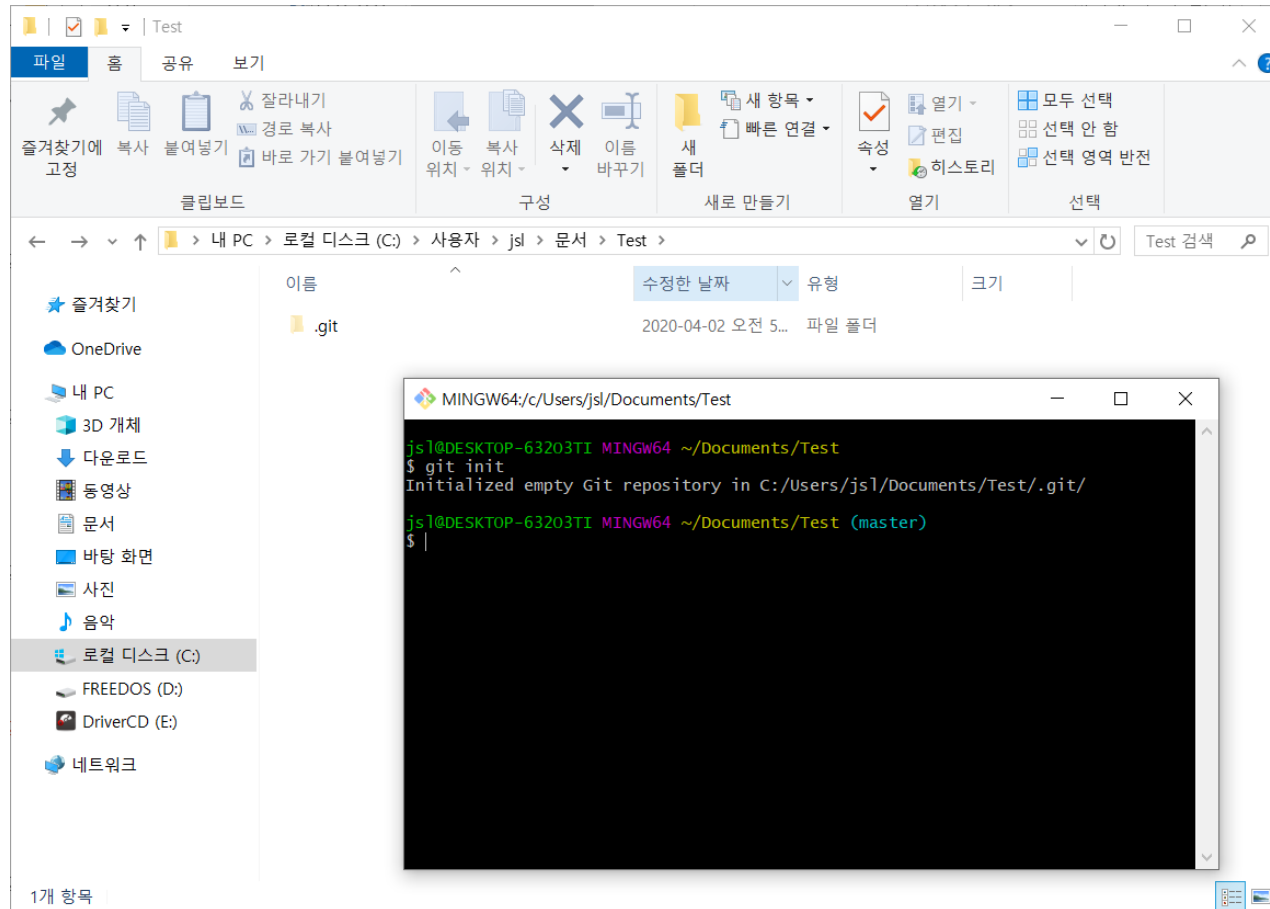
로컬 저장소 생성

- 로컬 저장소를 생성하고자 하는 폴더에서 git bash 실행



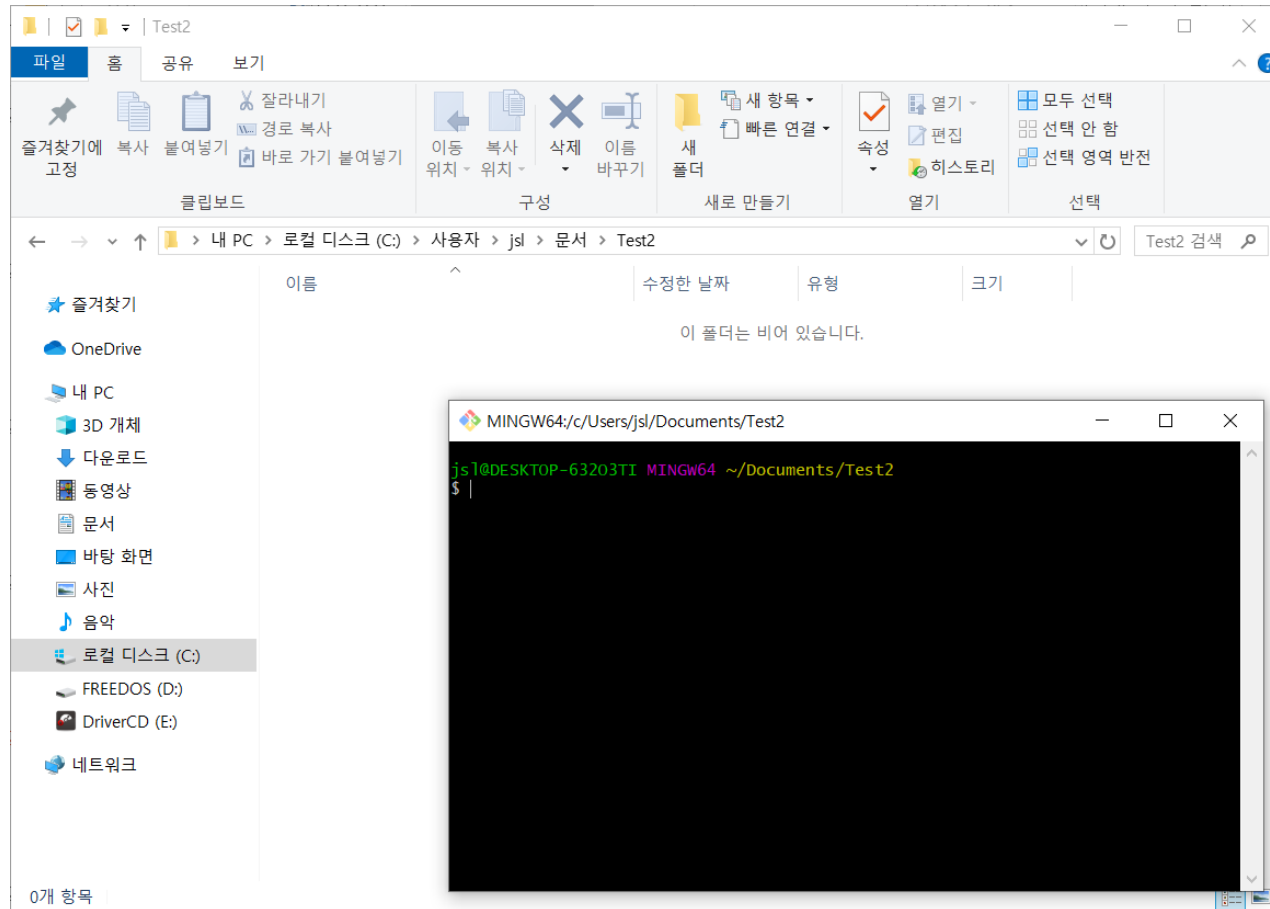
로컬 저장소 생성

■ git init 으로 로컬 저장소 생성 (초기화)



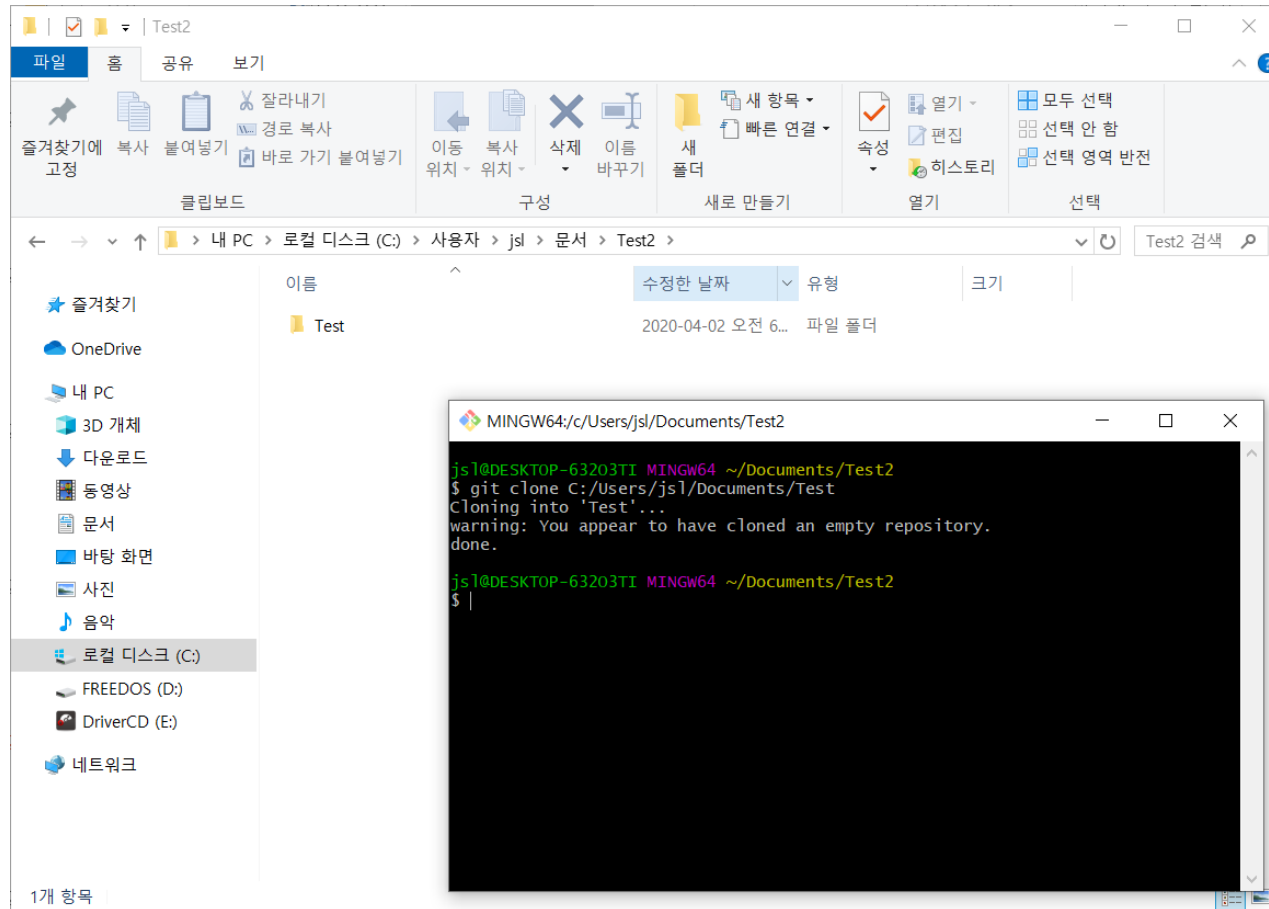
로컬 저장소 복제

- 저장소를 복제하기를 원하는 장소에서 git bash 실행



로컬 저장소 복제

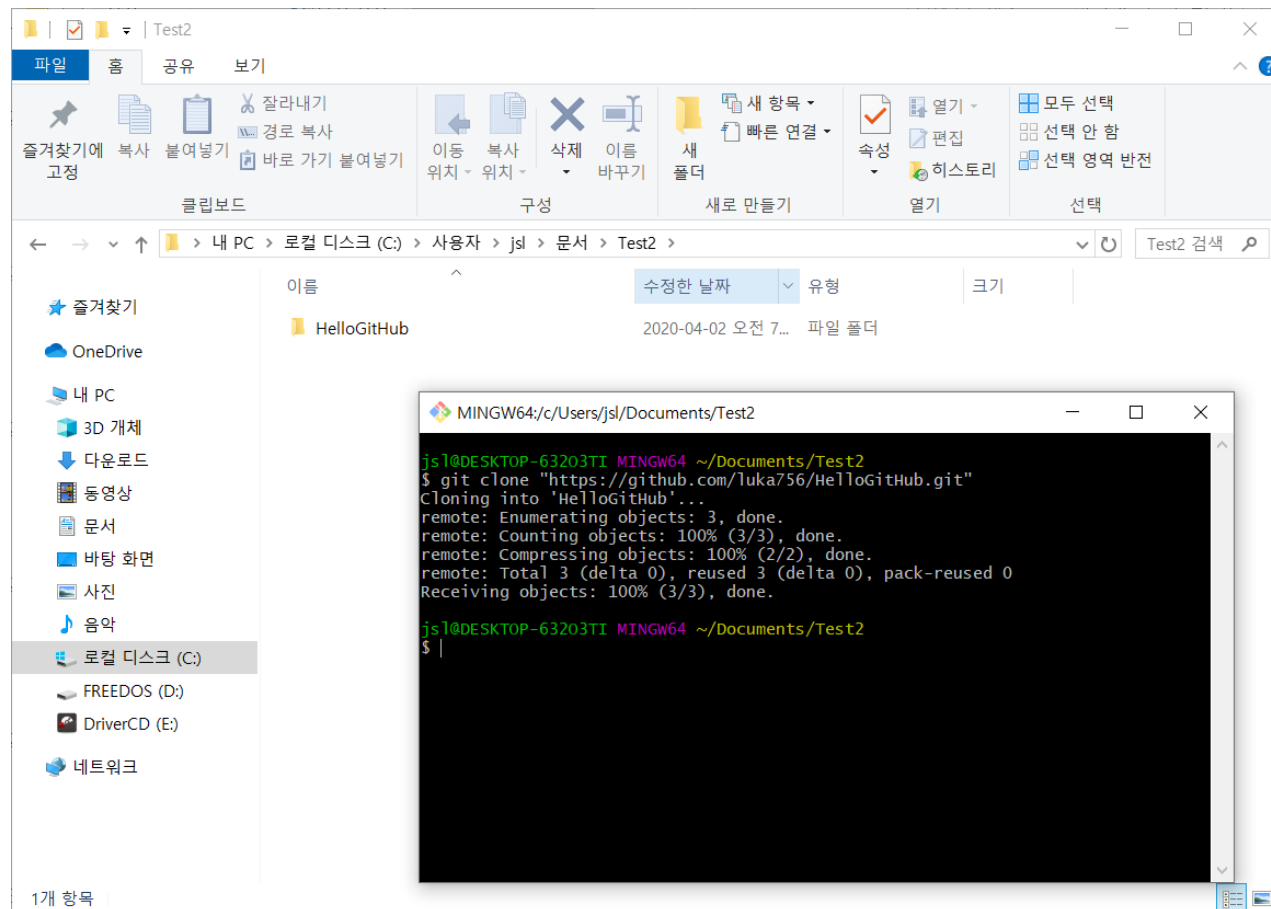
■ git clone <경로>



원격 저장소 가져오기

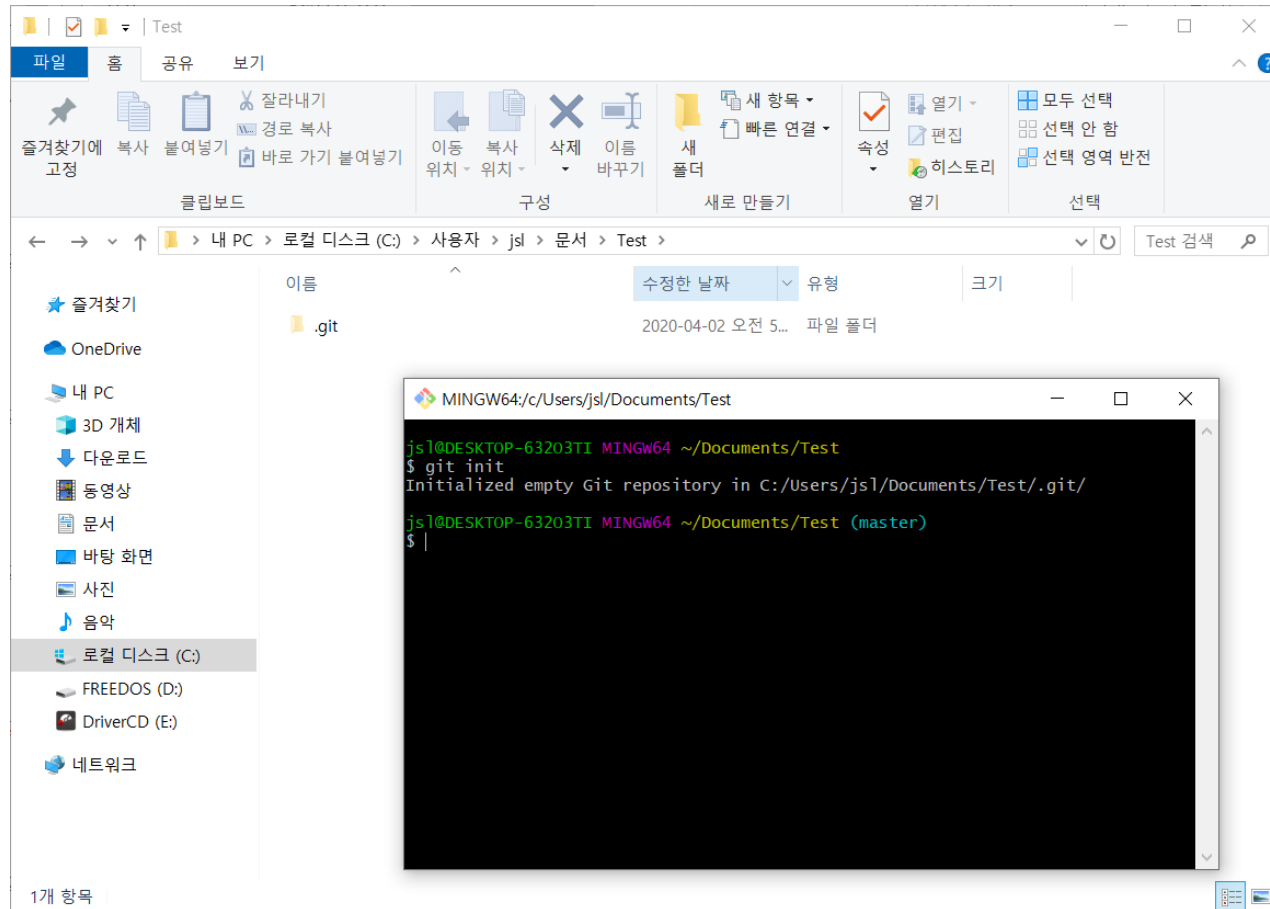
■ git clone “원격 저장소 주소”

◆ Ex) git clone "https://github.com/luka756/HelloGitHub.git"



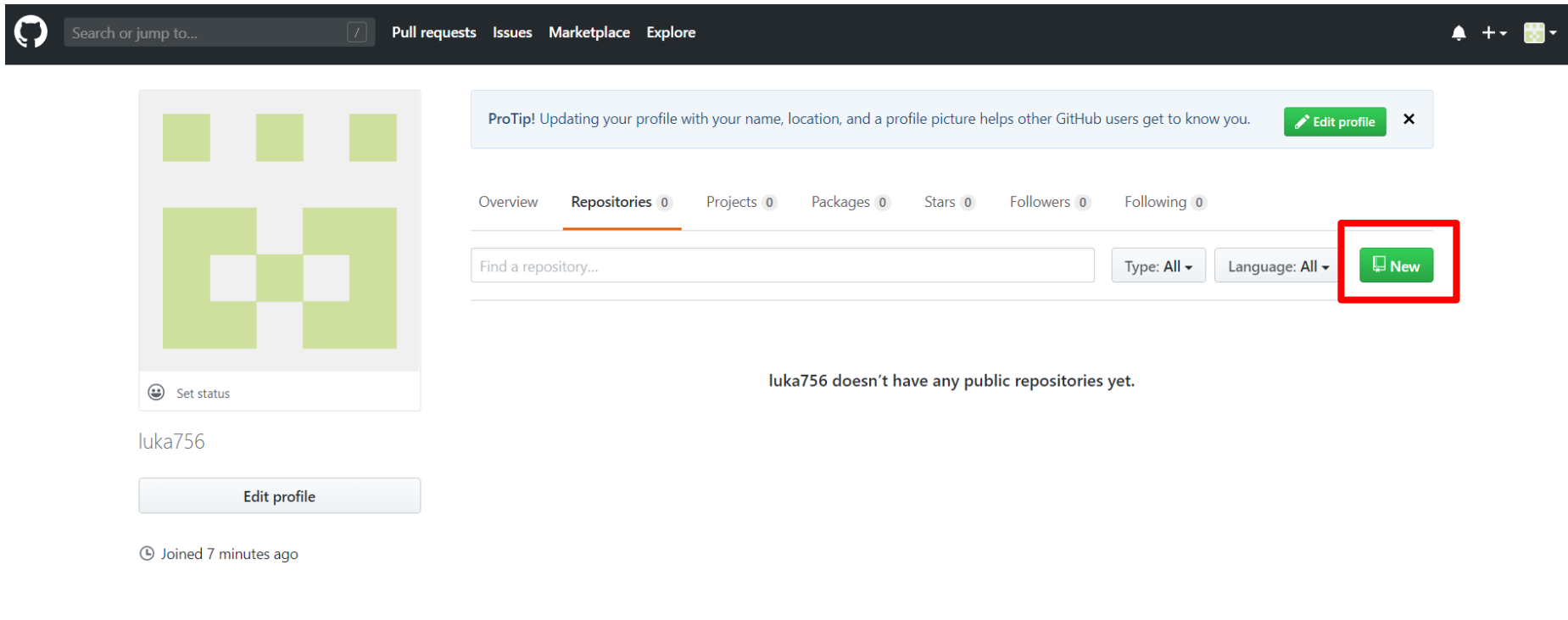
원격 저장소 가져오기

■ git init 으로 로컬 저장소 생성 (초기화)



원격 저장소 생성

■ Github repository 에서 New 선택



The screenshot shows the GitHub profile page for user 'luka756'. The page layout includes a top navigation bar with the GitHub logo, a search bar, and links for 'Pull requests', 'Issues', 'Marketplace', and 'Explore'. On the left side, there is a profile card for 'luka756' with a placeholder for a profile picture, a 'Set status' button, an 'Edit profile' button, and a note 'Joined 7 minutes ago'. The main content area features a 'ProTip!' banner, a tabbed interface with 'Overview', 'Repositories 0', 'Projects 0', 'Packages 0', 'Stars 0', 'Followers 0', and 'Following 0'. The 'Repositories' tab is selected, showing a search bar 'Find a repository...', filters for 'Type: All' and 'Language: All', and a green 'New' button highlighted with a red rectangle. Below this, a message states 'luka756 doesn't have any public repositories yet.'

원격 저장소 생성

■ repository 이름 설정 후 생성

Create a new repository

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

Owner

 luka756 ▾

Repository name *

HelloGitHub ✓

Great repository names are short and memorable. Need inspiration? How about **congenial-invention**?

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: None ▾

Add a license: None ▾



Create repository



원격 저장소 생성

■ repository 이름 설정 후 생성

The screenshot shows the GitHub interface for a repository named 'HelloGitHub' by user 'luka756'. The repository has 1 watch, 0 stars, and 0 forks. The 'Code' tab is selected, showing options for 'Set up in Desktop', 'HTTPS', and 'SSH'. The SSH URL 'https://github.com/luka756/HelloGitHub.git' is highlighted with a red box. Below this, there is a section for creating a new repository on the command line with a list of git commands.

luka756 / HelloGitHub

Unwatch 1 Star 0 Fork 0

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

Quick setup — if you've done this kind of thing before

원격 저장소 주소(경로)

Set up in Desktop or HTTPS SSH **https://github.com/luka756/HelloGitHub.git**

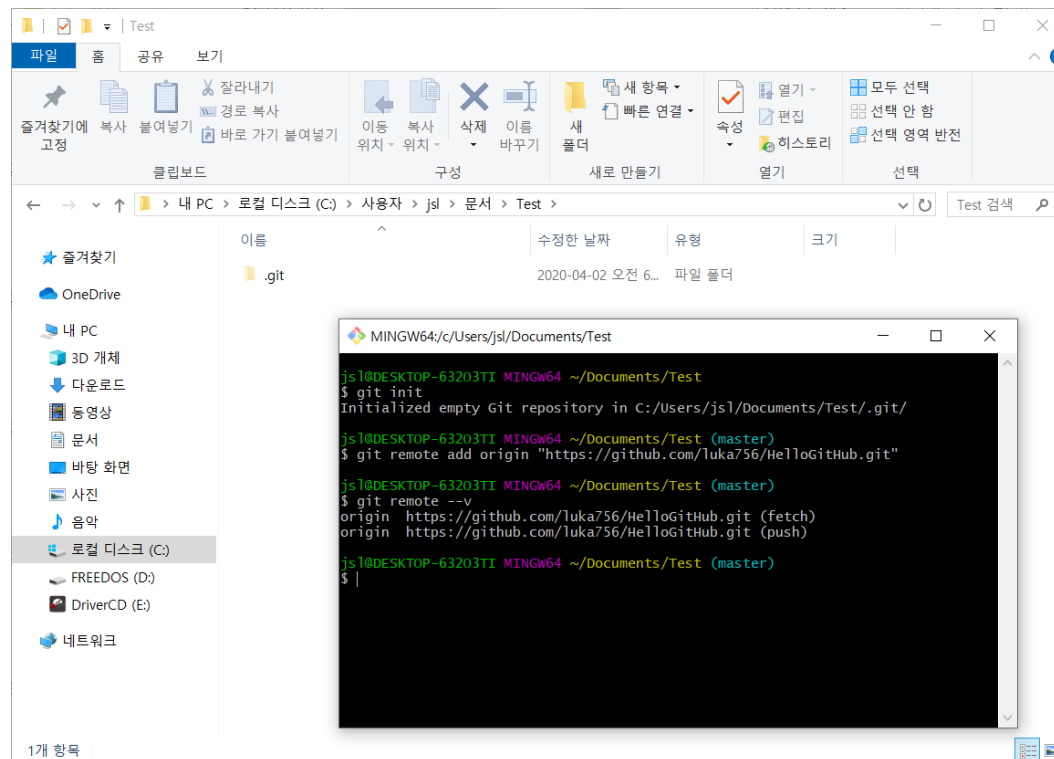
Get started by [creating a new file](#) or [uploading an existing file](#). We recommend every repository include a [README](#), [LICENSE](#), and [.gitignore](#).

...or create a new repository on the command line

```
echo "# HelloGitHub" >> README.md
git init
git add README.md
git commit -m "first commit"
git remote add origin https://github.com/luka756/HelloGitHub.git
git push -u origin master
```

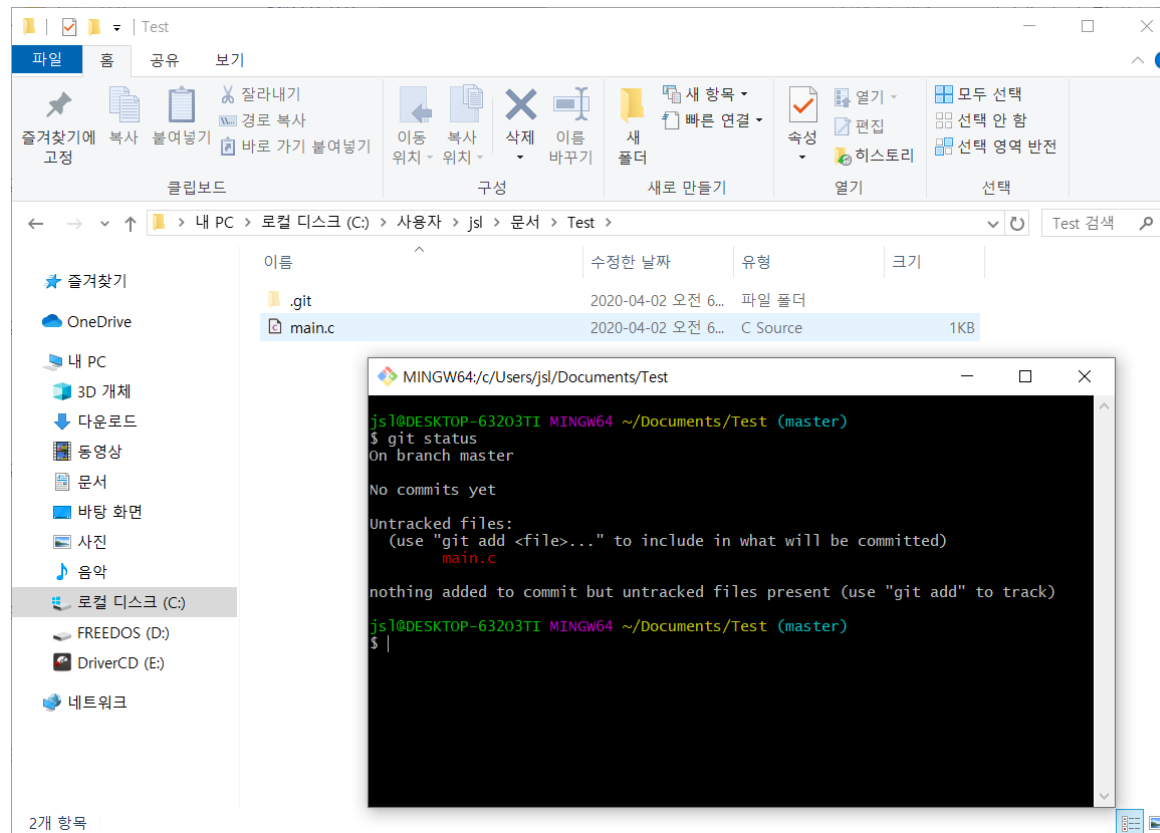
저장소 연결

- 로컬 저장소와 원격 저장소를 연결한다.
- `git remote add origin "원격 저장소 주소"`
 - ◆ Ex) `git remote add origin "https://github.com/luka756/HelloGitHub.git"`
- `git remove --v`로 확인



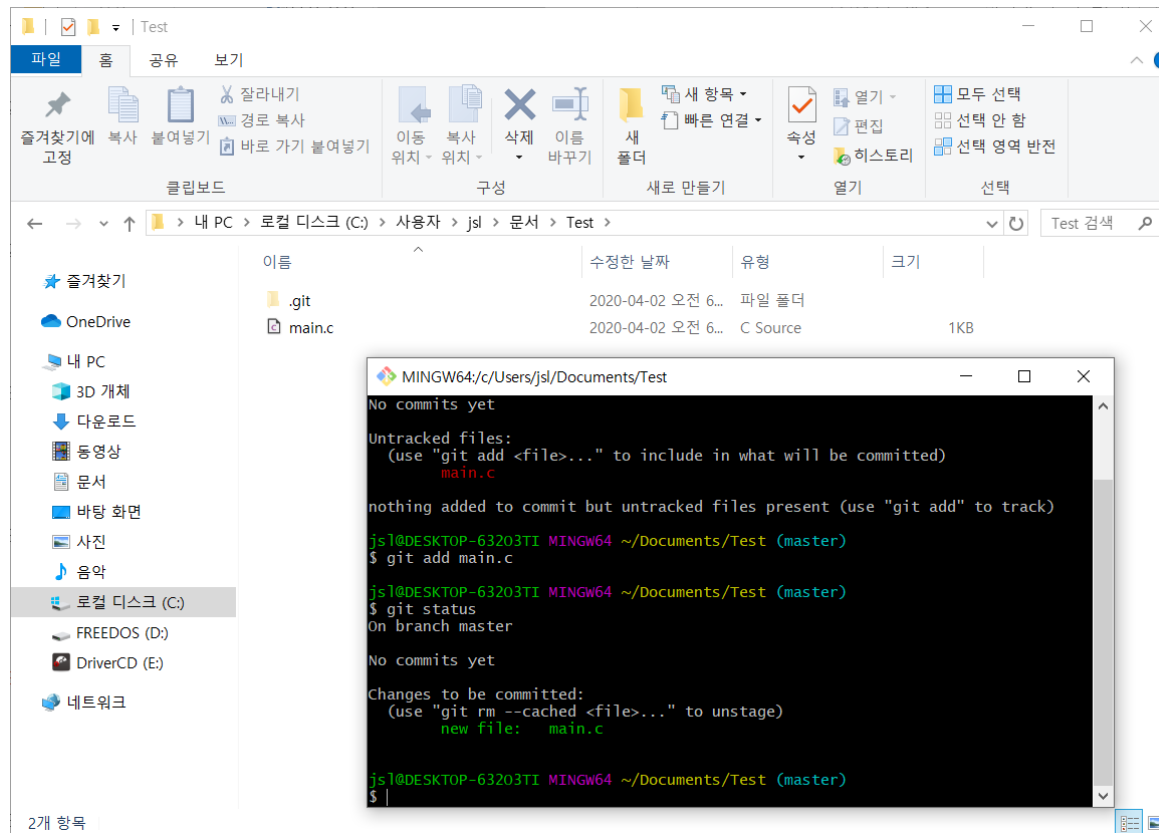
추가(add)

- 로컬 저장소가 생성된 곳에서 파일을 생성한다.
 - ◆ 소스코드(c, h 등), 리소스(txt, png 등) 등 모든 종류의 파일 가능
- git status 로 아직 반영되지 않은 파일(붉은 글씨)를 확인한다.



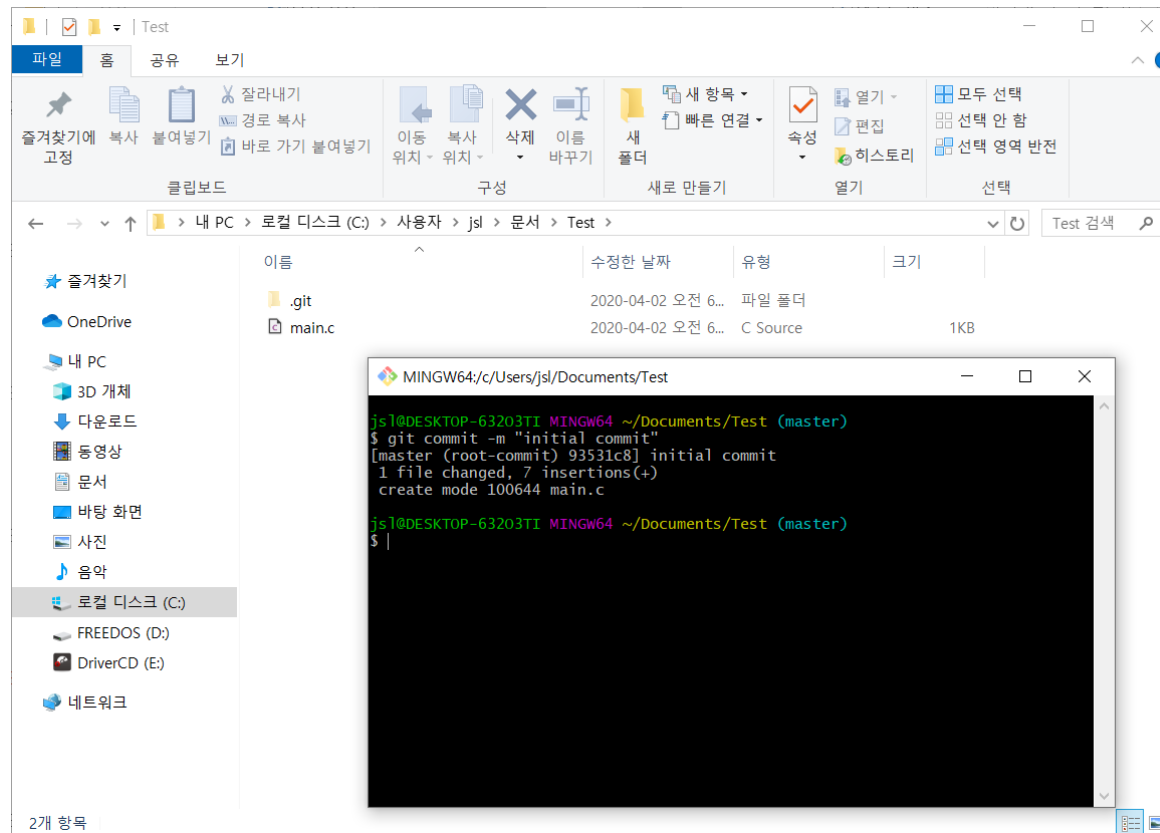
추가(add)

- git add <파일명>
 - ◆ 다수의 파일을 한꺼번에(모두) 추가할 경우 git add *
- git status 를 사용하여 전부 반영된 것을 확인



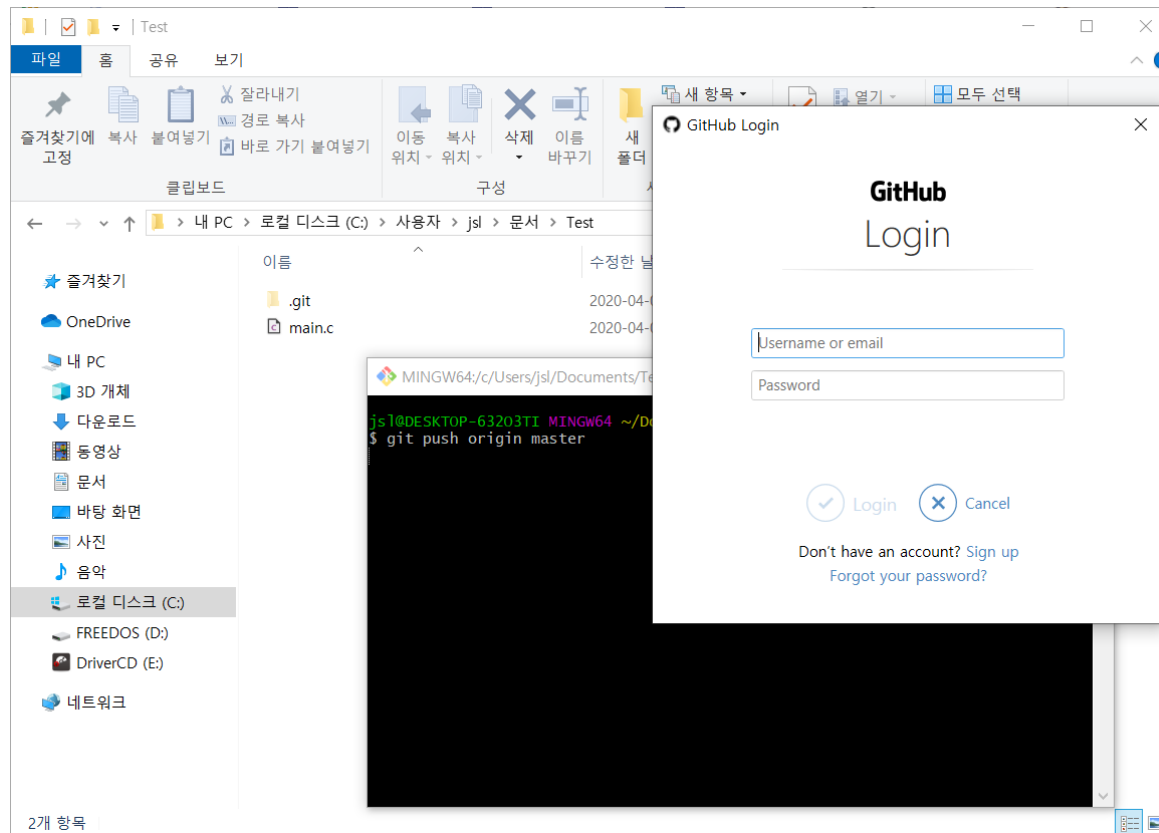
확정(commit)

- 변경된 내용을 저장소에 ‘새로운 버전’ 으로 확정시킨다.
 - ◆ Commit을 기준으로 버전이 관리된다. 과거에 commit한 시점으로 돌아갈 수 있다.
- `git commit -m “commit한 내용 설명”`
 - ◆ Ex) `git commit -m “initial commit”`



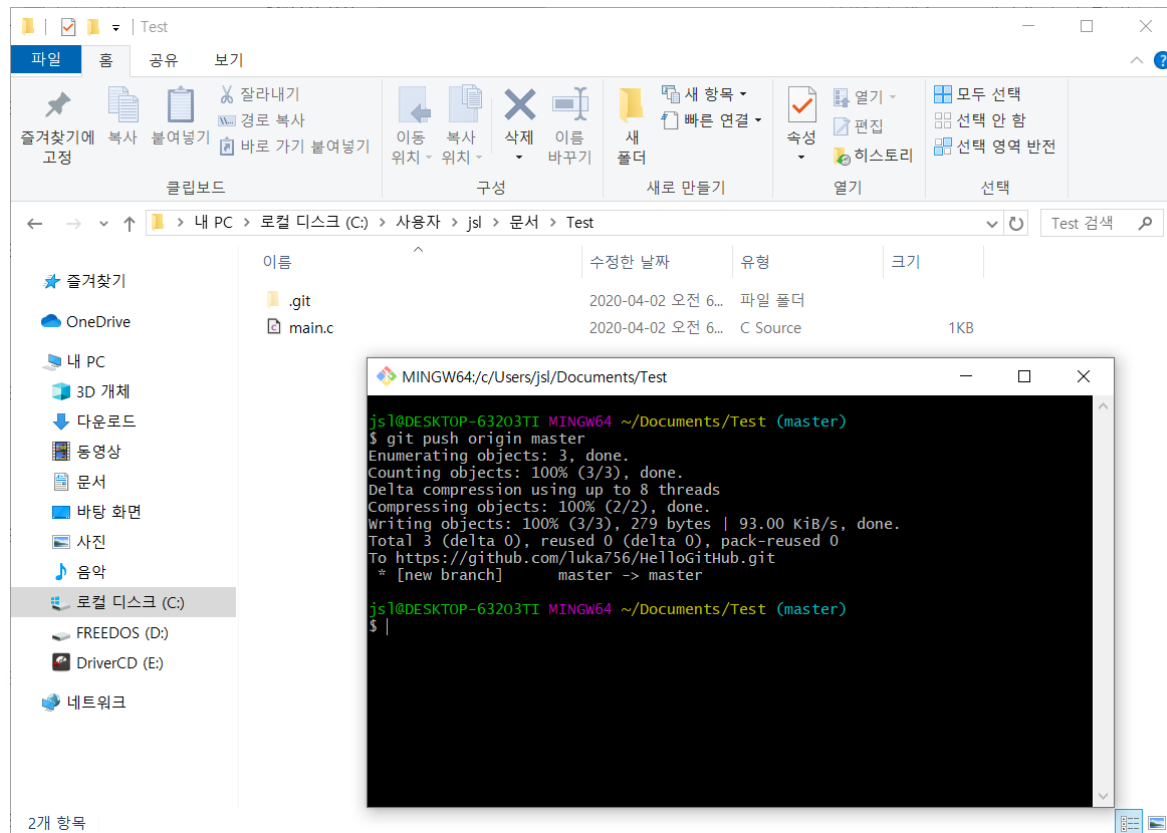
원격 저장소에 반영(push)

- Commit 한 내용을 원격 저장소(github)에 올린다.
- git push origin 'branch'
 - ◆ Ex) git push origin master. Branch에 대해서는 후에 설명한다.
 - ◆ 첫 사용 시 로그인 창이 뜬다.



원격 저장소에 반영(push)

- 로그인을 마친 후에는 로그와 함께 push가 완료된다.



원격 저장소에 반영(push)

- Github의 repository 에서 반영된 내용을 확인할 수 있다.

The screenshot shows the GitHub interface for a repository named 'HelloGitHub' by user 'luka756'. At the top, there are buttons for 'Unwatch' (1), 'Star' (0), and 'Fork' (0). Below this is a navigation bar with links for 'Code', 'Issues' (0), 'Pull requests' (0), 'Actions', 'Projects' (0), 'Wiki', 'Security', 'Insights', and 'Settings'. The 'Description' section has a text input field for a 'Short description of this repository' and a 'Website' input field for 'Website for this repository (optional)', with 'Save' and 'Cancel' buttons. Below the description is a 'Manage topics' section. A summary bar shows '1 commit', '1 branch', '0 packages', '0 releases', and '0 contributors'. Below this is a section for the 'master' branch, showing a 'New pull request' button and a 'Clone or download' button. The commit history shows a single commit 'jsl and jsl initial commit' with the latest commit hash '93531c8' and a timestamp of '36 minutes ago'. Below the commit history is a section for the 'main.c' file, showing an 'initial commit' with a timestamp of '36 minutes ago'. At the bottom, there is a prompt to 'Add a README' to help people understand the project.

luka756 / HelloGitHub

Unwatch 1 Star 0 Fork 0

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

Description Website

Short description of this repository Website for this repository (optional) Save or Cancel

Manage topics

1 commit 1 branch 0 packages 0 releases 0 contributors

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

jsl and jsl initial commit Latest commit 93531c8 36 minutes ago

main.c initial commit 36 minutes ago

Help people interested in this repository understand your project by adding a README. Add a README

원격 저장소에 반영(push)

- Github의 repository 에서 반영된 내용을 확인할 수 있다.

The screenshot shows the GitHub interface for a repository named 'HelloGitHub' by user 'luka756'. At the top, there are buttons for 'Unwatch' (1), 'Star' (0), and 'Fork' (0). Below this is a navigation bar with links for 'Code', 'Issues' (0), 'Pull requests' (0), 'Actions', 'Projects' (0), 'Wiki', 'Security', 'Insights', and 'Settings'. The 'Description' section has a text input field for a 'Short description of this repository' and a 'Website' input field for 'Website for this repository (optional)', with 'Save' and 'Cancel' buttons. Below the description is a 'Manage topics' section. A summary bar shows '1 commit', '1 branch', '0 packages', '0 releases', and '0 contributors'. Below this is a section for the 'master' branch, showing a 'New pull request' button and a 'Clone or download' button. The commit history shows a single commit 'jsl and jsl initial commit' with the latest commit hash '93531c8' and a timestamp of '36 minutes ago'. Below the commit history is a section for the 'main.c' file, showing an 'initial commit' with a timestamp of '36 minutes ago'. At the bottom, there is a prompt to 'Add a README' to help people understand the project.

luka756 / HelloGitHub

Unwatch 1 Star 0 Fork 0

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

Description Website

Short description of this repository Website for this repository (optional) Save or Cancel

Manage topics

1 commit 1 branch 0 packages 0 releases 0 contributors

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

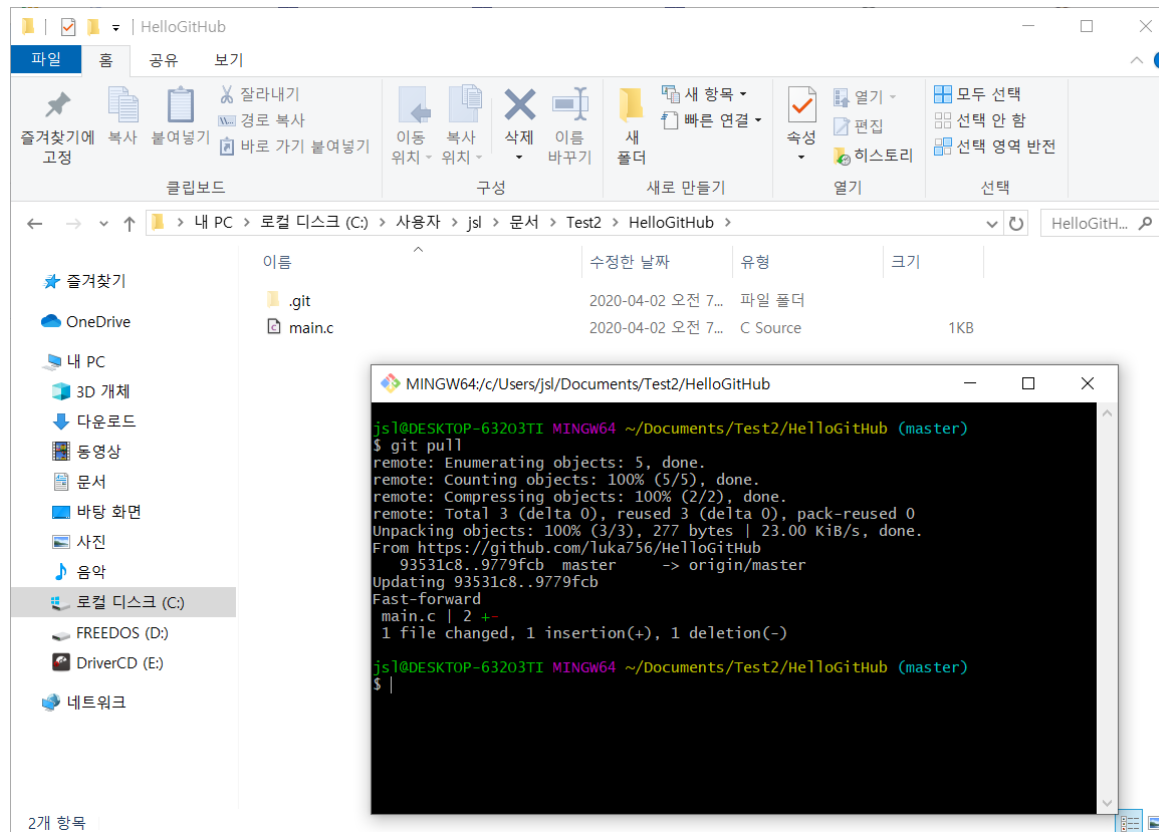
jsl and jsl initial commit Latest commit 93531c8 36 minutes ago

main.c initial commit 36 minutes ago

Help people interested in this repository understand your project by adding a README. Add a README

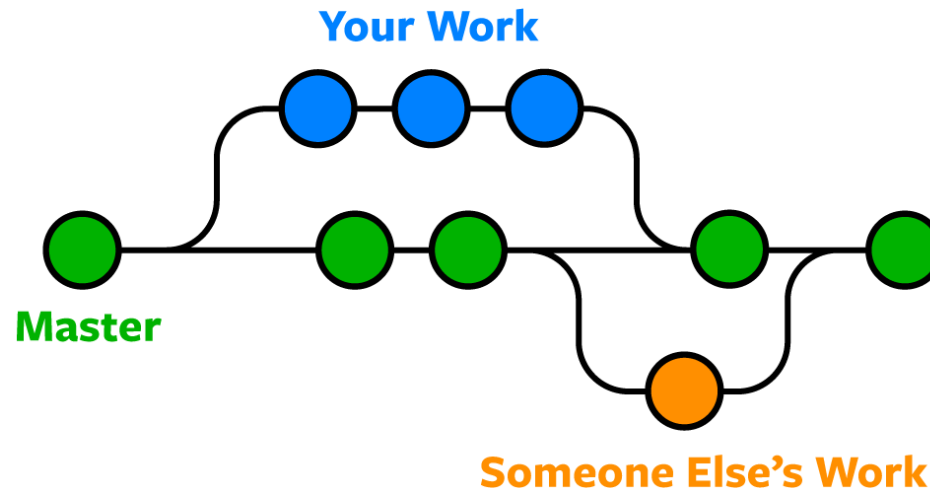
원격 저장소의 변경사항 적용(pull)

- 원격 저장소가 변경되었을 때, 해당 변경 사항을 로컬 저장소에 적용한다.
 - ◆ 변경 사항은 소스코드 수정, 리소스 변경, 파일 추가, 파일 삭제 등을 포함한다.
- 적용을 원하는 저장소(원격 저장소와 연결된)에서 사용한다.
- git pull



가지(branch)

- Git에서는 작업을 위해서 branch 라는 개념을 사용한다.
- 자료구조 변경 등 위험 부담이 있는 작업을 하기 전에, 원본을 보존하고 새로운 가지(branch)를 만들어 테스트해본 후 문제가 없으면(혹은 성능 하락이 없으면) 해당 변경 사항을 반영(merge)하는 작업이다.
- 이 작업은 여러 사용자가 협업하는 경우에 매우 유용하다.

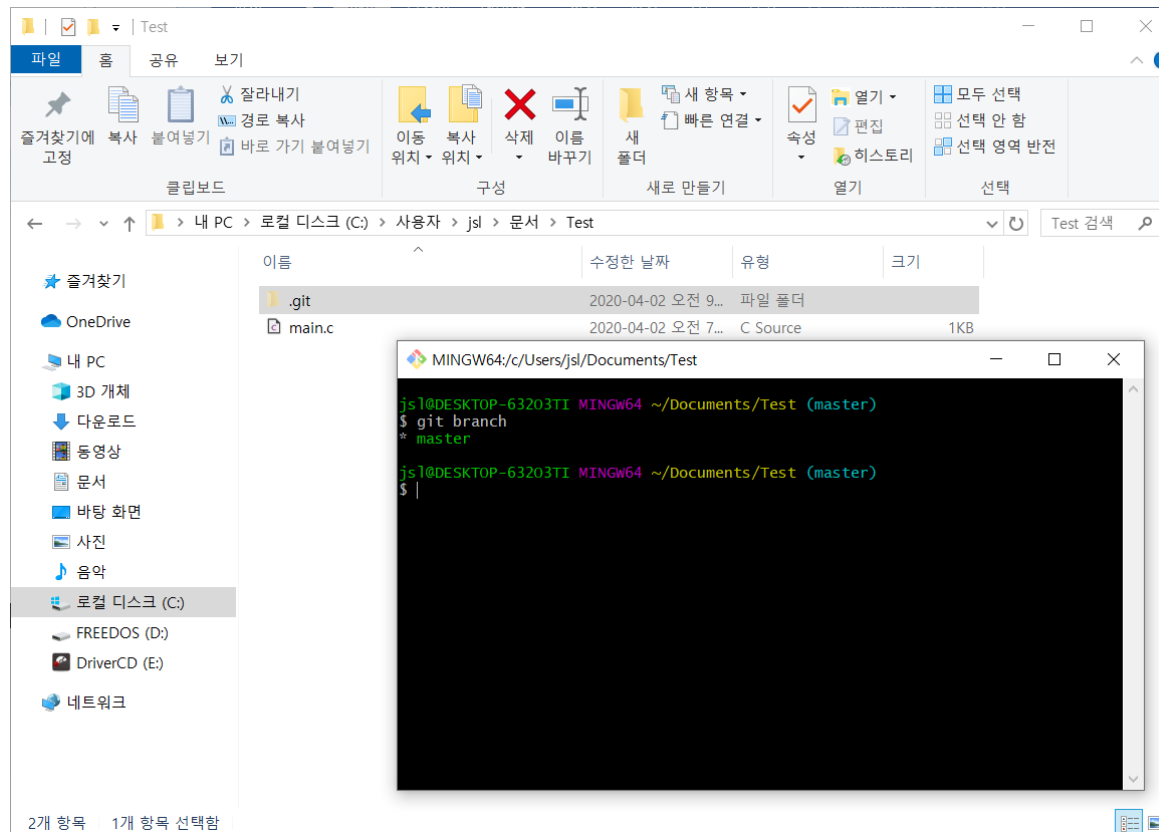


가지(branch)

- Branch는 생성과 변환이 자유롭다.
- 각 branch의 상태는 독립적이며, 로컬 저장소에서 branch를 전환하면 해당 branch의 상태로 파일이 변경된다.
- 이 기능을 이용해서 branch의 상태가 원하던 결과가 아닐 경우 이전의 branch로 돌아가서 변경된 내용을 전부 취소할 수 있다.

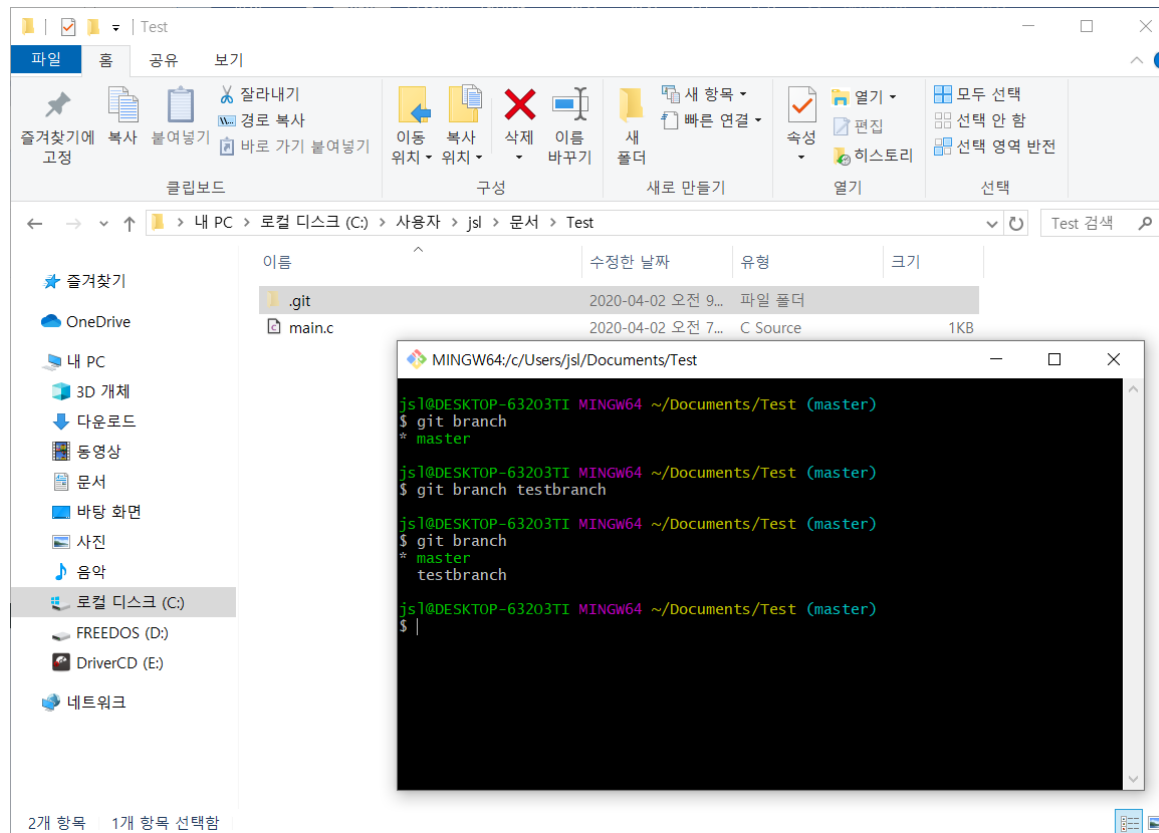
가지(branch)

- 저장소 생성시 master branch가 기본으로 생성된다.
- git branch 명령으로 현재 존재하는 branch의 목록을 확인할 수 있다.



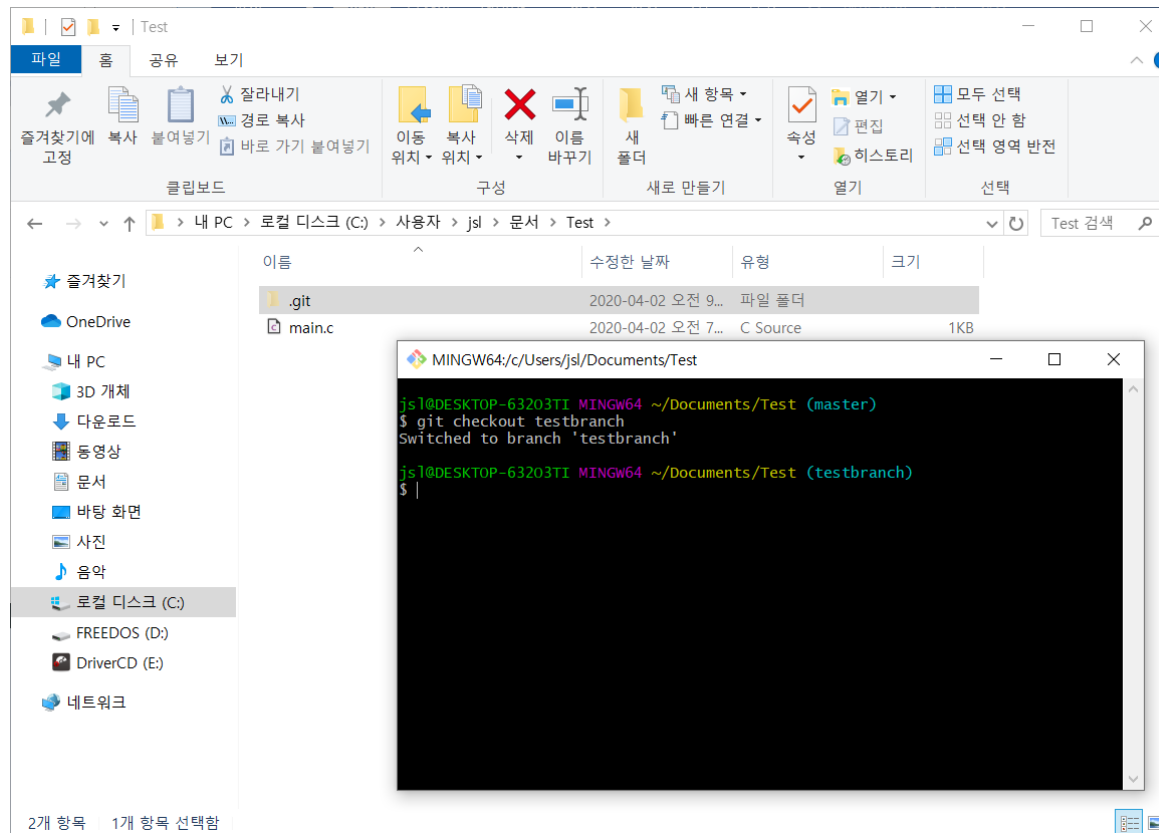
가지(branch)

- 새 branch 는 git branch <branchname> 으로 생성할 수 있다.
 - ◆ Ex) git branch testbranch



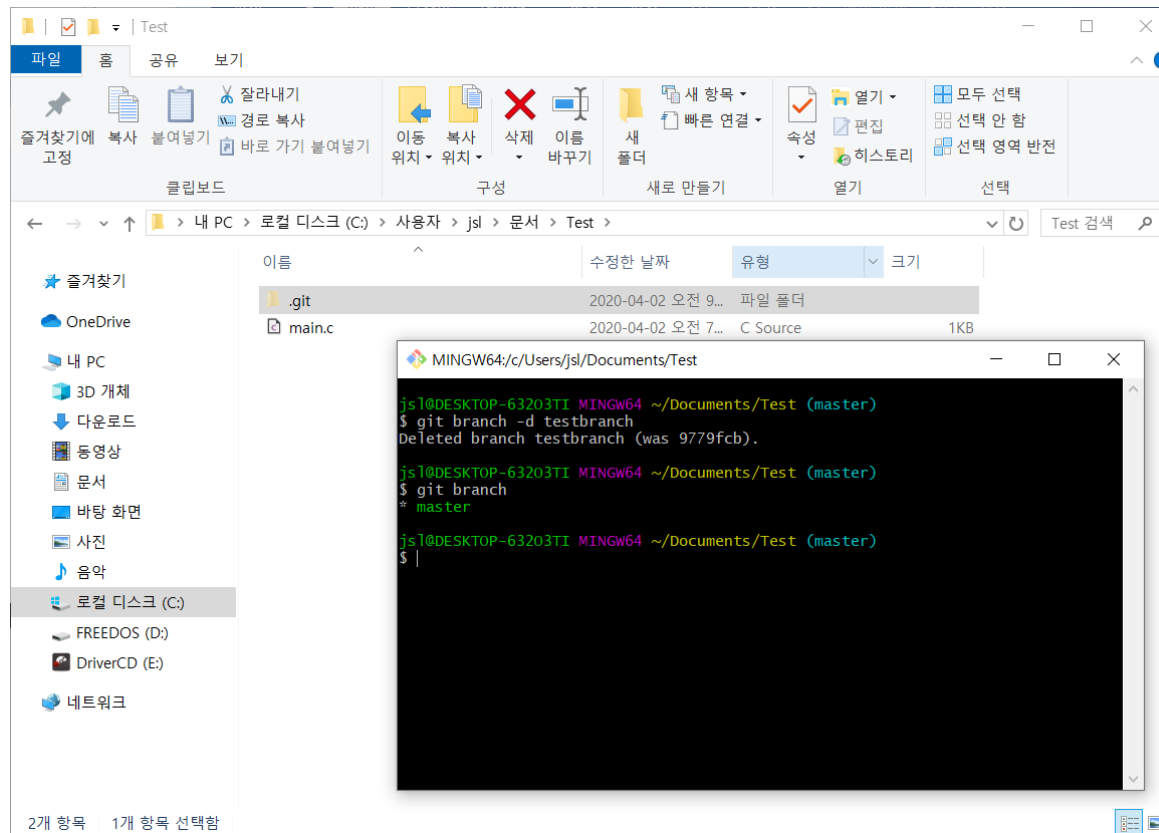
가지(branch)

- Branch간의 전환은 `git checkout <branchname>` 이다.
 - ◆ Ex) `git checkout testbranch`
 - ◆ `git checkout -b <branchname>`으로 생성+전환 가능



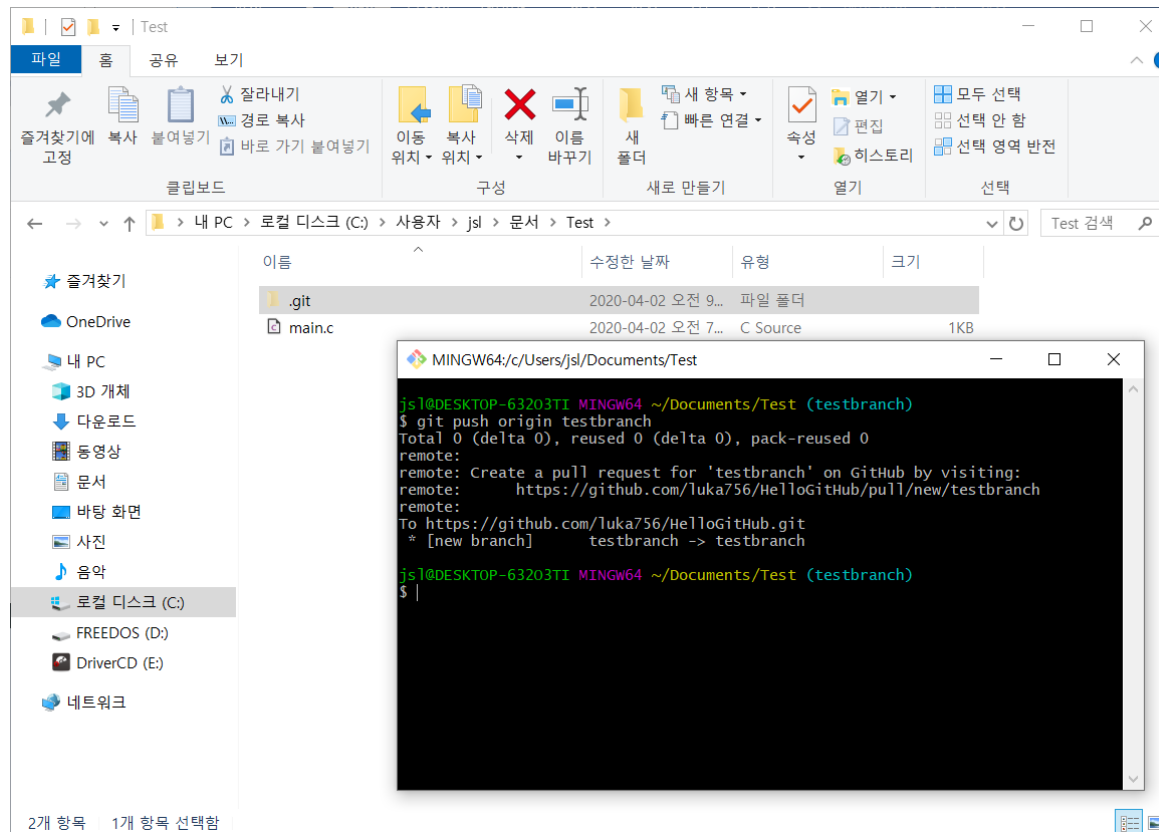
가지(branch)

- 더 이상 사용하지 않거나 필요없을 경우 branch를 삭제할 수 있다. `git branch -d <branchname>`
 - ◆ Ex) `git branch -d testbranch`



가지(branch)

- 새로 생성한 branch 이름으로 push할 수 있다.
 - ◆ Ex) git push origin testbranch



가지(branch)

- Github에서 branch의 생성을 확인할 수 있다.

luka756 / HelloGitHub

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

2 branches

0 packages

0 releases

0 contributors

Branch: master

New pull request

Create new file

Upload files

Find file

Clone or download

Switch branches/tags

Find or create a branch...

Branches

Tags

✓ master

default

testbranch

Latest commit 9779fcb 1 hour ago

Switch branches or tags

1 hour ago

Understand your project by adding a README.

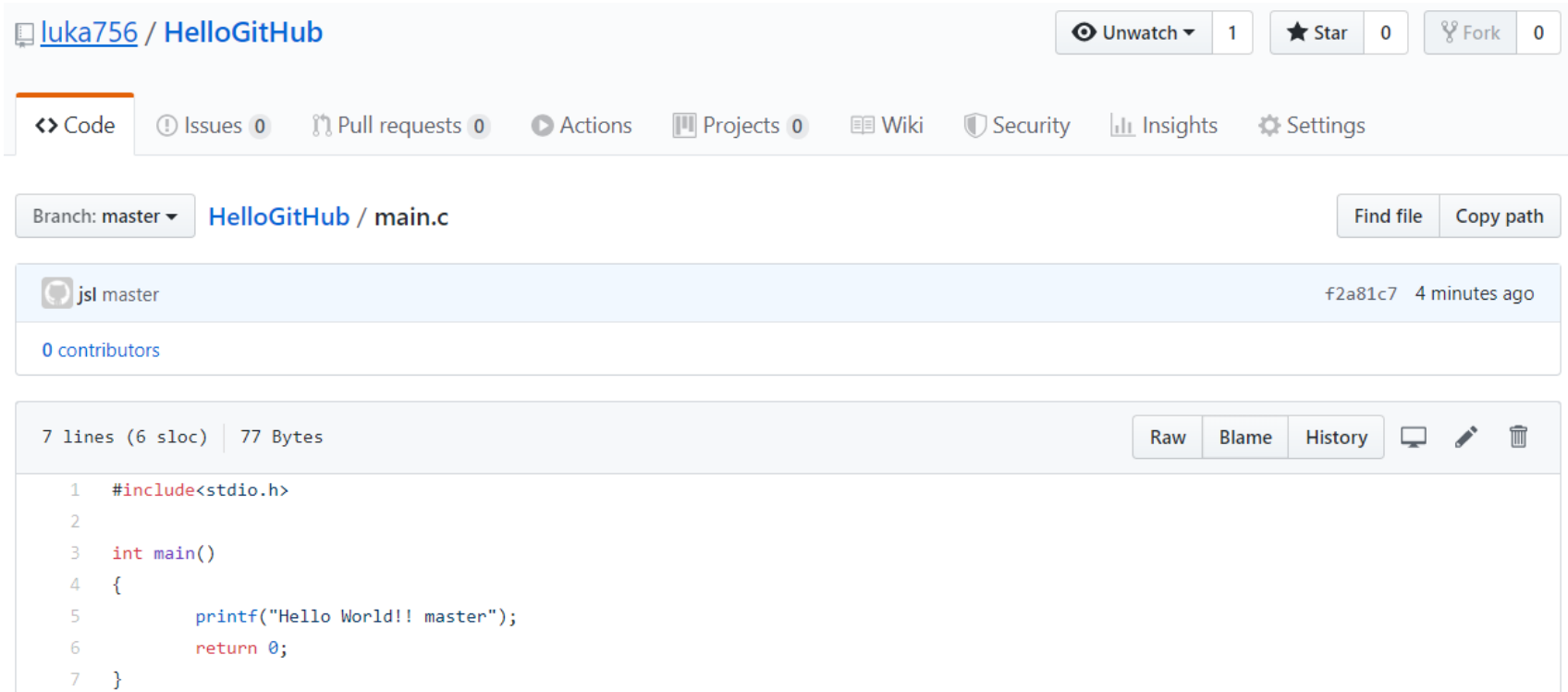
Add a README

병합(merge)

- 다른 가지의 변경 내용을 현재 가지에 병합할 수 있다.
 - ◆ `git merge <branchname>`
- 단 이때 동시에 적용될 수 없는 변경사항이 있을 때, 충돌(conflict)가 일어날 수 있다.

충돌(conflict)

- Master branch에서는 5번 줄을 master로 수정하였다.



The screenshot shows the GitHub interface for the repository 'luka756 / HelloGitHub'. At the top, there are buttons for 'Unwatch', 'Star', and 'Fork'. Below this is a navigation bar with links to 'Code', 'Issues', 'Pull requests', 'Actions', 'Projects', 'Wiki', 'Security', 'Insights', and 'Settings'. The main content area shows the 'main.c' file in the 'master' branch. A commit by 'jsl master' is displayed, with a commit hash of 'f2a81c7' and a timestamp of '4 minutes ago'. Below the commit information, the file content is shown, consisting of 7 lines of C code. The code includes a header file and a main function that prints 'Hello World!! master'.

```
1 #include<stdio.h>
2
3 int main()
4 {
5     printf("Hello World!! master");
6     return 0;
7 }
```

충돌(conflict)

- Testbranch에서도 5번 줄을 수정하였다.

luka756 / HelloGitHub

Unwatch 1 Star 0 Fork 0

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

Branch: testbranch HelloGitHub / main.c Find file Copy path

jsl testbranch a1d064e 5 minutes ago

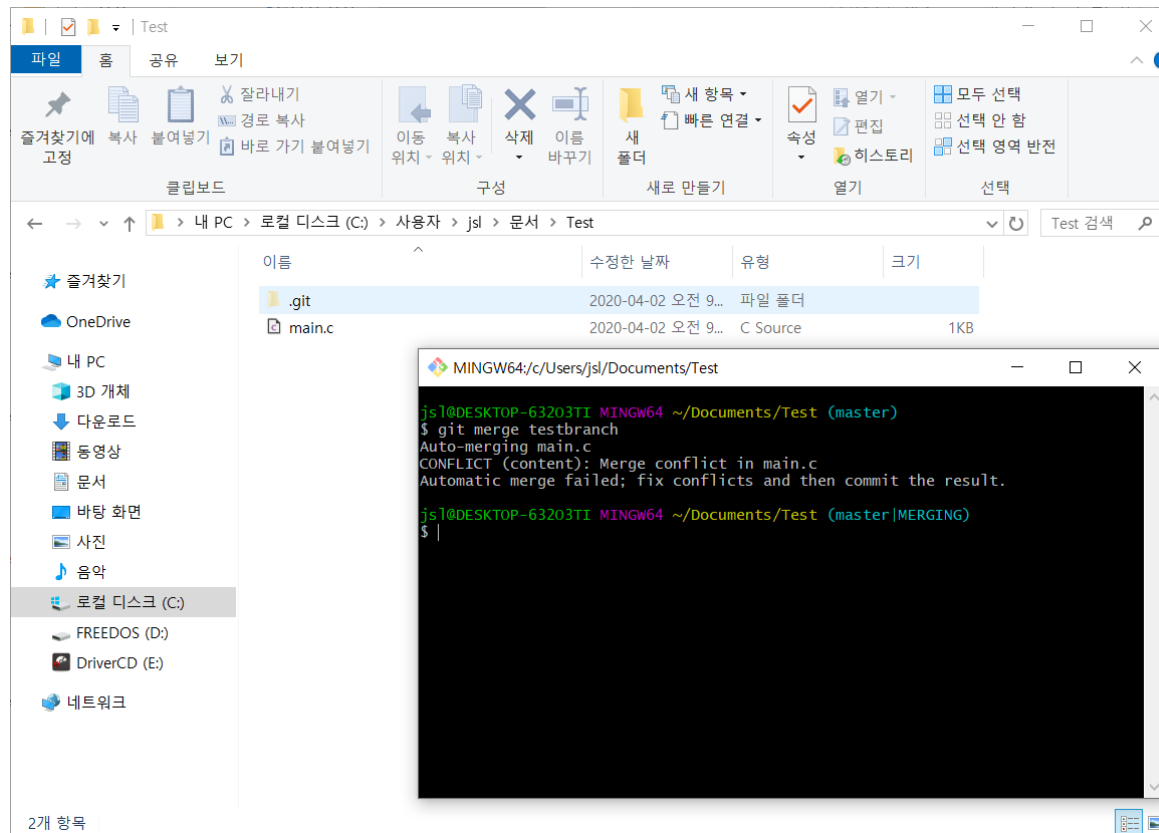
0 contributors

7 lines (6 sloc) 81 Bytes Raw Blame History

```
1 #include<stdio.h>
2
3 int main()
4 {
5     printf("Hello World!! testbranch");
6     return 0;
7 }
```

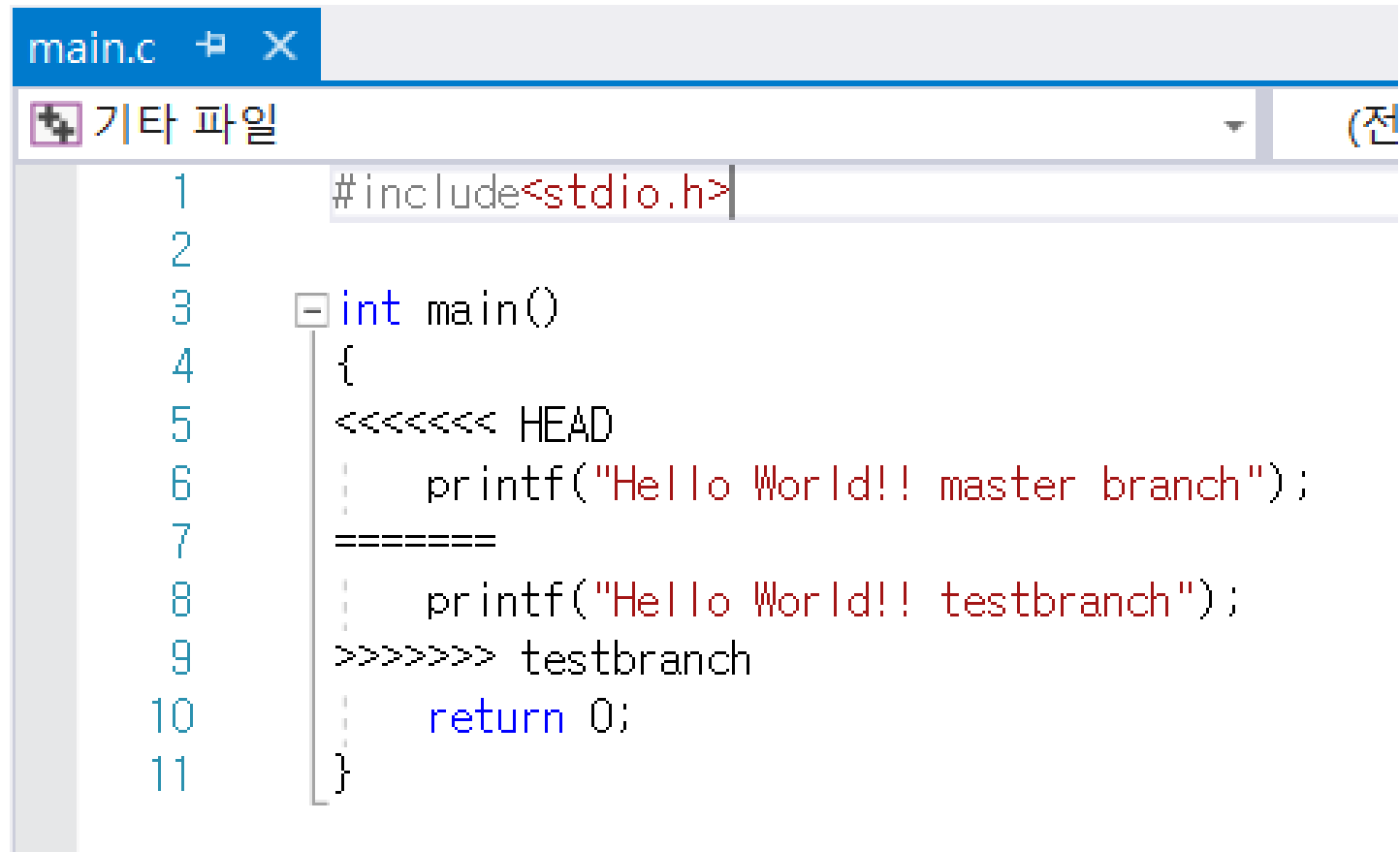
충돌(conflict)

- 충돌이 일어날 경우 아래와 같이 표시된다.
- 충돌이 일어난 파일을 열어 충돌이 없도록 수정해야 한다.



충돌(conflict)

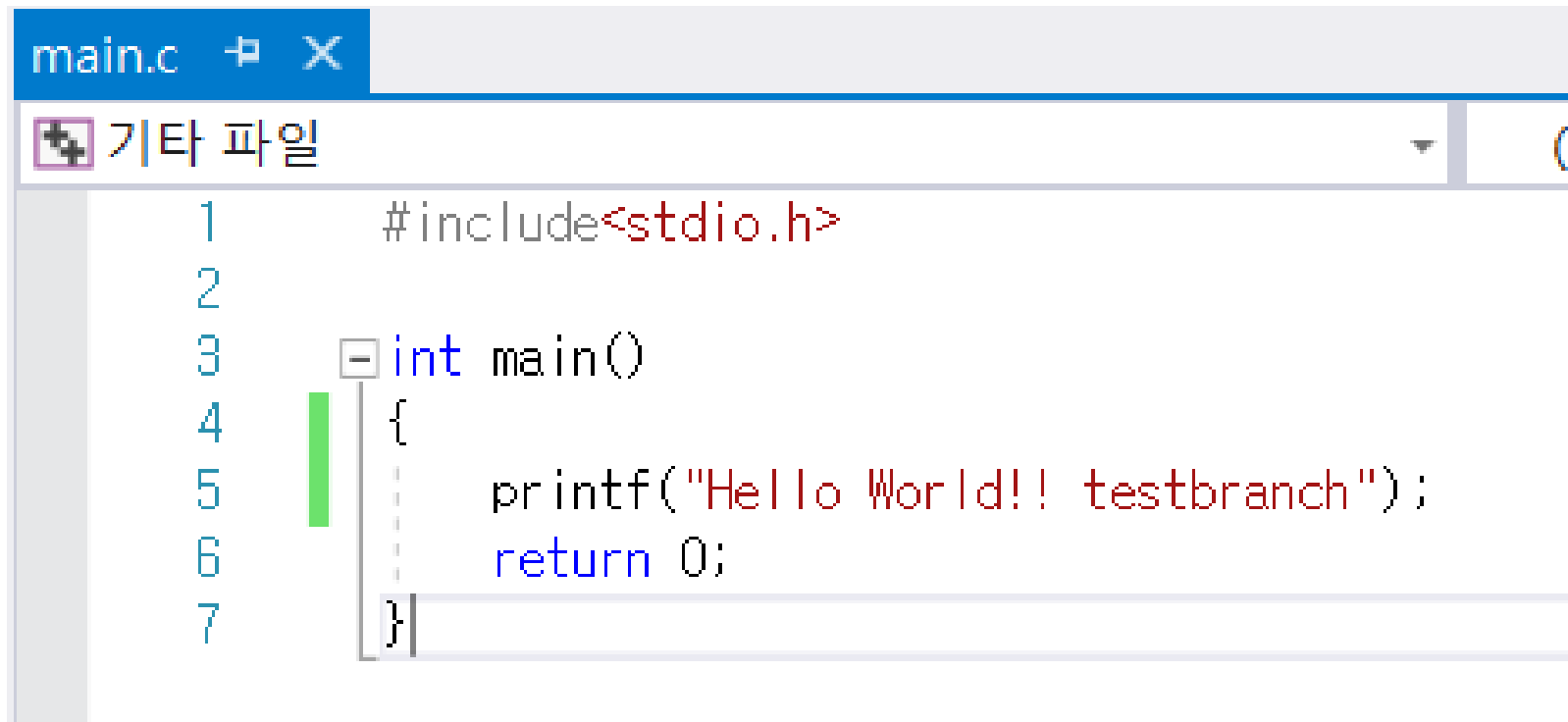
- 충돌이 발생한 파일은 아래와 같이 충돌 내용을 표시하여 준다.



```
main.c ✕
기타 파일 (전
1  #include<stdio.h>
2
3  int main()
4  {
5  <<<<<< HEAD
6  |     printf("Hello World!! master branch");
7  =====
8  |     printf("Hello World!! testbranch");
9  >>>>>> testbranch
10 |     return 0;
11 }
```

충돌(conflict)

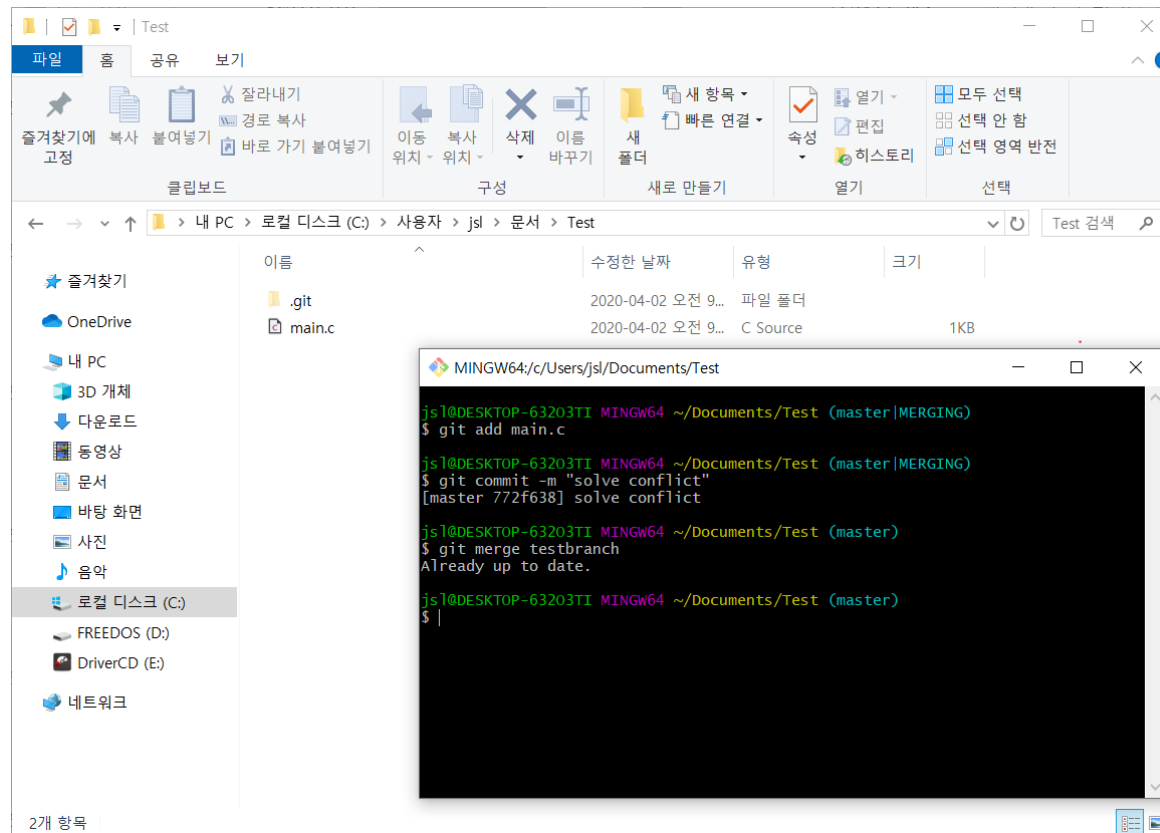
- 원하는 내용을 남기고 파일을 정리한다.



```
main.c  [icon] X
기타 파일
1      #include<stdio.h>
2
3      int main()
4      {
5          printf("Hello World!! testbranch");
6          return 0;
7      }
```

충돌(conflict)

- 수정한 파일을 add하고 commit 하면 병합이 완료된다.
- Branch 이름을 확인할 것.





5주차 실습 안내 **(Github 사용)**

5주차 예비 보고서

- Github 계정을 생성한 뒤 생성된 계정 화면(repository)을 촬영할 것.
- <https://github.com/luka756?tab=repositories> 와 같이 주소 첨부.
- Git을 설치한 뒤 CMD나 bash에서 git --version을 사용하여 설치된 것을 촬영할 것.

5주차 실습

- 저장소 생성, push
- 외부 저장소 가져오기

5주차 실습 1

- 지난 실습에서 작성하였던 소스코드가 저장된 폴더를 만든 뒤 해당 폴더에서 로컬 저장소를 생성하여라.
- 생성된 로컬 저장소를 github에 comsil_5_github 라는 이름의 원격 저장소에 push할 것.
- Commit message는 '실습 1'로 할 것.

5주차 실습 2

- 실습 1에서 작성한 원격 저장소로부터 다른 위치에 저장소를 복제하여라.
- 복제된 저장소에서 임의의 파일을 추가하고 push할 것.
- Commit message는 '실습 2'로 할 것

5주차 실습 제출물

- 아래 네 가지 화면을 캡처하여 제출할 것.
 - ◆ 실습 1에서 생성된 로컬 저장소
 - ◆ 실습 1에서 push 할 때의 git bash
 - ◆ 실습 2 에서 원격 저장소를 저장한 폴더
 - ◆ 실습 2에서 push 할 때의 git bash
- 생성된 원격 저장소의 주소

5주차 결과 보고서

- 앞의 실습 제출물들을 보고서에 포함할 것.
- 협업을 할 때 Github를 사용하는 것이 어떤 장점이 있는지 기술하여라.

참고자료

- <https://rogerdudler.github.io/git-guide/index.ko.html>
- https://backlog.com/git-tutorial/kr/intro/intro1_1.html
- <https://www.nobledesktop.com/learn/git>