경성대학교
KYUNGSUNG UNIVERSITY

깃과 친해지기

이름 : 전상훈
학번 : 20197750054

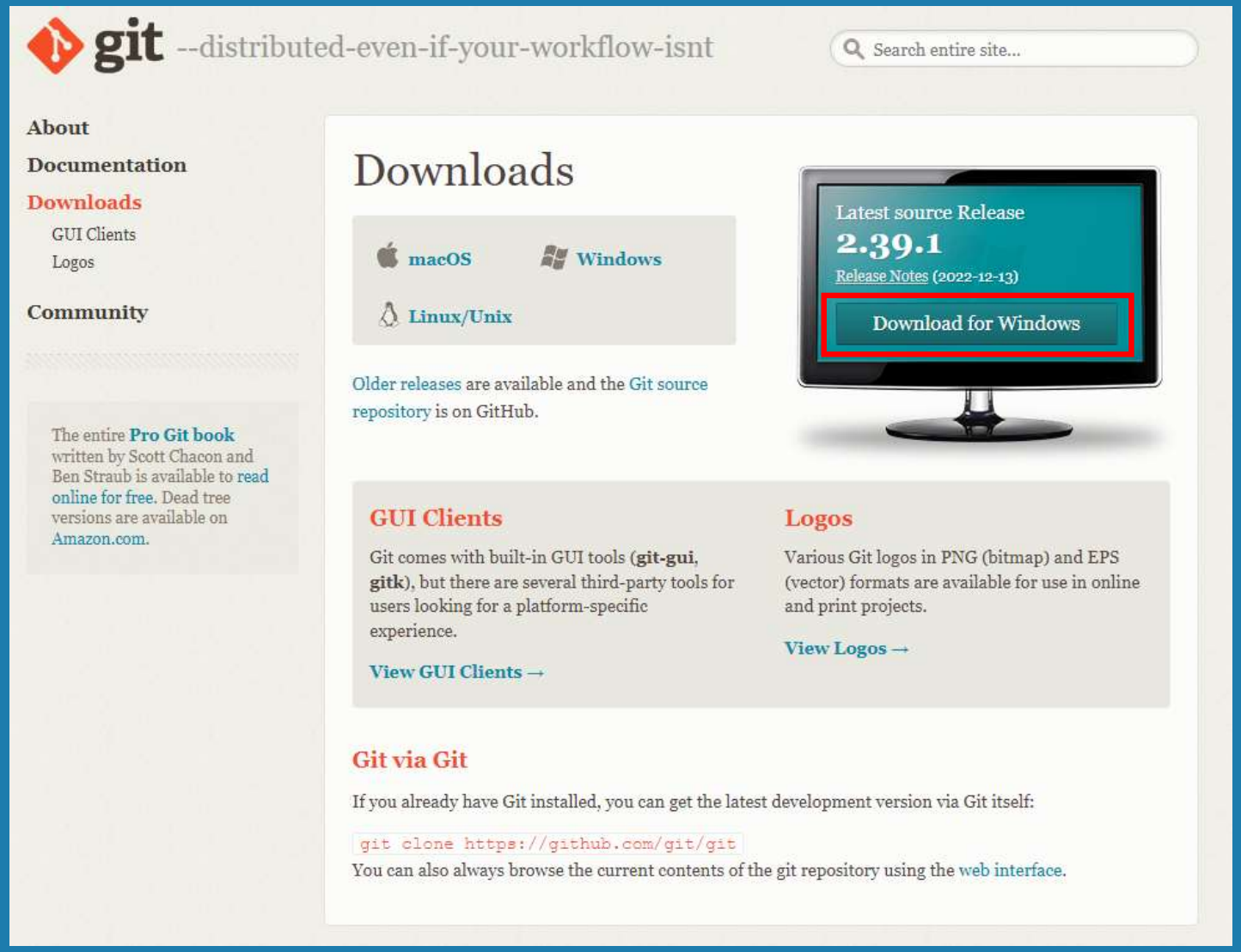
목차

Part 1,

깃 설치

깃 설치

https://git-scm.com/downloads



깃 설치

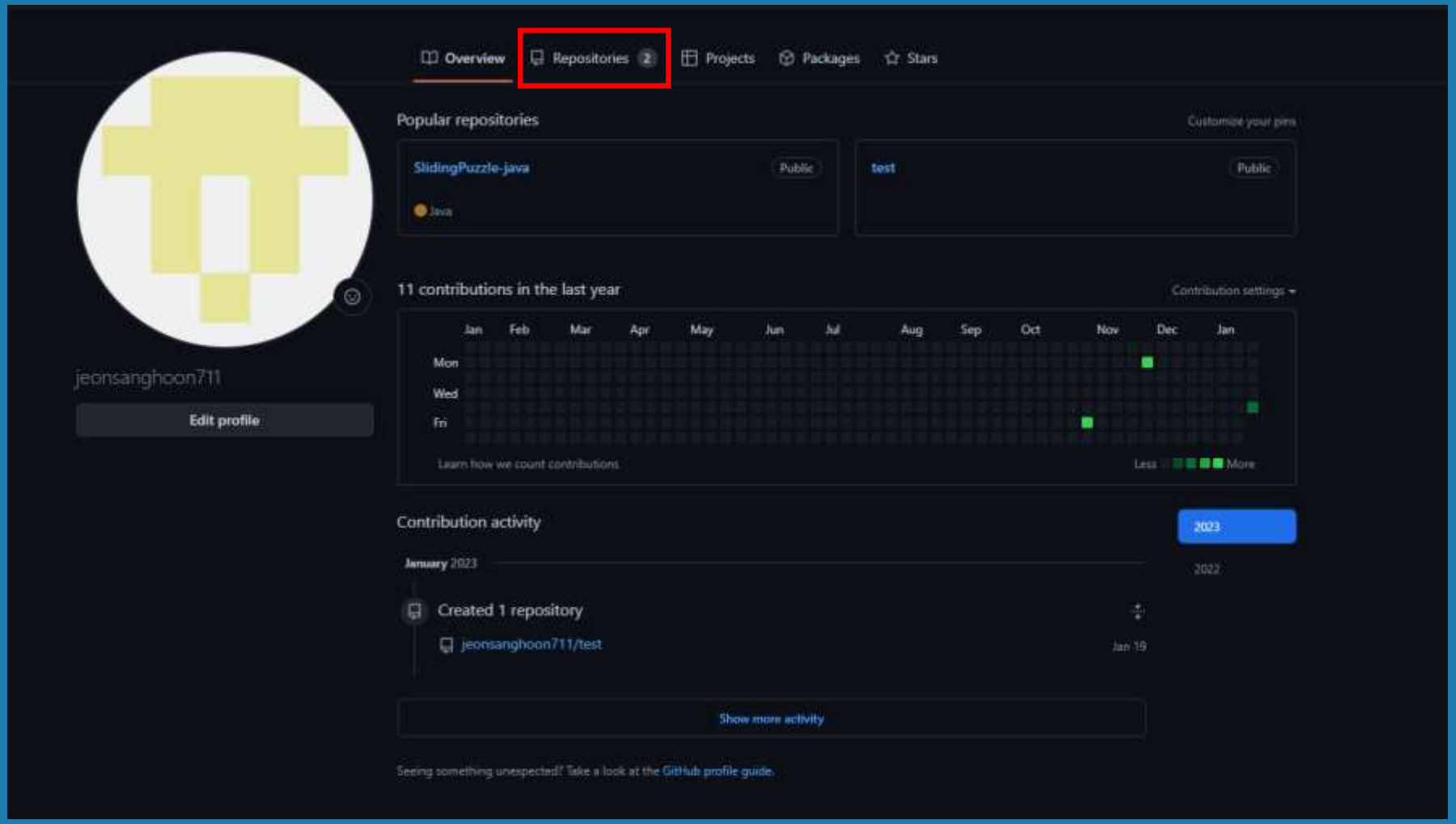
깃 설치 확인

```
C:₩Users₩User>git
usage: git [-v | --version] [-h | --help] [-C <path>] [-c <name>=<value>]
           [--exec-path[=<path>]] [--html-path] [--man-path] [--info-path]
           [-p | --paginate | -P | --no-pager] [--no-replace-objects] [--bare]
           [--git-dir=<path>] [--work-tree=<path>] [--namespace=<name>]
           [--super-prefix=<path>] [--config-env=<name>=<envvar>]
           <command> [<args>]

These are common Git commands used in various situations:

start a working area (see also: git help tutorial)
   clone     Clone a repository into a new directory
   init      Create an empty Git repository or reinitialize an existing one

work on the current change (see also: git help everyday)
   add       Add file contents to the index
   mv        Move or rename a file, a directory, or a symlink
```

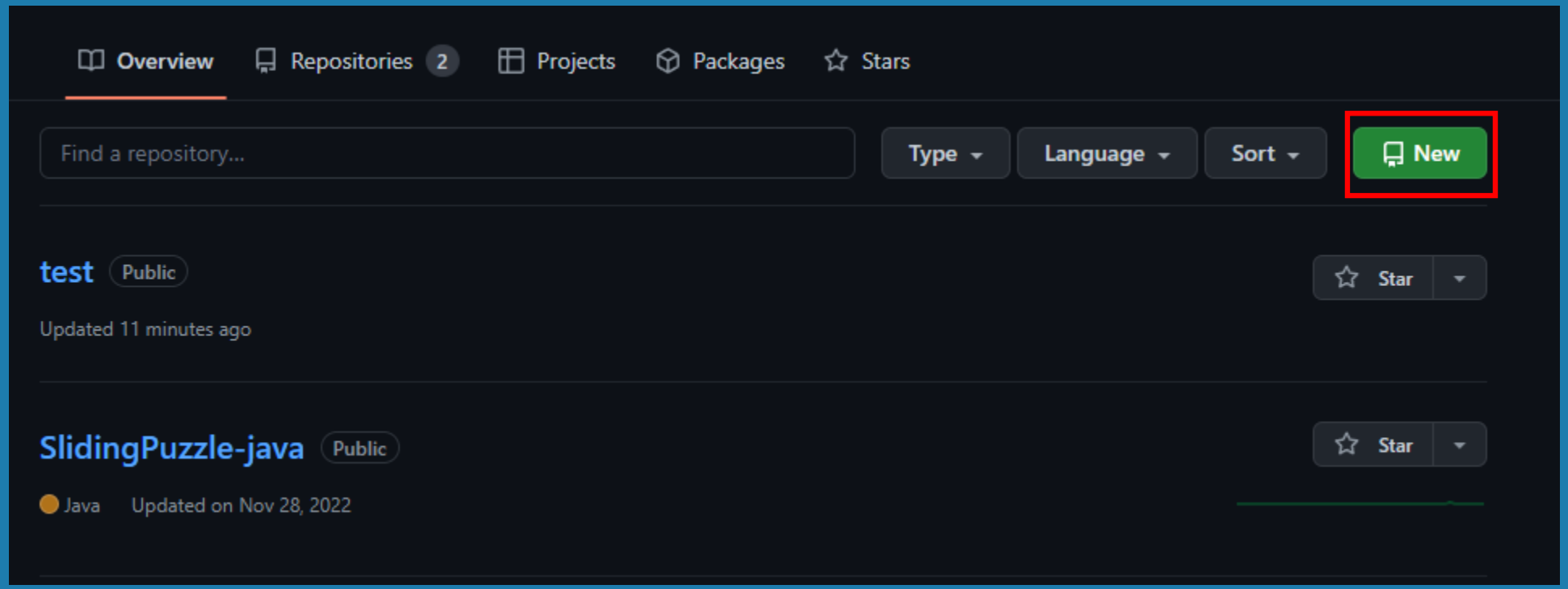
Part 2,

깃으로 파일 관리하기

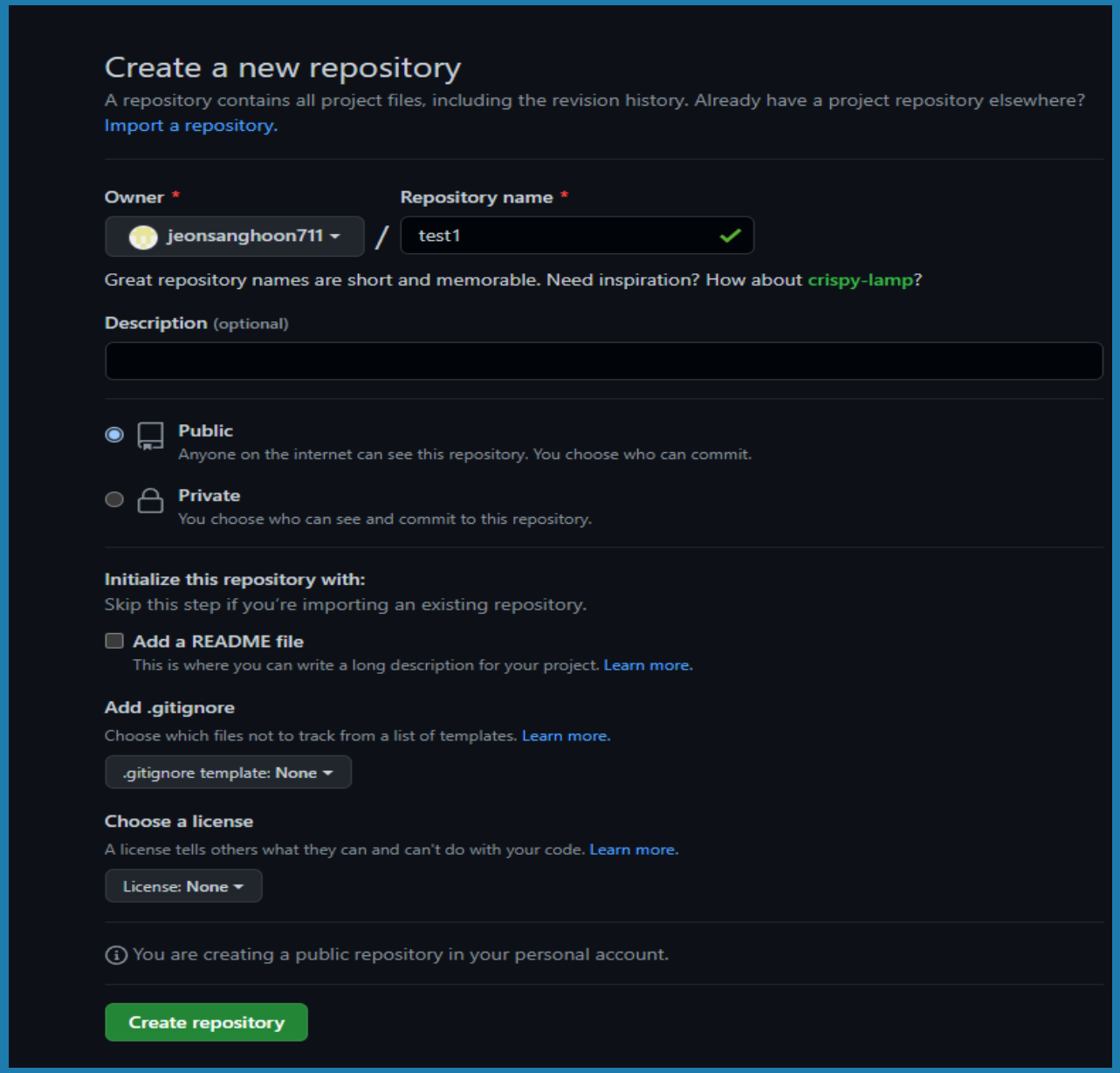
Git hub 접속



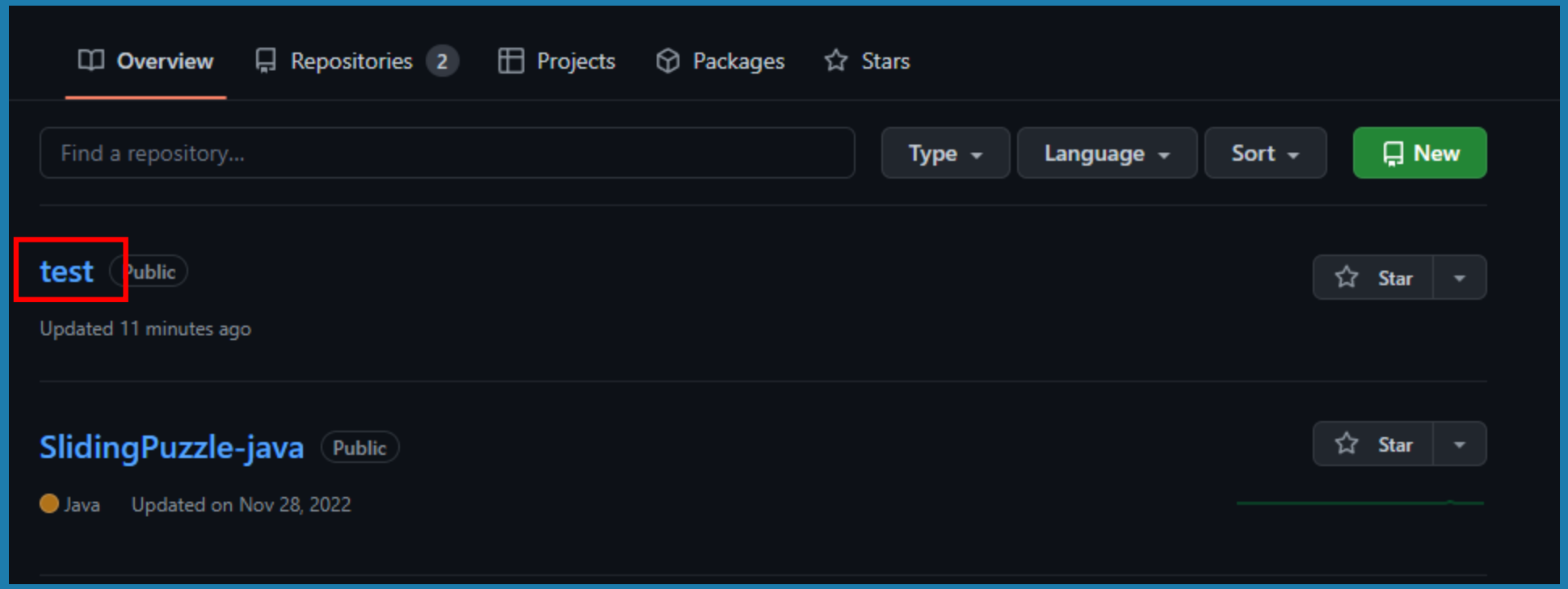
Git hub 접속



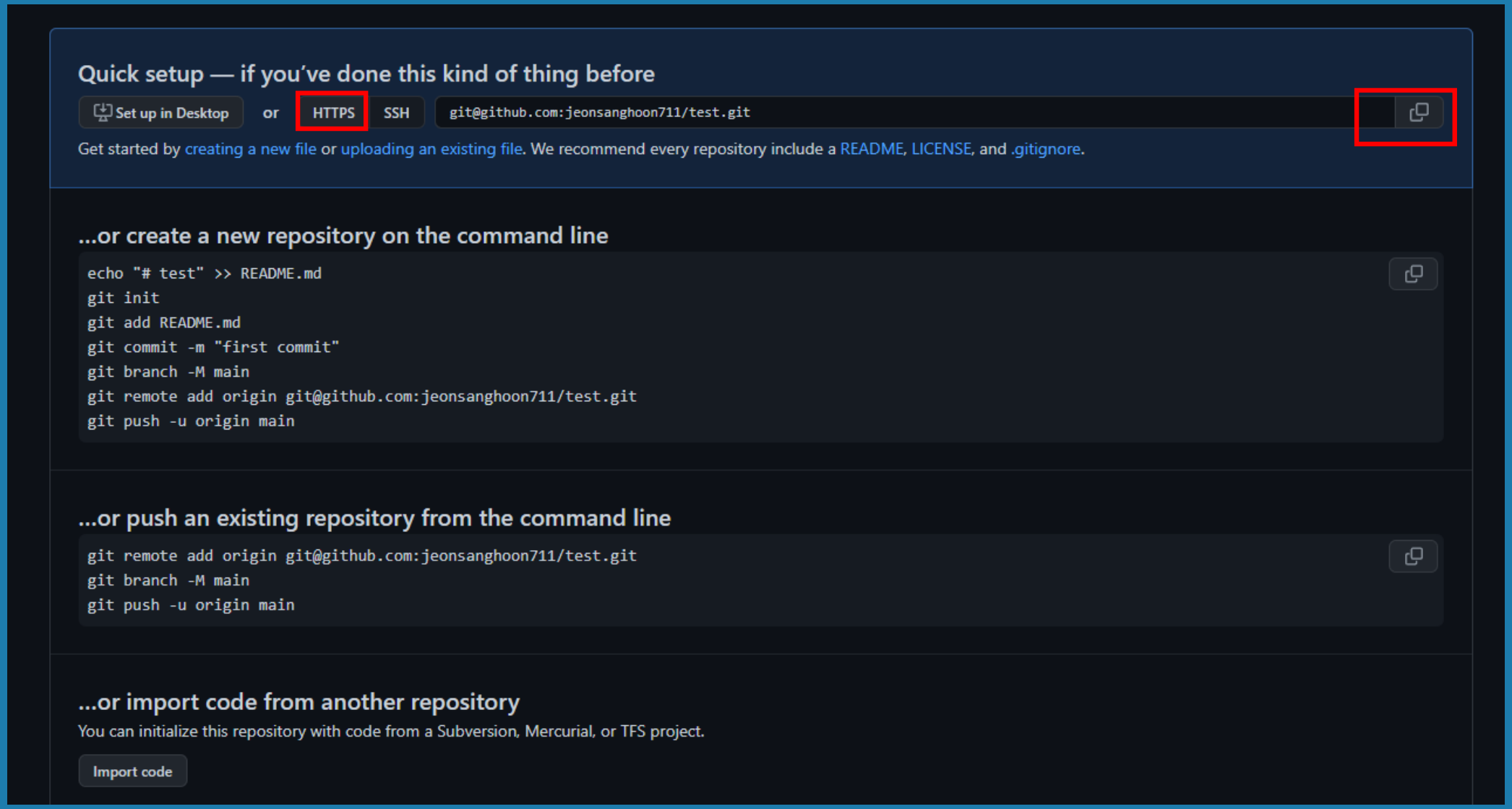
Git hub 접속



Git hub 접속



Git hub 접속



Git bash 실행

깃 복사 단축키 : Crtl + Ins
깃 붙여넣기 : Shift + Ins

```
git config -global init.defaultbranch main
```

```
git config --global user.name "사용자이름"
```

```
User@DESKTOP-M96ORTM MINGW64 ~
$ git config --global user.name "사 용 자 이 름 "
```

```
git config --global user.email "사용자이메일@...com"
```

```
User@DESKTOP-M96ORTM MINGW64 ~
$ git config --global user.email "사 용 자 이 메 일 @...com"
```

```
git config --list
```

```
User@DESKTOP-M96ORTM MINGW64 ~
$ git config --list
diff.astextplain.textconv=astextplain
filter.lfs.clean=git-lfs clean -- %f
filter.lfs.smudge=git-lfs smudge -- %f
filter.lfs.process=git-lfs filter-process
filter.lfs.required=true
http.sslbackend=openssl
http.sslcainfo=C:/Program Files/Git/mingw64/ssl/certs/ca-bundle.crt
core.autocrlf=true
core.fscache=true
core.symlinks=false
pull.rebase=false
credential.helper=manager-core
credential.https://dev.azure.com.usehttppath=true
init.defaultbranch=master
user.email=wjstkdgns711@naver.com
user.name=JeonSangHoon711
```

Git bash 실행

git init

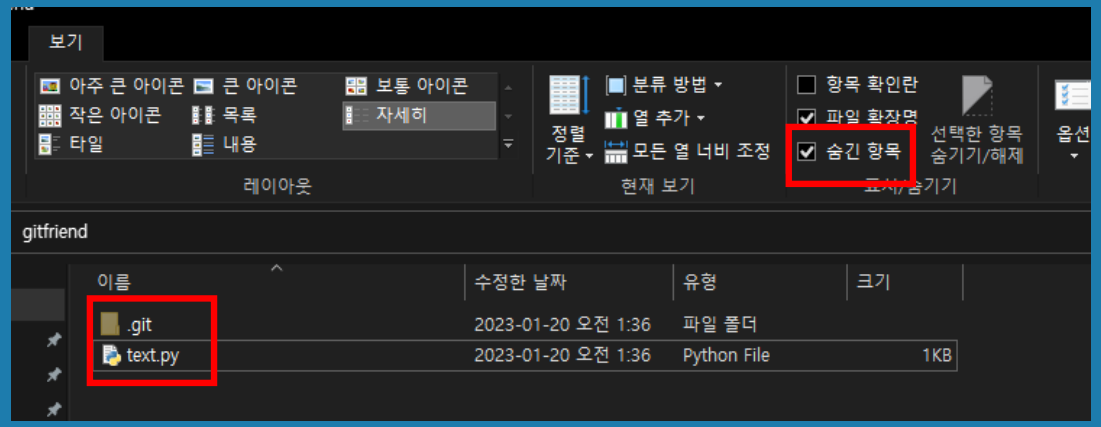
```
User@DESKTOP-M96ORTM MINGW64 ~/Desktop/gitfriend
$ git init
Initialized empty Git repository in C:/Users/User/Desktop/gitfriend/.git/
```

git remote add <원격저장소명> <url>

```
User@DESKTOP-M96ORTM MINGW64 ~/Desktop/gitfriend (master)
$ git remote add test https://github.com/jeonsanghoon711/test.git
```

git remote -v

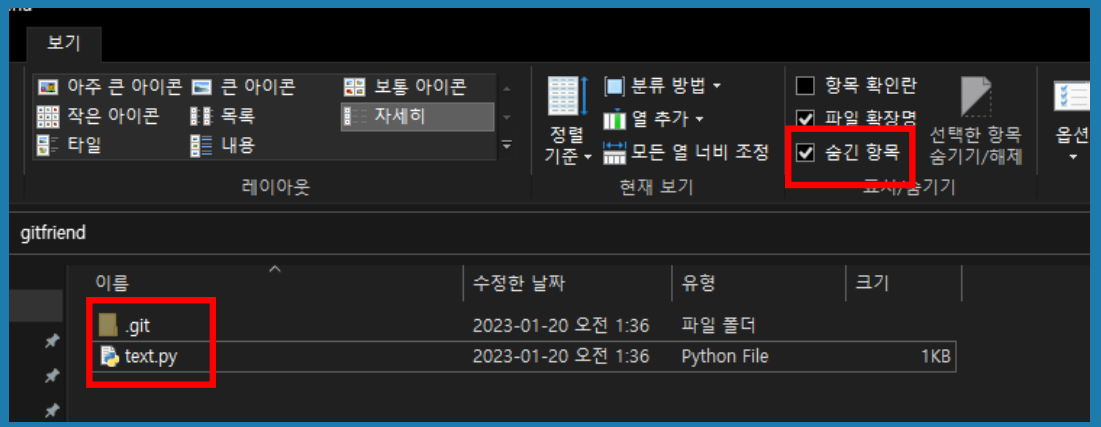
```
User@DESKTOP-M96ORTM MINGW64 ~/Desktop/gitfriend (master)
$ git remote -v
test    https://github.com/jeonsanghoon711/test.git (fetch)
test    https://github.com/jeonsanghoon711/test.git (push)
```



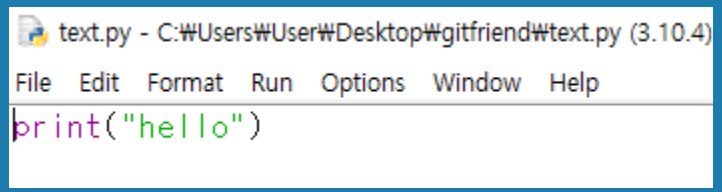
테스트용 파이썬 파일 생성

Git Push

보기 – 숨김 항목 체크 – .git 파일 확인



테스트용 파이썬 파일 생성



git add “파일이름”

```
User@DESKTOP-M96ORTM MINGW64 ~/Desktop/gitfriend (master)
$ git add text.py
```

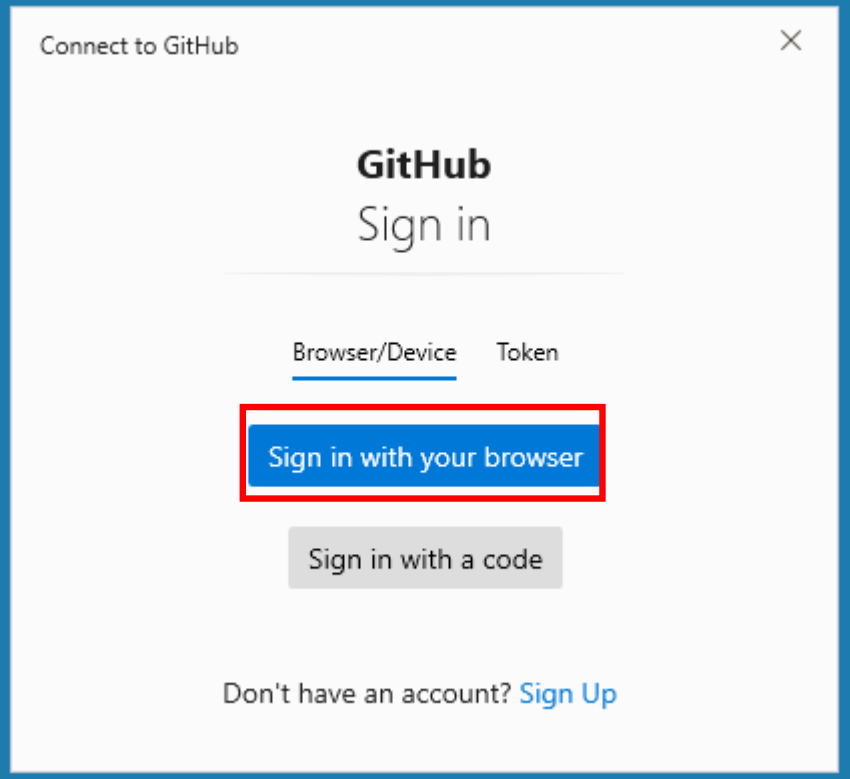
git commit –m “내용”

```
User@DESKTOP-M96ORTM MINGW64 ~/Desktop/gitfriend (master)
$ git commit -m "test"
[master (root-commit) 05f3acd] test
 1 file changed, 1 insertion(+)
 create mode 100644 text.py
```

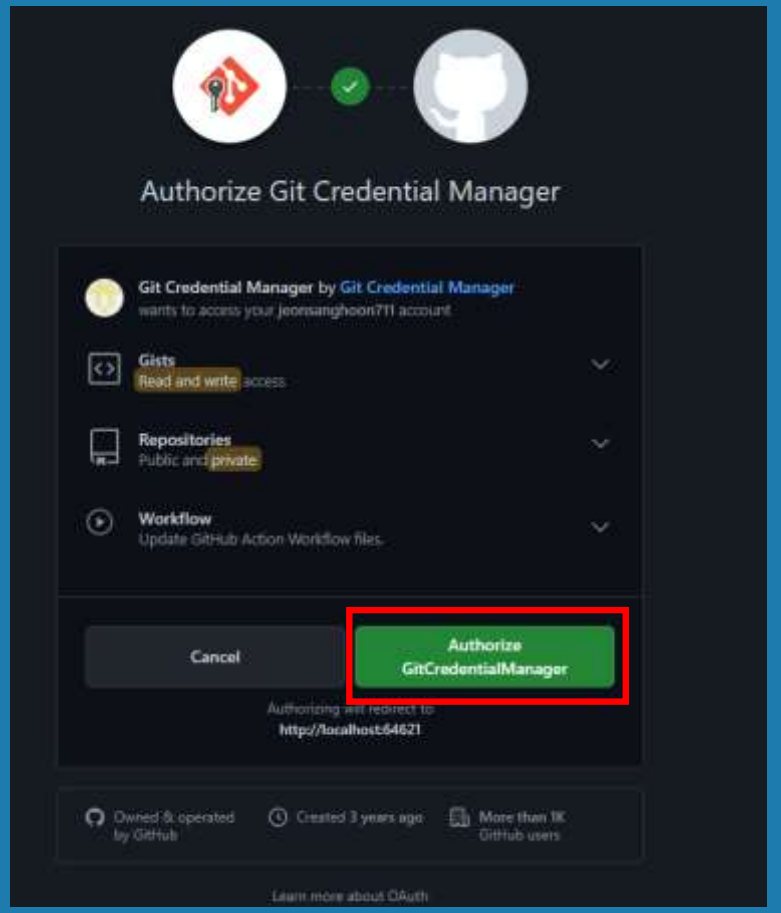
Git Push

`git push <원격저장소명> <브랜치명>`

```
User@DESKTOP-M960RTM MINGW64 ~/Desktop/friendgit (main)
$ git push test main
```

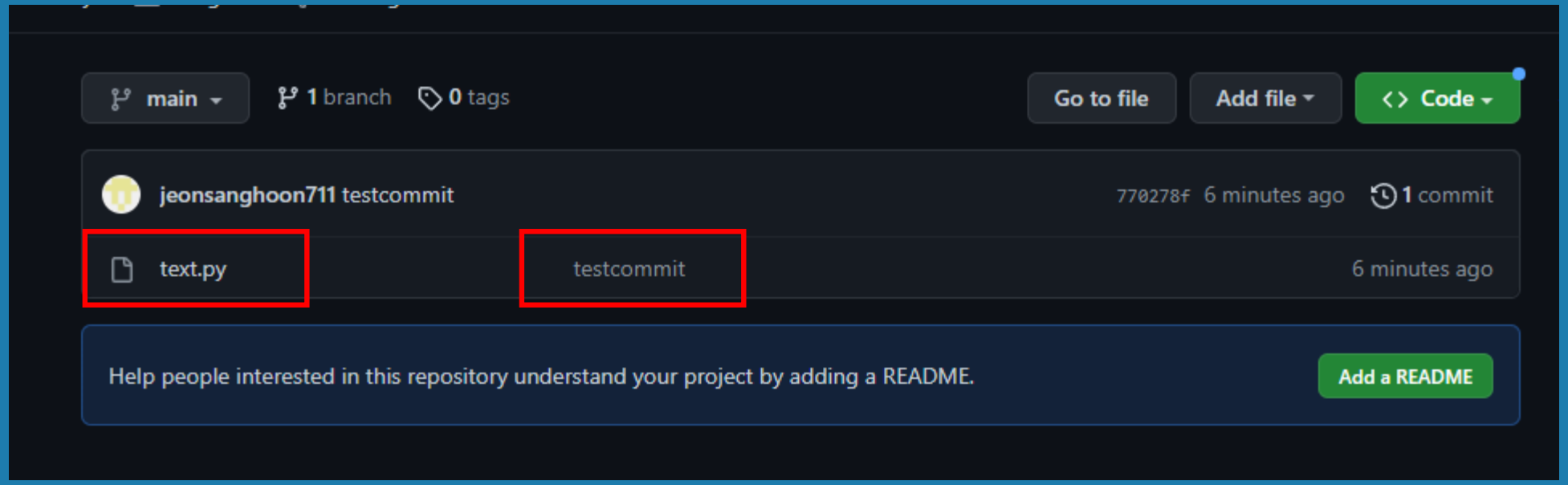


로그인



Git Push

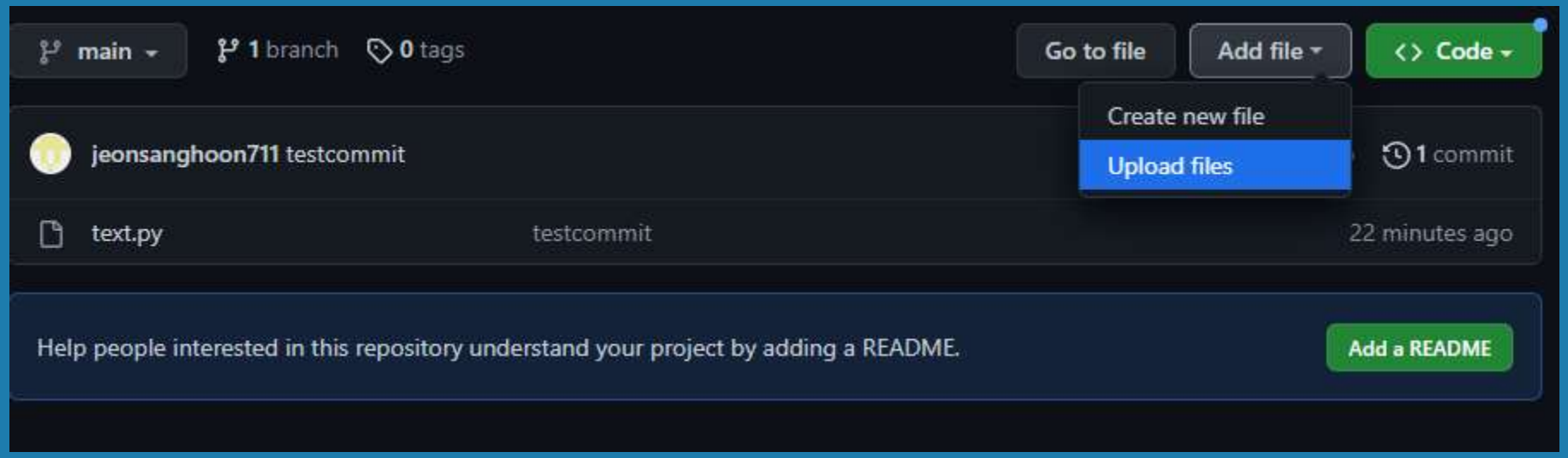
파일 업로드 확인



Git Pull

바탕화면에 테스트용 파일 생성

깃허브 업로드

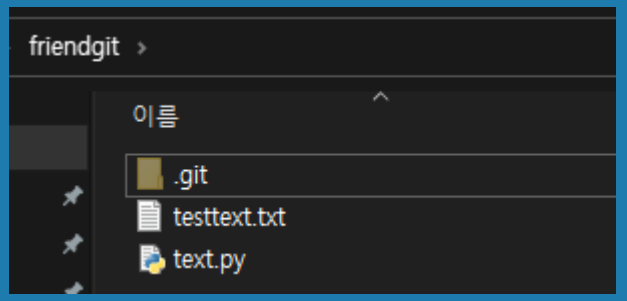


Git Pull

git pull <원격저장소명> <브랜치명>

```
User@DESKTOP-M960RTM MINGW64 ~/Desktop/friendgit (main)
$ git pull test main
remote: Enumerating objects: 4, done.
remote: Counting objects: 100% (4/4), done.
remote: Compressing objects: 100% (2/2), done.
remote: Total 3 (delta 0), reused 0 (delta 0), pack-reused 0
Unpacking objects: 100% (3/3), 676 bytes | 135.00 KiB/s, done.
From https://github.com/jeonsanghoon711/test
 * branch            main       -> FETCH_HEAD
   770278f..4d736ec  main       -> test/main
Updating 770278f..4d736ec
Fast-forward
 testtext.txt | 1 +
 1 file changed, 1 insertion(+)
 create mode 100644 testtext.txt
```

디렉토리 파일 확인

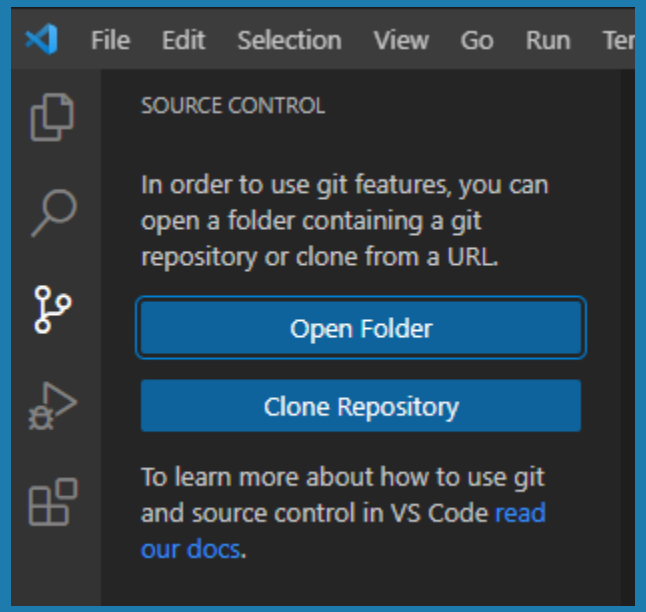


Part 3,

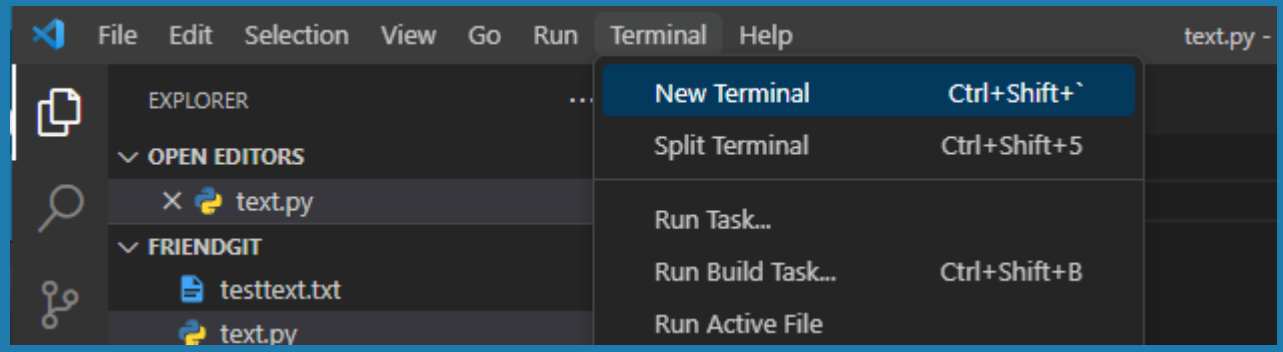
VS코드와 깃 연동

Vs코드와 깃

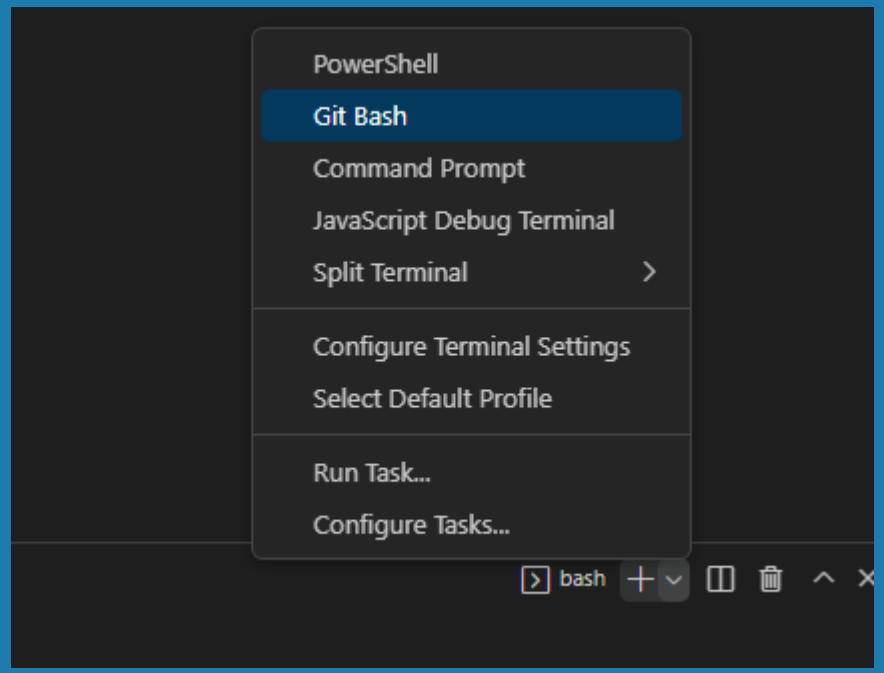
Open Folder



Terminal – New Terminal

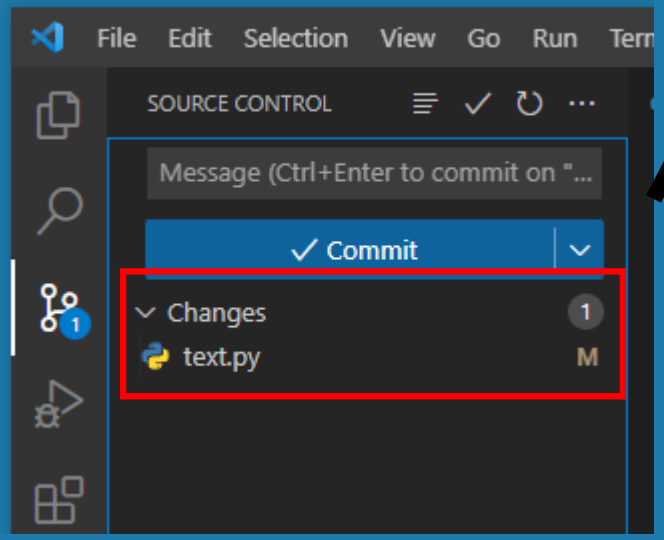


(우측하단) + -> Git Bash

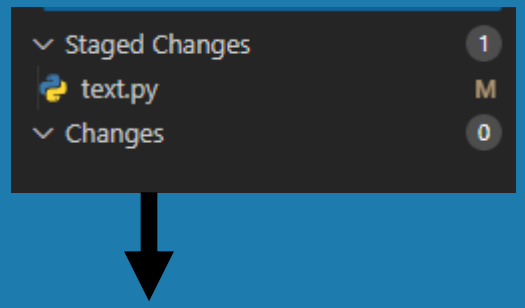


Vs코드 Push

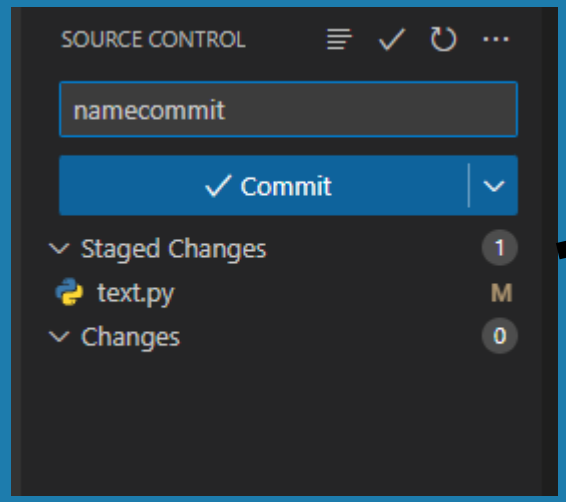
테스트 파일에 print(“이름”) 추가
-> Changes 에 파일 있는지 확인



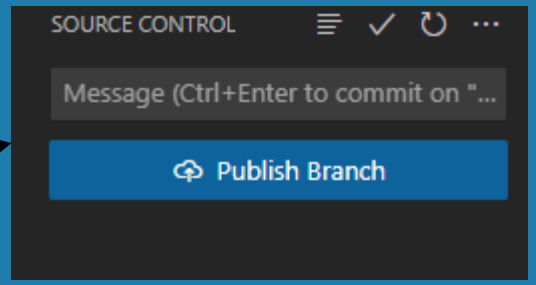
git add “파일명”
-> Staged Changes 이동 확인
-> git push 할 때 Stagged Changes에 있는 파일들이 Push 됨



Commit 입력 후 Commit 버튼 클릭

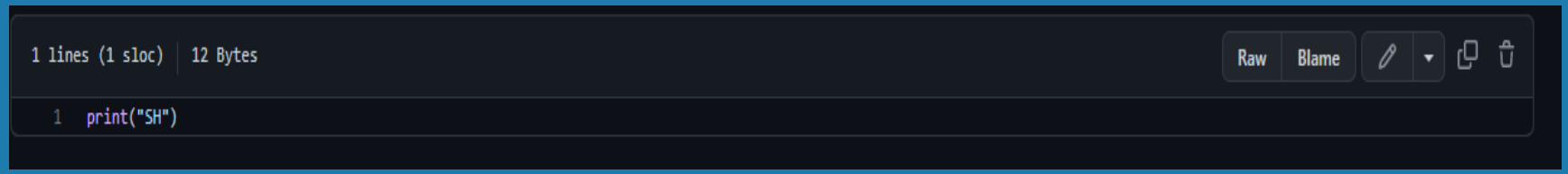


Publish Branch 클릭



Vs코드 Pull

깃 허브 에서 파일 수정



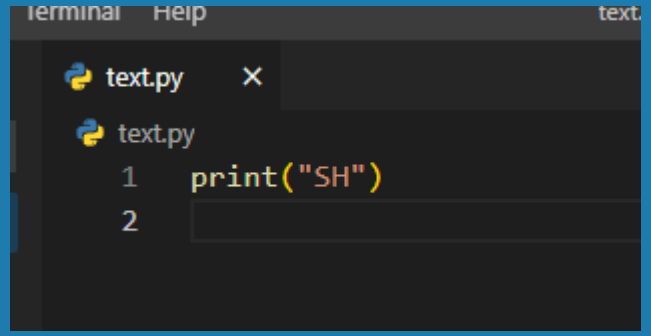
Vs코드 Pull

Vs코드에서

`git pull <원격저장소명> <브랜치명>`

```
User@DESKTOP-M960RTM MINGW64 ~/Desktop/friendgit (main)
$ git pull test main
```

파일 확인

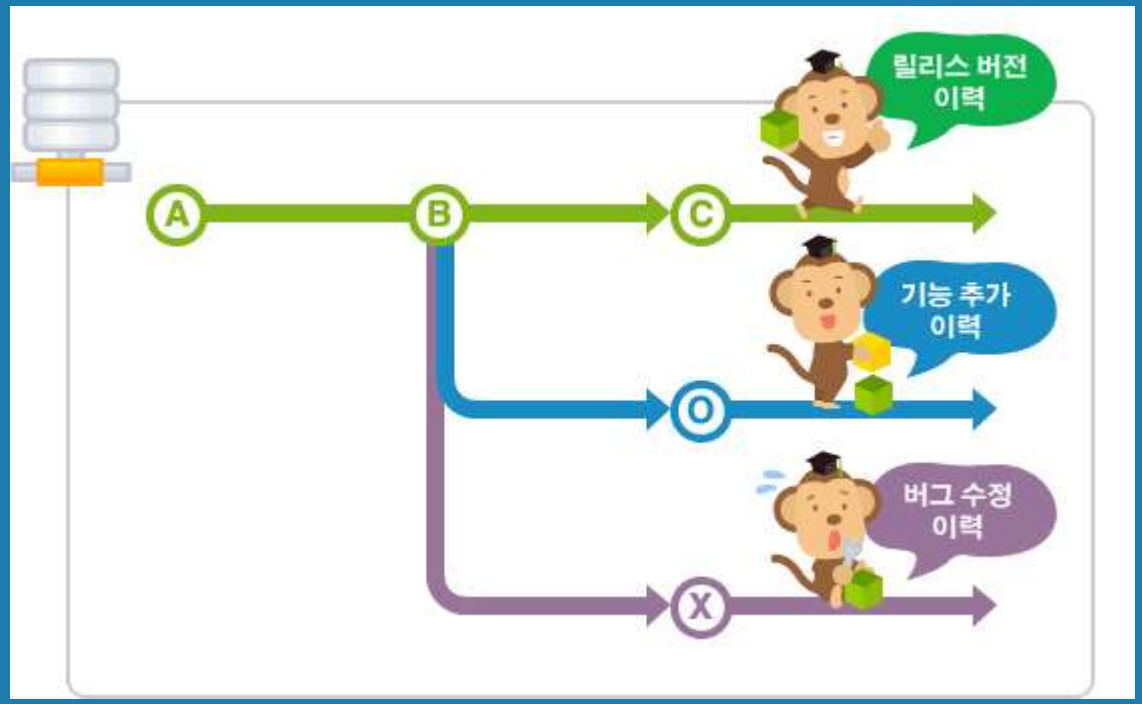


Part 4,

Branch 사용해보기

Branch

브랜치(branch)란?
브랜치란 독립적으로 어떤 작업을 진행하기 위한 개념입니다.
필요에 의해 만들어지는 각각의 브랜치는 다른 브랜치의 영향을 받지 않기 때문에,
여러 작업을 동시에 진행할 수 있습니다.



Branch

git branch <브랜치명>

```
User@DESKTOP-M960RTM MINGW64 ~/Desktop/friendgit
$ git branch JSH
```

Branch 이동 : git checkout <브랜치명>

```
User@DESKTOP-M960RTM MINGW64 ~/Desktop/friendgit (JSH)
```

git add “파일명”

```
User@DESKTOP-M960RTM MINGW64 ~/Desktop/friendgit (JSH)
$ git add text.py
```

git commit –m “commit내용”

```
User@DESKTOP-M960RTM MINGW64 ~/Desktop/friendgit (JSH)
$ git commit -m "새로운브랜치"
```

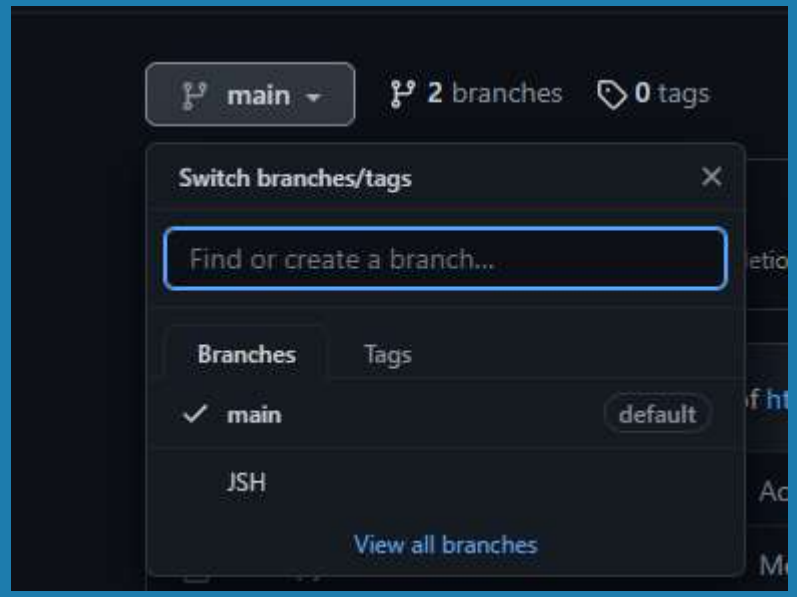
git push <원격저장소명> <브랜치명>

```
User@DESKTOP-M960RTM MINGW64 ~/Desktop/friendgit (JSH)
$ git push test JSH
Total 0 (delta 0), reused 0 (delta 0), pack-reused 0
remote:
remote: Create a pull request for 'JSH' on GitHub by visiting:
remote:      https://github.com/jeonsanghoon711/test/pull/new/JSH
remote:
To https://github.com/jeonsanghoon711/test.git
 * [new branch]      JSH -> JSH
```

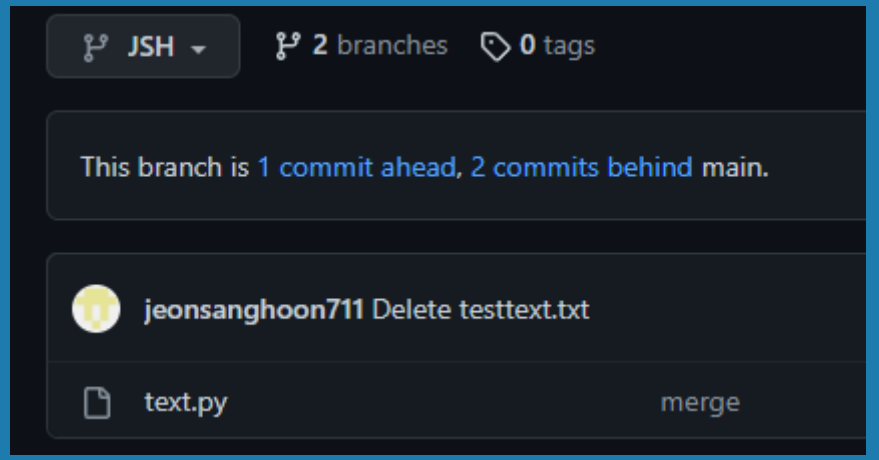
Branch

Git hub에 새로운 Branch 확인

Branch 이동



Branch 확인



Branch

파일 수정

text.py M

text.py

```
1  print("SH")
2  print("JSH")
3  print("asdfasdfasdf")
```

git add “파일명”

```
User@DESKTOP-M960RTM MINGW64 ~/Desktop/friendgit (JSH)
$ git add text.py
```

git commit -m “commit내용”

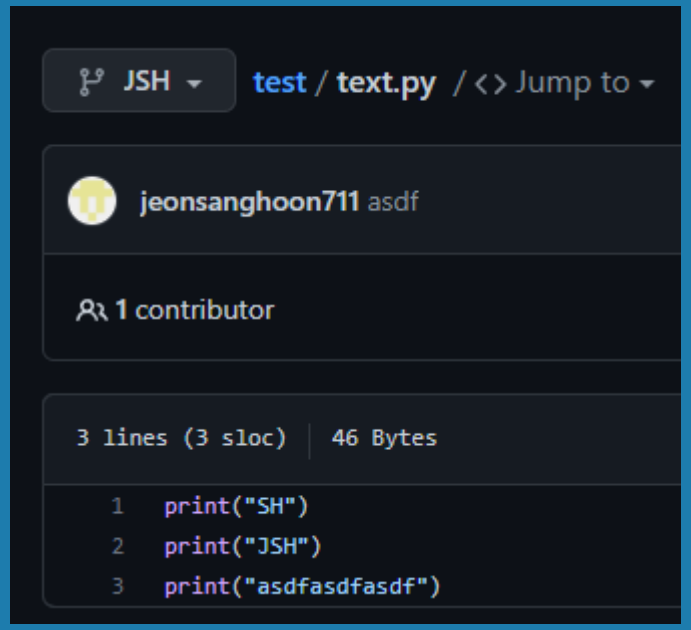
```
User@DESKTOP-M960RTM MINGW64 ~/Desktop/friendgit (JSH)
$ git commit -m "새로운브랜치"
```

git push <원격저장소명> <브랜치명>

```
User@DESKTOP-M960RTM MINGW64 ~/Desktop/friendgit (JSH)
$ git push test JSH
Total 0 (delta 0), reused 0 (delta 0), pack-reused 0
remote:
remote: Create a pull request for 'JSH' on GitHub by visiting:
remote:      https://github.com/jeonsanghoon711/test/pull/new/JSH
remote:
To https://github.com/jeonsanghoon711/test.git
 * [new branch]      JSH -> JSH
```

Merge

파일 확인



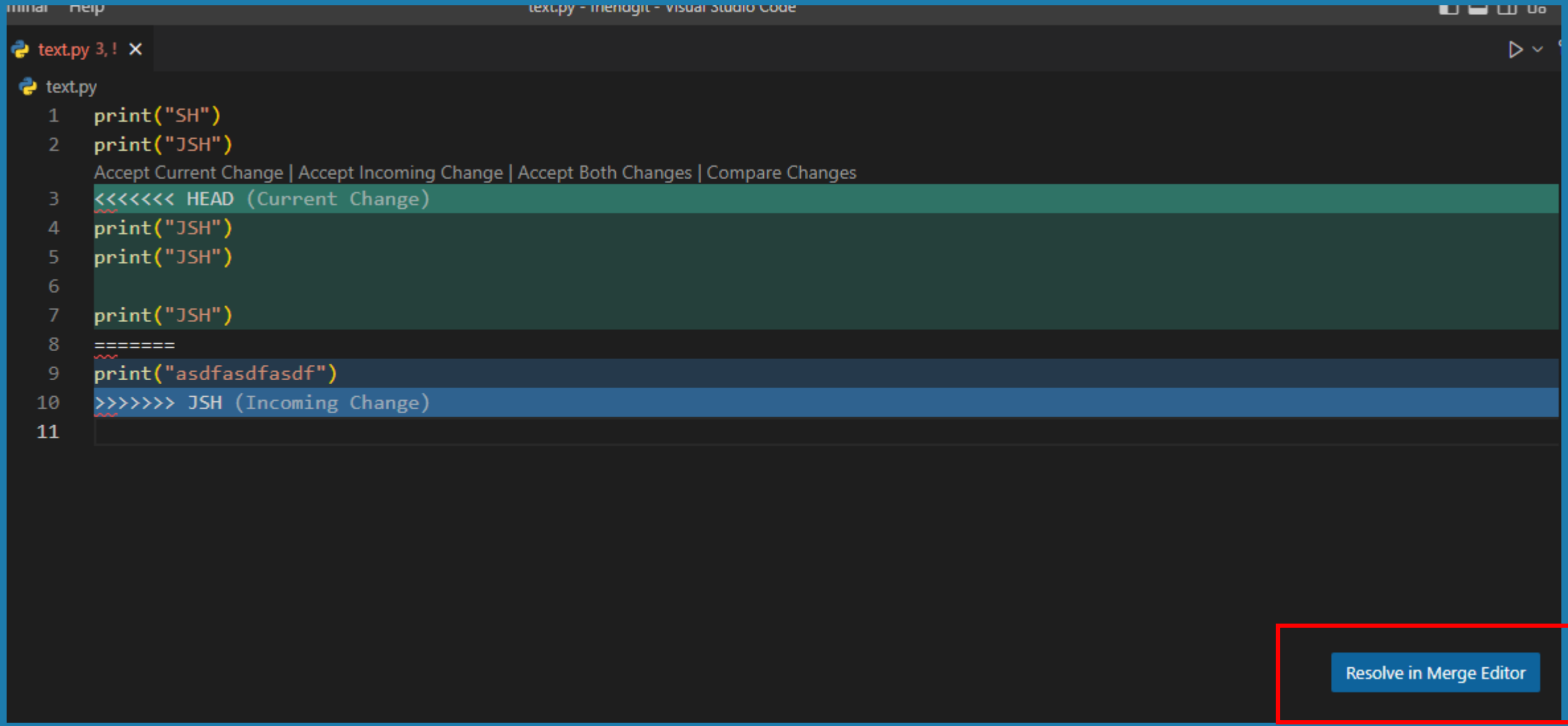
git checkout <브랜치명>

```
User@DESKTOP-M960RTM MINGW64 ~/Desktop/friendgit (JSH)
$ git checkout main
Switched to branch 'main'
Your branch is up to date with 'test/main'.
```

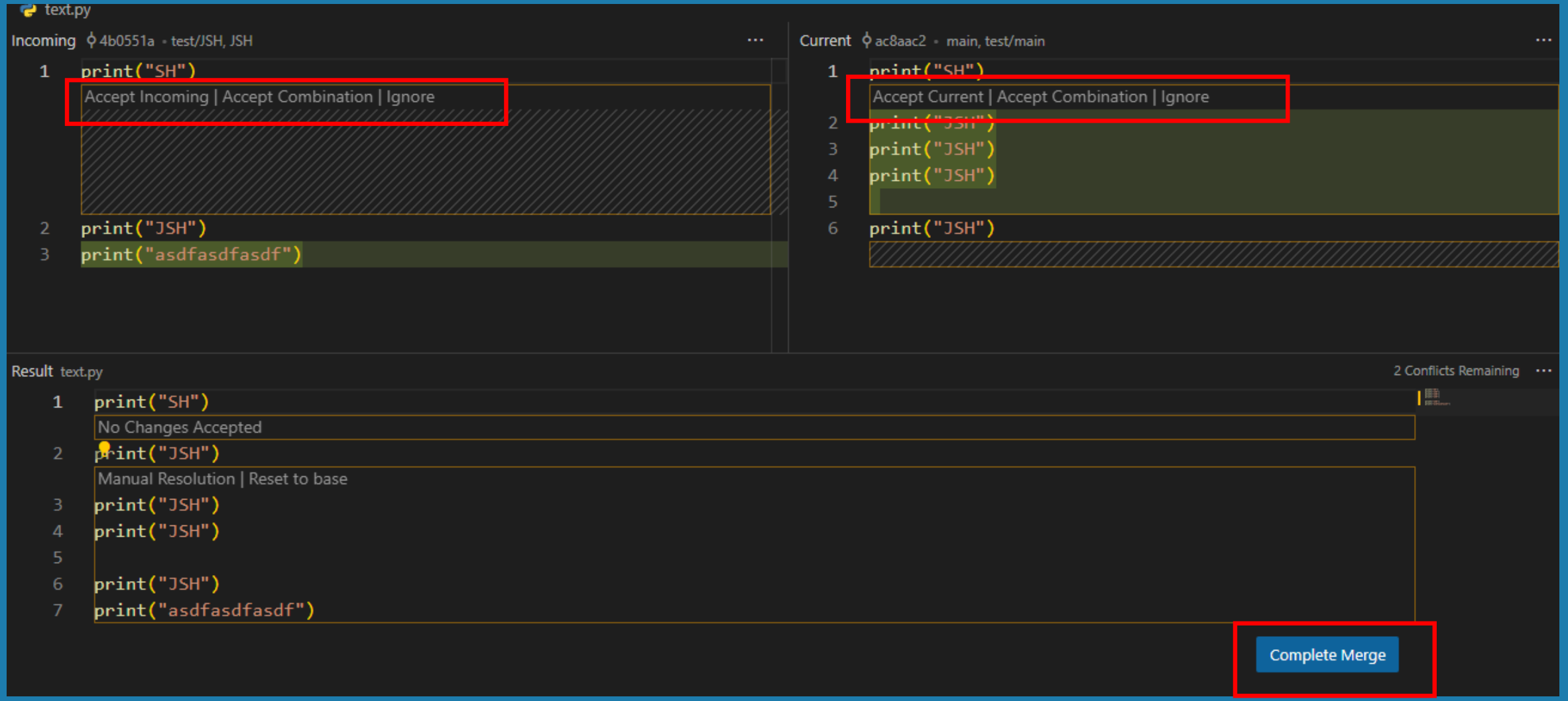
git merge <브랜치명>

```
User@DESKTOP-M960RTM MINGW64 ~/Desktop/friendgit (main)
$ git merge JSH
Auto-merging text.py
CONFLICT (content): Merge conflict in text.py
Automatic merge failed; fix conflicts and then commit the result.
```

Merge



Merge

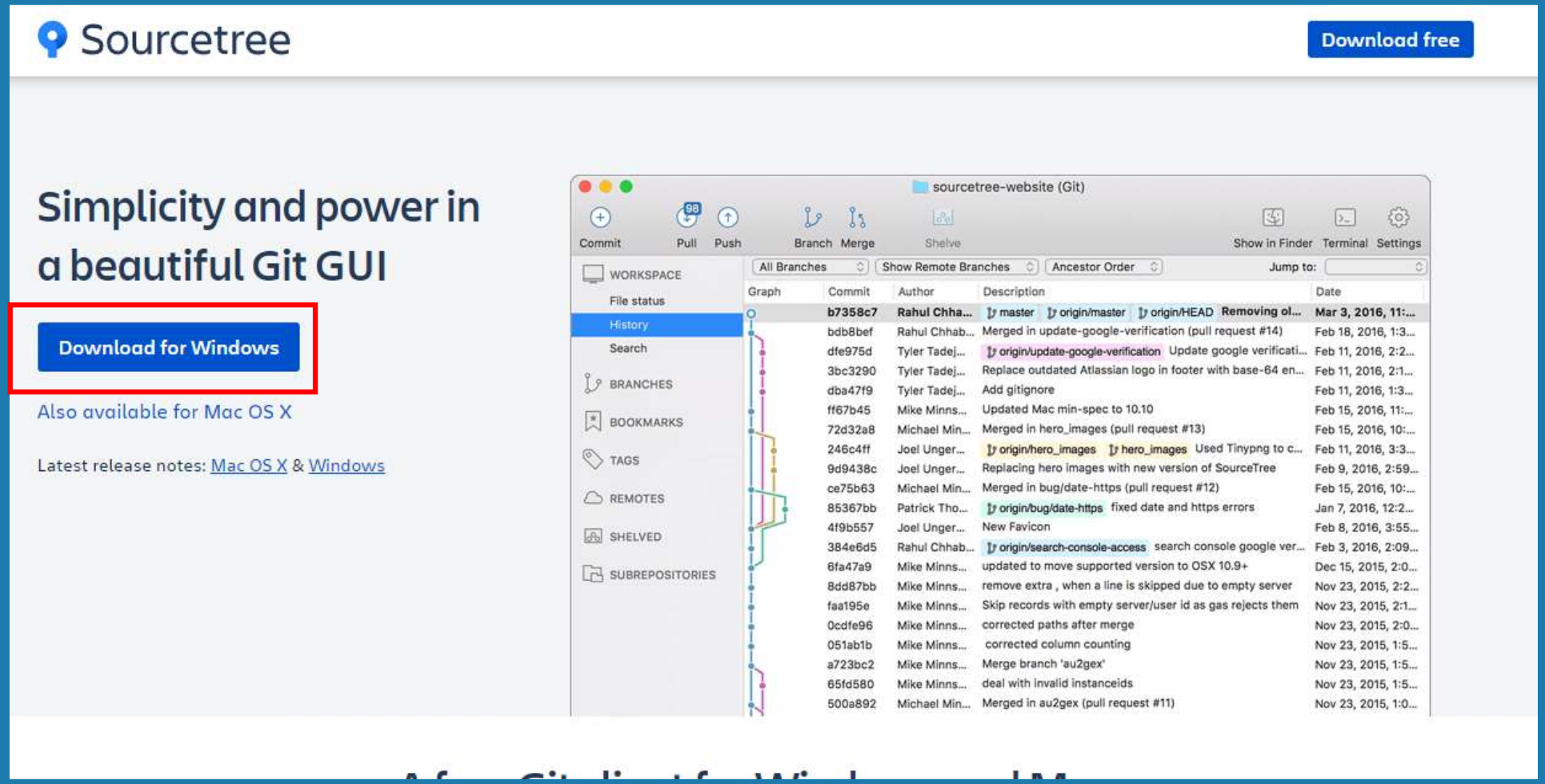


Part 4,

깃 소스트리 사용해보기

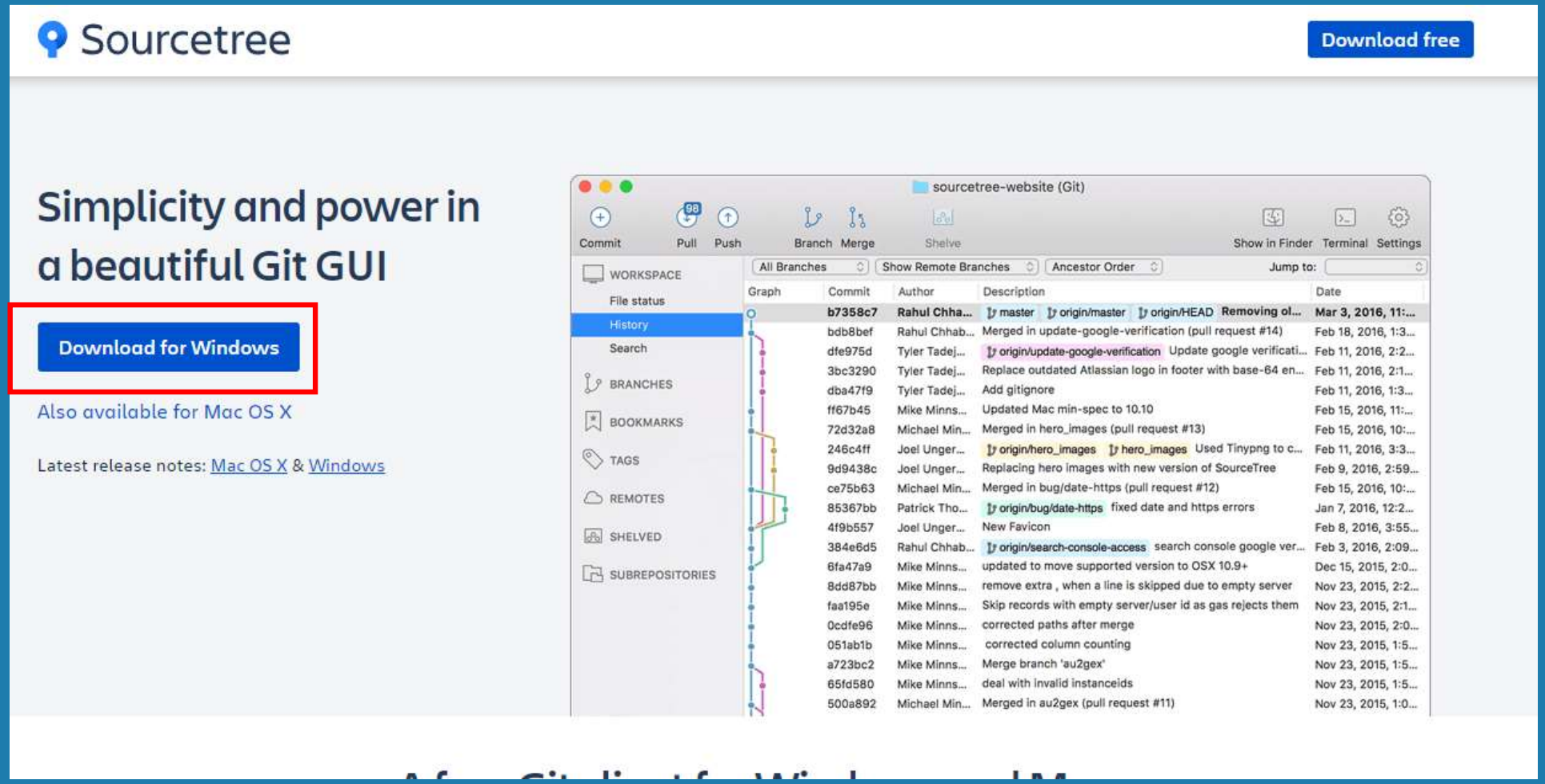
깃소스트리

https://www.sourcetreeapp.com/

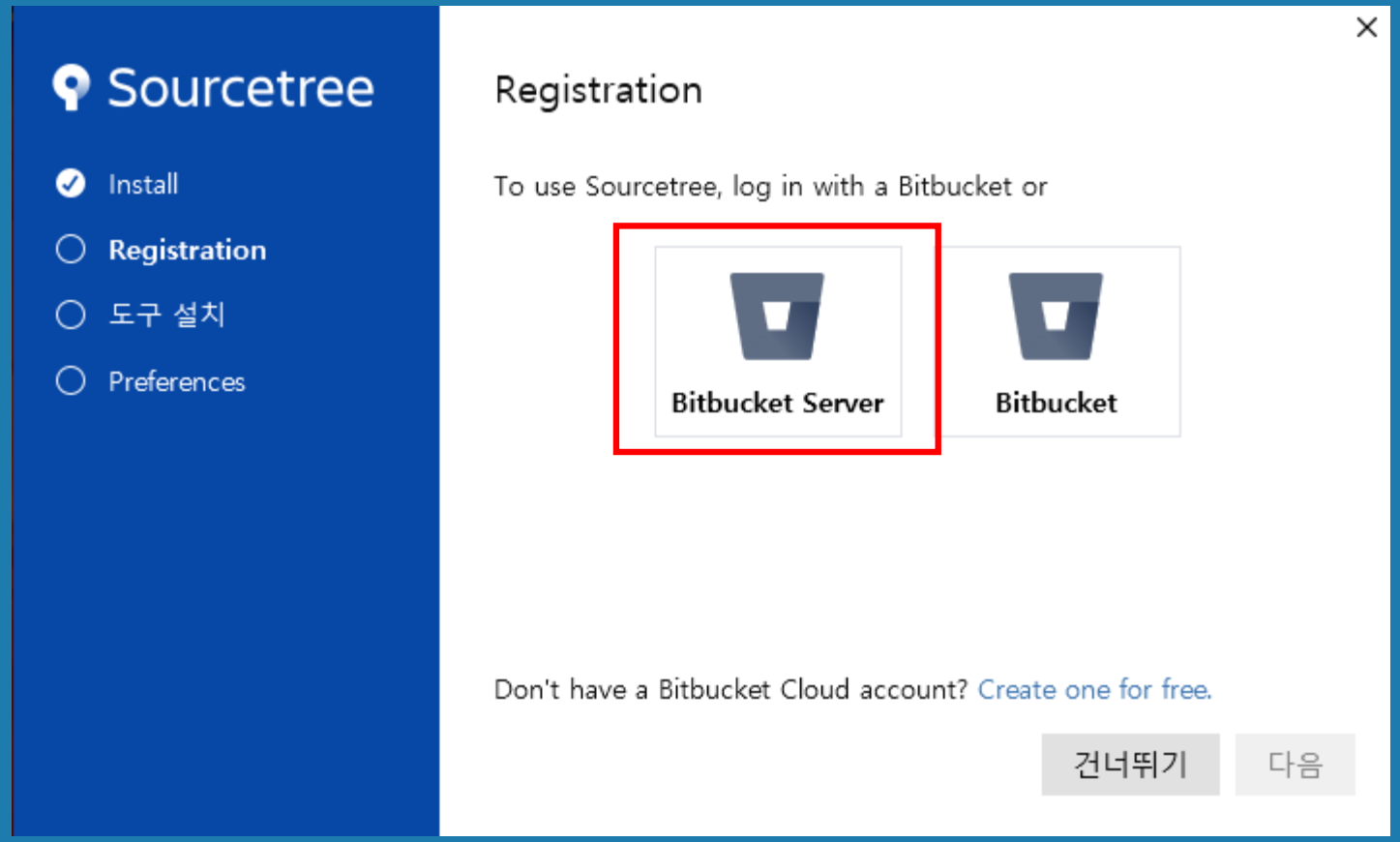


깃소스트리

https://www.sourcetreeapp.com/

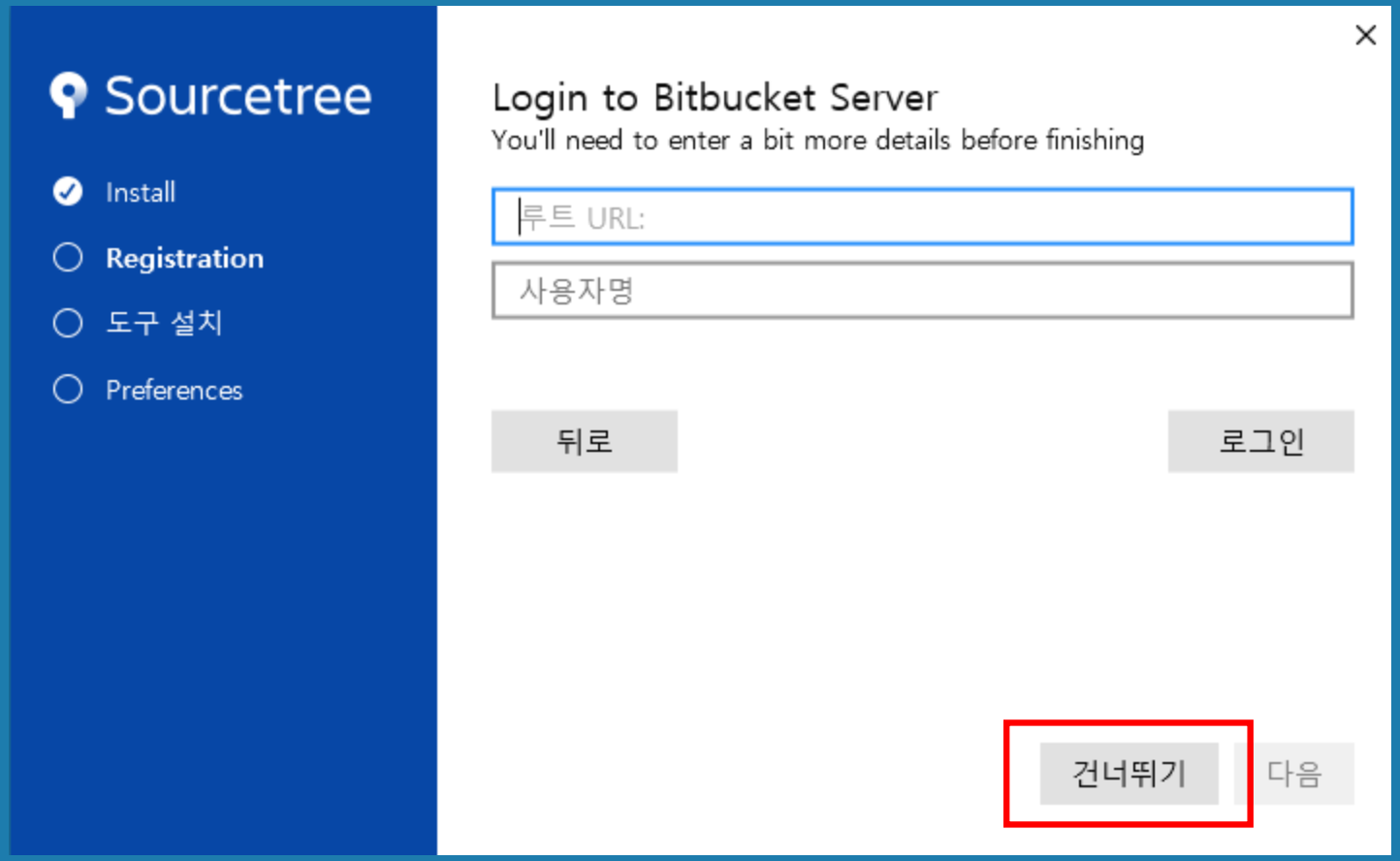


깃소스트리

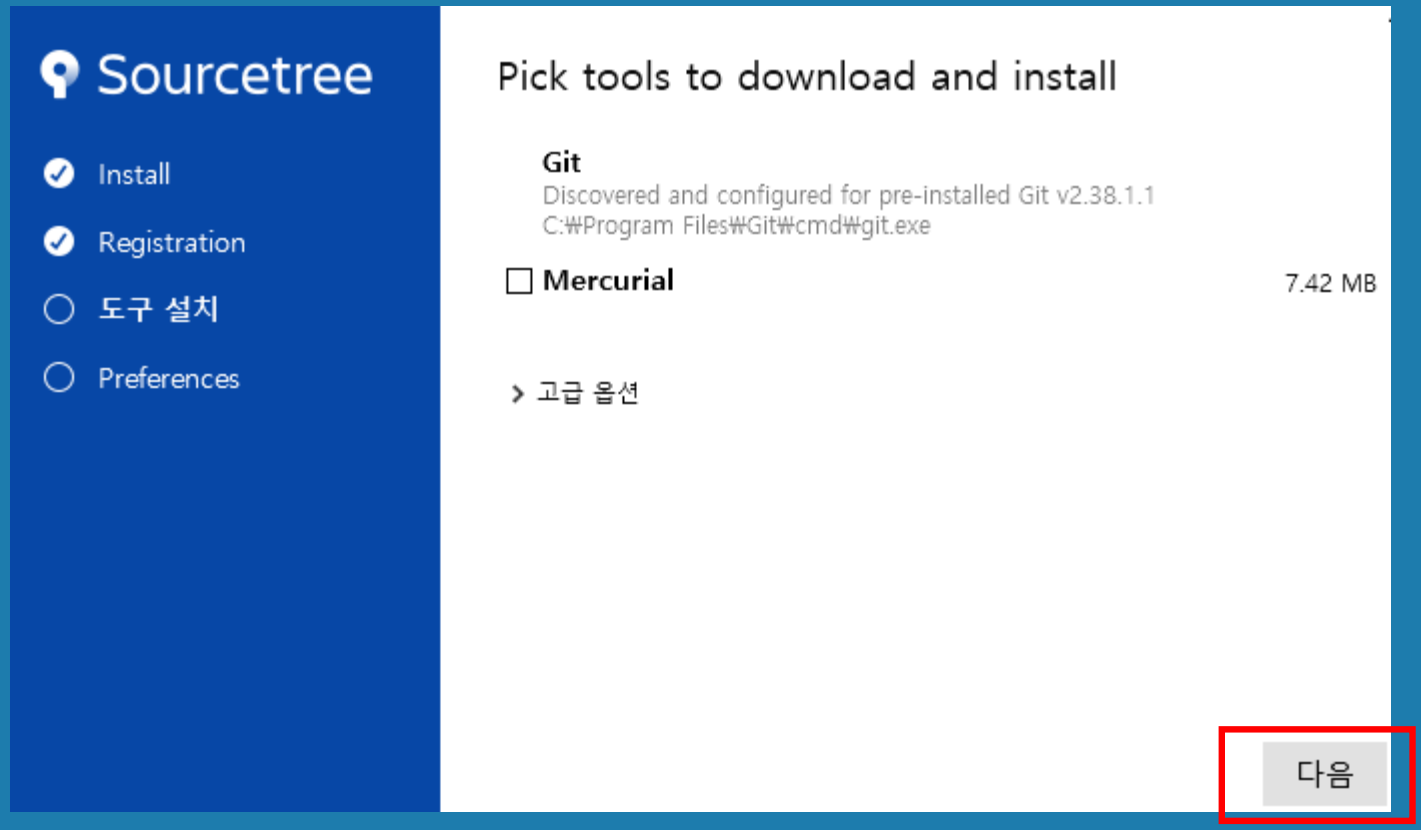


깃소스트리

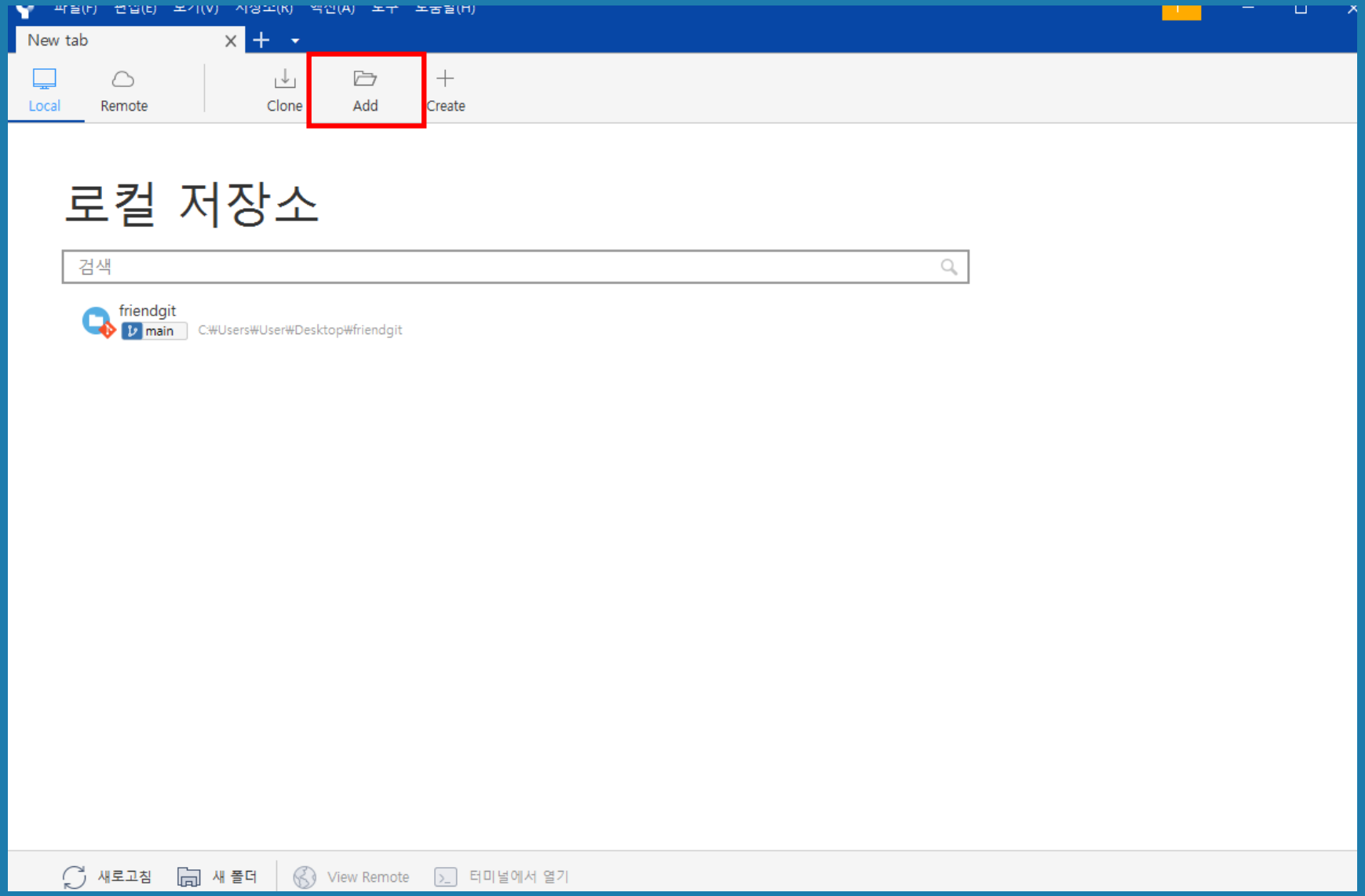
https://www.sourcetreeapp.com/



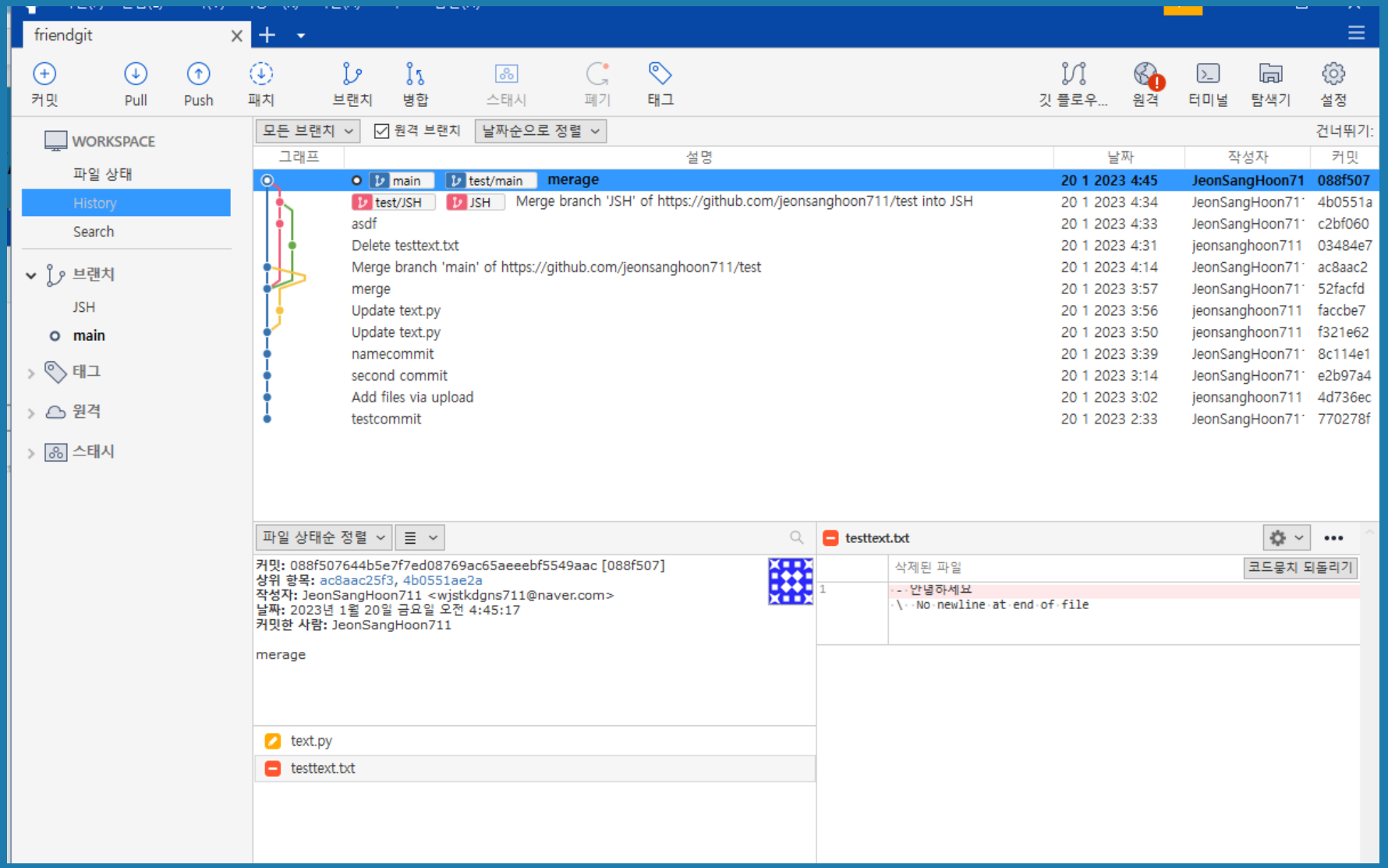
깃소스트리



깃소스트리



깃소스트리



출처

https://backlog.com/git-tutorial/kr/stepup/stepup1_1.html

감사합니다