

Git - 소스 코드 관리



GitHub



깃허브(Git Hurb)

■ 깃허브

분산 버전 관리 툴인 깃 저장소 호스팅을 지원하는 웹 서비스이다.

■ 깃허브 이용 필수 환경

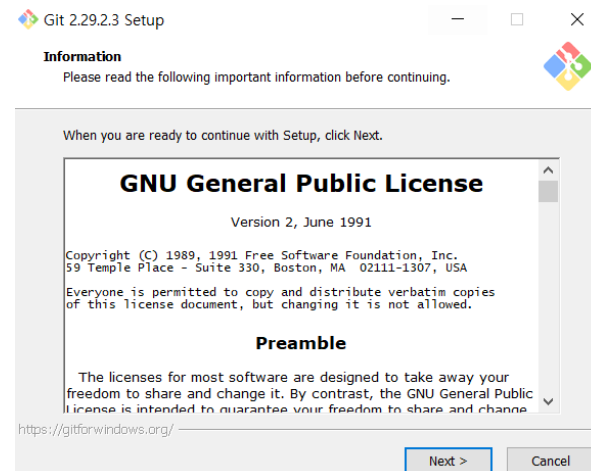
1. 깃 소프트웨어 설치(git-scm.com)
2. 깃허브 가입(github.com) 및 원격 저장소 생성
3. 명령 프롬프트 사용(CLI 프로그램)



깃 소프트웨어 설치

■ Git – 소프트웨어 설치

git-scm.com > 다운로드 후 설치



깃허브 원격 저장소 만들기

■ 깃허브 가입 및 Repository(저장소) 만들기

① Sign Up

Join GitHub

Create your account

Username *


Email address *

Password *


Make sure it's at least 15 characters OR at least 8 characters including a number and a lowercase letter.
[Learn more.](#)


Email preferences


☐ Send me occasional product updates, announcements, and offers.


 kiyongee2 ▾


② New 클릭

Repositories 

 kiyongee2/java



 kiyongee2/bt_car

 kiyongee2/db-access

 kiyongee2/gitstudy

③ 저장소 이름 만들기

Owner * Repository name *

 kiyongee2 ▾ / html 

Great repository names are short and memorable. Need inspiration? How about **cuddly-octo-sniffle**?

Description (optional)



명령 프롬프트 사용

■ 깃허브 사용 툴 - 명령 프롬프트

윈도우 - 검색 - cmd - 명령 프롬프트

C:W>git

C:W>git --version

```
C:\Users\김기용>git
usage: git [--version] [--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>]
        <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
  restore              Restore working tree files
  rm                   Remove files from the working tree and from the index
  sparse-checkout      Initialize and modify the sparse-checkout


examine the history and state (see also: git help revisions)
  bisect               Use binary search to find the commit that introduced a bug
  diff                 Show changes between commits, commit and working tree, etc
  grep                 Print lines matching a pattern
  log                  Show commit logs
  show                 Show various types of objects
  status               Show the working tree status
```



깃허브 가입

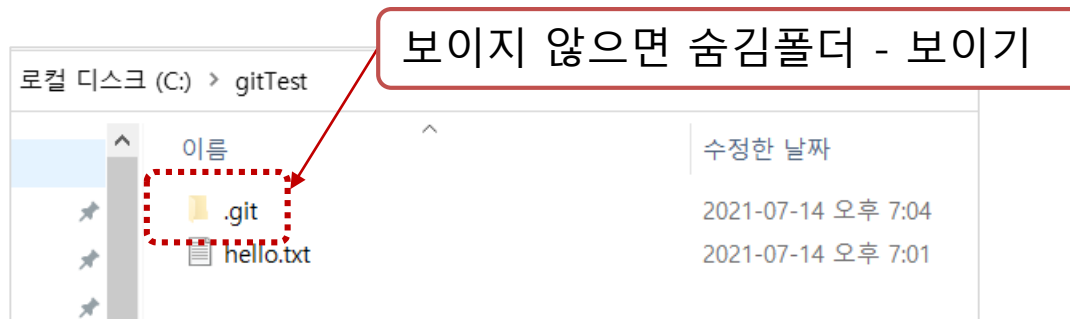
■ Git 초기 환경 설정

git config 명령은 컴퓨터 1대에서 처음 한번만 실행함

```
C:\WgitTest> git init
```

```
C:\W gitTest > git config --global user.name 본인 ID
```

```
C:\W gitTest > git config --global user.email 본인 Email
```



▶ user.name 확인하기

```
C:\Wgit config user.name
```



깃허브 가입

■ 깃에 파일 업로드하기 – 처음 업로드시

> git **status** (상태 확인)


> git **add** hello.txt / git **add** * (파일 여러 개 add)

> git **commit -m** "Add html+css file" (커밋 메모)

> git **remote add origin** http://github.com/kiyongee2/gitTest.git

> git **push -u** origin master

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

 Set up in Desktop or **HTTPS** **SSH** <https://github.com/kiyongee2/pyweb.git>

Get started by [creating a new file](#) or [uploading an existing file](#). We recommend every repository

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

```
echo "# pyweb" >> README.md
git init
git add README.md
git commit -m "first commit"
git branch -M main
git remote add origin https://github.com/kiyongee2/pyweb.git
git push -u origin main
```



깃허브 가입

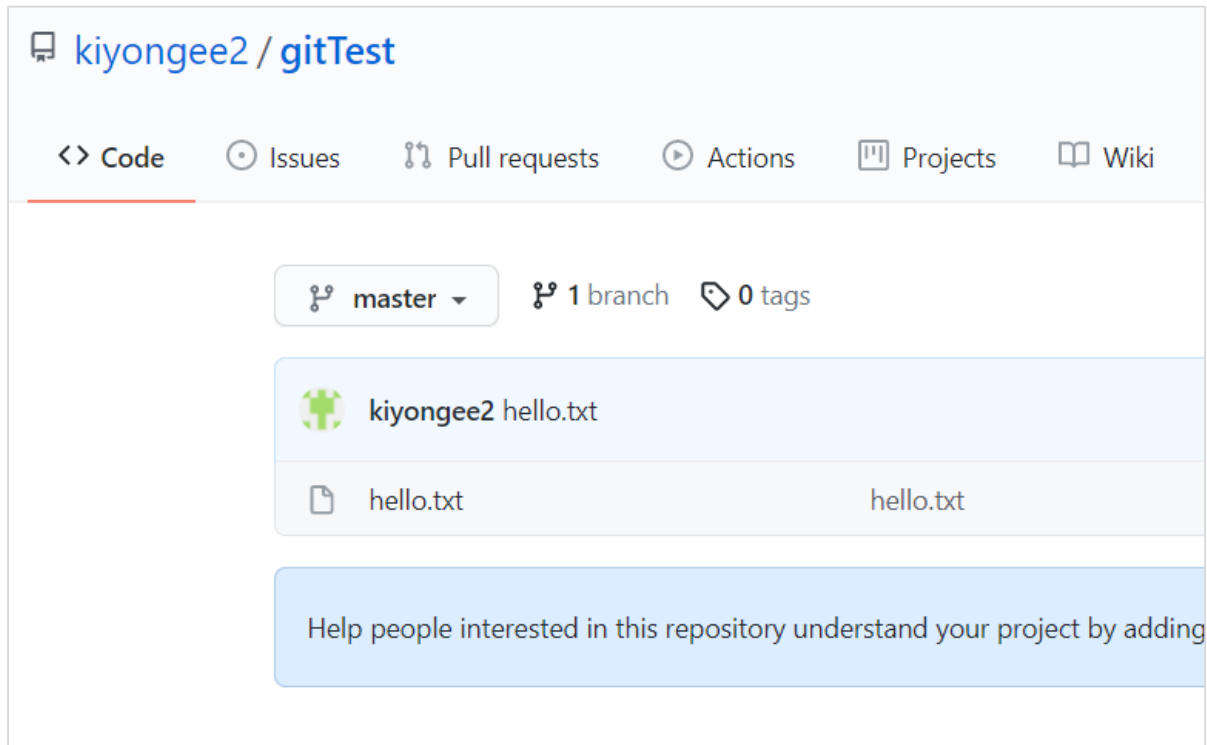
■ 깃에 파일 업로드하기 – 두번째 이후

- > git **status** 상태 확인
- > git **add** *
- > git **commit** -m "Add 추가 파일"
- > git **push**



깃허브 가입

■ 업로드된 파일 확인하기



깃허브 2개의 계정 사용

✓ 1대의 컴퓨터에서 2개의 계정을 사용할 경우 - 권한 에러 발생


```
c:\Wappjava3>git push -u origin main  
remote: Permission to sugu2100/appjava.git denied to kiyongee2
```

 > 제어판 > 사용자 계정 > 자격 증명 관리자

자격 증명 관리

웹 사이트, 연결된 응용 프로그램 및 네트워크에 대해 저장된 로그인 정보를 보고 삭제합니다.

 웹 자격 증명

 Windows 자격 증명

git:https://github.com

수정한 날짜: 오늘

인터넷 또는 네트워크 주소: git:https://github.com

사용자 이름: sugu2100

암호:

지속성: 로컬 컴퓨터

[편집](#) [제거](#)

일반 자격 증명 편집

입력한 사용자 이름과 암호를 사용하여 해당 위치에 액세스할 수 있는지 확인하십시오.

인터넷 또는 네트워크 주소: git:https://github.com

사용자 이름:

암호:



깃허브(GitHub)

■ origin과 master

- origin : 원격 저장소(repository)
- branch: 독립적인 작업 또는 기능 기본(default)는 master임

```
c:\w\projects>git remote -v  
origin https://github.com/kiyongee2/pybo.git (fetch)
```

```
c:\w\projects\mysite>git remote show origin  
* remote origin  
Fetch URL: https://github.com/kiyongee2/pybo.git  
Push URL: https://github.com/kiyongee2/pybo.git  
HEAD branch: master  
Remote branch:
```

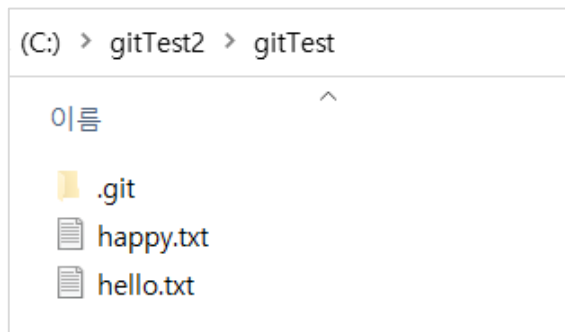


깃허브(GitHub)

■ 원격저장소에서 자료 가져오기

처음엔 **git clone** > 2번째 부터 **git pull** 사용

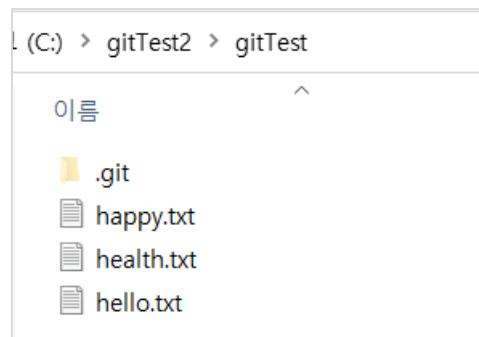
c:\gitTest2>**git clone** https://github.com/kiyongee2/gitTest



gitTest에서
health.txt - 업로드

2번째 부터 추가 파일이 있는 경우

c:\gitTest2>git pull



깃허브(GitHub)

- 브랜치 master -> main로 변경

```
c:\WgitTest>git branch
```

```
*master
```

```
c:\WgitTest>git branch -M main
```

```
*main
```

```
c:\WgitTest>git push -u origin main
```



깃허브(GitHub)

- 새 브랜치 만들기(새 폴더와 같은 의미)

git branch 새 브랜치 이름

```
c:\WgitTest>git branch develop
```

```
c:\WgitTest>git branch
```

```
*master
```

```
develop
```



깃허브(GitHub)

- 브랜치 이동하기

- git checkout 사용

```
c:\WgitTest>git checkout develop  
c:\WgitTest>git branch  
  
master  
* develop
```

- 새 브랜치에서 업로드하기

```
c:\WgitTest>git add *  
c:\WgitTest>git comit -m "추가"  
c:\WgitTest>git remote add develop 깃 레포지터리  
c:\WgitTest>git push develop
```



깃허브(GitHub)

- 브랜치 합치기

git merge 사용 – master로 이동 후 병합 수행

```
c:\WgitTest>git checkout master
```

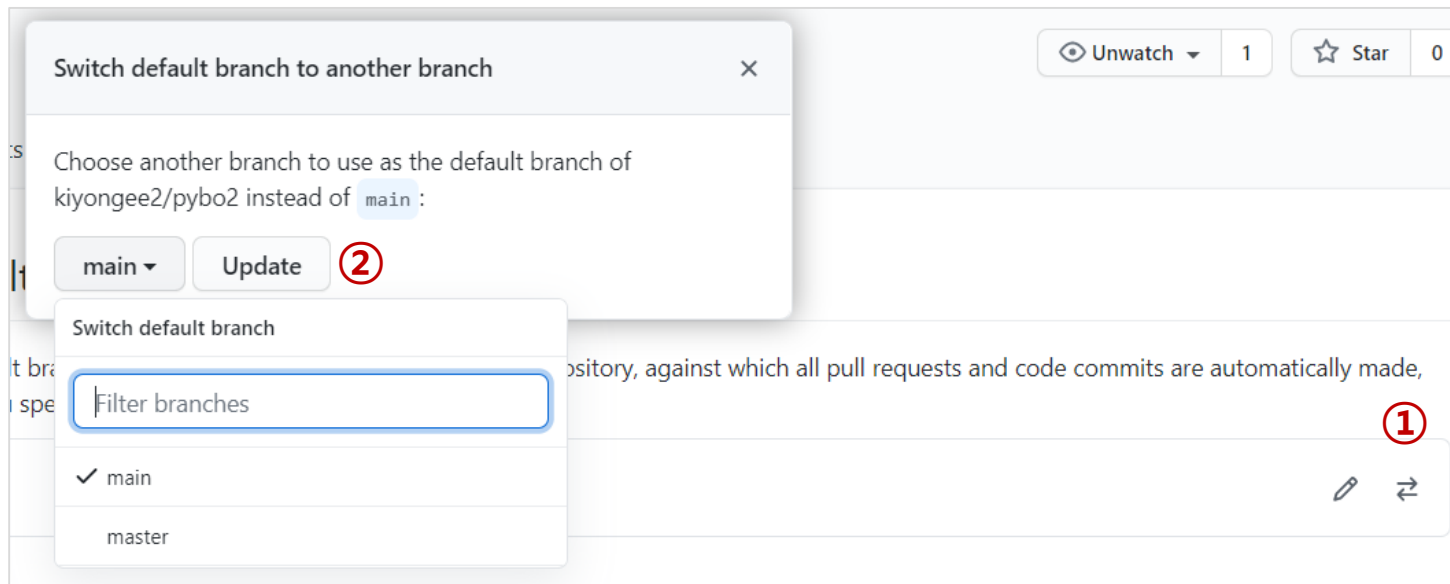
```
c:\WgitTest>git merge develop
```



깃허브(GitHub)

▪ branch 기본값(default) 변경

Settings > branches > 전환 > update



깃허브(GitHub)

■ 파일 삭제하기

>git **rm** 파일이름

>git **commit -m** "Delete 파일이름"

>git **push**

■ 디렉터리 삭제하기

>git **rm -rf** 디렉터리 이름

>git **commit -m** "Delete 디렉터리 이름"

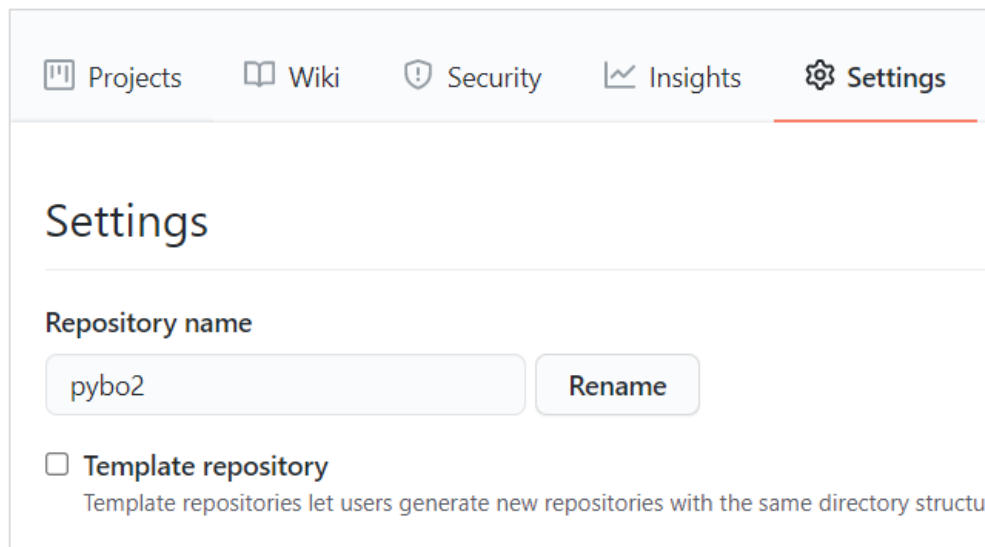
>git **push**



깃허브(GitHub)

■ 계정 이름 변경하기

Settings > 변경할 이름 > Rename



The screenshot shows the GitHub 'Settings' page for a repository. At the top, there is a navigation bar with icons and labels for 'Projects', 'Wiki', 'Security', 'Insights', and 'Settings'. The 'Settings' tab is selected and highlighted with a red underline. Below the navigation bar, the title 'Settings' is displayed. Under the 'Repository name' section, there is a text input field containing 'pybo2' and a 'Rename' button. Below this, there is a checkbox labeled 'Template repository' which is currently unchecked. A descriptive text below the checkbox states: 'Template repositories let users generate new repositories with the same directory structure'.



깃허브(GitHub)

■ 계정 삭제하기

Settings > Danger Zone

Are you absolutely sure?

×

Unexpected bad things will happen if you don't read this!

This action **cannot** be undone. This will permanently delete the **kiyongee2/gitTest** repository, wiki, issues, comments, packages, secrets, workflow runs, and remove all collaborator associations.

Please type **kiyongee2/gitTest** to confirm.

kiyongee2/gitTest

I understand the consequences, delete this repository



깃허브(GitHub)

■ 협업 하기

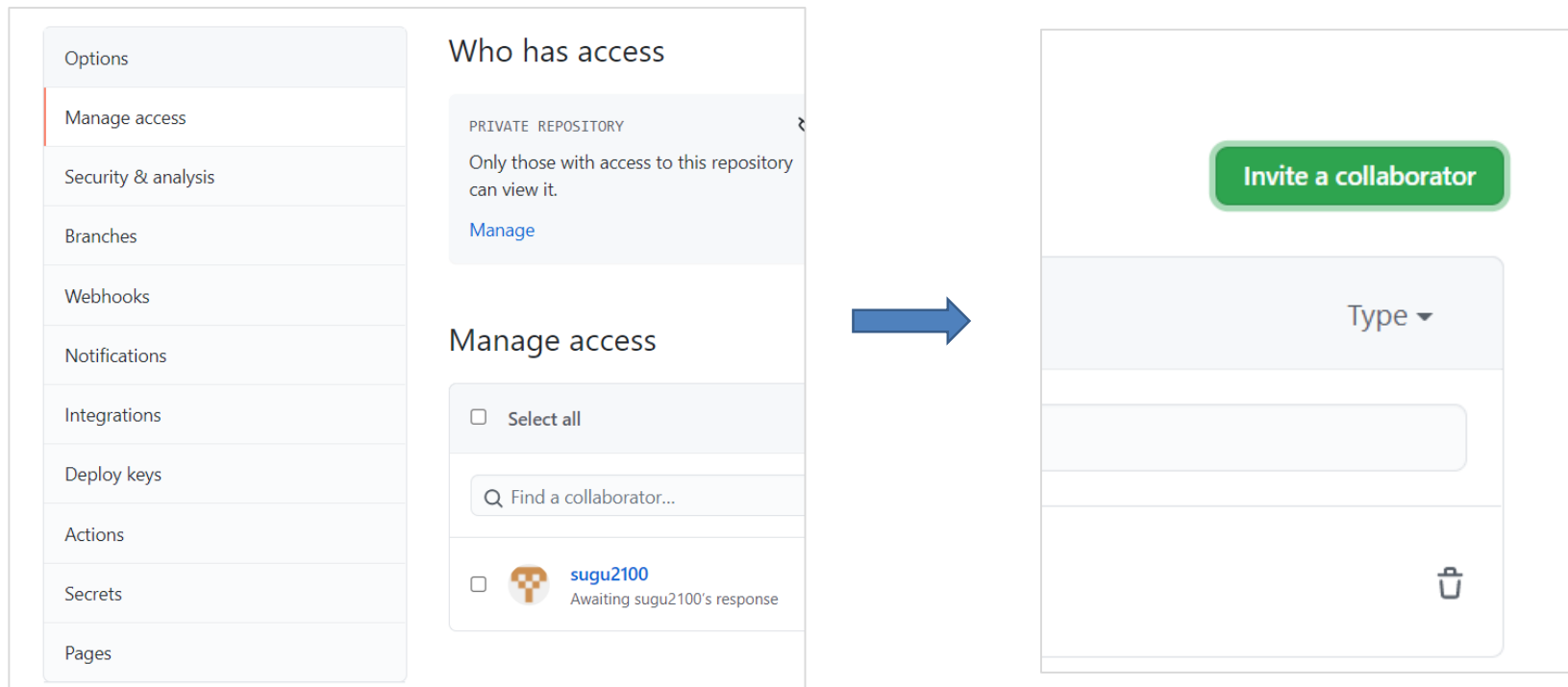
1. com1과 com2 준비
2. 모두 원격의 git-test에서 git clone으로 다운로드
3. com1>git-test 에서 hello.txt 변경



깃허브(GitHub)

▪ private – 협업 참여

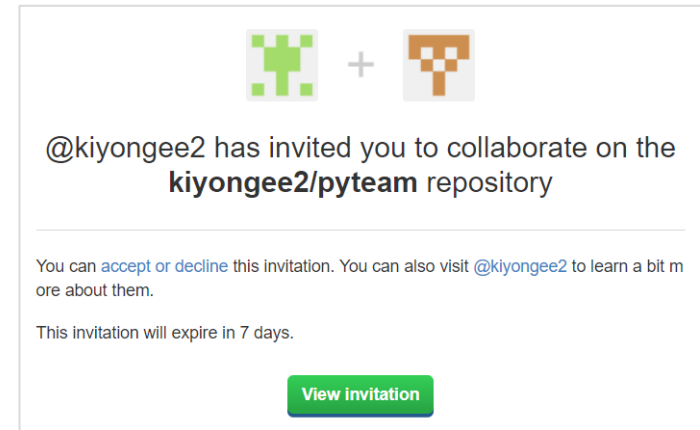
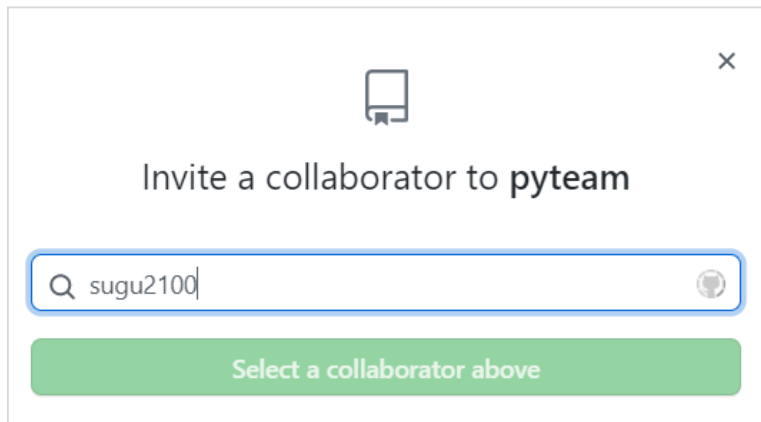
setting > Manage access > invite a collaborator



깃허브(GitHub)

- **private – 협업 참여**

setting > Manage access > invite a collaborator



이메일 확인

