

Git - 소스 코드 관리



GitHub



깃허브(Git Hurb)

■ 깃허브란?

분산 버전 관리 툴인 깃 저장소 호스팅을 지원하는 웹 서비스이다.

깃을 창시한 사람은 리눅스를 만든 리누즈 토발즈이고, 깃허브를 인수하여 운영하는 곳은 마이크로소프트(MS)사이다.

■ 깃허브 환경 구축

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 > 계속 next



Download for Windows

[Click here to download](#) the latest (2.34.1) 64-bit version or recent maintained build. It was released **about 1 month ago**.

Other Git for Windows downloads

[Standalone Installer](#)

[32-bit Git for Windows Setup.](#)

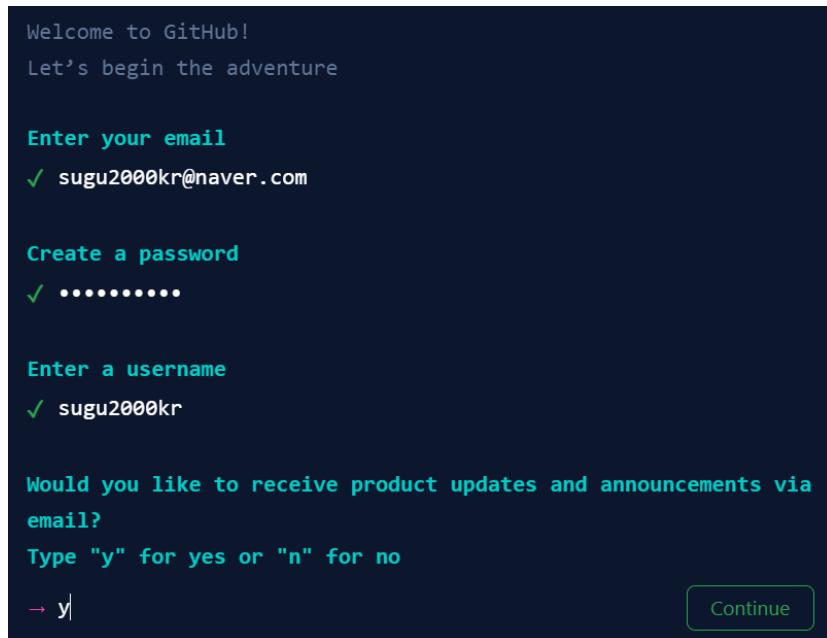
[**64-bit Git for Windows Setup.**](#)



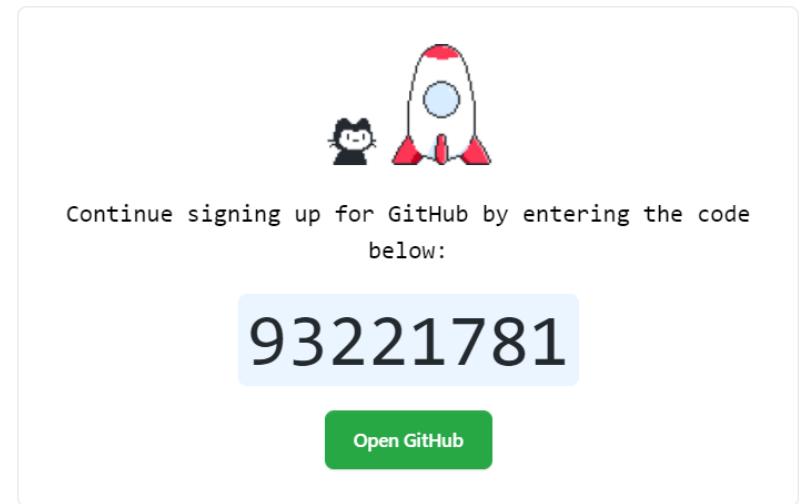
깃허브 원격 저장소 만들기

■ 깃허브 가입하기

Sign Up > 메일로 코드 확인



Here's your GitHub launch code, @sugu2000kr!



깃허브 원격 저장소 만들기

■ Repository(저장소) 만들기

Create your first project
Ready to start building? Create a repository for a new idea or bring over an existing repository to keep contributing to it.

Create repository **Import repository** ① 클릭

Recent activity
When you take actions across GitHub, we'll provide links to that activity here.

Owner * Repository name * ② 저장소 이름 만들기
sugu2000kr / gitTest ✓

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

Description (optional)
깃 테스트

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

③ 깃 명령어

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

[Set up in Desktop](#) or [HTTPS](#) [SSH](#) <https://github.com/sugu2000kr/gitTest.git>

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

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

```
echo "# gitTest" >> README.md
git init
git add README.md
git commit -m "first commit"
git branch -M main
git remote add origin https://github.com/sugu2000kr/gitTest.git
git push -u origin main
```



명령 프롬프트 사용

▪ 깃허브 사용 툴 – 명령 프롬프트

* 윈도우 – 검색 – cmd – 명령 프롬프트

C:\>git

C:\>git –version

* 사용자 확인

C:\>git config user.name

```
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
```



깃 환경 설정

▪ Git 초기 환경 설정

git config 명령은 컴퓨터 1대에서 처음 한번만 실행함

C:\gitTest> git config --user.name //git 계정확인

C:\gitTest > git config --global user.name "kiyongee2"(본인 ID)

C:\gitTest > git config --global user.email "kiyongee2@gmail.com"

C:\gitTest> git init #git 초기화하기



깃에 파일 업로드하기

▪ 깃에 파일 업로드하기 – 처음 업로드시

> git **status** (상태 확인)

> **git add hello.txt / git add .** (모든 파일 add * 도 가능) //git 추가하기

> git **commit -m "Add hello.txt"** //커밋

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

> git **push -u origin master**



깃에 파일 업로드하기

▪ 깃에 파일 업로드하기 – 두번째 이후

> git **status** 상태 확인

> git **add ***

> git **commit** -m "Add 추가 파일"

> git **push**



깃허브 레포지터리 보기

■ 업로드된 파일 확인하기

The screenshot shows a GitHub repository page for 'kiyongee2 / gitTest'. The 'Code' tab is selected. At the top, there are links for Issues, Pull requests, Actions, Projects, and Wiki. Below these, it shows 'master' branch, 1 branch, and 0 tags. A file named 'hello.txt' is listed. A blue box at the bottom encourages adding a project description.

kiyongee2 / gitTest

Code Issues Pull requests Actions Projects Wiki

master 1 branch 0 tags

kiyongee2 hello.txt

hello.txt hello.txt

Help people interested in this repository understand your project by adding



깃 파일 삭제

- 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**



깃 계정 이름 변경

▪ 계정 이름 변경하기

Settings > 변경할 이름 > Rename

The screenshot shows the GitHub Settings interface. At the top, there are navigation links: Projects, Wiki, Security, Insights, and Settings. The Settings link is highlighted with a red underline. Below the navigation, the word "Settings" is displayed. Underneath, there is a section for "Repository name" with a text input field containing "pybo2" and a "Rename" button next to it. A checkbox labeled "Template repository" is present, with a descriptive subtitle below it.

Repository name

pybo2

Rename

Template repository

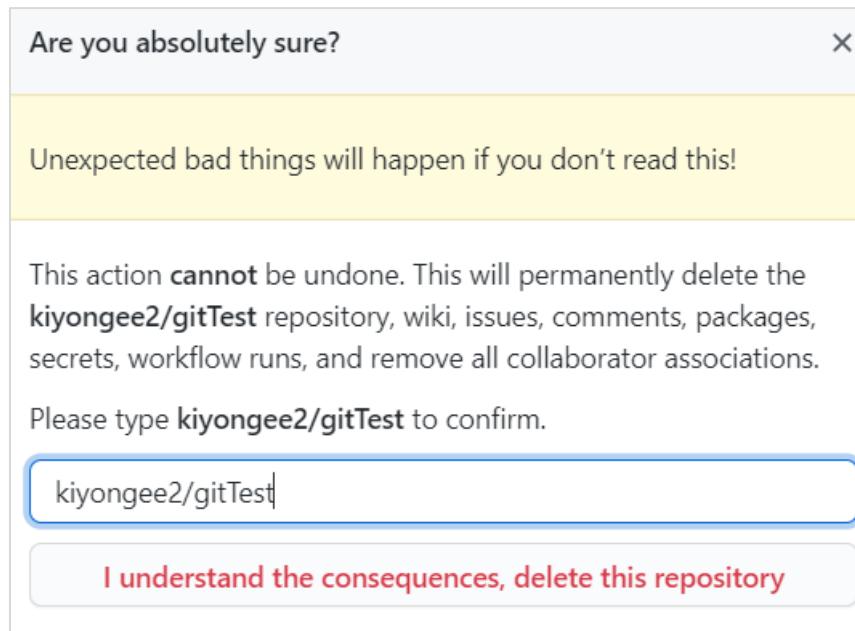
Template repositories let users generate new repositories with the same directory structure



깃 계정 삭제

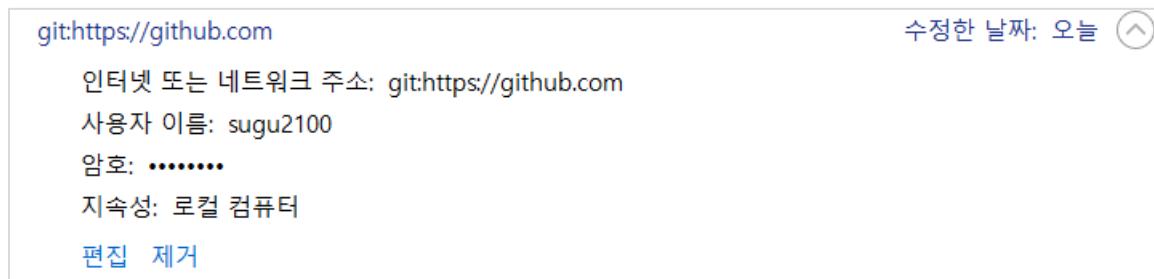
▪ 계정 삭제하기

Settings > Danger Zone



내 컴퓨터의 다른 사용자 계정 삭제하기

❖ 이미 사용중인 다른 사용자 계정 삭제하기

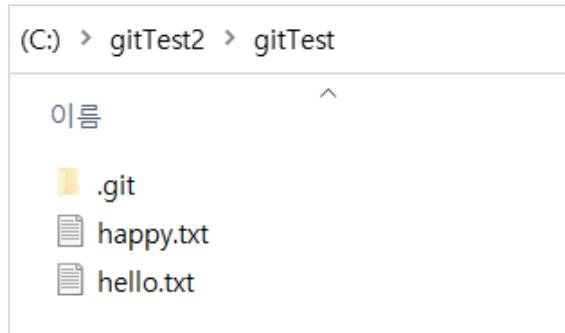


깃 클론(git clone)

■ 원격저장소에서 자료 가져오기

처음엔 git clone > 2번째 부터 git pull 사용

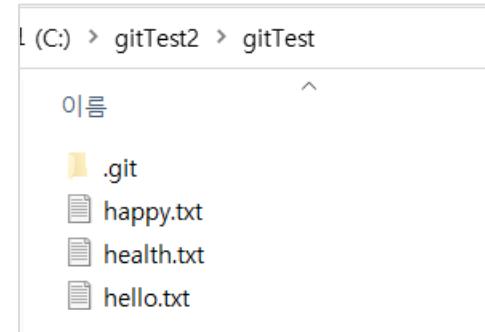
```
c:\gitTest2>git clone https://github.com/kiyongee2/gitTest
```



gitTest에서
health.txt - 업로드

2번째 부터 추가 파일이 있는 경우

```
c:\gitTest2>git pull
```



브랜치 이름 변경하기

- 브랜치 master -> main으로 변경

```
c:\gitTest>git branch  
*master  
c:\gitTest>git branch -M main  
*main  
c:\gitTest>git push -u origin main
```

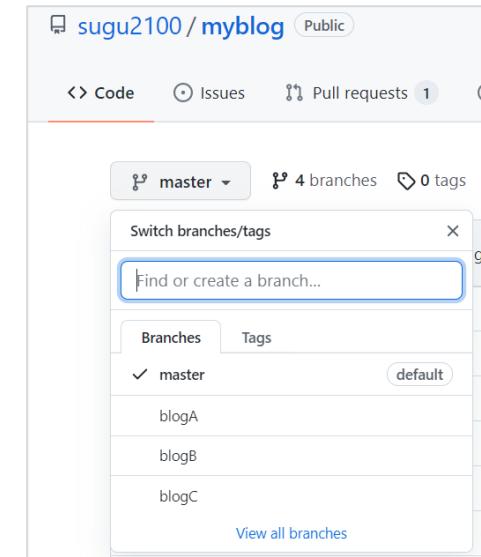


새 브랜치 만들기

▪ 새 브랜치 만들기

1. 새 브랜치 만들기 - **git branch** 브랜치 이름

```
c:\gitTest>git branch blogA  
c:\gitTest>git branch  
*master  
  
blogA
```



2. blogA 원격 계정에 추가하기

```
c:\gitTest>git remote add blogA https://github.com/sugu2100/myblog  
c:\gitTest>git push blogA
```



브랜치 이동하기

- **브랜치 이동하기**

blogA로 브런치 이동 – git checkout 브런치 이름

```
c:\gitTest>git checkout blogA
```

```
c:\gitTest>git branch
```

```
master
```

```
* blogA
```

- **자료 수정후 깃에 업로드하기**

```
c:\gitTest>git add *
```

```
C:\gitTest>git commit -m "추가"
```

```
C:\gitTest>git push blogA
```

