

1장. Python 개발 환경 구축 및 깃(GIT)



프로그래밍이란?

- **프로그래밍(Programming)이란?**

- 컴퓨터 프로그램을 만드는 일
- 컴퓨터에게 원하는 작업을 수행하도록 명령을 내리는 과정

- **프로그램(Program)**

- 컴퓨터에게 일을 시키는 명령의 집합 또는 프로그래밍한 작업의 결과.

- **프로그래밍 언어의 종류**

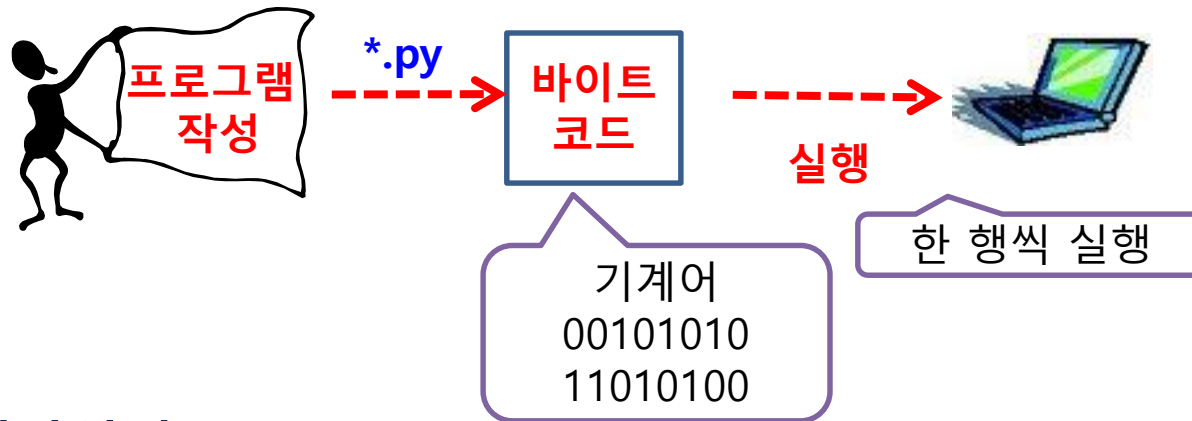
- C언어 , C++언어, Java, Python, JavaScript, C#

- **인터프리터, 컴파일러**

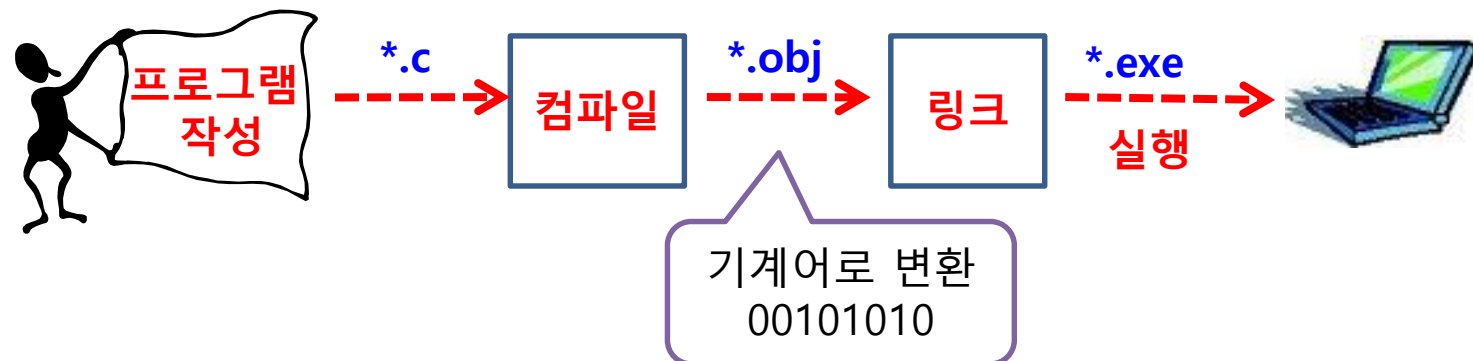
- 프로그램 언어를 컴퓨터가 알 수 있는 언어(기계어)로 바꿔 주는 프로그램
- **인터프리터(코드를 한 줄씩 변환하여 실행) – 파이썬, 자바스크립트**
- **컴파일러(전체 코드를 한 번에 변환후 실행) – C, C++, Java**

인터프리터와 컴파일러

➤ 인터프리터(Interpreter)



➤ 컴파일러(Compiler)



Python 언어

◆ 파이썬(Python) 창시

- 창시자 : 1990년 네델란드 암스테르담의 **귀도 반 로섬**
(이름의 유래 - 좋아하는 코미디 프로그램)
- 플랫폼에 독립적이고, 인터프리터 언어이며 객체지향 언어이다.

◆ 파이썬의 특징

- ✓ 사람이 사고하는 체계와 비슷하다.
- ✓ 문법이 간결하고 읽기 쉽다.
- ✓ 오픈 소스로 무료 - 누구나 자유롭게 사용하고 확장 가능
- ✓ 개발 속도가 빠르다.
- ✓ 다양하고 많은 라이브러리를 사용할 수 있다.

Python 언어

◆ 파이썬으로 할 수 있는 일

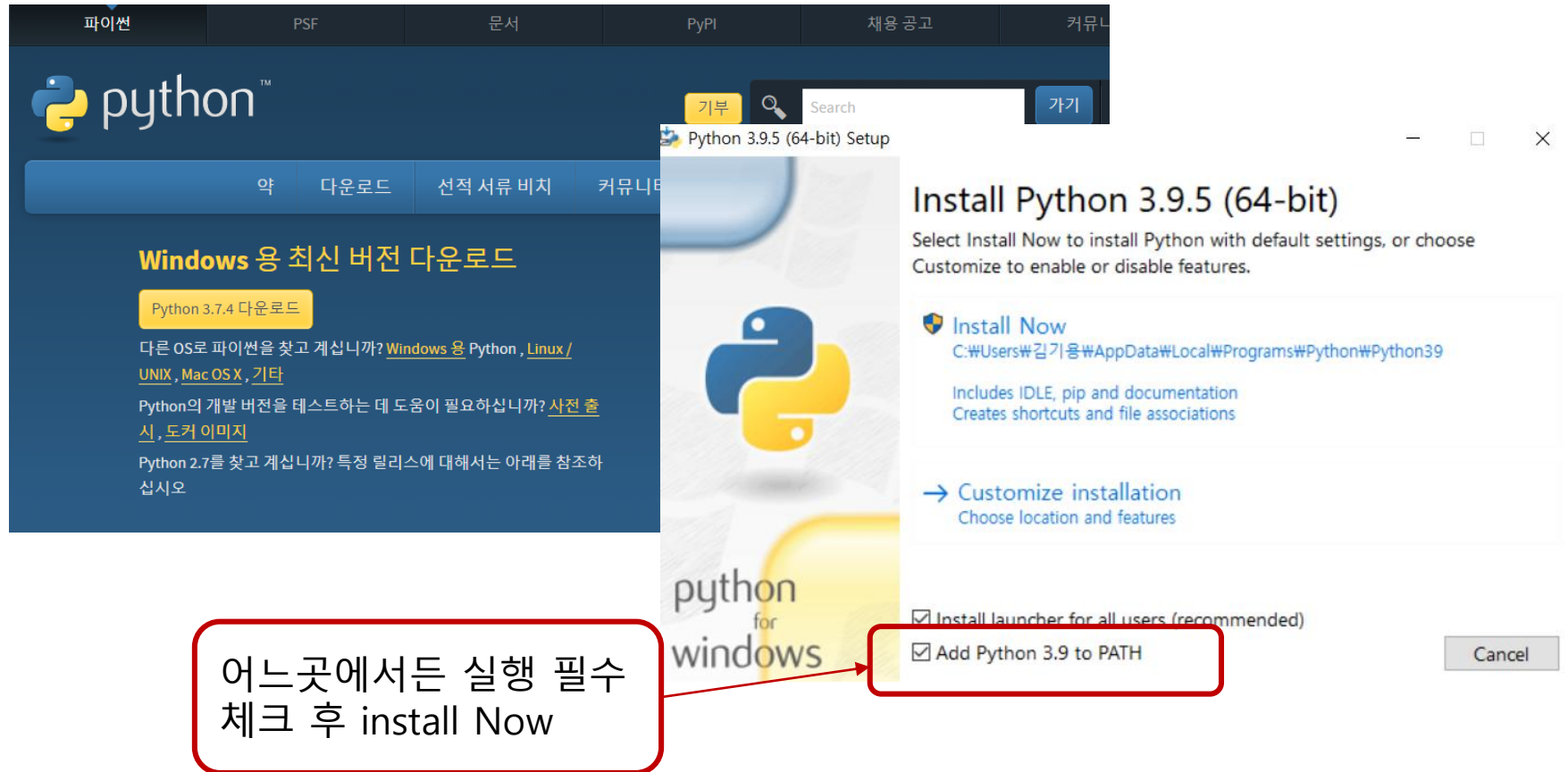
- **시스템 유틸리티 제작** – 윈도우 등에서 사용자에게 유용한 도구 (압축, 이미지뷰어 등)
- **웹 프로그래밍** – 웹 사이트 제작 등 웹 개발 (장고[Django] & 플라스크)
- **데이터 분석** – 데이터 수집, 분석 및 시각화(Pandas 모듈, matplotlib)
- **머신러닝(딥러닝)** – 인공지능 구현(텐서플로우, 케라스)
- **사물 인터넷** – IOT 구현(라즈베리파이[Raspberry Pi])

◆ 파이썬으로 할 수 없는 일

- **시스템 프로그래밍** – 운영체제 관련 제작(주로 C언어로 개발함)
- **모바일 프로그래밍** – 안드로이드, 아이폰 앱 등

파이썬 설치하기

- 파이썬 - www.python.org



파이썬 설치하기

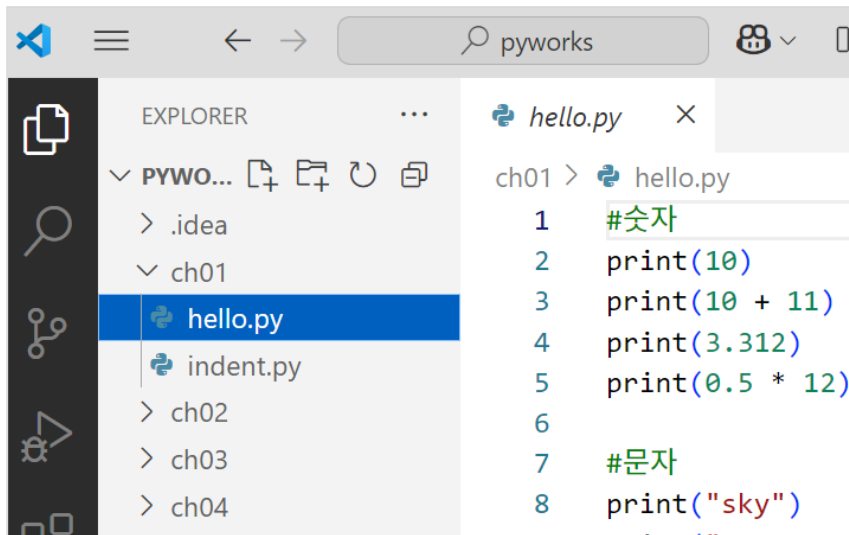
- 파이썬 설치 버전 확인 및 파이썬 실행하기

- 명령 프롬프트(cmd)

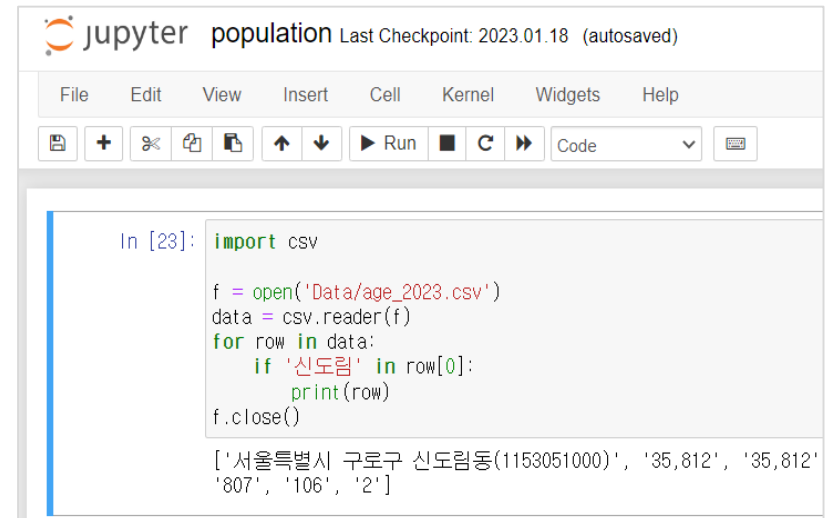
```
C:\Users\kiyon>python --version
Python 3.10.4

C:\Users\kiyon>python
Python 3.10.4 (tags/v3.10.4:9d38120, Mar 23 2022, 23:13:41)
Type "help", "copyright", "credits" or "license" for more in
>>> 10 + 20
30
>>> "안녕하세요"
'안녕하세요'
>>> a = 10
>>> b = 20
>>> a + b
30
```

파이썬 추천 에디터(Editor)



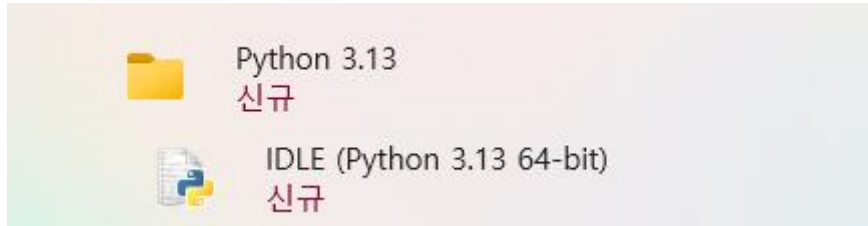
비주얼 스튜디오 코드(VS code)



주피터노트북(Jupyter Notebook)

파이썬 IDLE

■ 파이썬의 IDLE 실행

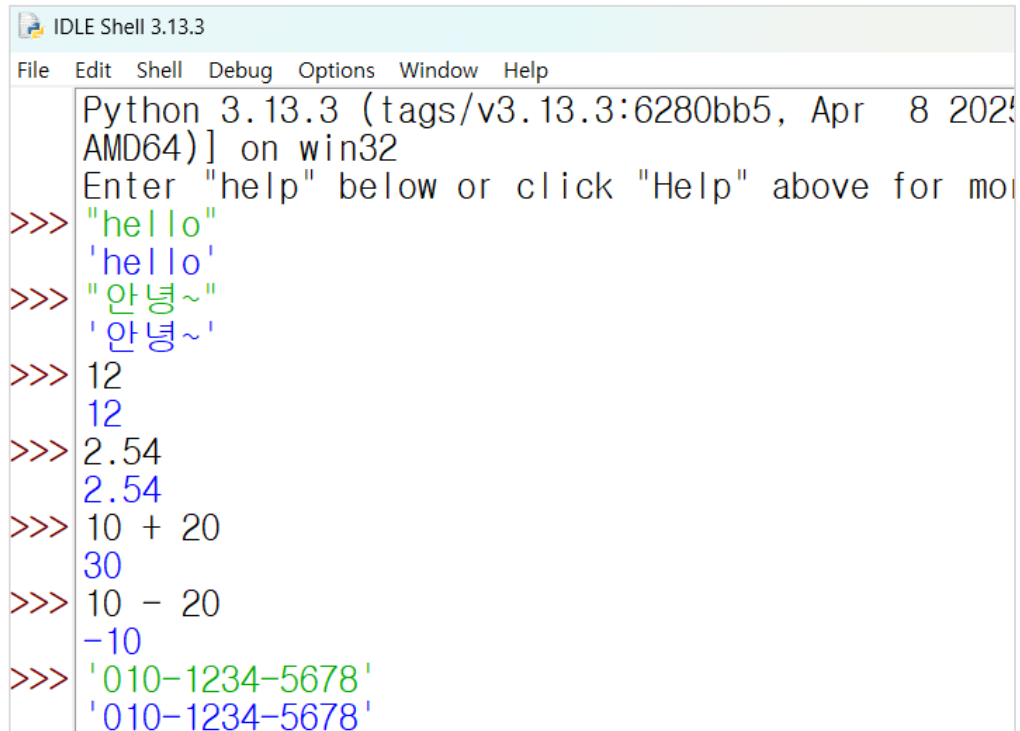


■ 파이썬의 셸

>>> 기호는 파이썬이 사용자가 입력하기를 기다리고 있다는 뜻이다.
사용자가 입력을 하면 바로 결과를 보여 주는데 이것을 대화형 셸(shell)이라고 함

파이썬 IDLE

■ 파이썬의 셸



```
Python 3.13.3 (tags/v3.13.3:6280bb5, Apr  8 2025, 10:00:00) on win32
Enter "help" below or click "Help" above for more
>>> "hello"
'hello'
>>> "안녕~"
'안녕~'
>>> 12
12
>>> 2.54
2.54
>>> 10 + 20
30
>>> 10 - 20
-10
>>> '010-1234-5678'
'010-1234-5678'
```

기초 문법 - 오류

✓ 구문 오류 및 변수 사용 오류

```
>>> 10 + 20
30
>>> 10 - 20
-10
>>> '010-1234-5678'
'010-1234-5678'
>>> 10 + 20 =
SyntaxError: cannot assign to expression
>>> hello
Traceback (most recent call last):
  File "<pyshe11#8>", line 1, in <module>
    hello
NameError: name 'hello' is not defined. Did you mean: 'help'?
>>>
```

에러 : 구문 오류

변수와 문자 구분

파이썬 IDLE - Editor

■ 에디터(Editor)

에디터는 프로그램을 작성하여 파일 형태로 저장하는 편집기이다.

1. File - > New File (새 파일)
2. 코드 작성
3. hello.py 로 저장 (pyworks 폴더 생성후)
4. Run -> Run Module(F5) : 실행

```
# print() 함수로 출력  
# 문자 출력  
print("Hello~ World!")  
print("안녕~ 세계야!")  
print('010-1234-5678')  
  
#숫자 출력  
print(12)  
print(2.54)  
print(10 + 20)  
print(10 - 20)
```

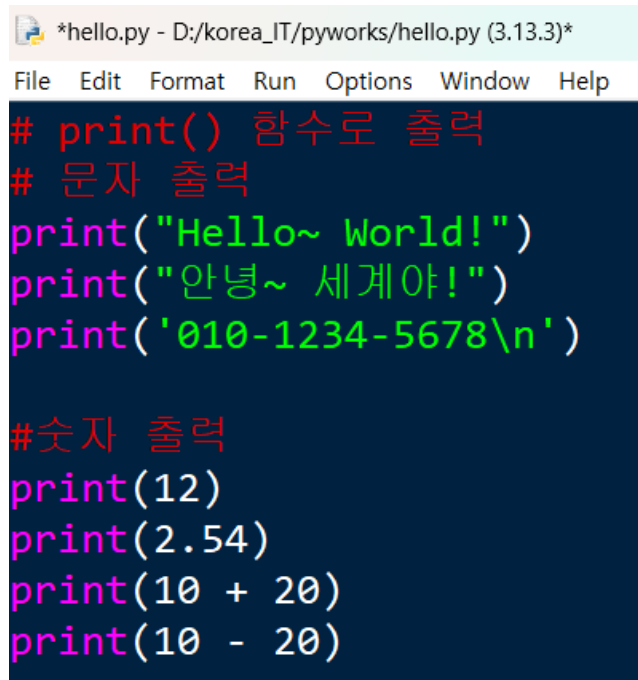
```
Hello~ World!  
안녕~ 세계야!  
010-1234-5678  
12  
2.54  
30  
-10
```

파이썬 IDLE – Editor

■ 에디터(Editor) 환경 설정

Options > Configure IDLE > Settings

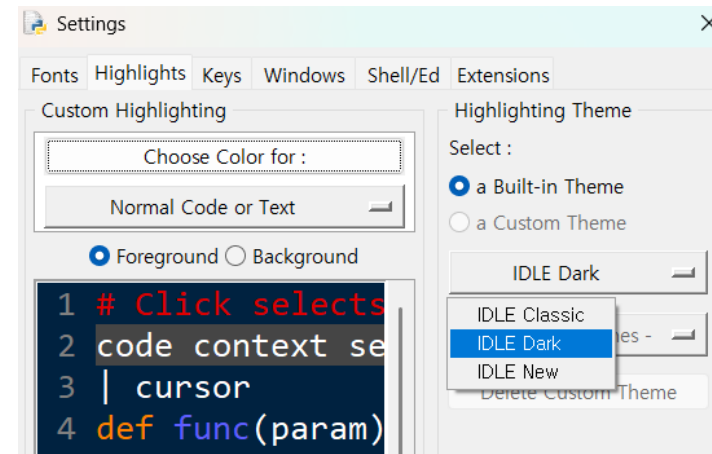
- 글꼴 변경 : (Fonts) - Consolas
- 화면 색상 변경 : (Highlights) – IDLE Dark



The screenshot shows the Python IDLE editor window with the title bar '*hello.py - D:/korea_IT/pyworks/hello.py (3.13.3)*'. The menu bar includes File, Edit, Format, Run, Options, Window, and Help. The code is as follows:

```
# print() 함수로 출력
# 문자 출력
print("Hello~ World!")
print("안녕~ 세계야!")
print('010-1234-5678\n')

#숫자 출력
print(12)
print(2.54)
print(10 + 20)
print(10 - 20)
```



기초 문법

■ 기본 문법

- **자료형**을 사용하지 않는다. ($n = 10$, $msg = 'hello'$)

- **주석**

한 줄 주석 : '#' 기호

여러 줄 주석 : `"""~"""`, (쌍따옴표 3번, 홀따옴표 3번 사용함)

- **들여쓰기(indent)**

4칸 들여쓰기

```
# 들여쓰기(indent)

n = 10
if n % 2 == 0:
    print("짝수")
else:
    print("홀수")
```

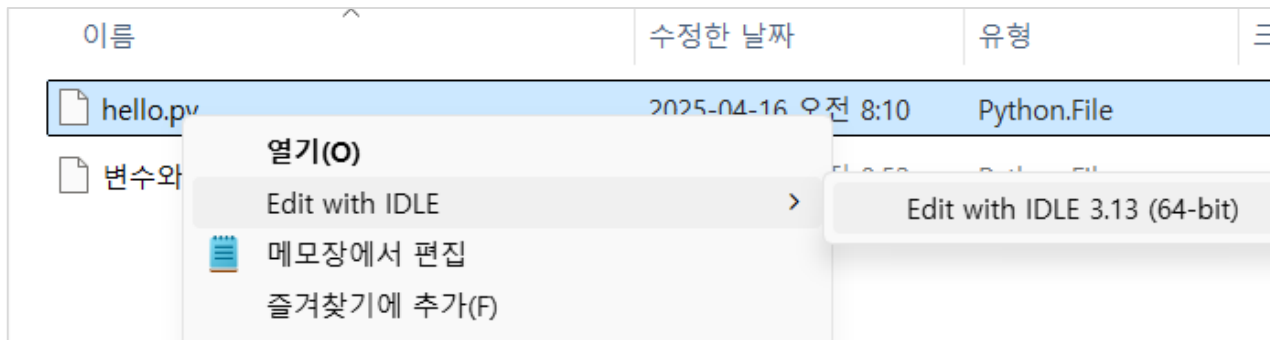
```
print('A')
print('B')
print('C')
```

1칸 들여쓰기로 에러

파이썬 파일 IDLE로 실행

◆ .py 파일 실행하기

파일 > 단축메뉴 > Edit with IDLE > Edit with IDLE 3.13



◆ 파이썬 Docs

- Documents > Python Docs > Tutorial(자습서)

Python 3.9.5 documentation

Welcome! This is the documentation for Python 3.9.5.

Parts of the documentation:

[What's new in Python 3.9?](#)

or all "What's new" documents since 2.0

[Tutorial](#)

start here

[Library Reference](#)

keep this under your pillow

[Language Reference](#)

describes syntax and language elements

[Python Setup and Usage](#)

how to use Python on different platforms

[Python HOWTOs](#)

in-depth documents on specific topics

[Installing Python Modules](#)

installing from the Python Package Index & other sources

[Distributing Python Modules](#)

publishing modules for installation by others

[Extending and Embedding](#)

tutorial for C/C++ programmers

[Python/C API](#)

reference for C/C++ programmers

[FAQs](#)

frequently asked questions (with answers!)

◆ 파이썬 Docs

▪ Tutorial(자습서) > Numbers

3.1.1. Numbers

The interpreter acts as a simple calculator: you can type an expression at it and it will write the value. syntax is straightforward: the operators `+`, `-`, `*` and `/` work just like in most other languages (for exam or C); parentheses `(())` can be used for grouping. For example:

```
>>> 2 + 2
4
>>> 50 - 5*6
20
>>> (50 - 5*6) / 4
5.0
>>> 8 / 5 # division always returns a floating point number
1.6
```

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 of the 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

Welcome to GitHub!
Let's begin the adventure

Enter your email
✓ sugu2000kr@naver.com


Create a password
✓

Enter a username
✓ sugu2000kr

Would you like to receive product updates and announcements via email?
Type "y" for yes or "n" for no
→ y

Continue

Here's your GitHub launch code, @sugu2000kr!



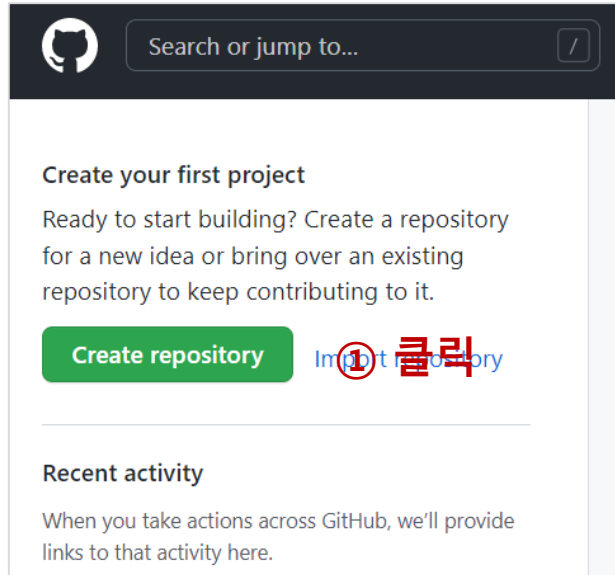
Continue signing up for GitHub by entering the code below:

93221781

Open GitHub

깃허브 원격 저장소 만들기

Repository(저장소) 만들기



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:\W>git

C:\W>git -version

* 사용자 확인

C:\W>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

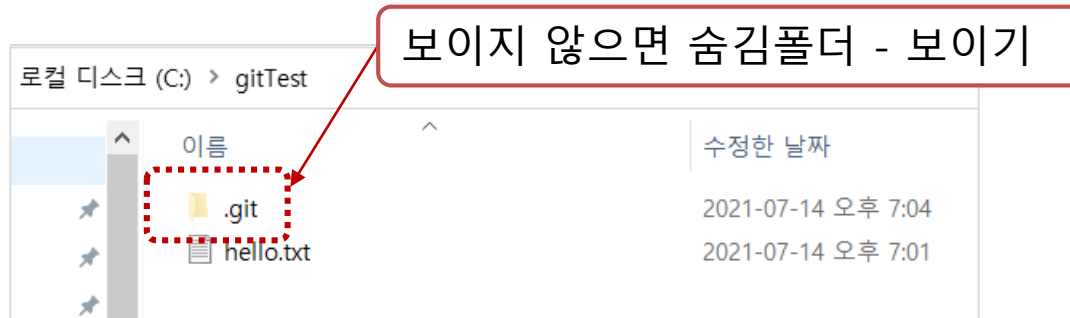
git config 명령은 컴퓨터 1대에서 처음 한번만 실행함

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

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

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

C:\W gitTest> git init #git 초기화하기



깃에 파일 업로드하기

■ 깃에 파일 업로드하기 – 처음 업로드시

> git **status** (상태 확인)

➤ git **add** hello.txt (파일 1개업로드시)
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

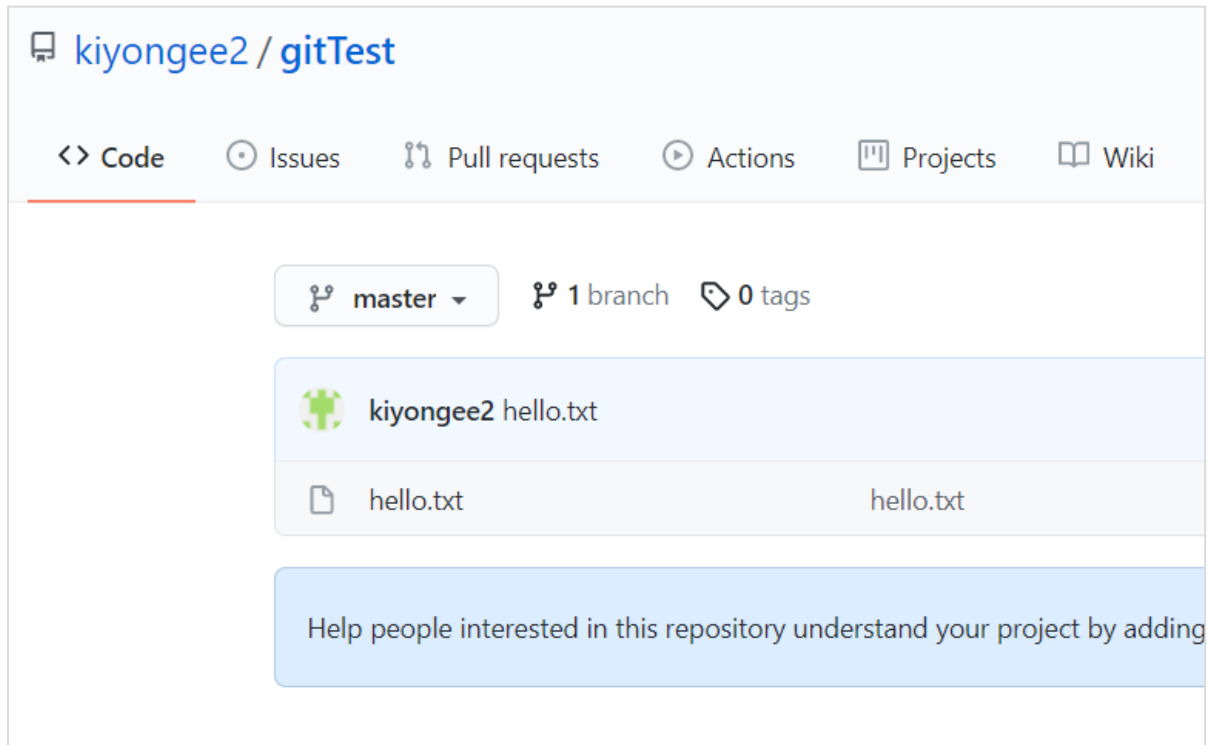
> git **add** *

> git **commit** -m "Add 추가 파일"

> git **push**

깃허브 레포지터리 보기

■ 업로드된 파일 확인하기



깃 파일 삭제

■ 파일 삭제하기

>git **rm** 파일이름

>git **commit -m** "Delete 파일이름"

>git **push**

■ 디렉터리 삭제하기

>git **rm -rf** 디렉터리 이름

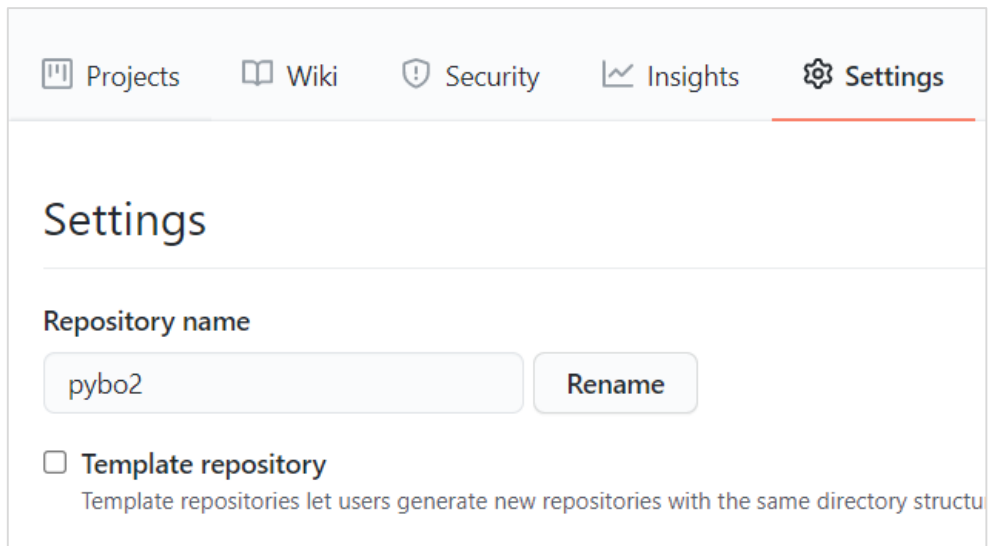
>git **commit -m** "Delete 디렉터리 이름"

>git **push**

깃 계정 이름 변경

■ 계정 이름 변경하기

Settings > 변경할 이름 > Rename



The screenshot shows the GitHub Settings page. 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 page title 'Settings' is displayed. Under the 'Settings' section, there is a 'Repository name' label. Below this label is a text input field containing the text 'pybo2'. To the right of the input field is a button labeled 'Rename'. Below the 'Repository name' section, there is a checkbox labeled 'Template repository'. The checkbox is currently unchecked. Below the checkbox, there is a descriptive text: 'Template repositories let users generate new repositories with the same directory structure'.

깃 계정 삭제

▪ 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


내 컴퓨터의 다른 사용자 계정 삭제하기


❖ 이미 사용중인 다른 사용자 계정 삭제하기

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

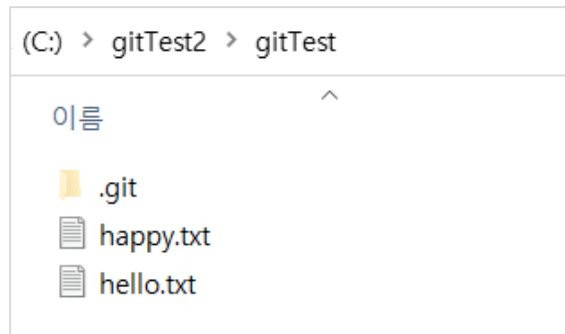
[편집](#) [제거](#)

깃 클론(git clone)

- 원격저장소에서 자료 가져오기

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

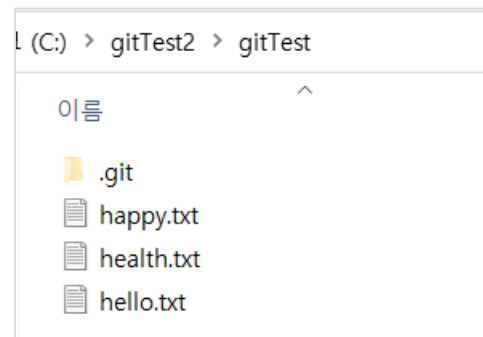
`c:\WgitTest2>git clone https://github.com/kiyongee2/gitTest`



gitTest에서
health.txt - 업로드

2번째 부터 추가 파일이 있는 경우

`c:\WgitTest2>git pull`



브랜치 이름 변경하기

■ 브랜치 master -> main으로 변경

개발을 하다 보면 코드를 여러 개로 복사해야 하는 일이 자주 생긴다.

코드를 통째로 복사하고 나서 원래 코드와는 상관없이 독립적으로 개발을 진행할 수 있는데, 이렇게 독립적으로 개발하는 것이 브랜치다.

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

```
*master
```

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

```
*main
```

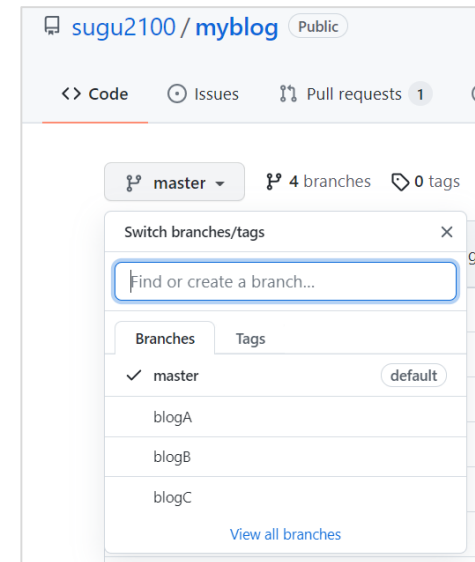
```
c:\WgitTest>git push -u origin main
```

새 브랜치 만들기

■ 새 브랜치 만들기

1. 새 브랜치 만들기 - **git branch** 브랜치 이름

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



2. blogA 원격 계정에 추가하기

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

브랜치 이동하기

- 브랜치 이동하기

blogA로 브랜치 이동 – git checkout 브랜치 이름

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

- 자료 수정후 깃에 업로드하기

```
c:\WgitTest>git add *
C:\WgitTest>git commit -m "추가"
C:\WgitTest>git push blogA
```