

1장. 자바(Java) 개발 환경 및 기본문법



JDK & Eclipse



프로그래밍과 자바

● 프로그래밍이란?

- 컴퓨터 프로그램을 만드는 일
- 컴퓨터에게 일을 하도록 명령어를 만드는 것

만약 사람이 원하는 바를 대략 말하면 컴퓨터가 알아서 프로그램을 만들어 준다면 좋은 것이다.

● 프로그래밍 언어의 종류

- C언어 , C++ , Java, Python, Javascript, C#

● 자바(Java)

- 1991년 제임스 고슬링을 비롯한 썬 마이크로시스템즈 연구원들이 개발했고, 당시에는 C, C++언어 사용되었고, 가전제품이나 휴대용 장치에 사용하는 독립적으로 작동하는 더 안정된 프로그래밍 언어가 필요했음
- 지금은 데이터베이스로 유명한 오라클사에 인수 되었음



자바란?

● 자바 언어의 특징

- 운영 체제에 독립적이다. – JVM(자바가상머신)이 가능하게 함
- 객체지향 언어이다. – 유지보수가 쉽고, 확장성이 좋다.
- 풍부한 기능이 제공되는 오픈 소스이다.
- 네트워크와 멀티 쓰레드를 지원하는 다양한 API(라이브러리)
- 안드로이드용 스마트폰 App(앱) 개발 언어로 사용되고 있다.



자바(Java)로 개발한 프로그램

▪ 웹 사이트(서버)

- 웹 사이트를 운영하려면 반드시 서버(server)가 필요하다.
- 검색 사이트, 쇼핑몰, 금융 사이트 등 자바로 개발한 웹 서버 프로그램으로 운영

• 안드로이드 앱

- 안드로이드 폰에서 사용하는 앱을 만들 수 있다.

• 게임

- 게임을 만들때는 C++, C를 주로 사용하지만 마인크래프트처럼 게임을 구현하는데도 사용된다.



자바 가상 머신(JVM)

◆ JVM(Java Virtual Machine)

- 자바 프로그램 실행 환경을 만들어 주는 소프트웨어
- 자바 코드를 컴파일한 .class(바이트 코드)는 JVM 환경에서 실행됨
- 컴퓨터의 운영체제에 맞는 자바 실행 환경(JRE)가 설치되어 있다면 자바 가상머신이 설치되어 있는 것이다. (JRE > JVM)

◆ JDK와 JRE

JDK(Java Development Kit) – 자바 개발을 위해 설치하는 라이브러리이다.

JRE(Java Runtime Environment) – 자바 프로그램이 실행되는 자바실행환경이다.

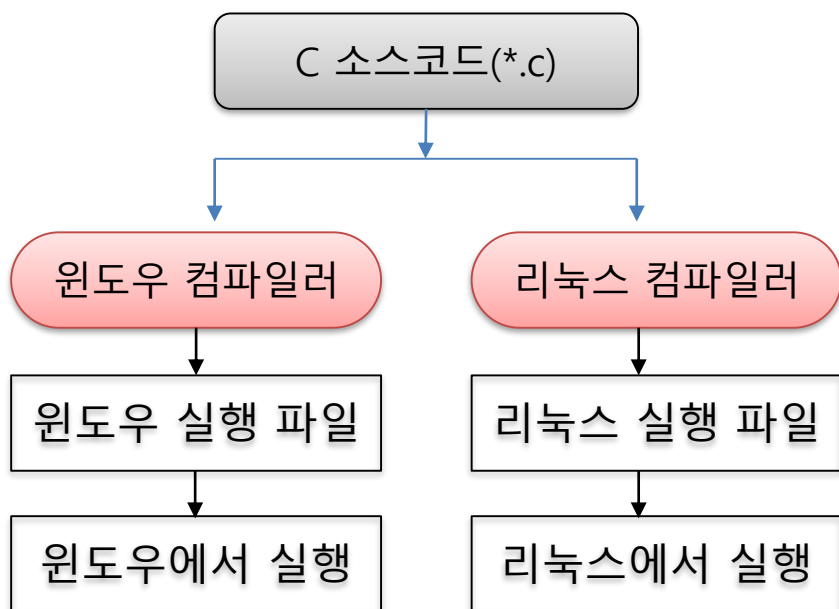
▶ 컴파일(Compile)과 컴파일러

컴파일은 프로그램(코드)를 컴퓨터가 알 수 있는 언어(기계어)로 바꿔 주는 일
컴파일러는 프로그램 언어를 기계어로 번역해 주는 프로그램으로 자바(JDK)를 설치하면 자바 컴파일러도 설치

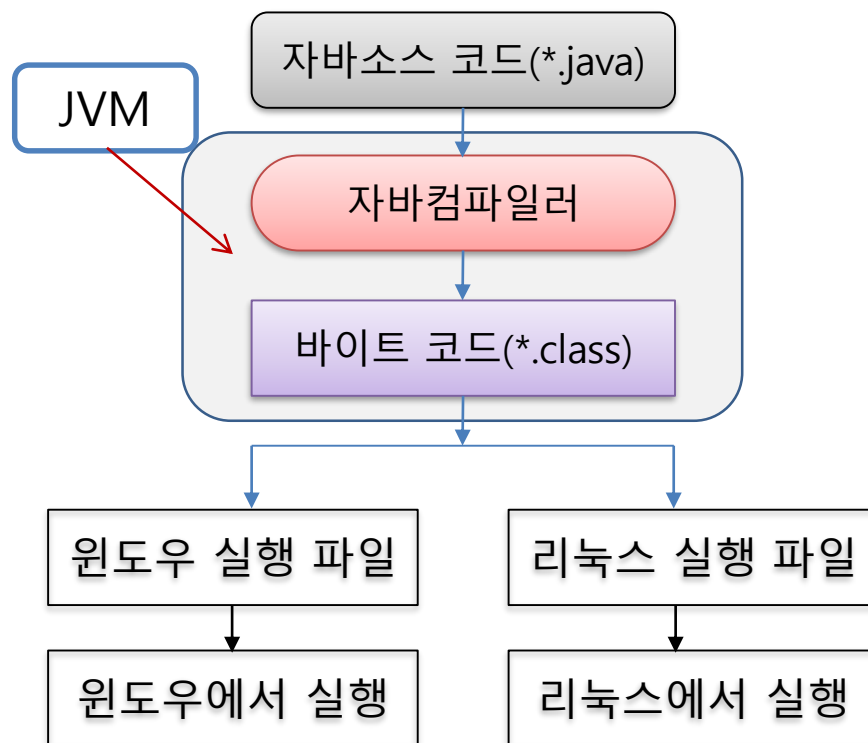


자바(Java) 언어

◆ JVM의 기능(역할)



구조적 언어-C언어



객체 지향 언어- Java, Python



자바 개발 환경 구축

◆ 자바 개발도구(JDK) 설치

- jdk 다운로드(검색)-> windows> Java SE21 다운로드 -> x64 인스톨러

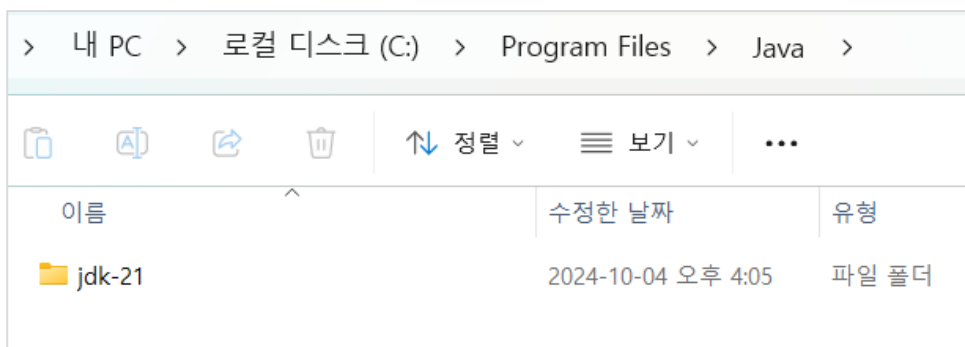
JDK 23	JDK 21	GraalVM for JDK 23	GraalVM for JDK 21
Java SE Development Kit 21.0.6 downloads			
JDK 21 binaries are free to use in production and free to redistribute, at no cost, under the Oracle No-Fee Terms and Conditions (NFTC).			
JDK 21 will receive updates under the NFTC, until September 2026, a year after the release of the next LTS. Subsequent JDK 21 updates will be licensed under the Java (OTN) and production use beyond the limited free grants of the OTN license will require a fee .			
Linux	macOS	Windows	
Product/file description		File size	Download
x64 Compressed Archive		185.92 MB	https://download.oracle.com/java/21/latest/jdk-21_windows-x64_bin.zip (sha256)
x64 Installer		164.31 MB	https://download.oracle.com/java/21/latest/jdk-21_windows-x64_bin.exe (sha256)
x64 MSI Installer		163.06 MB	https://download.oracle.com/java/21/latest/jdk-21_windows-x64_bin.msi (sha256)



자바 개발 환경 구축

◆ 자바 개발도구(JDK) 설치

- 설치 경로 : C:> Program Files > Java -> jdk-21



- 명령 프롬프트(cmd)로 설치 확인

```
C:\Users\LG>java -version
java version "21.0.4" 2024-07-16 LTS
Java(TM) SE Runtime Environment (build 21.0.4+8-LTS-274)
Java HotSpot(TM) 64-Bit Server VM (build 21.0.4+8-LTS-274, mixed mode, sharing)
```



자바 개발 환경 구축

◆ 자바 Documentation 설치

Java Api 를 설명하고 있는 문서

Documentation Download

Release information

- [Online Documentation](#)
- [Installation Instructions](#)
- [Release Notes](#)
- [Documentation License](#)

Java® Platform, Standard Edition & Java Development Kit Version 11 API Specification

This document is divided into two sections:

Java SE

The Java Platform, Standard Edition (Java SE) APIs define the core Java platform.

JDK

The Java Development Kit (JDK) APIs are specific to the JDK and will not necessarily have counterparts in the Java SE APIs whose names start with `jdk`.

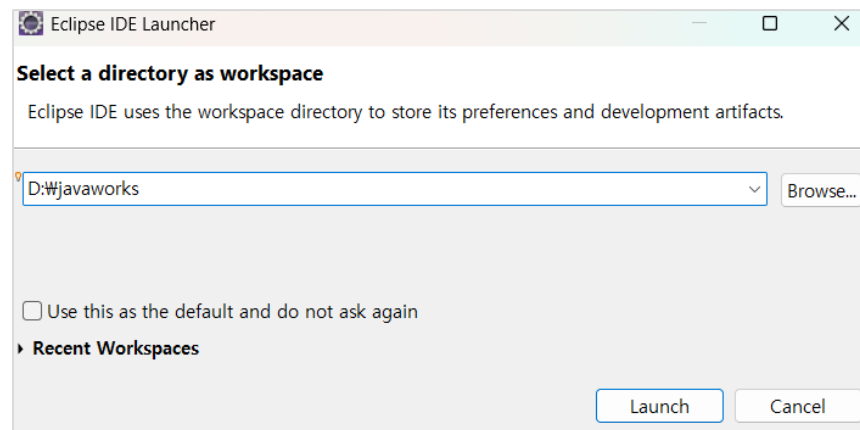
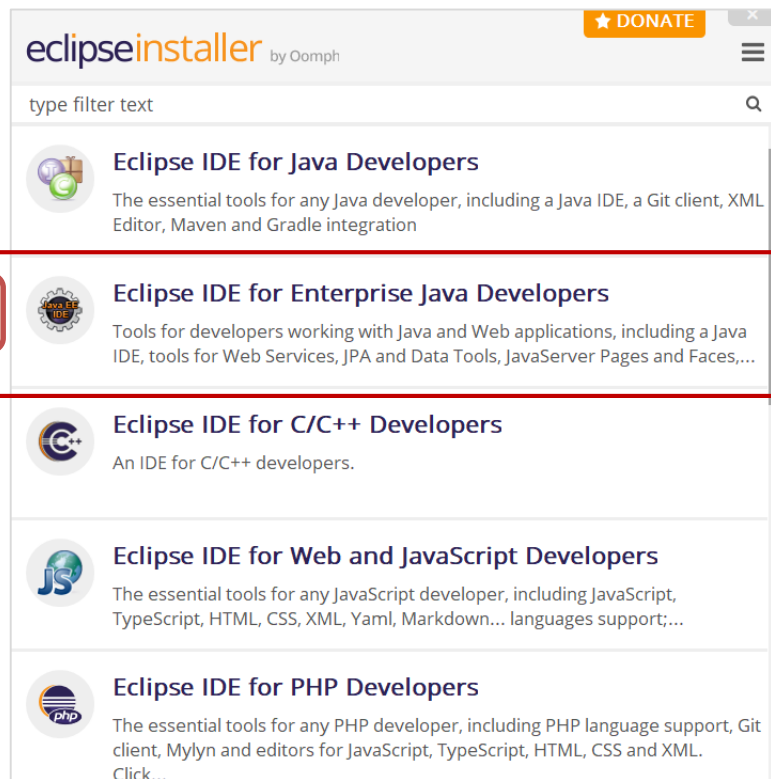
All Modules	Java SE	JDK	Other Modules
Module	Description		
<code>java.base</code>	Defines the foundational APIs of the Java SE Platform.		
<code>java.compiler</code>	Defines the Language Model, Annotation Processing, and Java Compiler APIs.		
<code>java.datatransfer</code>	Defines the API for transferring data between and within applications.		
<code>java.desktop</code>	Defines the AWT and Swing user interface toolkits, plus APIs for a subset of the Java 2D and JavaFX APIs.		



이클립스(Eclipse) IDE 설치

◆ 이클립스 IDE(통합개발환경) 설치

- 검색 _ 이클립스(<https://www.eclipse.org/downloads/>)
- 버전 - Eclipse IDE 2024-09



workspace(작업영역) – C:\jvaworks
폴더 생성

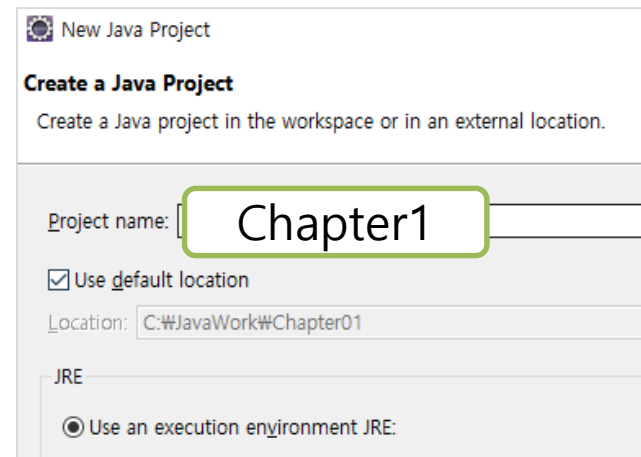
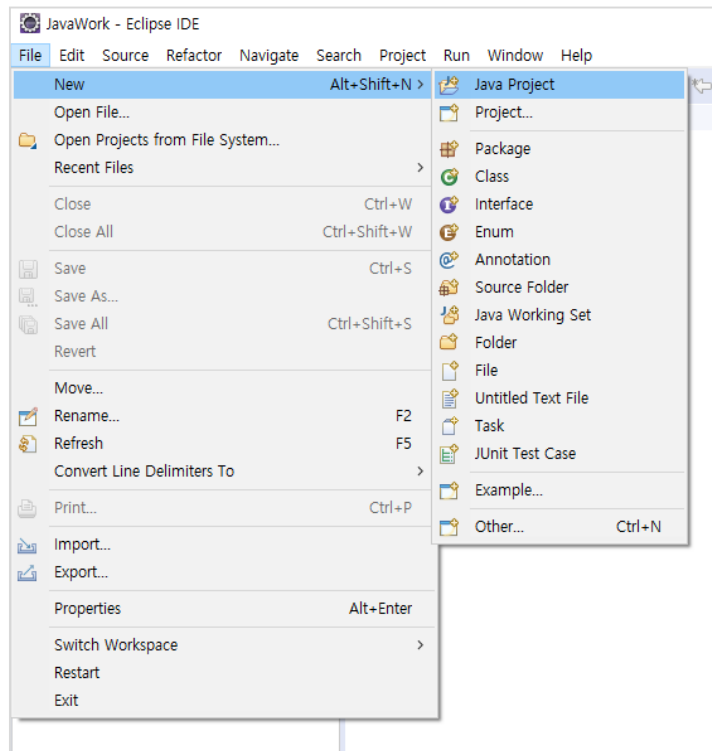


프로젝트 만들기

- 첫 자바 프로젝트(Project) 만들기

File->New->Java Project

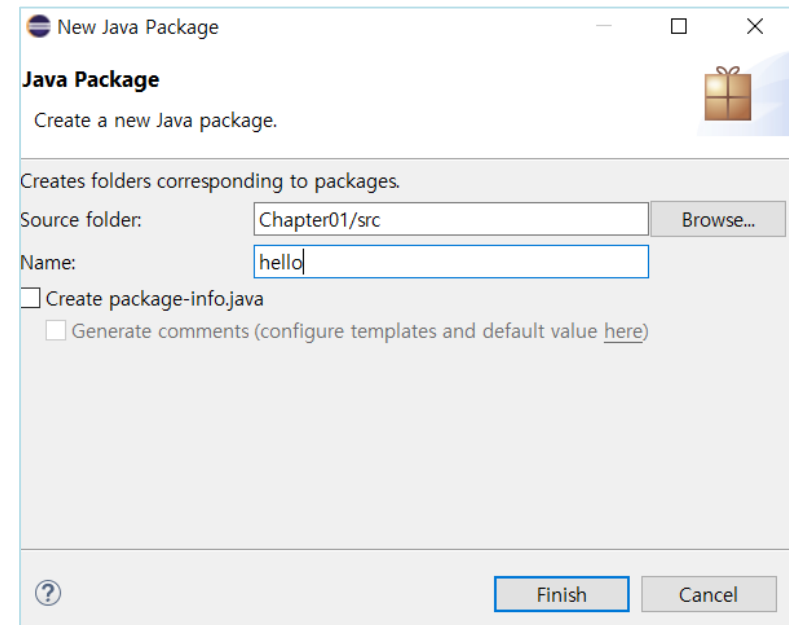
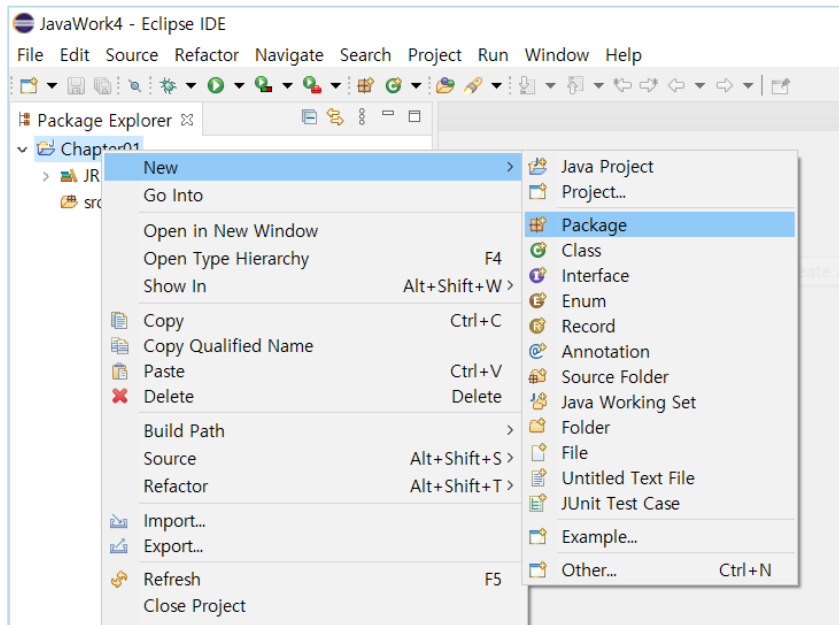
Project Name : Chapter01



첫번째 패키지 만들기

- 첫번째 패키지 만들기

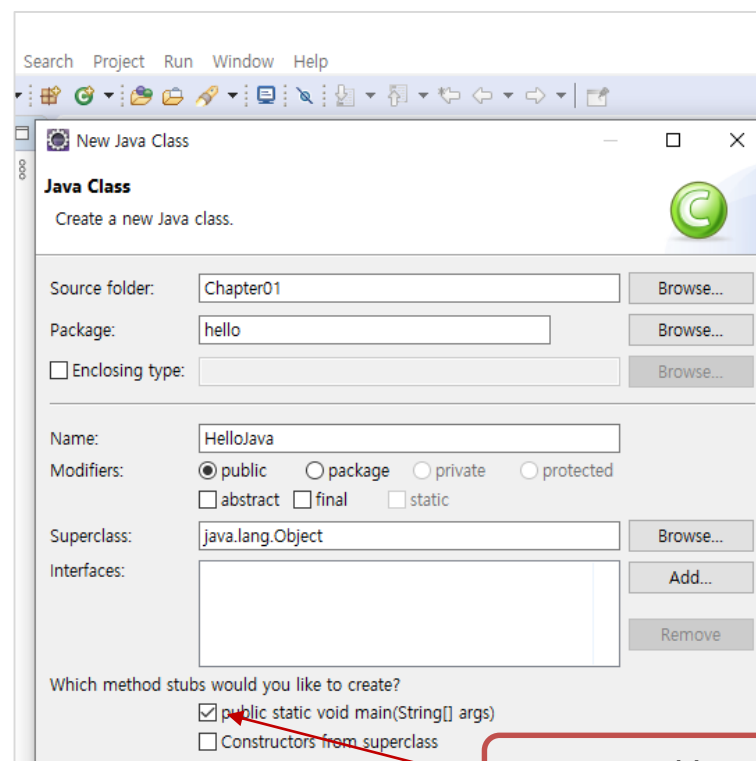
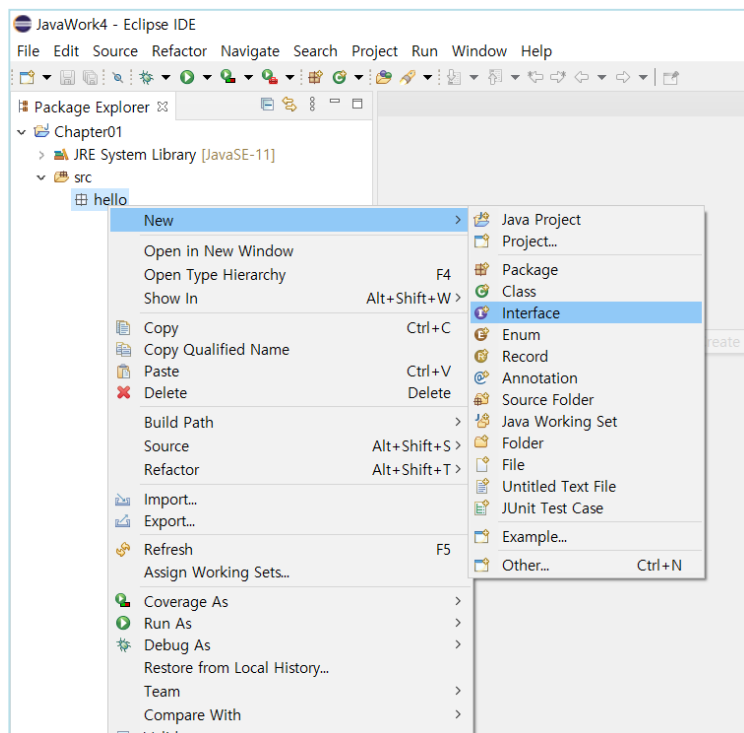
패키지 만들기 : Chapter01(마우스우측) -> New package -> Name : hello



첫번째 클래스 만들기

● 첫번째 클래스 만들기

클래스 만들기 : hello(우측)->New class->Name : HelloJava



main()함수 체크



첫번째 클래스 만들기

1. java 코드 작성

- 파일 이름 : HelloJava.java
- 클래스 : System , 패키지: out, 메서드(함수) : main(), print(), println()
- 파일 실행하려면 main() 메서드가 필요함

```
package hello;
```

패키지 이름

```
public class HelloJava{  
    public static void main(String[] args) {  
        System.out.println("Hello~ Java");  
    }  
}
```

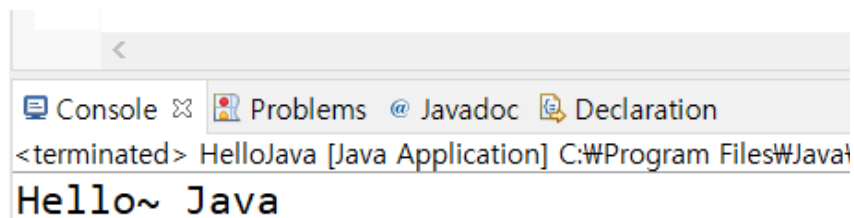
main()메서드



첫번째 클래스 만들기

2. 컴파일 및 실행

- 컴파일 하기 : 빌드 자동화 옵션 지정 -> 클래스 파일 생성
- 실행 : Run -> Run as -> Java Application [실행 단추(▶) 클릭]
- 실행 결과 : 콘솔(Console)



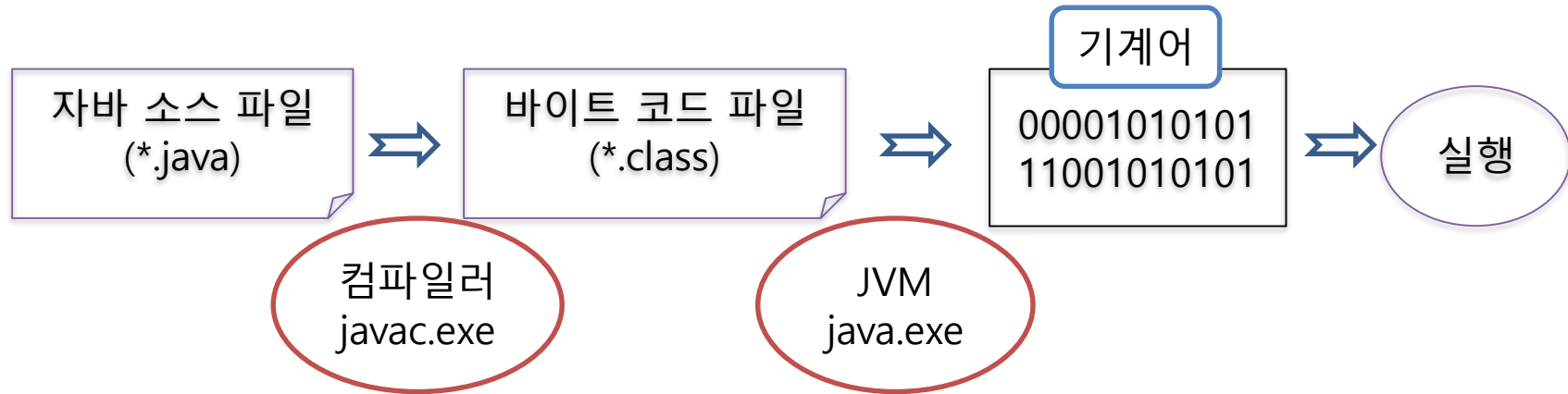
※ 클래스(.class) 파일의 위치는 어디일까?

내 PC > 로컬 디스크 (C:) > JavaWork4 > Chapter01 > bin > hello					hello 검색	
	이름	수정한 날짜	유형	크기		
	HelloJava.class	2020-12-10 오전 8:59	CLASS 파일	1KB		
	PrintData.class	2020-12-10 오전 9:00	CLASS 파일	1KB		



컴파일과 빌드

✓ 컴파일(compile)



▷ 바이트 코드 파일은 완전한 기계어가 아니므로 바로 실행할 수 있는 파일이 아니다.

✓ 빌드(build)

컴파일과 링크된 코드들을 실행가능한 파일로 만드는 일련의 과정으로 전처리, 컴파일, 패키징, 배포등이 포함된다.

Java 빌드 툴로는 Maven, Gradle 등이 있다.



주석 , 블록, 세미콜론(;)

● 기초 문법

- 주석은 소스코드에 설명을 추가하거나 특정 코드가 컴파일되지 않도록 처리할때 사용
 - 한줄 주석 : 문장 앞에 '//' 표시
 - 여러 줄 주석 : /*~ */ 기호 사용
- 문장이 종료는 세미콜론(;)을 사용
- { } 블록 안에 코드 작성



주식 , 블록, 세미콜론(;)

● 기초 문법

```
package hello;

/*
 * 파일명 - HelloJava.java
 * 만든이 - 김기용
 * 프로그램 - Hello~ World를 출력하는 프로그램
 */

public class HelloJava {

    public static void main(String[] args) {

        //System-클래스, out-패키지, println()-함수
        //print(), println()-줄바꿈
        //System.out.print("Hello~ World");
        System.out.println("Hello~ World");

        System.out.println("안녕~ 자바");
    }
}
```



데이터(data) 출력하기

```
package hello;

public class PrintData {

    public static void main(String[] args) {
        //데이터 처리 - 숫자, 문자, 불리언
        //정수, 실수
        System.out.println(100);
        System.out.println(3.3);

        //문자
        System.out.println('A');
        System.out.println('가');
        System.out.println("apple");

        //연산
        System.out.println(4 + 5);
        System.out.println(4 + "5");

        //불리언(참/거짓- true/false)
        System.out.println(true);
        System.out.println(5 < 4);
    }
}
```



System 클래스

Module `java.base`

Package `java.lang`

Class `System`

`java.lang.Object`
`java.lang.System`

```
public final class System
extends Object
```

The `System` class contains several useful class fields and methods. It provides access to system resources, output streams; access to externally defined properties and environment variables.

Since:
1.0

`java.base` > `java.lang` > `System`

<code>void</code>	<code>print(Object obj)</code>
<code>void</code>	<code>print(String s)</code>
<code>PrintStream</code>	<code>printf(String format, Object... args)</code>
<code>PrintStream</code>	<code>printf(Locale l, String format, Object... args)</code>
<code>void</code>	<code>println()</code>
<code>void</code>	<code>println(boolean x)</code>
<code>void</code>	<code>println(char x)</code>



연습 문제

실습 문제 : Java 개발 환경 구축

() 안에 들어갈 적당한 말을 맞춰보세요.

1. 프로그램(코드)을 기계가 이해할 수 있는 언어로 바꾸는 일을 ()이라고 한다.
2. 자바로 만든 프로그램은 ()이 설치되어 있으면 운영체제와 상관없이 실행할 수 있다.
3. 자바 개발을 위해 설치하는 자바 라이브러리를 () 라고 한다.

1.컴파일 2.JVM(자바가상머신) 3.JDK



깃(Git) 과 깃허브(Git Hub)사용



Git & Git Hub



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:

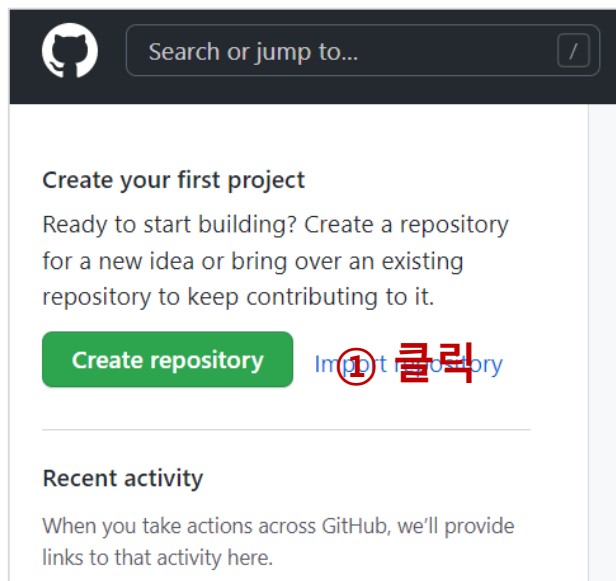
93221781

Open GitHub



깃허브 원격 저장소 만들기

Repository(저장소) 만들기



Search or jump to...

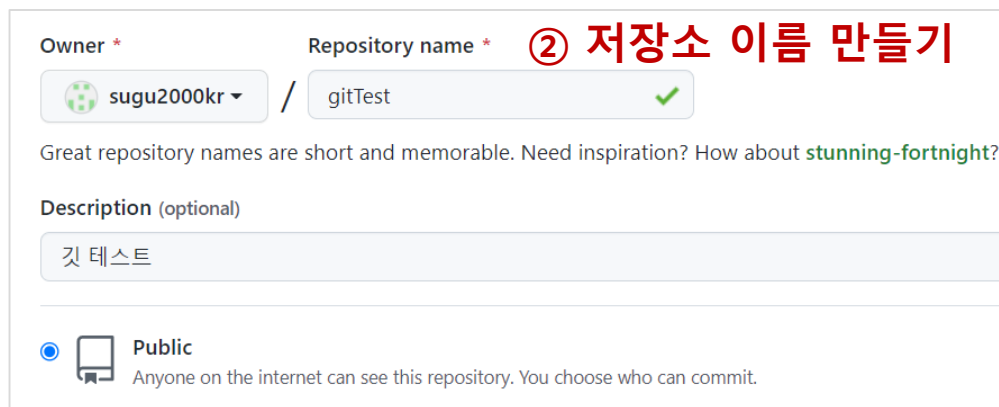
Create your first project

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

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: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

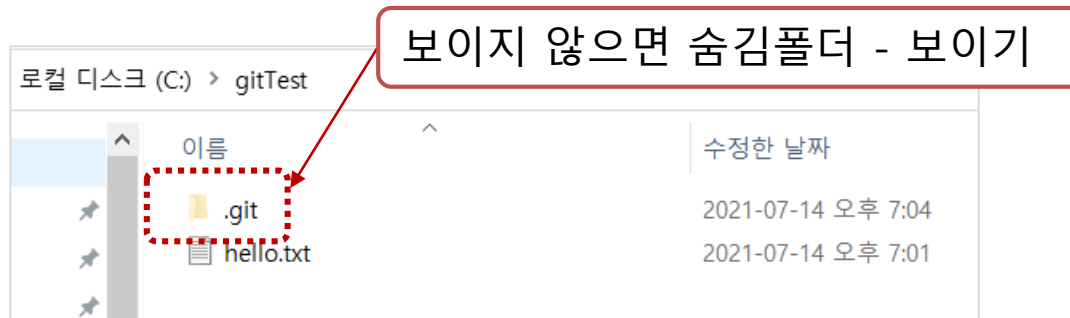
git config 명령은 컴퓨터 1대에서 처음 한번만 실행함

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** *

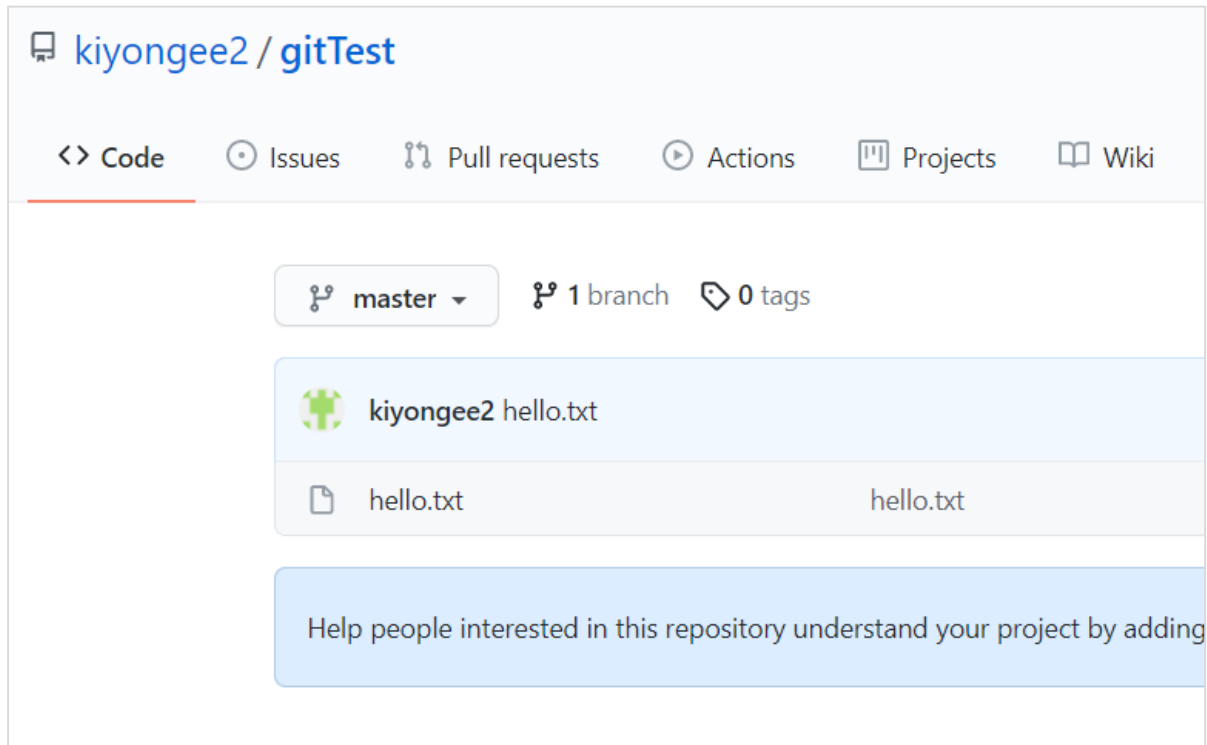
> git **commit** -m "Add 추가 파일"

> 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"

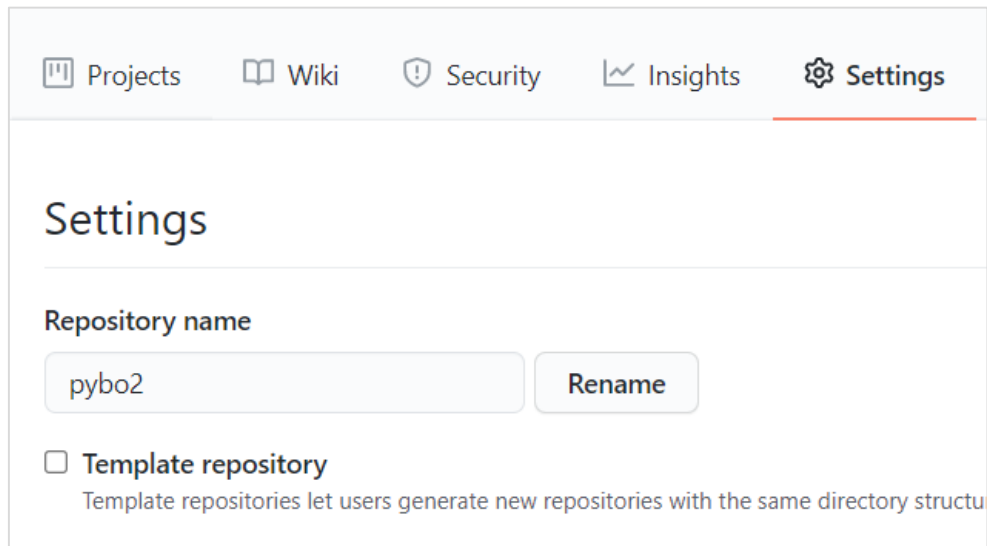
>git **push**



깃 계정 이름 변경

■ 계정 이름 변경하기

Settings > 변경할 이름 > Rename



The screenshot shows the GitHub 'Settings' page 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 page 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'.



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




기존 계정 삭제하기


■ 이미 사용중인 다른 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.com수정한 날짜: 오늘 ^

인터넷 또는 네트워크 주소: git:https://github.com
사용자 이름: sugu2100
암호:
지속성: 로컬 컴퓨터
[편집](#) [제거](#)

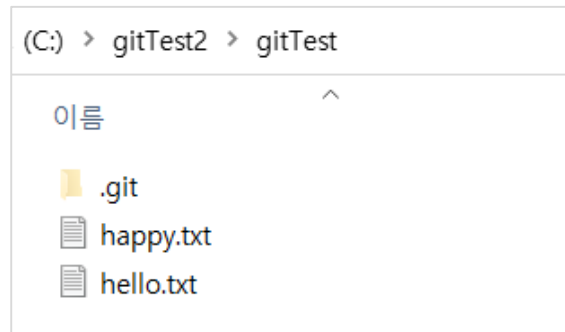


깃 클론(git clone)

- 원격저장소에서 자료 가져오기

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

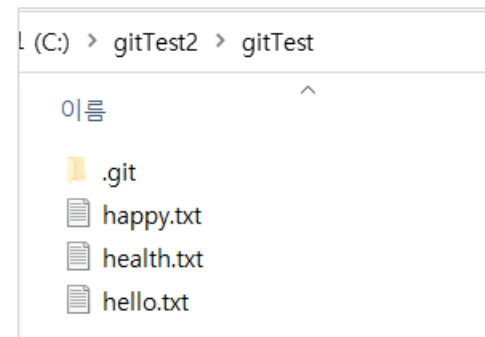
c:\WgitTest2> **git clone** https://github.com/kiyongee2/gitTest



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
```

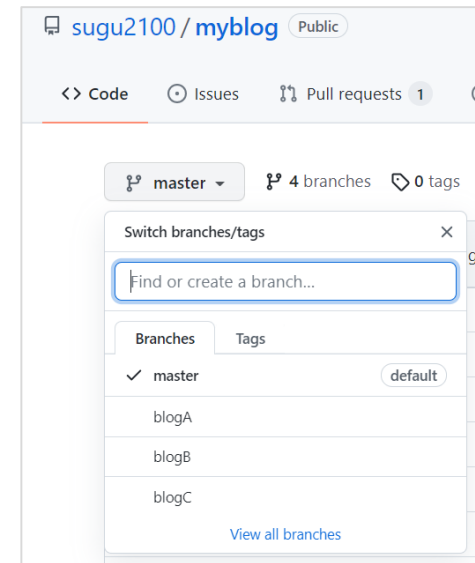


새 브랜치 만들기

■ 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
```



실습문제 1 – 깃(git) 설치

- 실습 예제

깃(git) 소프트웨어 설치하고, 깃허브 리포지터리 계정에서 다운로드 받기

URL: https://github.com/kiyongee2/korea_it_java-.git

