

JAVA 기초

프로그램은 사람이 이해하는 코드를 작성.
느려도 꾸준하면 경기에서 이긴다.

Content

1. 프로그램 기초

1. 프로그램이란
2. JAVA
3. JAVA 설치
4. 개발 Tool 설치
5. 첫번째 프로그램
6. 자바 프로그램 구조
7. JVM
8. 기본 입출력

1. 프로그램이란

1. 프로그램 기초 1-1. 프로그램이란

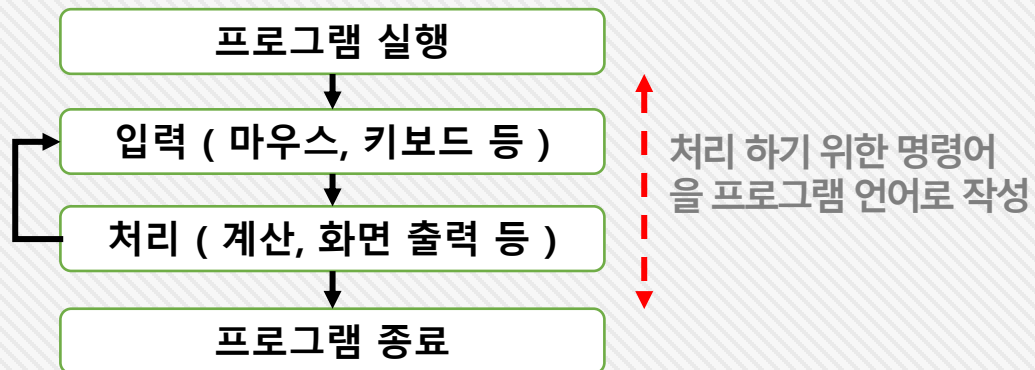
“ 어떤 목적을 달성 하기 위해서 프로그램 언어로 진행 순서를 작성 한 것 ”

사전적 의미

- 목록, 순서, 예정 계획 이란 뜻
- Rapp & Poertner(1992) - 특정 목표를 달성하기 위한 활동의 집합체
- York(1983) - 목표를 달성하기 위한 일련의 상호 의존적인 활동
- Smith(1989) - 특정 목표를 달성 하기 위해서 만들어진 조직적인 활동

프로그램 파일 (.exe)

- 어떤 목적을 수행 하기 위해 만든 파일



컴퓨터 프로그램

- 어떤 작업을 하기 위한 일련의 순서를 컴퓨터에게 알려 주기 위한 파일
- 일련의 순서를 컴퓨터가 이해 할 수 있는 명령어들의 모음
- 컴퓨터가 이해 할 수 있는 명령어는 0, 1로 되어 있는데 사람이 이해 할 수 있는 언어를 프로그램 언어라 한다.
- 두산백과 사전 : 컴퓨터를 실행 시키기 위해 차례대로 작성된 명령어 모음

프로그램 종류

- 시스템 프로그램
 - 컴퓨터 시스템과 하드웨어들을 제어 및 관리 하는 프로그램
 - 예) 윈도우, 리눅스, 장치 드라이버, 컴파일러 등
- 응용 프로그램
 - 사용자가 원하는 기능을 제공하는 프로그램
 - 엑셀, 게임, 워드 등

프로그램과 소프트웨어

- 프로그램
 - 컴파일된 결과물뿐만 아니라, 프로그래머가 작성한 소스 코드까지도 포함.
- 소프트웨어
 - 프로그램뿐만 아니라 CD, 설명서, 제품 포장 등 패키지 전체.

2. 프로그램 언어

1. 프로그램 기초 1-1. 프로그램 이란

“ 사람이 이해 할 수 있는 표현법을 사용 하여 프로그래밍 할 수 있는 언어 ”

의미

- 컴퓨터 시스템을 구동 시키는 소프트웨어를 작성하기 위한 형식언어
- 컴퓨터를 이용하여 특정 문제를 해결하기 위한 프로그램을 작성하기 위해 사용되는 언어

```
b8 21 0a 00 00
a3 0c 10 00 06
b8 6f 72 6c 64
a3 08 10 00 06
b8 6f 2c 20 57
a3 04 10 00 06
b8 48 65 6c 6c
a3 00 10 00 06
b9 00 10 00 06
ba 10 00 00 00
bb 01 00 00 00
b8 04 00 00 00
cd 80
b8 01 00 00 00
cd 80
```

```
MONITOR FOR 6802 1.4      9-14-80  TSC ASSEMBLER  PAGE    2

C000                      ORG      ROM+$0000 BEGIN MONITOR
C000 8E 00 70  START      LDS      #STACK

*****
* FUNCTION: INITA - Initialize ACIA
* INPUT: none
* OUTPUT: none
* CALLS: none
* DESTROYS: acc A

0013      RESETA  EQU      %00010011
0011      CTLREG  EQU      %00010001

C003 86 13      INITA    LDA  A  #RESETA  RESET ACIA
C005 B7 80 04          STA  A  ACIA
C008 86 11          LDA  A  #CTLREG   SET 8 BITS AND 2 STOP
C00A B7 80 04          STA  A  ACIA

C00D 7E C0 F1          JMP      SIGNON  GO TO START OF MONITOR
```

언어 종류

• 저급 언어

- 기계어

- 컴퓨터가 직접 이해할 수 있는 언어
- 0과 1의 2진수 형태로 표현되며 수행시간이 빠르다.
- CPU에 내장된 명령들을 직접 사용하는 것으로, 프로그램을 작성하고 이해하기가 어렵다.
- 기종마다 기계어가 다르므로 언어의 호환성이 없다.

- 어셈블리어

- 기계어와 1:1로 대응되는 기호로 이루어진 언어로, 니모닉(Mnemonic) 언어
- 하드웨어 제어에 주로 사용되며, 언어의 호환성이 없다.
- 컴퓨터가 직접 이해할 수 없으므로 어셈블리어로 작성된 프로그램은 어셈블러를 사용하여 기계어로 번역해주어야 한다.

• 고급 언어

- 컴파일러 언어 라고도 하며, 인간이 실생활에서 사용하는 자연어와 비슷한 형태 및 구조를 가지고 있다.
- 하드웨어에 대한 깊은 지식이 없어도 프로그램 작성과 수정이 용이
- 컴퓨터가 이해할 수 있는 기계어로 번역하기 위해 컴파일러나 인터프리터가 사용
- 기계어와 어셈블리어를 제외한 C, JAVA, Python등의 언어가 고급언어

2. 프로그램 언어

1. 프로그램 기초 1-1. 프로그램 이란

“ 사람이 이해 할 수 있는 표현법을 사용 하여 프로그래밍 할 수 있는 언어 ”

컴파일러

- 컴파일러는 고급 언어로 작성된 프로그램 전체를 목적 프로그램으로 번역한 후, 링킹 작업을 통해 컴퓨터에서 실행 가능한 실행 프로그램을 생성
- 번역 실행 과정을 거쳐야 하기 때문에 번역 과정이 번거롭고 번역 시간이 오래 걸리지만, 한번 번역한 후에는 다시 번역하지 않으므로 실행 속도가 빠르다.
- 컴파일러를 사용하는 언어에는 C언어 Java 등

컴파일러와 인터프리터 차이점

구분	컴파일러	인터프리터
번역단위	전체	행(줄)
목적 프로그램	생성함	생성하지 않음
실행속도	빠름	느림
번역속도	느림	빠름
관련언어	C, JAVA	Python, BASIC, LISP, APL, SNOBOL

인터프리터

- 인터프리터는 고급 언어로 작성된 프로그램을 한 줄 단위로 받아들여 번역하고, 번역과 동시에 프로그램을 한 줄 단위로 즉시 실행시키는 프로그램.
- 프로그램이 직접 실행되므로 목적 프로그램은 생성되지 않음
- 줄 단위로 번역, 실행되기 때문에 시분할 시스템에 유용하며 원시 프로그램의 변화에 대한 반응이 빠르다
- 번역 속도는 빠르지만 프로그램 실행 시 매번 번역해야 하므로 실행 속도는 느리다.
- CPU의 사용시간의 낭비가 크다.
- 인터프리터를 사용하는 언어에는 Python, BASIC, SNOBOL, LISP, APL 등

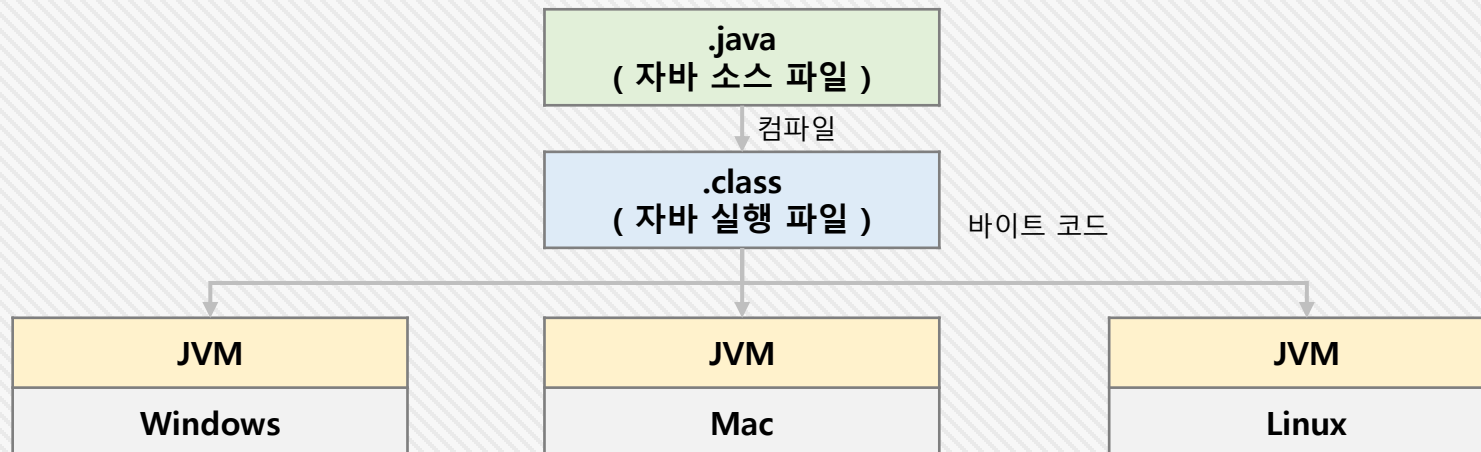
“ 한 번 적성하면 어느 플랫폼에서나 실행 (플랫폼 독립성) ”

JAVA

- 썬 마이크로시스템즈의 제임스 고슬링(James Gosling)과 다른 연구원들이 개발한 객체 지향적 프로그래밍 언어
- 컴파일된 코드가 플랫폼 독립적
- 자바로 개발된 프로그램은 CPU나 운영 체제의 종류에 관계없이 JVM(Java Virtual Machine)을 설치할 수 있는 시스템에서는 어디서나 실행

철학 (자바 언어의 5가지 핵심 목표)

- 객체 지향 방법론을 사용해야 한다.
- 같은 프로그램(바이트코드)이 여러 운영 체제(마이크로프로세서)에서 실행될 수 있어야 한다.
- 컴퓨터 네트워크 접근 기능이 기본으로 탑재되어 있어야 한다.
- 원격 코드를 안전하게 실행할 수 있어야 한다.
- 다른 객체 지향 언어들의 좋은 부분만 가지고 와서 사용하기 편해야 한다.



“ 자바 개발 도구와 개발 실행 환경 ”

JDK(Java Development Kit)

- Java 환경에서 돌아가는 프로그램을 개발하는 데 필요한 툴들을 모아 놓은 소프트웨어 패키지
- JRE(Java Runtime Environment)와 Java 바이트코드 컴파일러, Java 디버거 등을 포함하는 개발 도구를 포함 하고 있다.
- Oracle JDK는 상업적 이용을 할 경우 유료
- 무료로 이용하고자 할 경우
 - Oracle의 OpenJDK 빌드, Zulu JDK, AdoptOpenJDK 등의 OpenJDK 기반 빌드를 이용
 - OpenJDK는 GPL 라이선스이지만 classpath exception이 적용되므로 Oracle의 지원이 메이저 버전이 올라가는 6개월마다 끊기는 점 외에는 자유롭게 사용

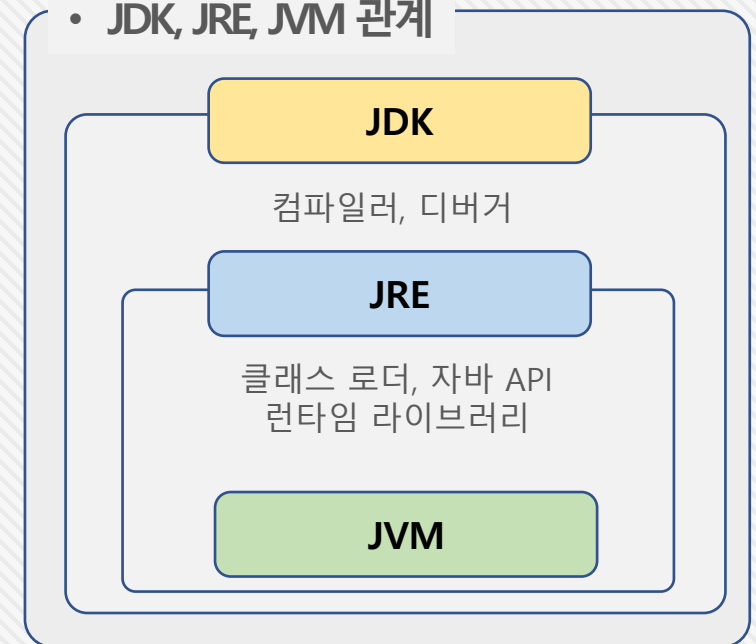
JRE(Java Runtime Environment)

- 컴퓨터의 운영체제 소프트웨어 상에서 실행되고 클래스 라이브러리 및 특정 Java 프로그램이 실행해야 하는 기타 리소스를 제공하는 소프트웨어 계층
- JDK를 사용하여 작성된 Java 코드를 JVM에서 이의 실행에 필요한 필수 라이브러리와 결합한 후 결과 프로그램을 실행하는 JVM의 인스턴스를 작성
 - 수정 없이도 어떤 운영체제에 서든 Java 프로그램을 실행

JVM(Java Virtual Machine)

- 자바 바이트코드를 실행하는 실행기
- 개발자가 이해하는 자바 언어를 JVM이 이해하는 자바 바이트코드로 번역

• JDK, JRE, JVM 관계



1. 자바 설치

1. 프로그램 기초 1-3. JAVA 설치

JAVA 설치

1. OPEN JDK 다운로드

* 다운로드 : <http://jdk.java.net/java-se-ri/11>

jdk.java.net

GA Releases
JDK 17
JMC 8
Early-Access Releases
JDK 19
JDK 18
Loom
Metropolis
Panama
Valhalla
Reference Implementations
Java SE 17
Java SE 16
Java SE 15
Java SE 14
Java SE 13
Java SE 12
Java SE 11
Java SE 10
Java SE 9
Java SE 8
Java SE 7
Feedback
Report a bug
Archive

Java Platform, Standard Edition 11 Reference Implementations

The official Reference Implementation for Java SE 11 (JSR 384) is based solely upon open-source code available from the [JDK 11 Project](#) in the [OpenJDK Community](#). This Reference Implementation applies to both the Final Release of JSR 384 (Sep 2018) and Maintenance Release 1 (Mar 2019).

The binaries are available under the GNU General Public License version 2, with the [Classpath Exception](#).

These binaries are for reference use only!

These binaries are provided for use by implementers of the Java SE 11 Platform Specification and are for reference purposes only. This Reference Implementation has been approved through the Java Community Process. Production-ready binaries under the GPL are available from Oracle; and will be in most popular Linux distributions.

RI Binary (build 11+28) under the GNU General Public License version 2

- Linux/x64 Java Development Kit (sha256) 178.9 MB
- ① Windows/x64 Java Development Kit (sha256) 178.7 MB**

RI Source Code

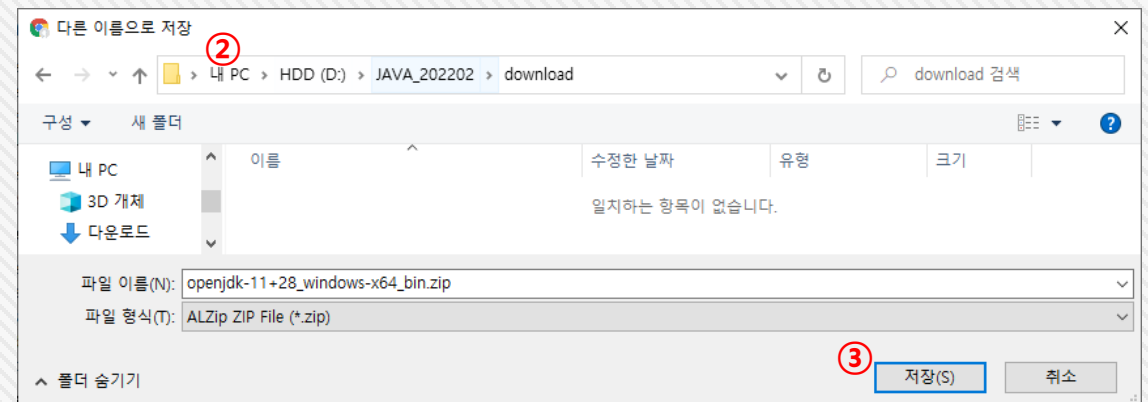
The source code of the RI binaries is available under the GPLv2 in a single zip file (sha256) 178.1 MB.

International use restrictions

Due to limited intellectual property protection and enforcement in certain countries, the JDK source code may only be distributed to an authorized list of countries. You will not be able to access the source code if you are downloading from a country that is not on this list. We are continuously reviewing this list for addition of other countries.

- ① Windows/64 java Development 클릭
- ② D:\JAVA_202202\download 선택
- 폴더가 없으면 생성
- ③ 저장 버튼 클릭

➤ 설치 폴더 : D:\JAVA_202202\download



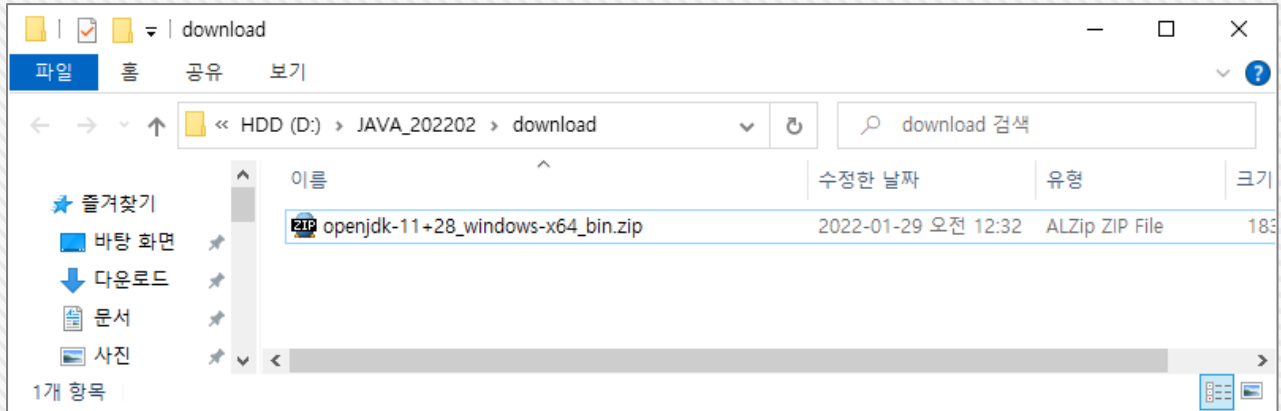
1. 자바 설치

1. 프로그램 기초 1-3. JAVA 설치

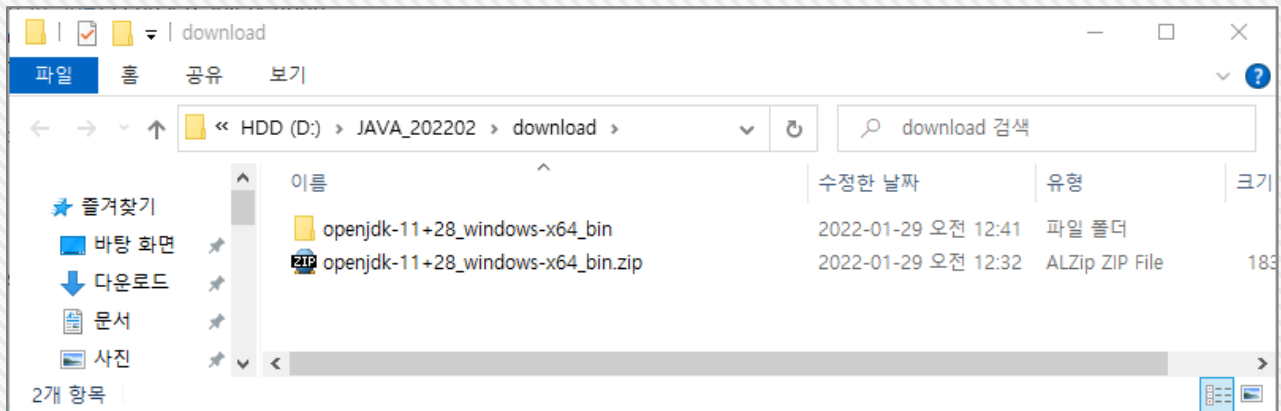
JAVA 설치

2. 다운로드 받은 파일 압축 풀기

① 다운 받은 폴더를 탐색기로 연다.



② 압축 해제를 한다.



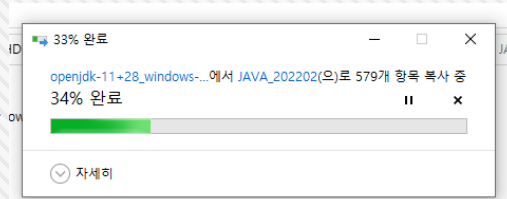
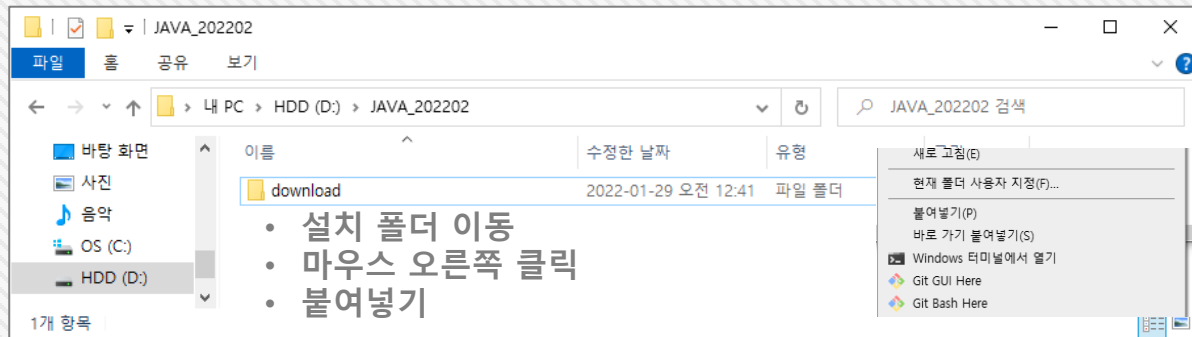
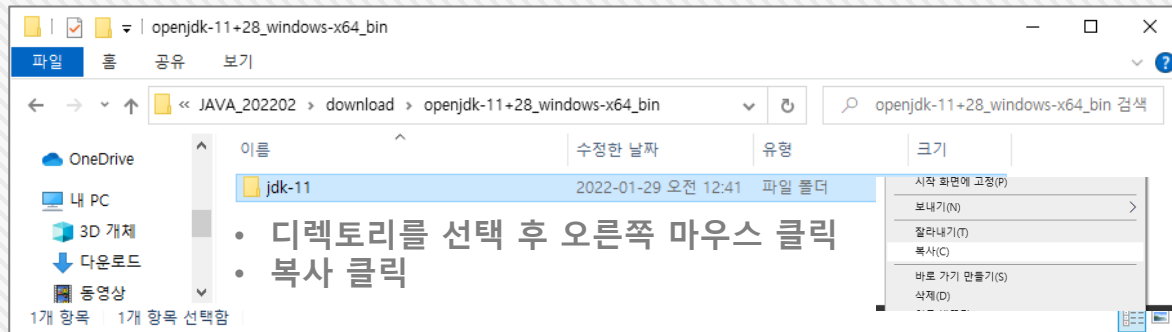
1. 자바 설치

1. 프로그램 기초 1-3. JAVA 설치

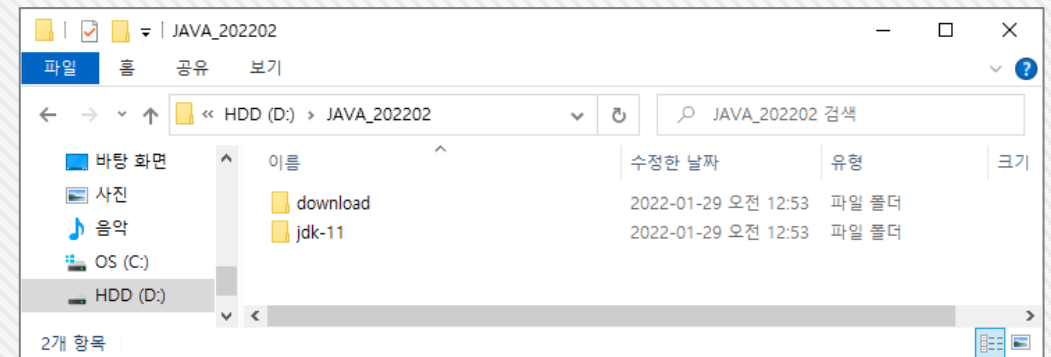
JAVA 설치

3. JAVA 설치

① D:WJAVA_202202Wjdk-11 로 copy 한다.



* 최종 모습



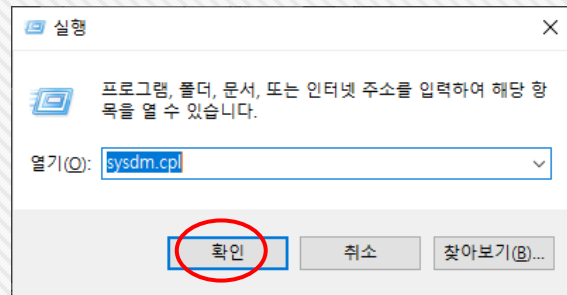
1. 자바 설치

1. 프로그램 기초 1-3. JAVA 설치

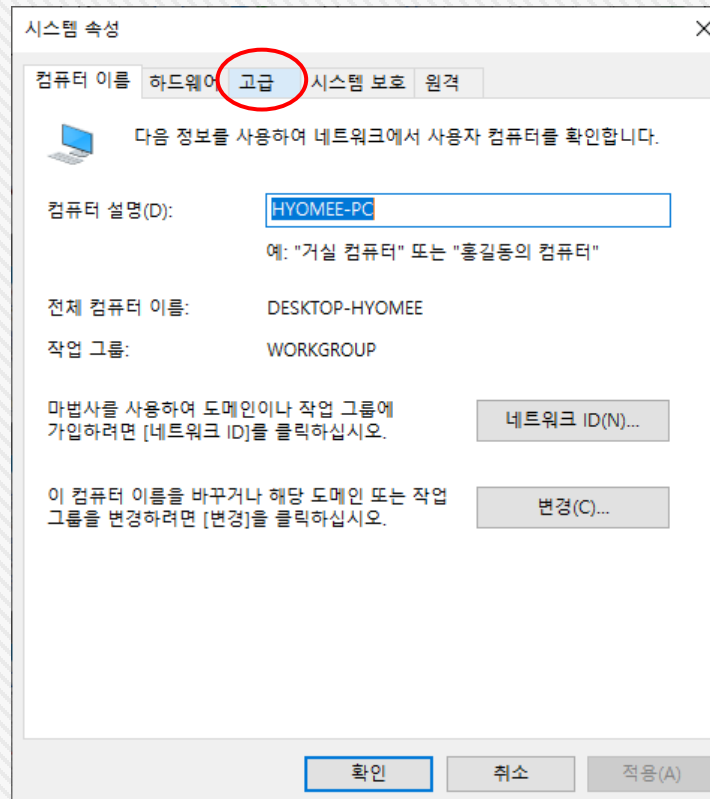
JAVA 설치

4. Windows 환경 설정

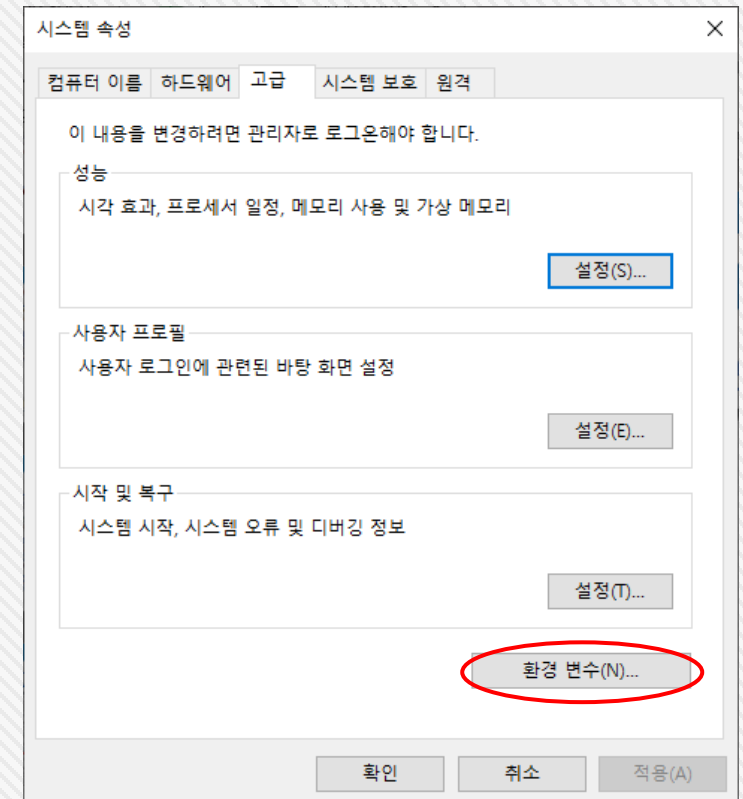
- ① 윈도우 + R 클릭 후 실행 창에서
sysdm.cpl 입력 후 확인 클릭



- ② 시스템 속성 창 에서 고급 탭 클릭



- ③ 환경 변수 버튼 클릭



1. 자바 설치

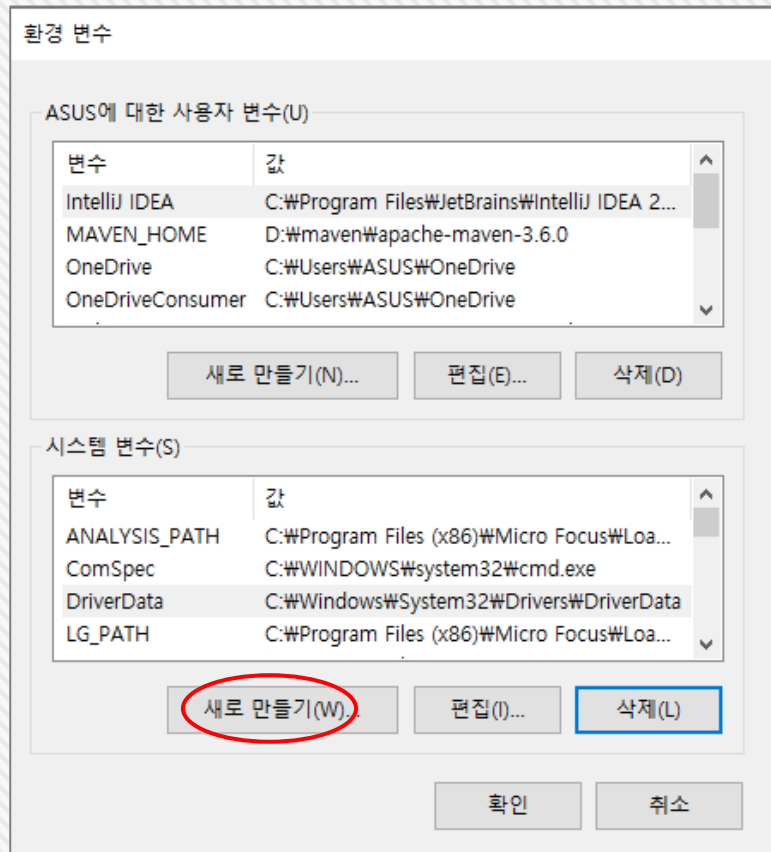
1. 프로그램 기초

1-3. JAVA 설치

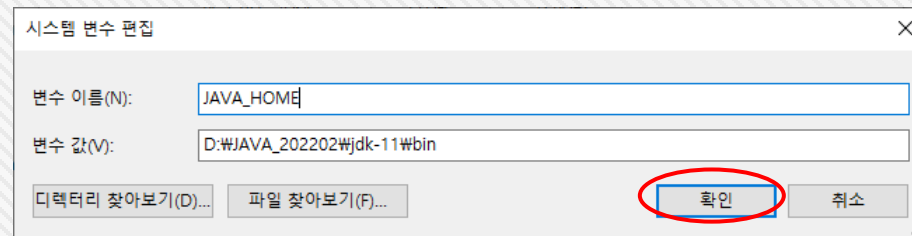
JAVA 설치

4. Windows 환경 설정

④ 환경 변수 창에서 시스템 변수 새로 만들기 클릭



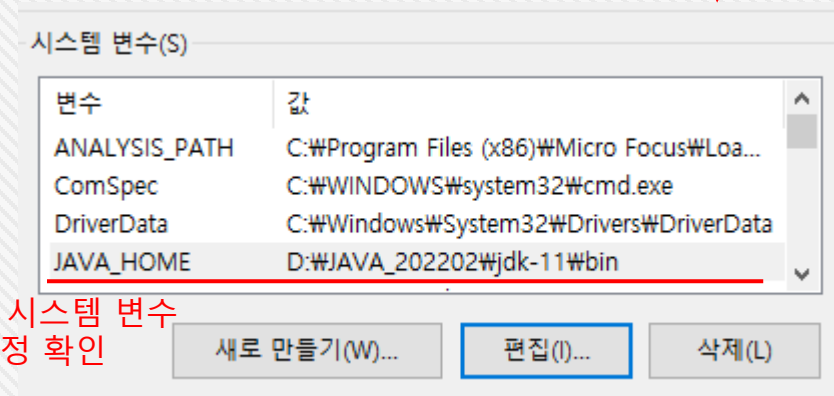
⑤ 시스템 변수 창에서 JAVA_HOME path 설정



변수 이름 : JAVA_HOME

변수 값 : D:\JAVA_202202\jdk-11\bin

환경 변수 창 > 시스템 변수
JAVA_HOME 설정 확인



1. 자바 설치

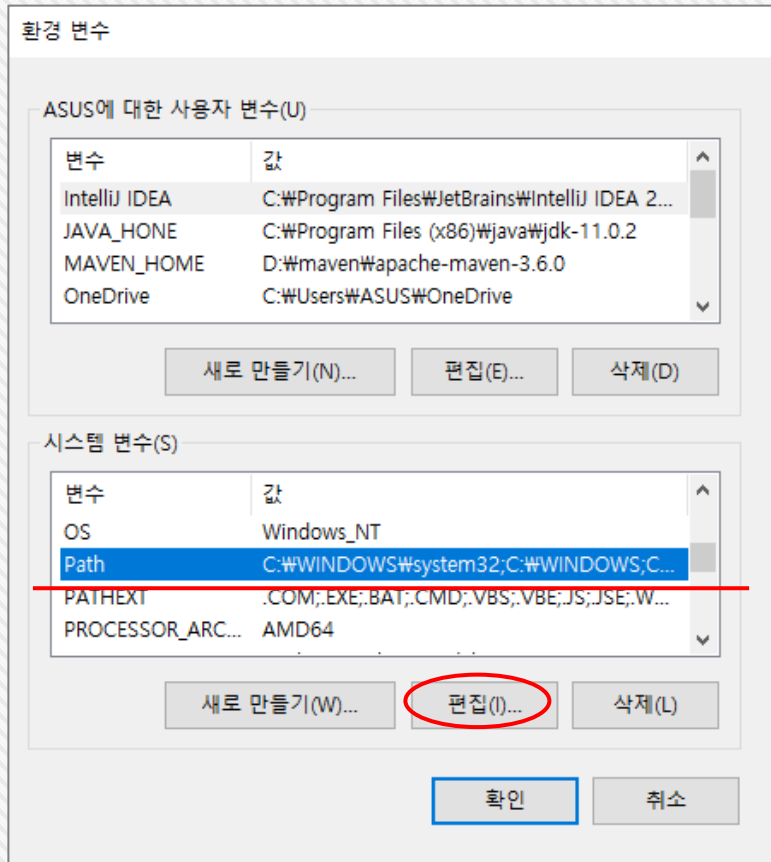
1. 프로그램 기초

1-3. JAVA 설치

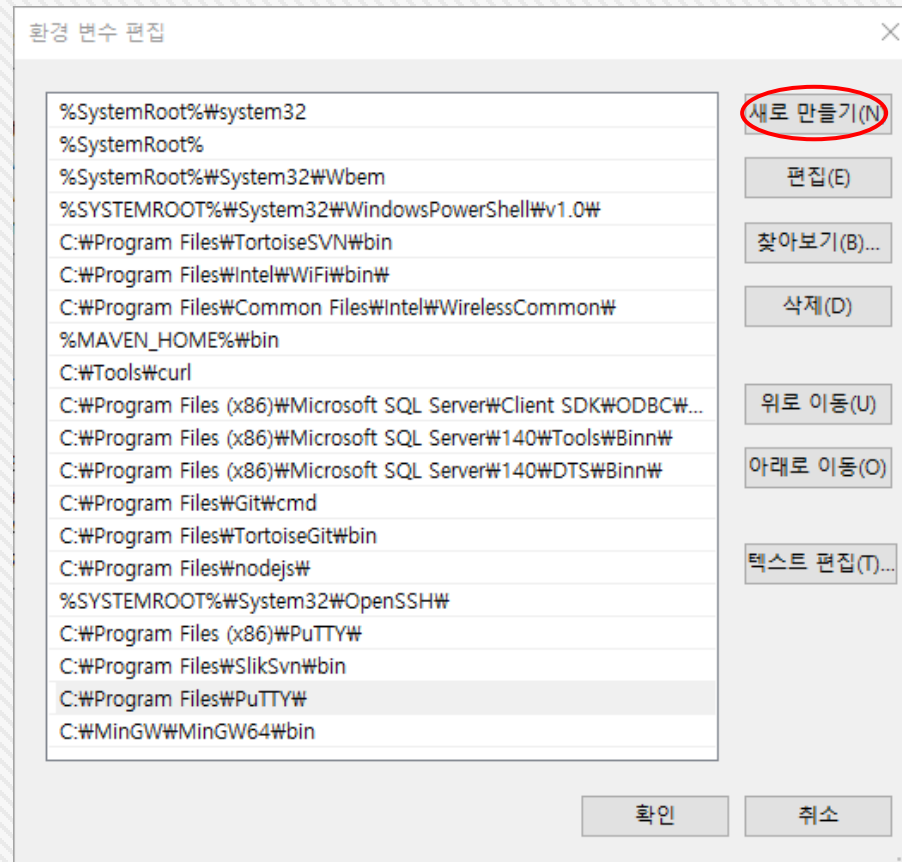
JAVA 설치

4. Windows 환경 설정

⑥ 시스템 변수 영역 에서 Path 선택 후 편집 클릭



⑤ 환경 변수 편집 창에서 새로 만들기 선택



1. 자바 설치

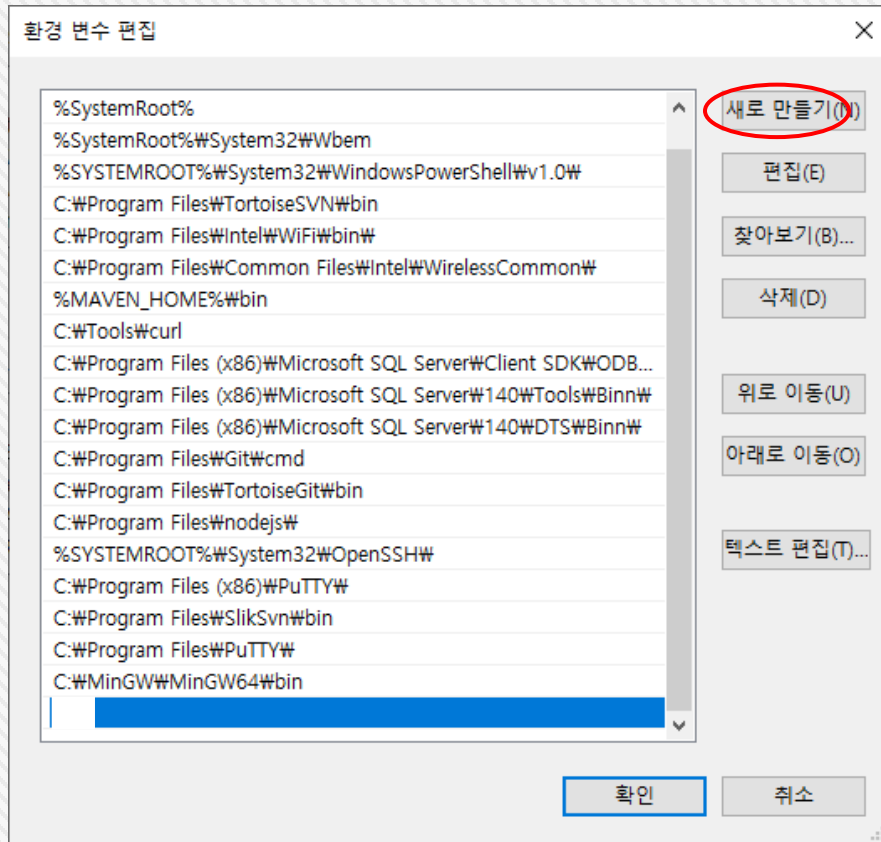
1. 프로그램 기초

1-3. JAVA 설치

JAVA 설치

4. Windows 환경 설정

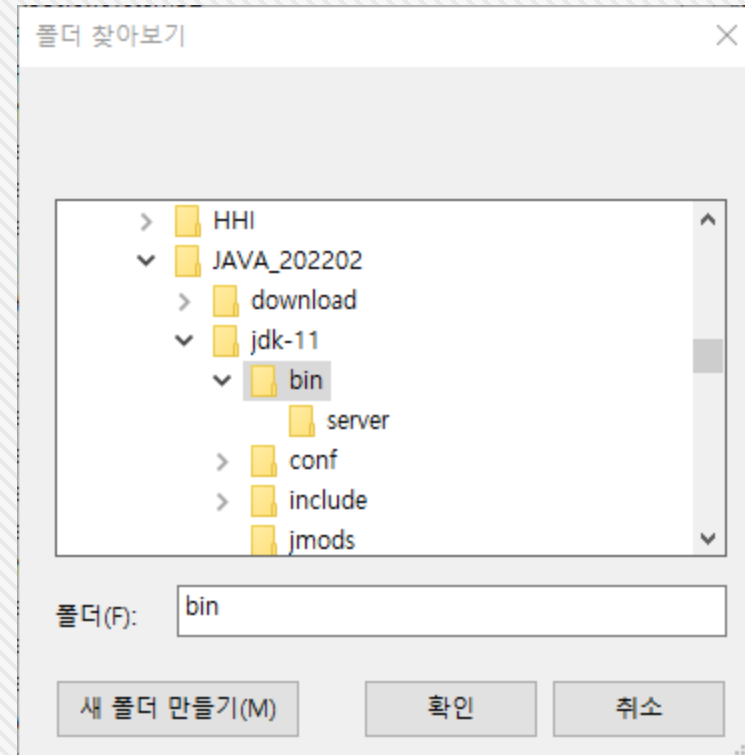
⑥ 아래 부분에 행이 추가 됨 , 찾아보기 클릭



%JAVA_HOME%\bin 입력 후 확인 버튼 클릭
-> 다음 페이지 로 이동 (8)번 이동

%JAVA_HOME%\bin 입력 하지 않은 경우만 실행

⑦ 폴더 찾아보기 창에서 java 설치 폴더 선택



D:\JAVA_202202\jdk-11\bin

1. 프로그램 기초

1-3. JAVA 설치

4. Windows 환경 설정

The screenshot shows the '환경 변수 편집' (Edit Environment Variables) window. The '시스템 변수' (System variables) tab is active. A list of system variables is displayed, including %SystemRoot%, %SystemRoot%\System32\Wbem, %SYSTEMROOT%\System32\WindowsPowerShell\v1.0\, C:\Program Files\TortoiseSVN\bin, C:\Program Files\Intel\WiFi\bin\, C:\Program Files\Common Files\Intel\WirelessCommon\, %MAVEN_HOME%\bin, C:\Tools\curl, C:\Program Files (x86)\Microsoft SQL Server\Client SDK\ODBC..., C:\Program Files (x86)\Microsoft SQL Server\140\Tools\Binn\, C:\Program Files (x86)\Microsoft SQL Server\140\DTDS\Binn\, C:\Program Files\Git\cmd, C:\Program Files\TortoiseGit\bin, C:\Program Files\nodejs\, %SYSTEMROOT%\System32\OpenSSH\, C:\Program Files (x86)\PuTTY\, C:\Program Files\SlikSvn\bin, C:\Program Files\PuTTY\, C:\MinGW\MinGW64\bin, and D:\JAVA_202202\jdk-11\bin. The '확인' (OK) button is circled in red.

환경 변수

ASUS에 대한 사용자 변수(U)

변수	값
IntelliJ IDEA	C:\Program Files\JetBrains\IntelliJ IDEA 2...
JAVA_HOME	C:\Program Files (x86)\java\jdk-11.0.2
MAVEN_HOME	D:\maven\apache-maven-3.6.0
OneDrive	C:\Users\ASUS\OneDrive

새로 만들기(N)... 편집(E)... 삭제(D)

시스템 변수(S)

변수	값
NUMBER_OF_PRO...	8
OS	Windows_NT
Path	C:\WINDOWS\system32;C:\WINDOWS;C...
PATHEXT	.COM;.EXE;.BAT;.CMD;.VBS;.VBE;.JS;.JSE;.W...

새로 만들기(W)... **편집(I)...** 삭제(L)

확인 취소

1. 자바 설치

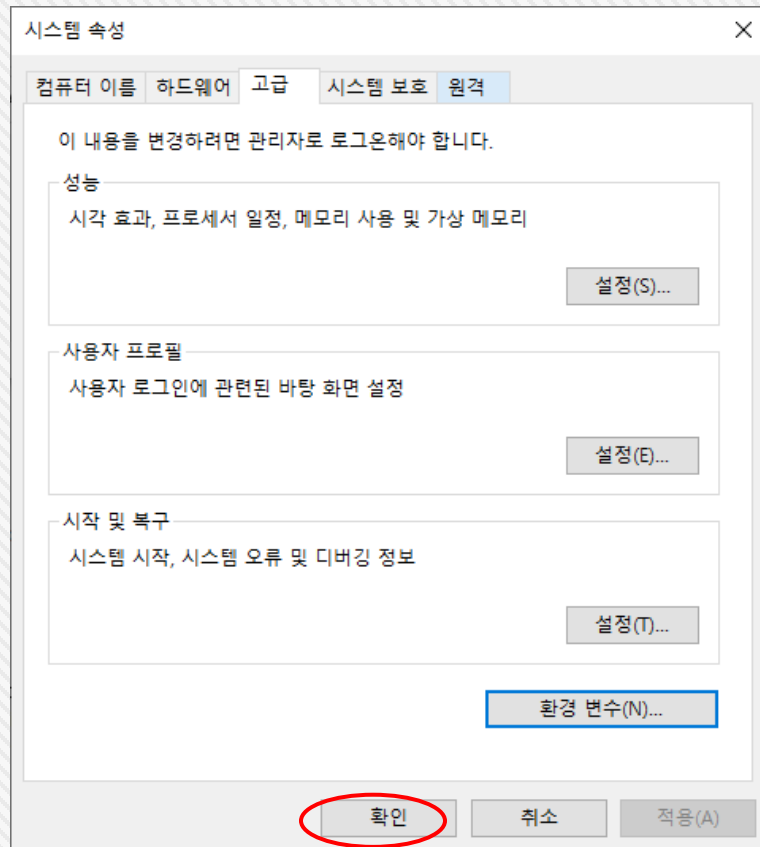
1. 프로그램 기초

1-3. JAVA 설치

JAVA 설치

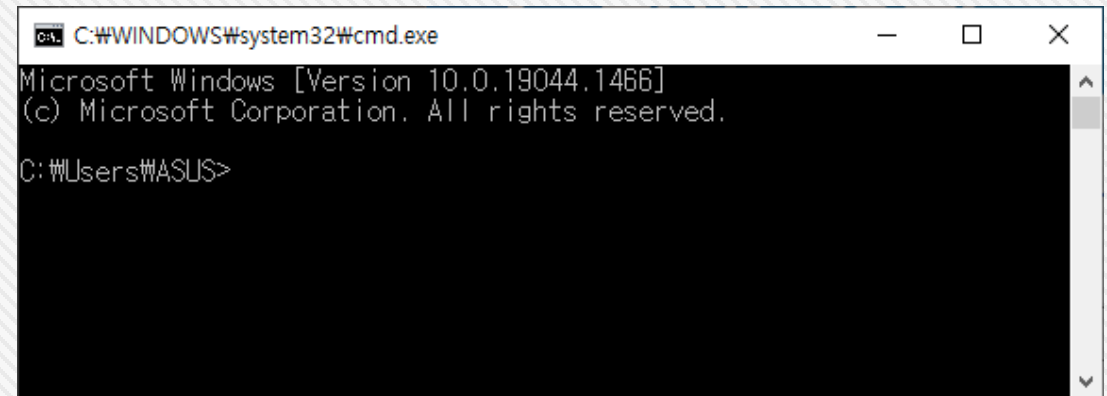
4. Windows 환경 설정

⑪ 시스템 속성 창에서 확인 버튼 클릭

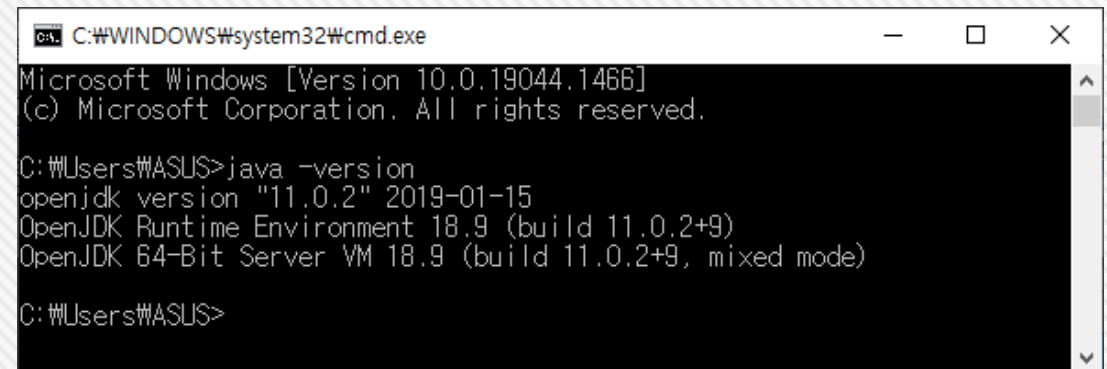


⑫ 설치 확인

- 윈도우 + R 클릭 후 실행 창에서 cmd 입력 후 확인 클릭



- Java -version 입력 후 엔터



1. 개발 Tool 설치


1. 프로그램 기초 1-4. 개발 Tool 설치

개발 Tool 설치

1. Eclipse 설치

- 다운로드 : <https://www.eclipse.org/downloads/packages/release/kepler/sr1/eclipse-ide-java-developers>

This package was released on 09/26/2013. A newer package is available [here](#).



Eclipse IDE for Java Developers

Package Description

The essential tools for any Java developer, including a Java IDE, a CVS client, Git client, XML Editor, Mylyn, Maven integration and WindowBuilder

This package includes:

- Code Recommenders Developer Tools
- Eclipse Git Team Provider
- Eclipse Java Development Tools
- Maven Integration for Eclipse
- Mylyn Task List
- WindowBuilder Core
- Eclipse XML Editors and Tools

► Detailed features list

Maintained by: Eclipse Mylyn Project

Download Links

Windows 32-bit | x86_64
macOS 32-bit | x86_64 Linux
32-bit | x86_64

Downloaded 1,048,954 Times

► Checksums...


Bugzilla

► Open Bugs: 35


► Resolved Bugs: 126

[File a Bug on this Package](#)

New and Noteworthy




Enterprise Pack for Eclipse



Download

The Eclipse Installer 2021-12 R now includes a JRE for macOS, Windows and Linux.



Get Eclipse IDE 2021-12

Install your favorite desktop IDE packages.

[Download x86_64](#)

1. 개발 Tool 설치

1. 프로그램 기초 1-4. 개발 Tool 설치

개발 Tool 설치

1. STS 설치

- 다운로드 : <https://spring.io/tools>

Spring Tools 4 for Eclipse

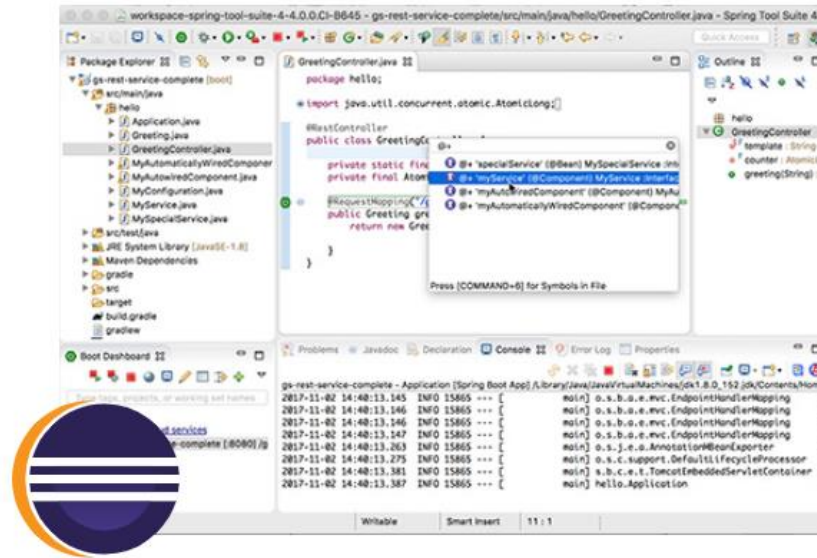
The all-new Spring Tool Suite 4.
Free. Open source.

4.13.0 - LINUX X86_64

4.13.0 - MACOS X86_64

4.13.0 - MACOS ARM_64

4.13.0 - WINDOWS X86_64



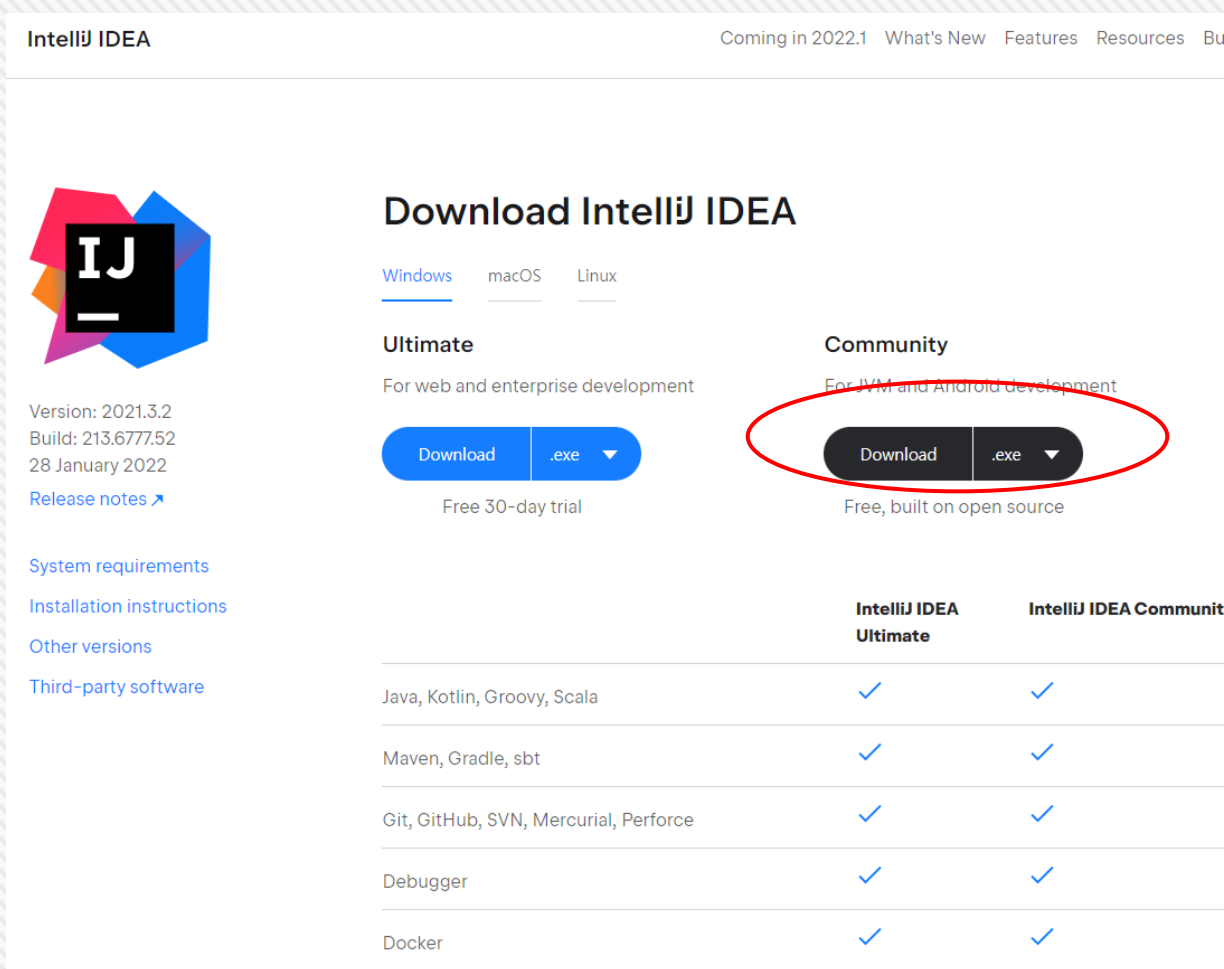
1. 개발 Tool 설치

1. 프로그램 기초 1-4. 개발 Tool 설치

개발 Tool 설치

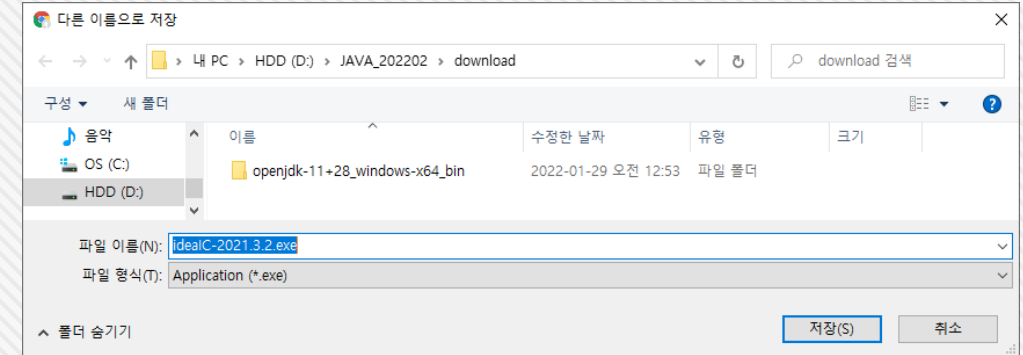
1. STS 설치

- 다운로드 : <https://www.jetbrains.com/idea/download/#section=windows>



The screenshot shows the IntelliJ IDEA download page. The 'Community' download button is circled in red. Below the download buttons is a table comparing features for Ultimate and Community editions.

	IntelliJ IDEA Ultimate	IntelliJ IDEA Community
Java, Kotlin, Groovy, Scala	✓	✓
Maven, Gradle, sbt	✓	✓
Git, GitHub, SVN, Mercurial, Perforce	✓	✓
Debugger	✓	✓
Docker	✓	✓



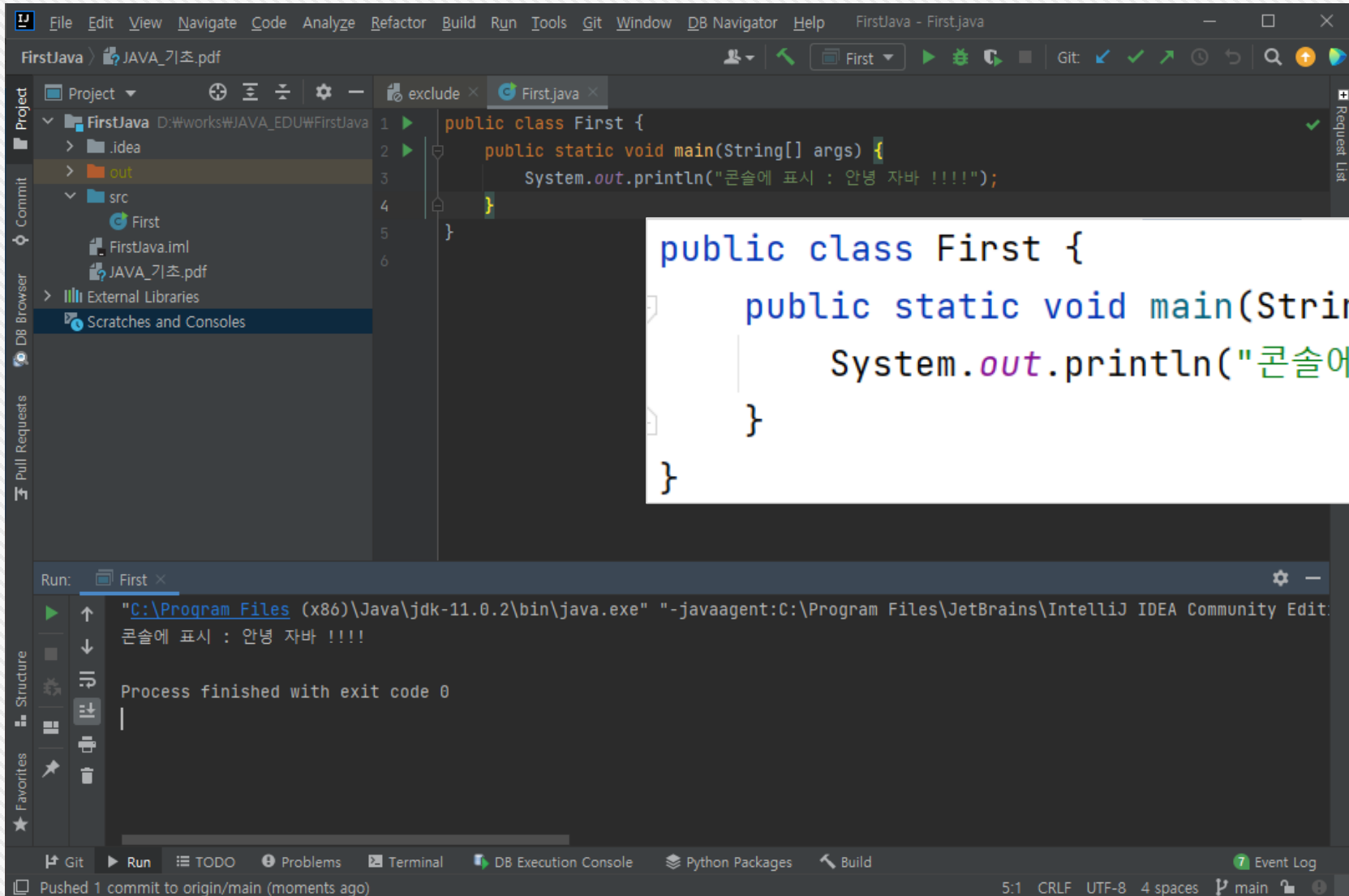
다운로드 : D:\JAVA_202202\download

Install 폴더 : D:\JAVA_202202\tool\IntelliJ

1. 첫 프로그램

1. 프로그램 기초 1-5. 첫번째 프로그램

첫 번째 프로그램



The screenshot shows the IntelliJ IDEA IDE with a project named 'FirstJava'. The source code for 'First.java' is displayed in the editor, showing a public class 'First' with a static 'main' method that prints a greeting to the console. The code is as follows:

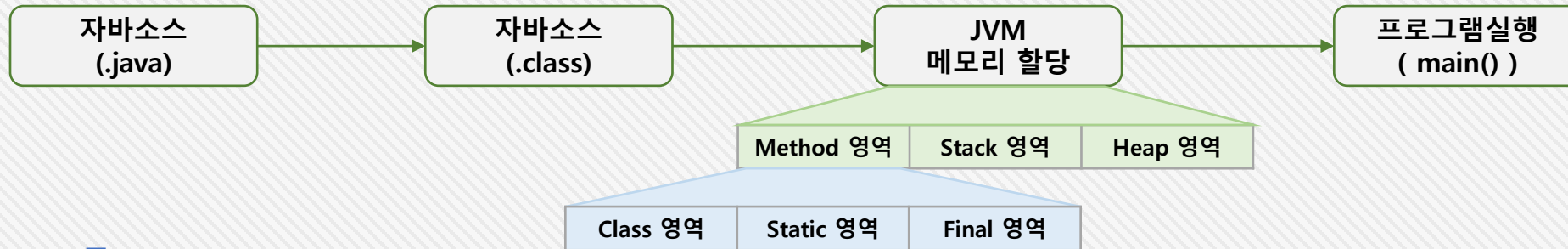
```
public class First {  
    public static void main(String[] args) {  
        System.out.println("콘솔에 표시 : 안녕 자바 !!!!");  
    }  
}
```

The 'Run' window at the bottom shows the execution of the program, displaying the output '콘솔에 표시 : 안녕 자바 !!!!' and indicating that the process finished with exit code 0. The status bar at the bottom shows 'Pushed 1 commit to origin/main (moments ago)' and '5:1 CRLF UTF-8 4 spaces main'.

1. 자바 프로그램 구조

1. 프로그램 기초 1-6. 자바 프로그램 구조

자바 실행 과정



JAVA File 구조

- 자바 소스 파일의 확장자는 java
- 자바 파일명은 접근 지정자가 public인 Top Level Class가 있다면 Class Name으로 되어야 함, 없다면 아무 이름으로 사용 할 수 있음
- public Class 가 main Method를 가진다.
- package 가 있다면, 해당 자바파일은 반드시 패키지명의 폴더에 존재해야 한다.

```
package com.hyomee;

/**
 * 여러 줄 주석
 */
public class First {

    public static void main(String[] args) {
        // 한 줄 주석
        System.out.println("콘솔에 표시 : 안녕 자바 !!!!");
    }

}
```

package 선언 : 소스 파일의 위치

/** ~ */ : 여러 줄 주석

public : 다른 패키지에서도 사용 (공개)
class : 클래스를 가리키는 자바 키워드, 파일명과 동일
- 내부 구성 요소 : field, method, constructor, inner class

자바로 만든 Application의 시작점
- Application에 하나 존재 해야 함 , 여러 개 존재 시 실행 시 지정 해 주어야 한다.

01. JVM

- JRE(Java Runtime Enviroment)는 크게 API, JVM으로 구성 됨
- JVM(자바 가상 머신, Java Virtual Machine)은 클래스 로더를 통해 자바 클래스를 메모리로 로드하여 자바API를 이용하여 실행한다.
- Method안에서 선언한 로컬 데이터는 Thread로 부터 안전 하다는 의미는 JVM Stack에 저장 된 데이터는 해당 Thread에서만 사용 할 수 있기 때문 이다,
- 객체는 new연산자에 의해 메모리 heap에 생성 되고 JVM의 GC(Garbage Collector)에 의해 자동으로 Heap 메모리에서 해제 됨.



02. 오류

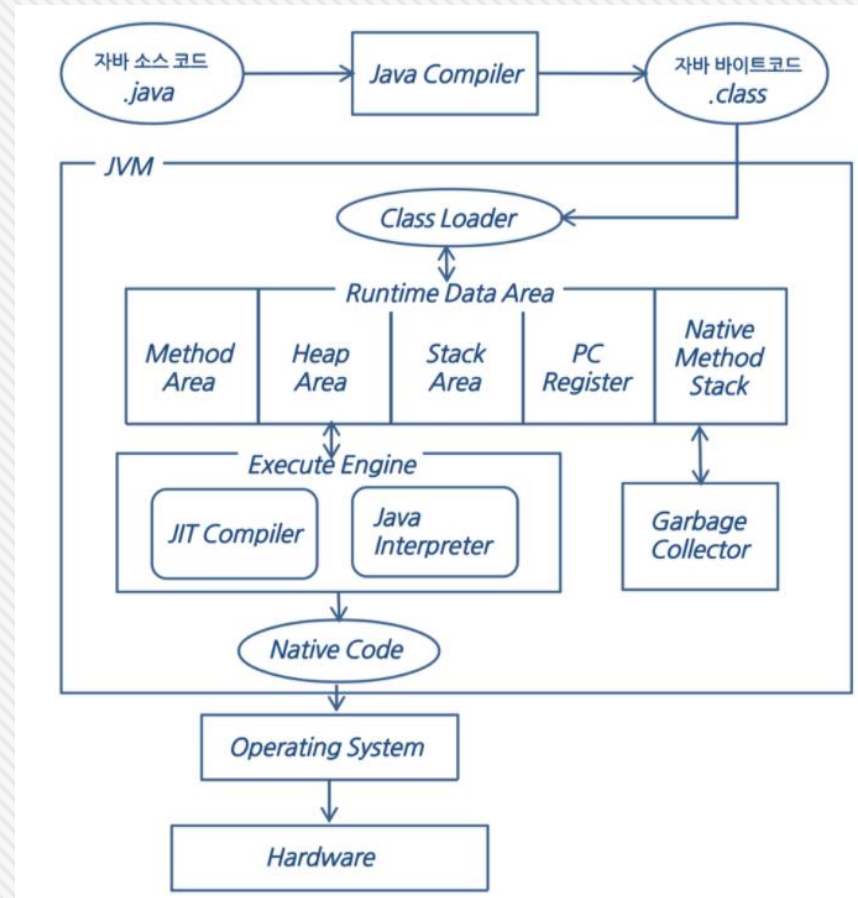
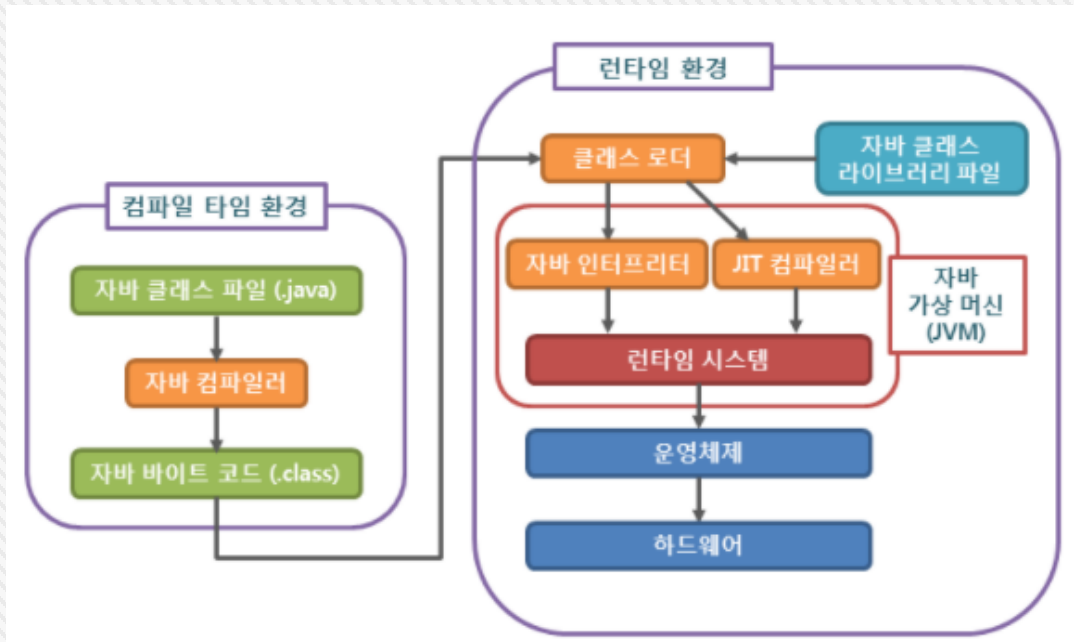
- `StrackOverflowError`
 - : Stack Frame에 Method를 추가 할 공간이 없을 때 발생
 - : JVM -Xss 옵션을 사용 하여 크기 조정
- `OutOfMemoryError`
 - : 실행 중인 Thread가 많아서 JVM Stack를 할당 할 수 없을 때 발생

03. JVM Data Type

기본 자료형 4Byte -> 플랫폼 독립성 보장

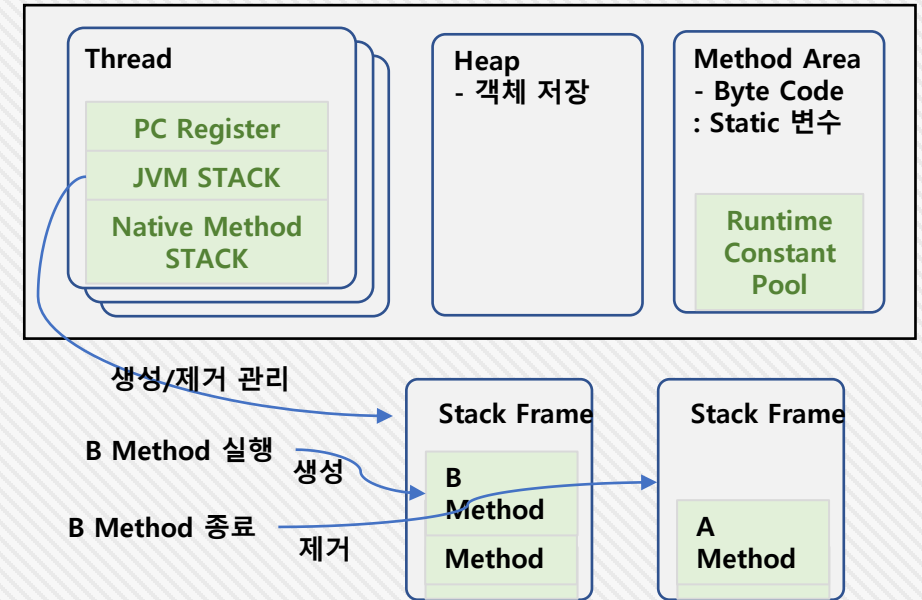
04. 실행 과정

자바프로그램을 실행하면 JVM의 클래스 로더가 컴파일 된 자바 바이트코드(.class 파일)을 런타임 데이터 영역(Runtime Data Area)의 Method Area에 로드 하고 실행 엔진(Execution Engine)이 이를 기계어로 번역 하면서 실행.



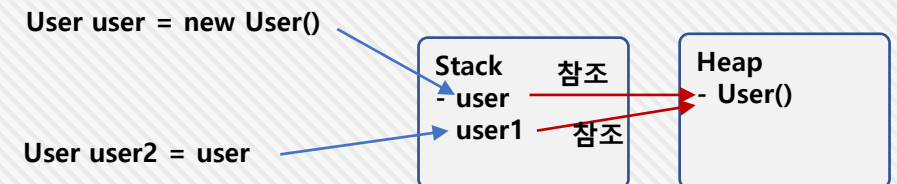
05. JVM Runtime Data Area - 운영체제로 부터 부여 받은 메모리 영역

- **Method Area**
: 모든 스레드가 공유 하는 영역, JVM이 시작 할 때 생성, 클래스와 인터페이스 Method에 대한 바이트 코드, 전역변수, 런타임 상수 등이 저장됨
-> Main Method가 컴파일 된 Byte Code가 있음
- **Heap Area**
: 객체를 저장 할 때 사용 하는 영역 => 성능 고려 필요
- **JVM Stack (임시 메모리)**
: 실행 시 Stack Frame이라는 각 스레드 마다 하나씩 할당
: 실행되는 메소드의 **Stack Frame에는 지역변수, 메소드의 인자, 메소드의 리턴 값, 리턴 변수 등이 저장되고** Stack Frame은 메소드가 끝나면 사라짐
- **Program Counter Register**
: 스레드마다 하나씩 존재 : JVM의 명령어 주소
- **Runtime Constant Pool**
: Method Area에 할당, 상수, 메소드, 필드를 저장
: 자바 프로그램이 참조 할 경우 메모리 주소를 찾아서 참조함
- **Native Method Stack**
: 자바 이외의 언어로 작성된 코드를 위한 Stack (C, C++ 등)



02. JVM Runtime Data Area

- 메소드 내에서 객체 참조 하면 선언한 변수는 지역변수로 Stack에 위치 하여 Heap에 저장된 객체에 대한 참조 값을 가짐
- New 연산자는 Heap 메모리에 객체를 만들고 그 객체의 참조 값을 반환 함



2. JVM

03. JVM 상태

```
class ClassMain {
    static int counter;

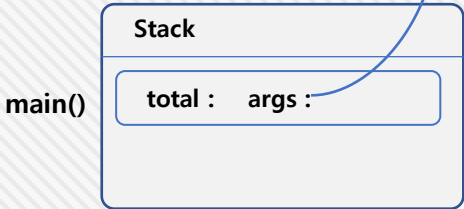
    public static void main(String[] args) {
        int total = sum(10, 30);
    }

    static int sum(int i, int j) {
        int sum = i + j;
        counter = counter + 1;
        return sum;
    }
}
```

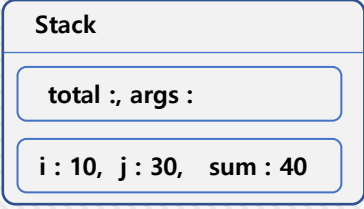
1. ClassMain Class가 시작 할 때 할당 됨



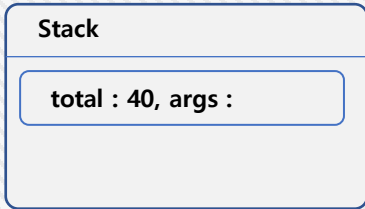
1. 메인 실행



2. SUM 실행



3. SUM 종료



* MAIN 종료 후 모두 사라짐

1. 콘솔에 출력

1. 프로그램 기초 1-8. 기본 입출력

콘솔에 문자열 출력

01. System.out.print()

- `public void print(String s)`
- 자료형 별로 Method가 선언 되어 있음 - Overloading
- 모든 출력을 한 줄로 출력

System : `java.lang.System`

Out : `System.out` :

- `public static final PrintStream out = null`

print : `java.io.print()`

- `public void print(String s)`

```
public class SystemPrint {  
    public static void main(String[] args) {  
        System.out.print("안녕");  
        System.out.print("자바");  
        System.out.print("다음행 출력\n");  
        System.out.print(2.1);  
    }  
}
```

안녕자바다음행 출력
2.1

Overloading : 동일한 이름에 매개변수 자료형이 틀린 것

02. System.out.printf()

- `printf(String format, Object ... args)`
- `format` : 출력 포맷, `...args` : 가변 인자
- 출력 포맷을 지정 하여 문자 출력 - Overloading
- 가변 인자 개수 만큼 출력 포맷 지정

```
public class SystemPrintf {  
    public static void main(String[] args) {  
        System.out.printf("%d\n", 10);           // 10 진수  
        System.out.printf("%o\n", 7);            // 8 진수  
        System.out.printf("%x\n", 15);           // 16 진수  
        System.out.printf("%s\n", "안녕");        // 문자열  
        System.out.printf("%f\n", 2.2);           // 실수 ( 소수점 6자)  
        System.out.printf("%.2f\n", 2.2);  
        System.out.printf("%d, %.2f\n", 2, 2.2);  
    }  
}
```

10
7
f
안녕
2.200000
2.20
2, 2.20

콘솔에 문자열 출력

03. System.out.println()

- `public void println(String x)`
- 자료형 별로 Method가 선언 되어 있음 - Overloading
- 출력 후 자동 개행

```
public class SystemPrintln {  
    public static void main(String[] args) {  
        System.out.println("안녕" + "자바");  
        System.out.println(3 + "자바");  
        System.out.println(3.8 + "자바");  
        System.out.println(3.8 + 2);  
        System.out.println(3.8 + 2 + "자바");  
        System.out.println("3.8" + "2");  
    }  
}
```

안녕자바
3자바
3.8자바
5.8
5.8자바
3.82

2. 콘솔 입력

1. 프로그램 기초 1-8. 기본 입출력

콘솔 입력

01. Scanner()

- `public Scanner(InputStream source) : java.util`
- `source` : 입력 장치
- 지정된 입력 스트림에서 입력(스캔)된 값을 `scanner` 객체를 구성

```
public class Input {  
    public static void main(String[] args) {  
        Scanner sc = new Scanner(System.in);  
        System.out.print("문자를 입력 하세요.");  
        String str = sc.next();  
        System.out.println("입력한 문자 : " + str);  
        float fo = sc.nextFloat();  
        System.out.printf("입력한 실수는 %f : ", fo);  
    }  
}
```

문자를 입력 하세요.: *안녕 21.3*
입력한 문자 : 안녕
입력한 실수는 21.299999 :

➤ 자료형에 따라서 사용 하는 Method가 틀림

```
sc.nextBigDecimal();  
sc.nextBigInteger();  
sc.nextBoolean();  
sc.nextInt();  
sc.nextLong();  
sc.nextLine();  
....
```

```
import java.util.Scanner;  
  
public class InputLine {  
    public static void main(String[] args) {  
        Scanner sc = new Scanner(System.in);  
        System.out.print("문장을 입력 하세요.");  
        String str = sc.nextLine();  
        System.out.println("입력한 문자 : " + str);  
    }  
}
```

문장을 입력 하세요.: *안녕하세요 10+5=15 입니다.*
입력한 문자 : 안녕하세요 10+5=15 입니다.