

Java

개발 환경 구축

JDK

개념

- JDK(Java Development Kit)
 - 자바 애플리케이션을 개발하고 실행하는 데 필요한 도구들의 모음
 - a. 자바 프로그래밍 언어의 개발 환경을 제공
 - 개발자들이 자바 애플리케이션을 컴파일하고, 디버깅하고, 실행할 수 있도록 하는 필수적인 소프트웨어
 - 자바 애플리케이션을 개발하는 데 필요한 다양한 도구와 라이브러리들로 구성되어 있다.
- JDK의 버전 종류
 - Java SE(Standard Edition) JDK가 일반적으로 가장 많이 사용
 - EE(Enterprise Edition)
 - ME(Micro Edition)

JDK

구성 요소

- JVM (Java Virtual Machine)
 - 자바 바이트코드를 실행하는 가상 머신
 - a. 자바 프로그램이 운영체제와 무관하게 실행될 수 있도록 해주는 중요한 역할을 한다.
- JRE (Java Runtime Environment)
 - 자바 프로그램을 실행하는데 필요한 런타임 환경
- javac (자바 컴파일러)
 - 자바 소스 파일(.java)을 바이트코드(.class)로 변환하는 컴파일러
- jar (Java Archive 툴)
 - 여러 자바 클래스 파일과 관련 리소스를 하나의 아카이브 파일(.jar)로 패키징하는 도구
- java (애플리케이션 런처)
 - 컴파일된 자바 바이트코드를 실행하는 명령어

JDK

구성 요소

- Javadoc
 - 자바 소스 코드에 포함된 주석을 기반으로 자동으로 API 문서를 생성하는 도구
- JDB (Java Debugger)
 - 자바 프로그램을 디버깅하는 데 사용하는 도구
- JVM 라이브러리 (Java Class Libraries)
 - 자바 표준 라이브러리들이 포함되어 있으며, 개발자는 이를 활용하여 여러 기능(입출력, 네트워킹, 데이터 구조, 알고리즘 등)을 쉽게 구현할 수 있다.

JRE

개념

- JRE(Java Runtime Environment)
 - 자바 프로그램이 실행될 수 있는 환경을 제공하는 소프트웨어 패키지
 - 자바 애플리케이션을 실행하는데 필요한 구성 요소와 라이브러리들을 포함하고 있다.
 - 자바 애플리케이션을 개발하지 않고 실행만 하고자 하는 사용자들에게 필요한 환경

CobinCabin
iRaCha

JRE

구성 요소

- JVM (Java Virtual Machine)
 - 자바 코드는 JVM 상에서 바이트코드로 실행되기 때문에, JVM은 자바 코드와 운영 체제 간의 중재자 역할을 한다.
- Java Class Libraries (자바 클래스 라이브러리)
 - 자바 애플리케이션 개발에 필수적인 표준 라이브러리들로 구성되어 있다.
- Java API (Application Programming Interface)
 - 자바 표준 라이브러리를 통해 제공되는 API

JVM

개념

- JVM(Java Virtual Machine)
 - 자바 프로그램을 실행하는 가상 환경
- JRE와 JVM의 차이점
 - JRE
 - a. JVM뿐만 아니라, 자바 애플리케이션 실행에 필요한 라이브러리와 파일들을 포함한 전체적인 실행 환경
 - JVM
 - a. 자바 바이트코드를 해석하고 실행하는 가상 머신으로, 자바 애플리케이션의 실행을 담당.

JVM

구성 요소

- 클래스 로더 (Class Loader Subsystem)
 - 자바 프로그램에서 필요로 하는 클래스들을 동적으로 로드하는 역할을 한다.
 - 자바 프로그램이 실행될 때 사용되는 .class 파일을 메모리에 적재하는 작업을 수행

CobinCabin
iRaCha

JVM

구성 요소

- 메모리 영역 (Runtime Data Area)
 - JVM은 자바 프로그램이 실행되는 동안 여러 가지 데이터를 저장하기 위해 다양한 메모리 영역을 사용한다.
 - 메서드 영역 (Method Area)
 - a. 주로 프로그램 실행 중에 필요한 메타데이터와 관련된 정보를 관리
 - b. 모든 스레드에서 공유되는 공간
 - 힙 영역 (Heap Area)
 - a. 객체와 배열을 저장하는 메모리 공간
 - 스택 영역 (Stack Area)
 - a. 각 스레드에 대한 실행 정보를 저장하는 영역

JVM

구성 요소

- PC 레지스터 (Program Counter Register)
 - a. 각 스레드가 실행 중인 명령어의 주소를 저장하는 공간
- 네이티브 메서드 스택 (Native Method Stack)
 - a. 네이티브 코드(자바 외의 다른 언어로 작성된 코드, 예: C/C++)를 실행할 때 사용되는 메모리 공간.

CobinCabin
iRaCha

JVM

구성 요소

- 실행 엔진 (Execution Engine)
 - 자바 바이트코드를 실제로 실행 시킨다.
 - JVM 실행 엔진은 인터프리터 방식과 JIT(Just-In-Time) 컴파일러 방식을 사용하여 바이트코드를 기계어로 변환하고 실행 한다.
- 실행 엔진의 구성 요소
 - a. 인터프리터 (Interpreter)
 - b. JIT 컴파일러 (Just-In-Time Compiler)
 - c. Garbage Collector (GC)
- 네이티브 메서드 인터페이스 (Native Method Interface)
 - 네이티브 라이브러리와 JVM 간의 API를 제공.
- 네이티브 라이브러리 (Native Method Libraries)
 - 자바 프로그램이 호출할 수 있는 네이티브 메서드들이 포함된 라이브러리

JVM

동작 방식

- JVM의 전체 동작
 - 자바 소스 코드(.java)를 바이트코드(.class)로 컴파일.
 - 클래스 로더가 바이트코드를 로드하고 메모리에 적재.
 - 실행 엔진이 바이트코드를 인터프리터 또는 JIT 컴파일러를 통해 실행.
 - **Garbage Collector**가 힙 메모리를 관리하여 불필요한 객체를 정리.
 - 필요 시 네이티브 코드 실행을 위해 **JNI(Java Native Interface)**를 사용.

CobinCabin
iRaCha

JVM

Garbage Collection

- 가비지 컬렉션(GC, Garbage Collection)
 - 힙 영역에서 사용되지 않는 객체들을 찾아 메모리를 해제 한다
 - a. 객체가 더 이상 참조되지 않으면 해당 객체는 "가비지"로 간주되며, 이를 해제 한다.
- 동작 방식
 - Mark and Sweep
 - a. GC는 메모리에서 참조되는 객체를 "마킹"하고, 참조되지 않는 객체들을 "청소"하는 두 단계로 동작 한다
 - Stop-the-World와 같은 메커니즘을 사용하여 모든 스레드를 일시 정지하고 GC 작업을 수행

JVM

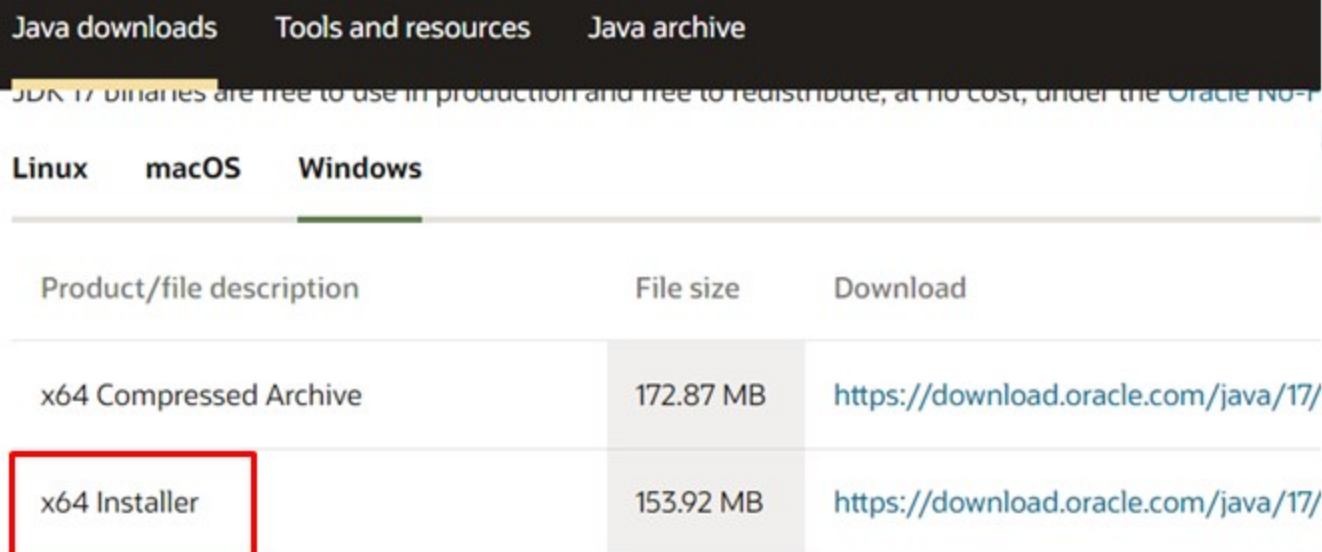
Garbage Collection

- Eden 영역
 - 새로 생성된 객체가 먼저 할당되는 공간
- Survivor 영역
 - Eden 영역에서 살아남은 객체가 이동하는 영역
- Old 영역
 - 여러 번의 Minor GC에서 살아남은 객체들이 이동하는 공간

개발환경 설정

Java 설치

- JDK 17 다운로드
 - <https://www.oracle.com/java/technologies/downloads/>

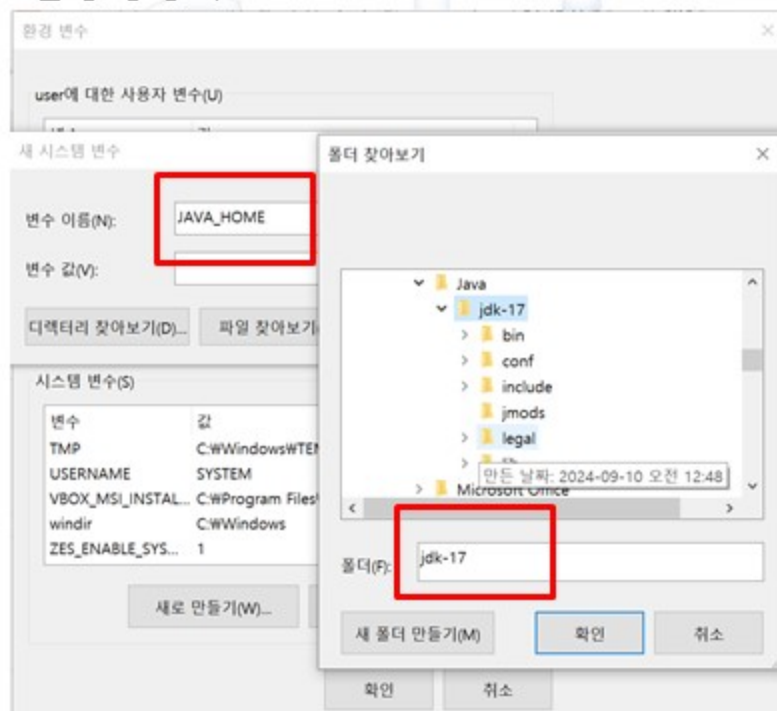


Java downloads Tools and resources Java archive		
JDK 17 binaries are free to use in production and free to redistribute, at no cost, under the Oracle No-FEE license.		
Linux macOS Windows		
Product/file description	File size	Download
x64 Compressed Archive	172.87 MB	https://download.oracle.com/java/17/
x64 Installer	153.92 MB	https://download.oracle.com/java/17/

개발 환경 설정

Java 설치

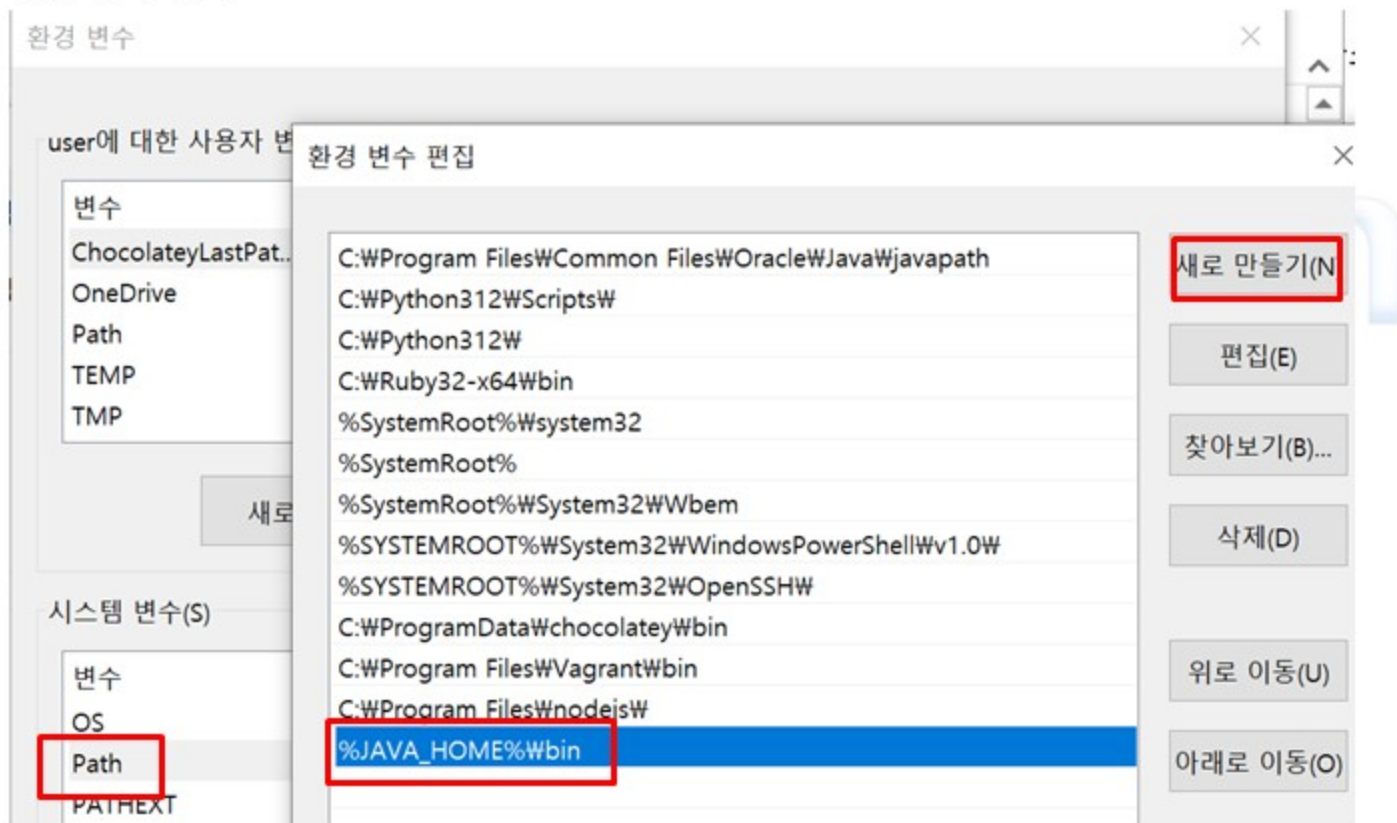
- 환경 변수 설정
 - sysdm.cpl
 - 자바 홈 디렉토리 : JAVA_HOME
 - 실행 명령어 : bin



개발 환경 설정

Java 설치

- 환경 변수 설정



개발환경 설정

Java 설치

- 설치 확인



A screenshot of a Windows Command Prompt window titled "선택 C:\Windows\system32\cmd.exe". The window shows the following text:

```
Microsoft Windows [Version 10.0.19045.4780]
(c) Microsoft Corporation. All rights reserved.

C:\Users\Wuser>java -version
java version "17.0.12" 2024-07-16 LTS
Java(TM) SE Runtime Environment (build 17.0.12+8-LTS-286)
Java HotSpot(TM) 64-Bit Server VM (build 17.0.12+8-LTS-286, mixed mode, sharing)

C:\Users\Wuser>javac -version
error: invalid flag: -version
Usage: javac <options> <source files>
use --help for a list of possible options

C:\Users\Wuser>
```

The commands `java -version` and `javac -version` are highlighted with red boxes. The output of `java -version` shows Java 17.0.12 LTS is installed. The output of `javac -version` shows an error: "error: invalid flag: -version".

개발환경 설정

VSCode 설치

- VS Code 다운로드
 - <https://code.visualstudio.com/download>

Download Visual Studio Code

Free and built on open source. Integrated Git, debugging and extensions.



↓ Windows

Windows 10, 11



↓ .deb

Debian, Ubuntu

↓ .rpm

Red Hat, Fedora, SUSE



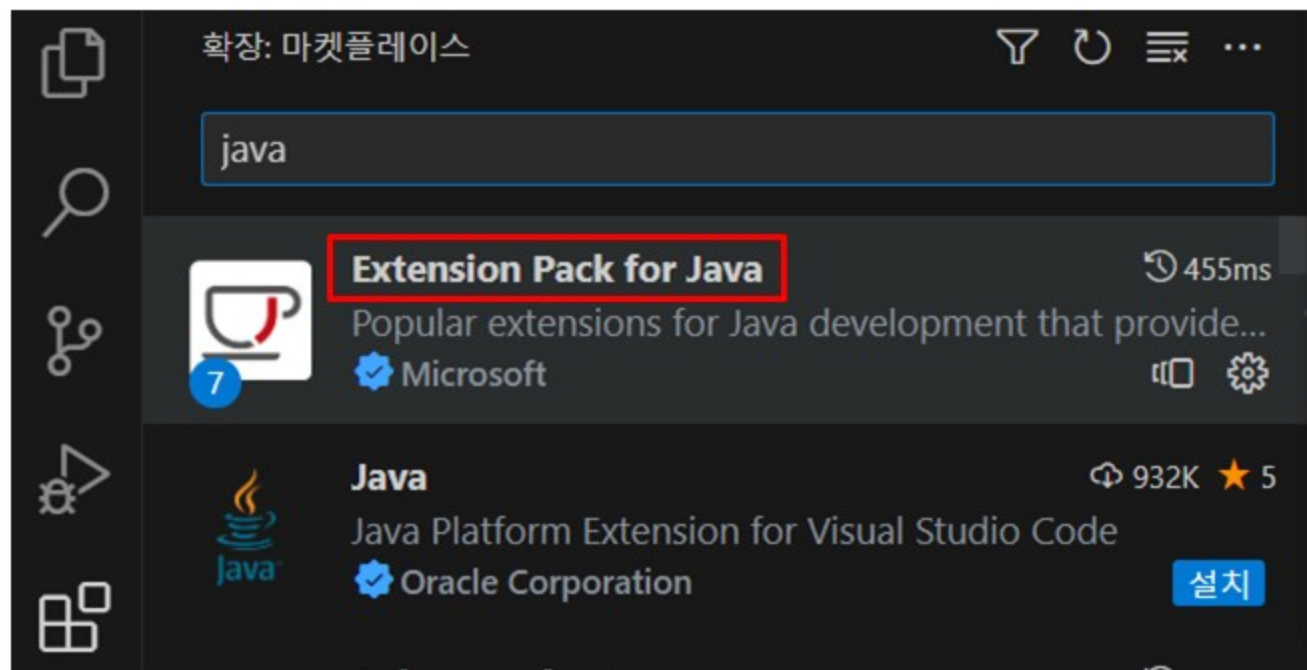
↓ Mac

macOS 10.15+

개발환경 설정

VSCode 확장팩 설치

- Java Extension Pack
 - 자바 개발에 필요한 주요 도구를 모두 포함한 패키지



개발 환경 설정

자동 import

- 파일을 저장할 때마다 자동으로 import가 정리
 - VSCode에서 Ctrl + Shift + P를 눌러 Preferences: Open Settings (JSON) 명령어를 실행.
 - settings.json 수정
 - "java.saveActions.organizeImports": true,

>preferences: open|

기본 설정: 기본 설정 열기(JSON)

Preferences: Open Default Settings (JSON)

기본 설정: 설정 열기(UI)

Preferences: Open Settings (UI)

기본 설정: 기본 바로 가기 키 열기(JSON)

Preferences: Open Default Keyboard Shortcuts (JSON)

개발환경 설정

단축키

- 파일 내에서 정의된 함수 및 변수 보기
 - Ctrl + Shift + O (Mac: Cmd + Shift + O)
- 이름 바꾸기(리팩토링)
 - F2
- 라인 복사
 - Shift + Alt + Down/Up (Mac: Shift + Option + Down/Up)
- 라인 삭제
 - Ctrl + Shift + K (Mac: Cmd + Shift + K) / Ctrl + X
- 라인 이동
 - Alt + Down/Up (Mac: Option + Down/Up)
- 라인으로 커서 이동
 - Ctrl + G (Mac: Cmd + G)

개발 환경 설정

단축키

- 자동 완성
 - Ctrl + Space (Mac: Cmd + Space)
- 주석 처리
 - Ctrl + / (Mac: Cmd + /)
- 다중 커서
 - Alt + Click (Mac: Option + Click)
- 코드 자동 정렬
 - Ctrl + Shift + I (Mac: Cmd + Shift + I)
- 패키지 자동 import
 - Alt + Shift + O
- 빠른 수정 (Quick Fix)
 - Ctrl + . (Mac: Cmd + .)