

프로그래밍기초

Chapter 01. Introduction to Java

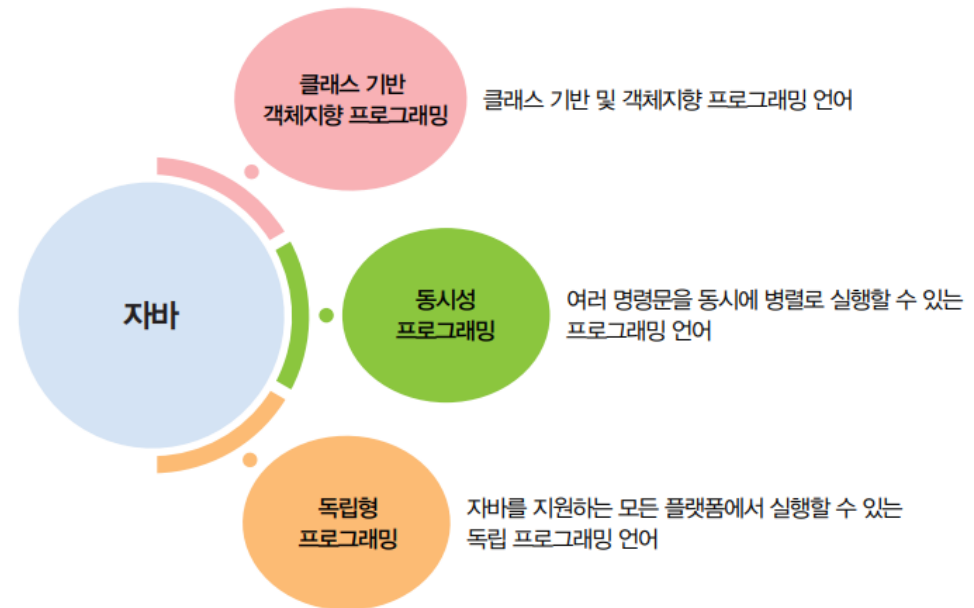
Section 01

자바의 개요

자바의 개요

■ 자바의 개요

- 1995년 Sun Microsystems가 출시한 언어
- 다양한 플랫폼에서 사용할 수 있는 객체지향 프로그래밍(Object-Oriented Programming, OOP) 언어

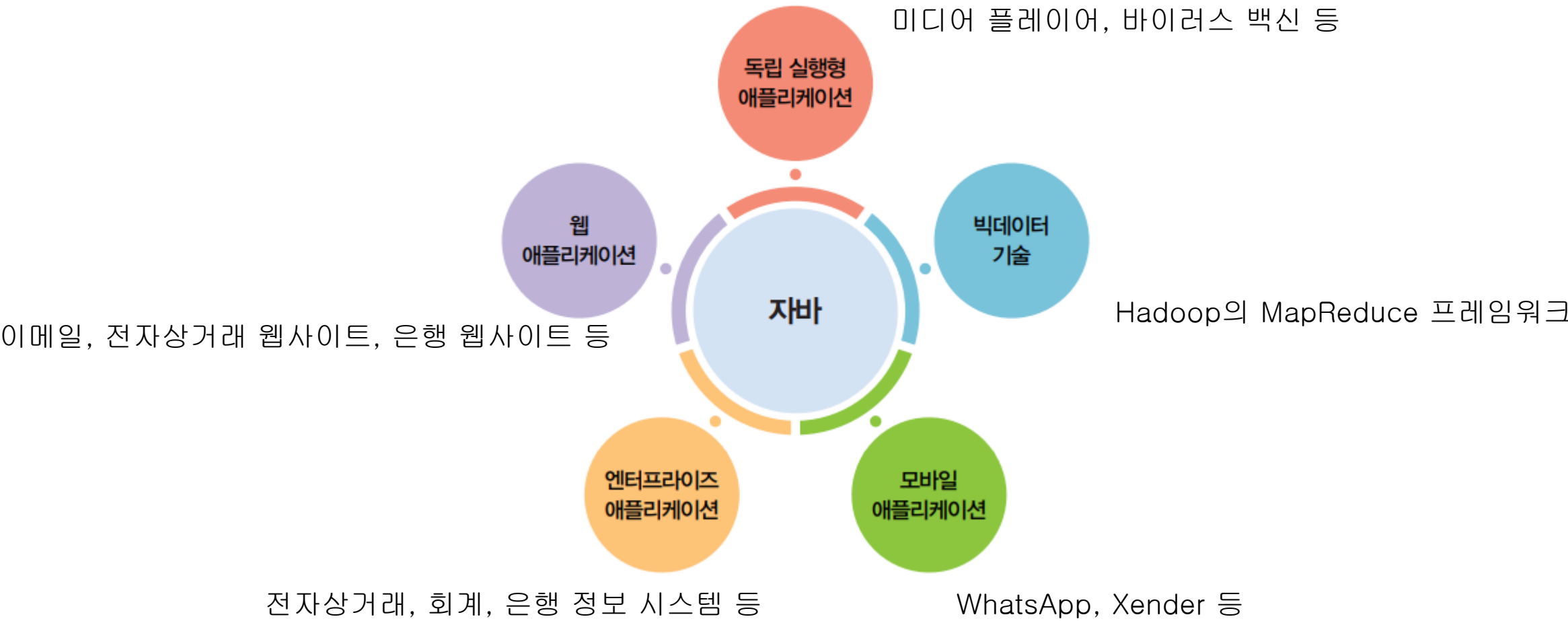


자바를 배워야 하는 이유

■ 자바를 배워야 하는 이유

- 자바는 현재 가장 널리 사용되는 프로그래밍 언어 중 하나임
- 자바의 구문과 기능은 C, C++ 등의 프로그래밍 언어와 유사함
- 자바는 플랫폼에 독립적임
 - 자바 코드는 먼저 바이트코드로 컴파일되며 자바 가상 머신(JVM)이 있는 모든 시스템에서 실행할 수 있음
 - ✓ 따라서 자바를 사용하면 코드를 한 번만 작성하고 원하는 곳에서 실행할 수 있음
- 자바는 객체지향 프로그래밍 언어임
 - 객체지향 프로그래밍: 프로그래밍 문제를 서로 상호 작용하는 객체로 분해하는 프로그래밍 접근 방식
 - ✓ 따라서 프로그램 구성상 실제 세계의 객체와 쉽게 연결할 수 있음

자바 애플리케이션의 유형



자바의 주요 특징

- 배우기 쉬운 표준적인 **기본 문법**
- 대규모 개발을 지원하는 **객체지향 프로그래밍**
- 풍부한 표준 명령어
- 다양한 컴퓨터에서 똑같은 동작을 하는 **범용성**
- 개발자가 메모리 관리에 신경쓰지 않아도 됨

Section 02

자바의 구성 요소

자바의 구성 요소

■ 자바 플랫폼

- 자바 프로그래머는 고급 프로그래밍 언어(즉 소스 코드)로 프로그램을 작성함
그런데 컴퓨터의 CPU나 칩은 기계어 코드만 이해할 수 있음
→ 다시 말해 기계어 코드는 CPU 수준에서 실행되며, CPU 모델에 따라 기계어 코드가 다름
→ 그러나 자바 프로그래밍 언어의 경우 기계어 코드에 대해 걱정할 필요 없음
- 컴파일러가 실행 가능한 기계어 코드로 번역하기 때문임



자바의 구성 요소

■ 자바 플랫폼

- 이러한 처리 기능은 자바 플랫폼 구성 요소인 자바 개발 키트(JDK), 자바 런타임 환경(JRE), 자바 가상 머신(JVM) 내에서 발생함



자바의 구성 요소

■ 자바 개발 키트(JDK)

- JDK : 애플릿과 자바 애플리케이션을 만드는 데 사용되는 소프트웨어 개발 환경
 - 윈도우(Windows), 맥(MacOS), 솔라리스(Solaris), 리눅스(Linux)에서 JDK를 사용할 수 있음
 - JDK는 자바 프로그램을 코딩하고 실행하는데 도움이 되며, 동일한 컴퓨터에 둘 이상의 JDK 버전을 설치할 수도 있음
- JDK의 특징
 - JDK에는 자바 프로그램을 작성하는 데 필요한 도구와 이를 실행하는 JRE가 포함되어 있음
 - JDK에는 컴파일러, 자바 애플리케이션 실행기, 애플릿뷰어(Appletviewer) 등이 포함됨
 - 컴파일러는 자바로 작성된 코드를 바이트코드로 변환함
 - 자바 애플리케이션 런처는 JRE를 열어 필요한 클래스를 로드하고 기본 메서드를 실행함

자바 개발 키트(JDK)
컴파일러, 디버거, 애플릿뷰어 등

자바 런타임 환경(JRE)
클래스 로더, 자바 API, 런타임 라이브러리

자바 가상 머신(JVM)
JIT 컴파일러, 자바 인터프리터

자바의 구성 요소

■ 자바 런타임 환경(JRE)

- JRE : 다른 소프트웨어를 실행하도록 설계된 소프트웨어를 의미함
- JRE에는 런타임 라이브러리, 클래스 로더, JVM이 포함됨
 - 간단히 말해서 자바 프로그램을 실행하려면 JRE가 필요함
- JRE의 특징
 - JRE에는 런타임 라이브러리, JVM 및 기타 지원 파일이 포함되어 있음
 - 디버거, 컴파일러 등의 자바 개발 도구는 포함되어 있지 않음
 - Math, Swing, Util, Lang, Awt, 런타임 라이브러리와 같은 중요한 패키지 클래스를 사용함
 - 자바 애플릿을 실행해야 하는 경우 시스템에 JRE가 설치되어 있어야 함

자바 개발 키트(JDK)

컴파일러, 디버거, 애플릿뷰어 등

자바 런타임 환경(JRE)

클래스 로더, 자바 API, 런타임 라이브러리

자바 가상 머신(JVM)

JIT 컴파일러, 자바 인터프리터

자바의 구성 요소

■ 자바 가상 머신(JVM)

- JVM : 자바 코드 또는 애플리케이션을 구동하기 위한 런타임 환경을 제공하는 엔진
 - 자바 바이트코드를 기계어로 변환함
 - JVM은 JRE의 일부임
 - 다른 프로그래밍 언어에서 컴파일러는 특정 시스템에 대한 기계어 코드를 생성하지만, 자바 컴파일러는 자바 가상 머신으로 알려진 가상 머신용 코드를 생성함
- JVM의 특징
 - JVM은 플랫폼 독립적인 자바 소스 코드 실행 방법을 제공함
 - JVM에는 수많은 라이브러리, 도구, 프레임워크가 있음
 - ✓ 자바 프로그램을 실행하면 모든 플랫폼에서 실행할 수 있고 많은 시간이 절약됨
 - JVM은 자바 소스 코드를 저수준 기계어로 변환하는 JIT 컴파일러와 함께 제공됨
 - ✓ 따라서 일반 응용 프로그램보다 빠르게 실행됨

자바 개발 키트(JDK)

컴파일러, 디버거, 애플릿뷰어 등

자바 런타임 환경(JRE)

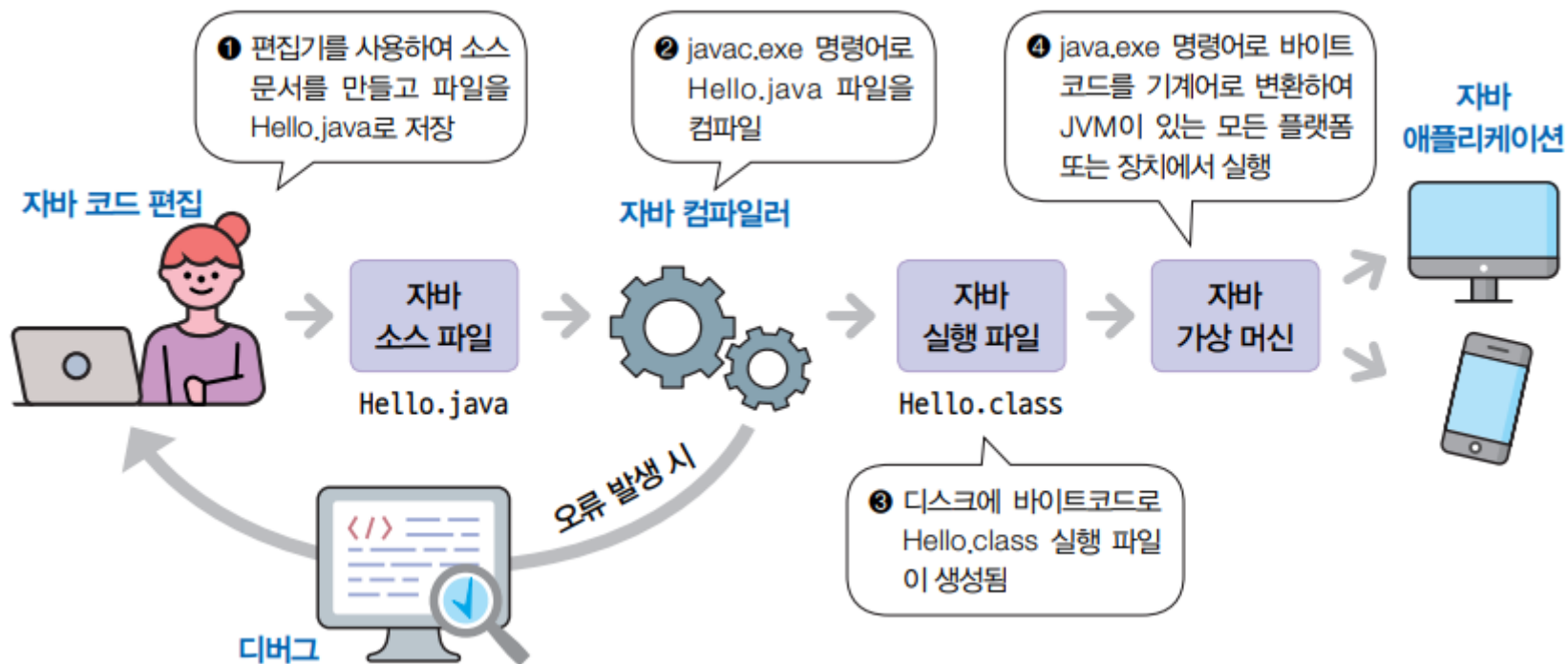
클래스 로더, 자바 API, 런타임 라이브러리

자바 가상 머신(JVM)

JIT 컴파일러, 자바 인터프리터

자바의 구성 요소

■ 자바 프로그램 실행 과정



Section 03

자바 개발 환경 구축

자바 개발 환경 구축

■ 자바 개발 환경 도구

- 자바 개발 환경 : JDK

→ 자바 코드를 작성하려면 자바 개발 도구인 JDK가 반드시 설치되어 있어야 한다.

- 통합 개발 환경 : 이클립스

→ 자바 코드를 작성하고 이를 컴파일하여 오류를 검사하고 실행 결과를 확인할 수 있는 통합 개발 환경(IDE)으로서 개발자에게 가장 인기 있는 이클립스(Eclipse)를 선택하여 설치한다.

자바 개발 환경 구축

■ 자바 개발 환경 구성

- JDK 설치
- JRE 설치
- Eclipse 설치

자바 개발 환경 구축

■ 자바 설치 및 환경 설정

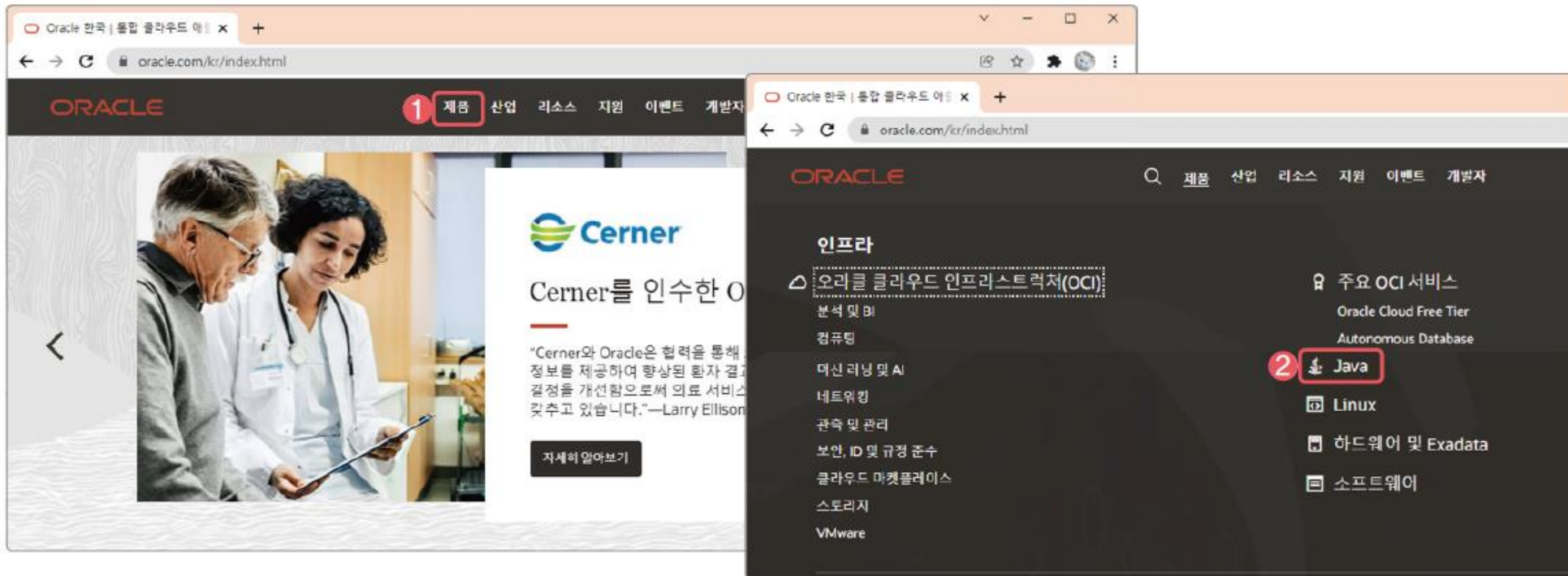
- 모든 자바 에디션에는 JRE와 JDK가 포함되어 있음
- 개발한 자바 프로그램을 실행하기만 할 때는 JRE로도 충분하지만 프로그램을 개발할 때는 JDK를 설치해야 함

자바 개발 환경 구축

■ JDK 설치하기

• 01. 오라클 사이트에 접속하기

→ <http://www.oracle.com/kr> 에 접속하여 [제품]-[Java]를 선택

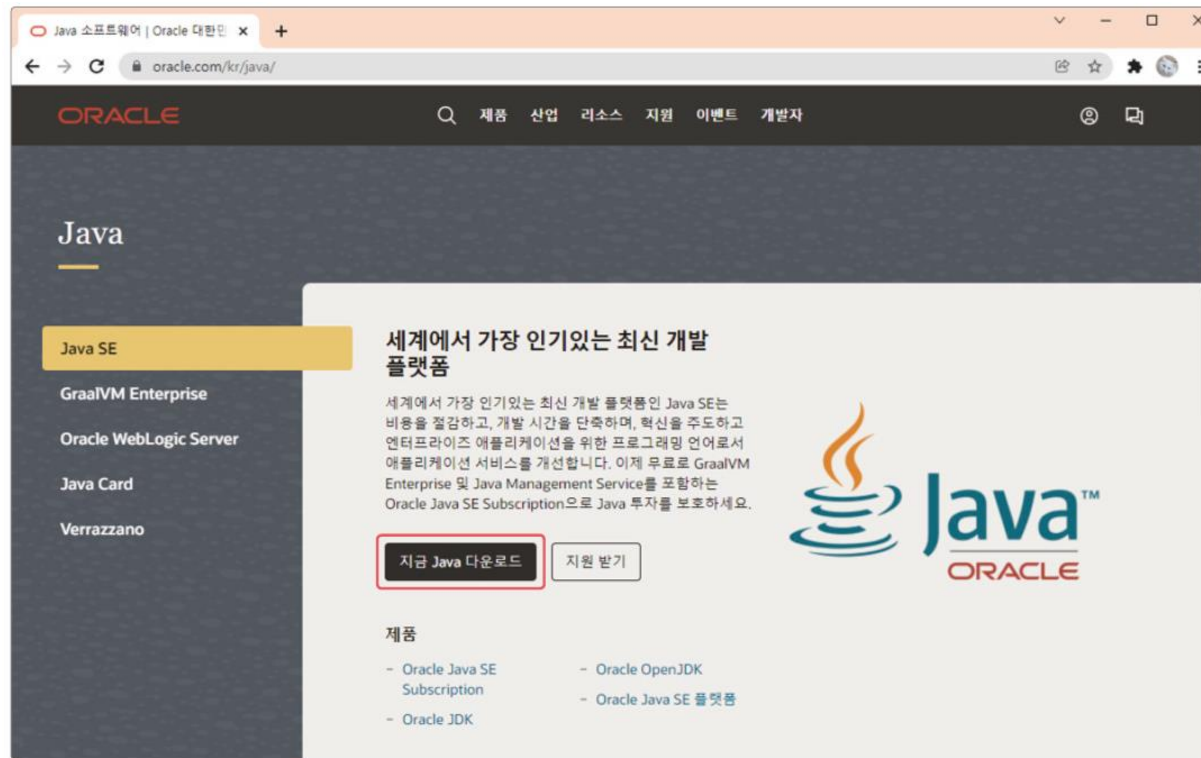


자바 개발 환경 구축

■ JDK 설치하기

• 02. JDK 선택하기

→ Java SE에서 <지금 Java 다운로드>를 클릭



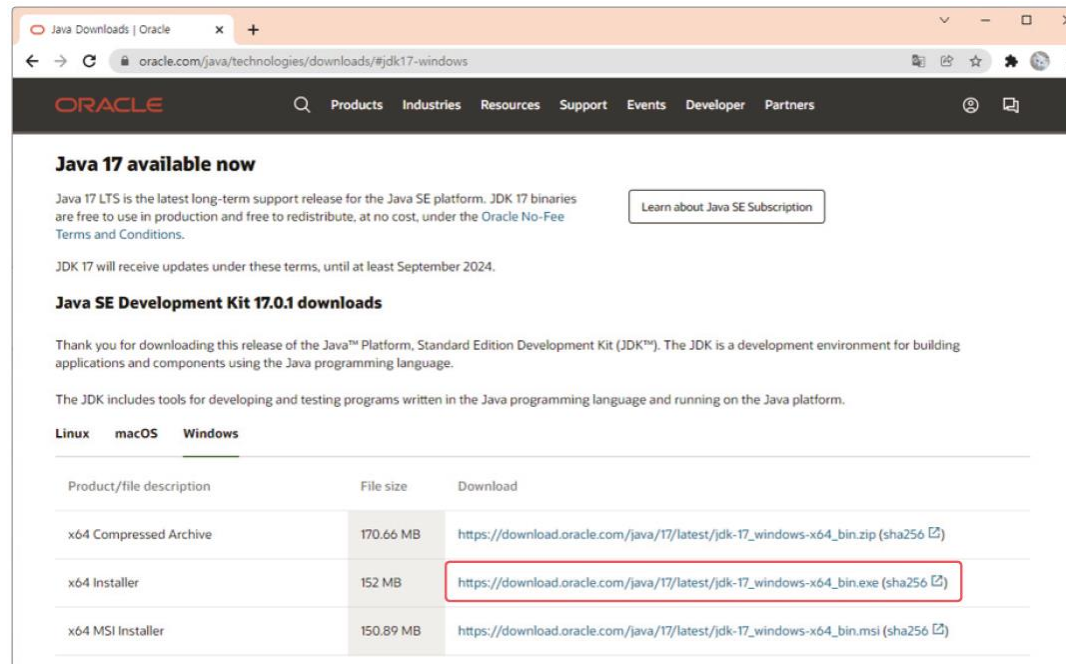
자바 개발 환경 구축

■ JDK 설치하기

• 03. 운영체제 버전에 맞는 설치 파일 선택하기

→ 운영체제에 맞는 버전을 찾아 클릭하여 다운로드하기

✓ 64비트 윈도우용인 'Windows x64' 선택



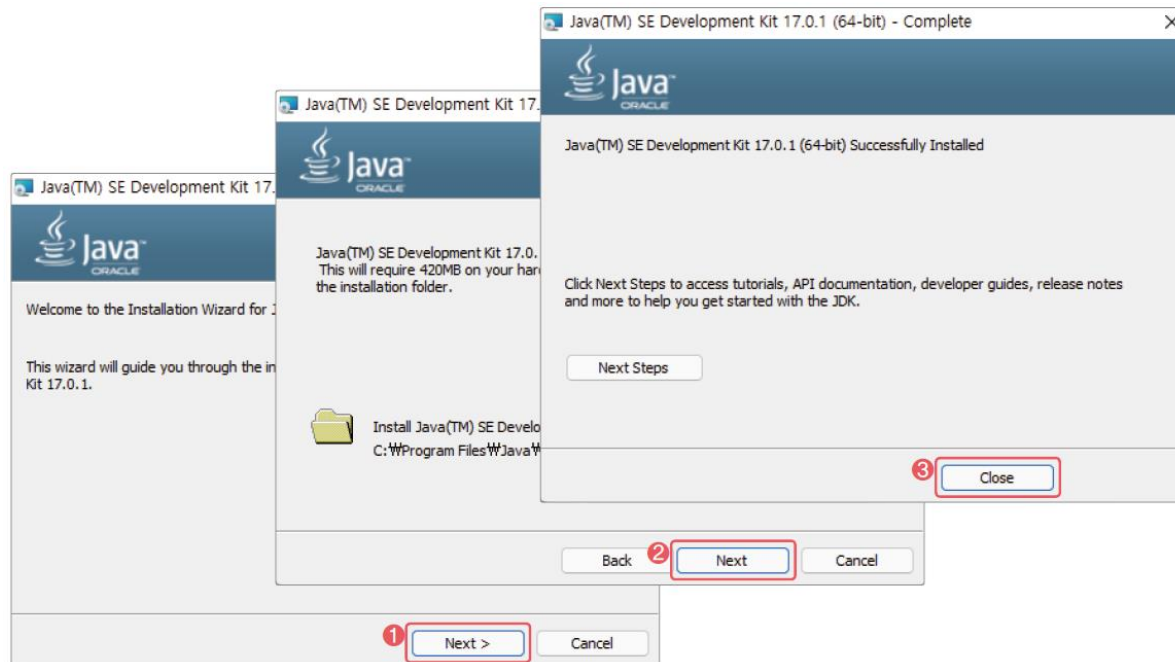
자바 개발 환경 구축

■ JDK 설치하기

• 04. 자바 설치하기

→설치를 시작하면 각 설치 단계마다 기본 설정을 그대로 두고 <Next> 또는 <다음>을 클릭

→마지막 화면에서 <Close>를 클릭하면 설치가 완료됨



자바 개발 환경 구축

■ JDK 설치하기

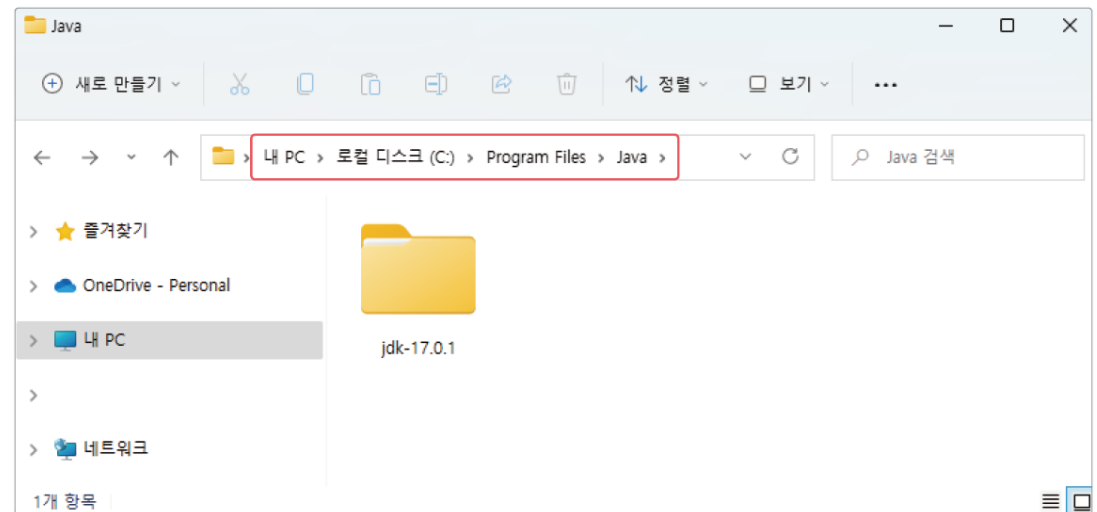
• 05. 자바 설치 위치 확인하기

→ 기본 설정으로 자바를 설치하면 'C:\Program Files\Java'에 설치됨

→ 'Java\jdk버전번호\bin' 폴더 내부 파일

✓ javac.exe : 자바 컴파일러 파일

✓ java.exe : JVM을 구동하는 파일

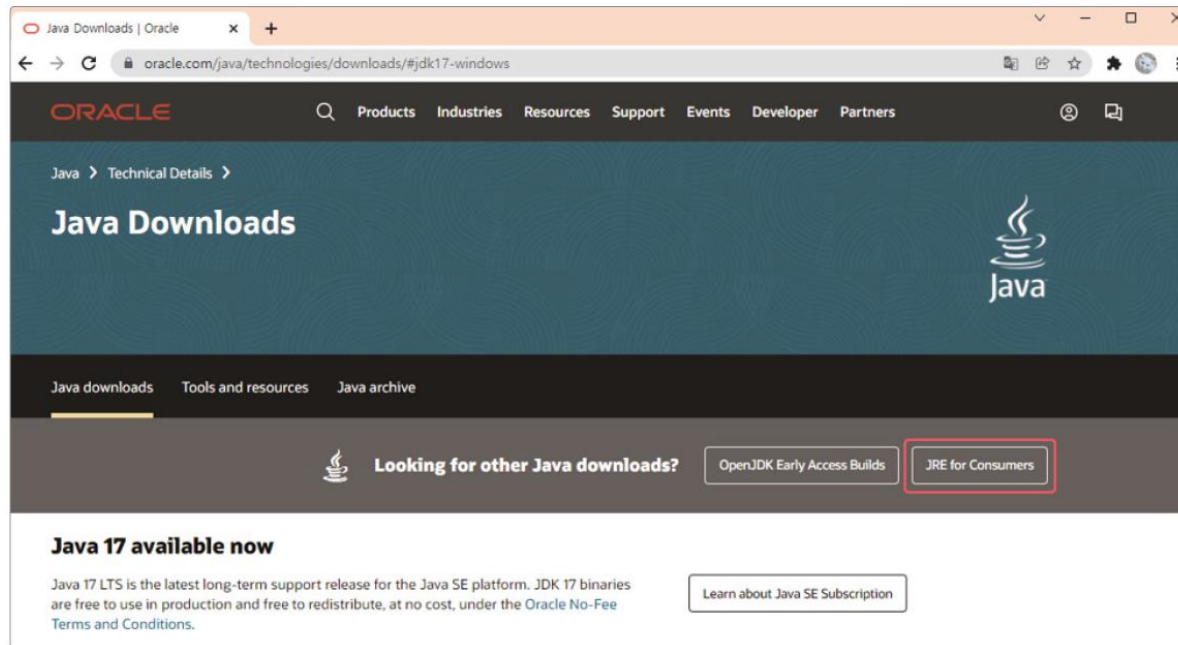


자바 개발 환경 구축

■ JRE 설치하기

• 01. JRE 선택하기

→오라클 사이트의 자바 다운로드(<https://www.oracle.com/java/technologies/downloads/>) 화면 상단에서 <JRE for Consumers>를 클릭하기

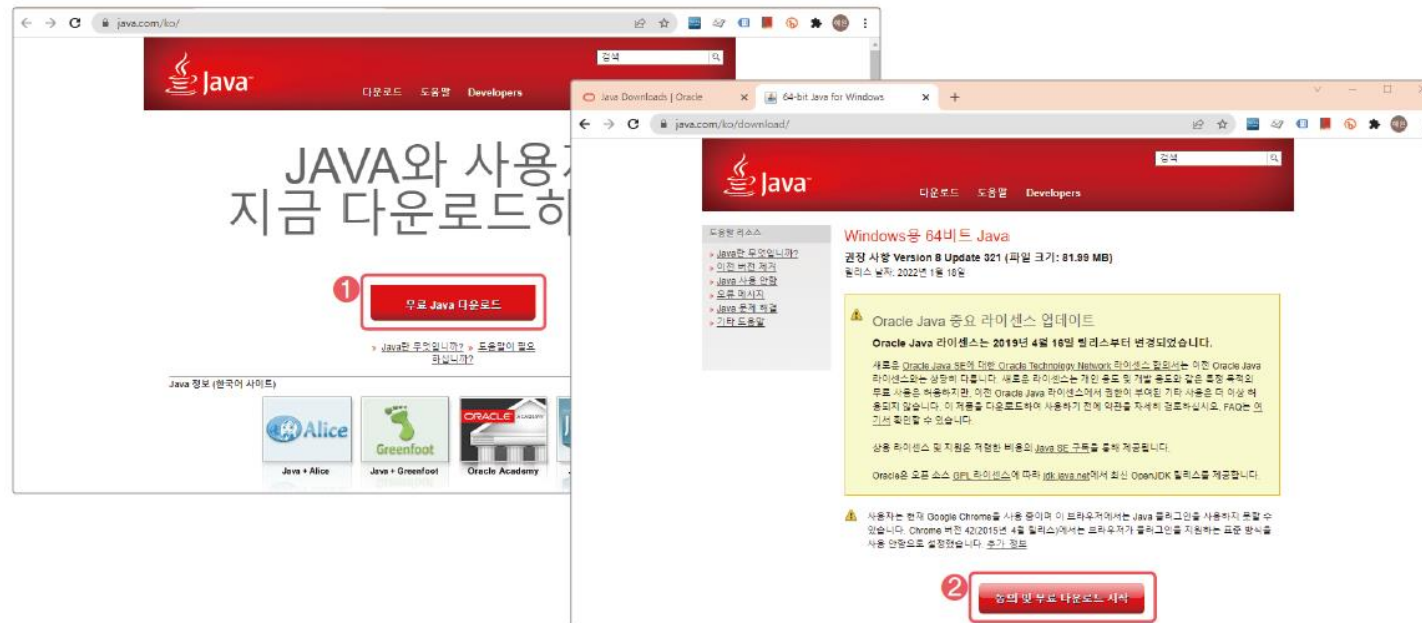


자바 개발 환경 구축

■ JRE 설치하기

• 02. JRE 다운로드하기

→ <무료 Java 다운로드>를 클릭한 뒤, 이어지는 화면에서 <동의 및 무료 다운로드 시작>을 클릭하여 다운로드하기



자바 개발 환경 구축

■ JRE 설치하기

• 03. JRE 설치하기

→ 설치를 시작하여 설치 첫 단계에서 <설치>를 클릭하고 마지막 화면에서 <닫기>를 클릭하여 설치 완료하기

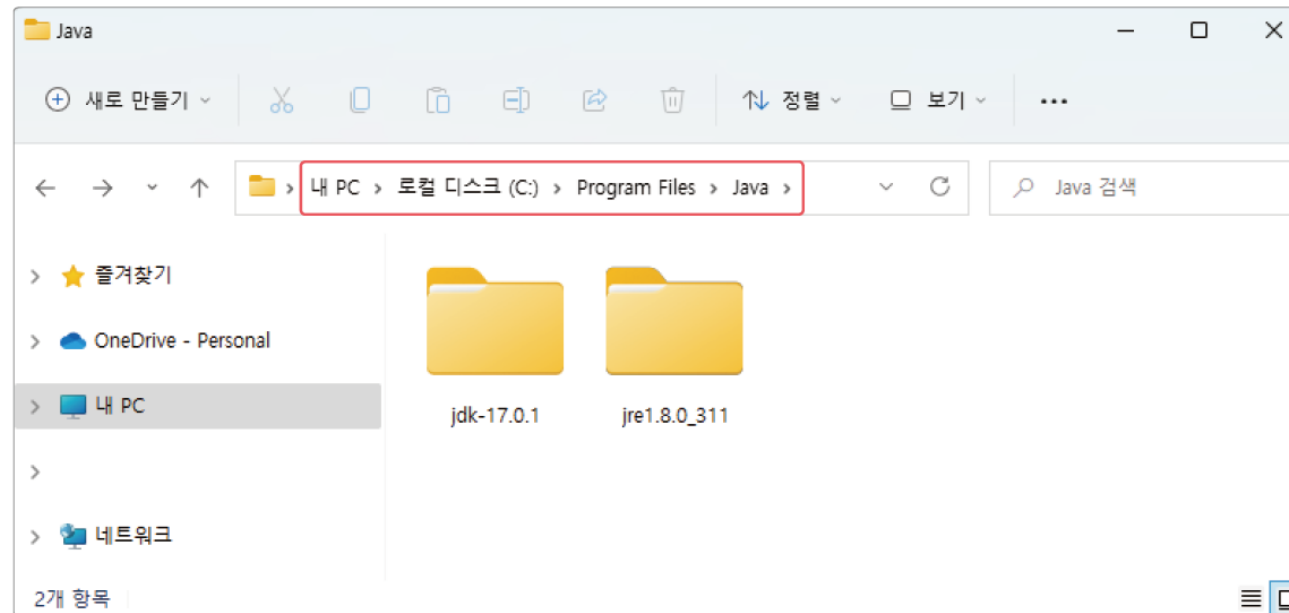


자바 개발 환경 구축

■ JRE 설치하기

• 04. 자바 설치 위치 확인하기

→ 기본 설정으로 자바를 설치하면 C:\Program Files\Java'에 설치됨



자바 개발 환경 구축

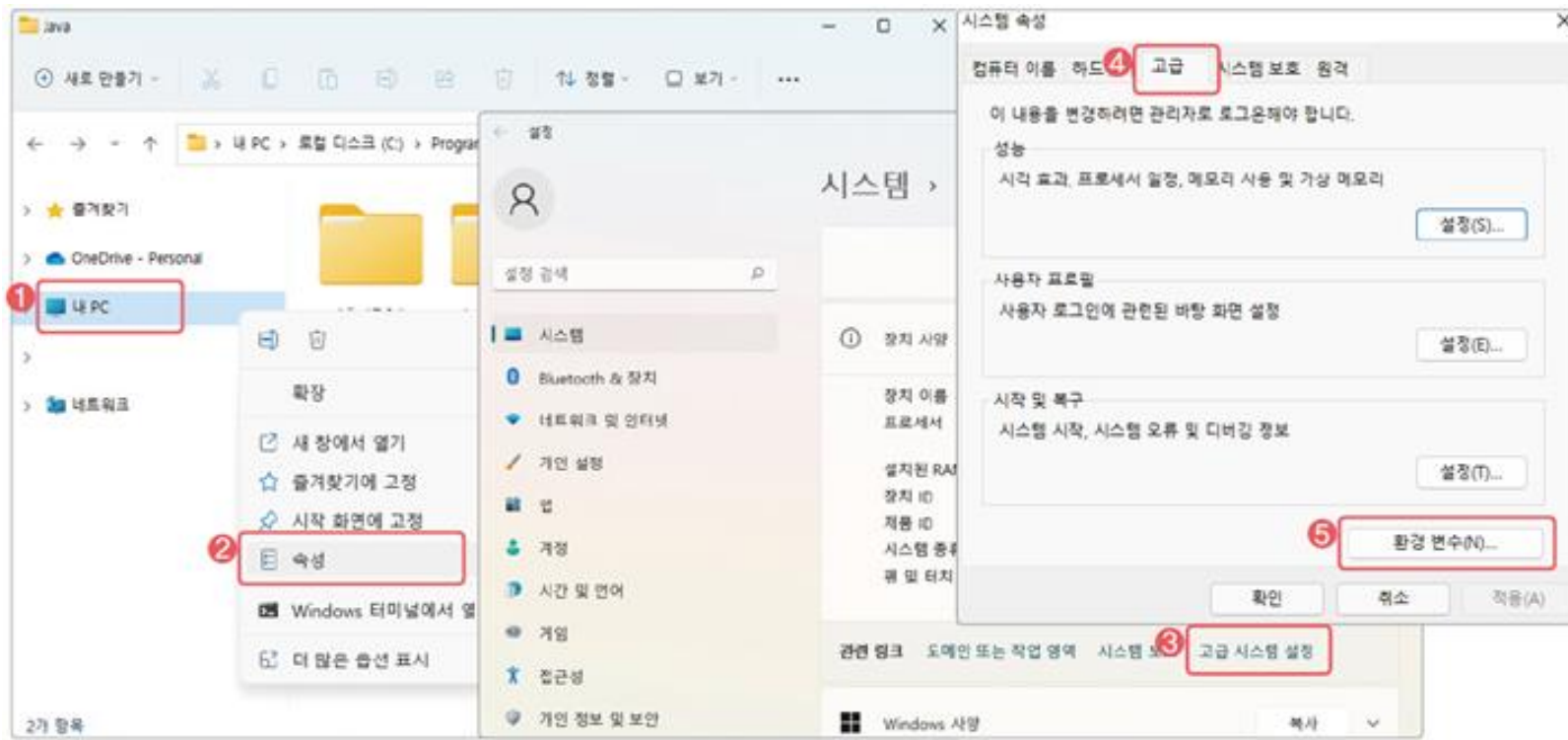
■ JRE 설치하기

- cmd 콘솔 명령 프롬프트에서 javac로 컴파일하거나 다른 프로그램에서 자바 JDK를 참조하려면 윈도우 환경 변수를 지정해야 함
 - 01. [환경 변수] 대화상자 열기
 - ❶ 윈도우 탐색기의 [내 PC]에서 마우스 오른쪽 버튼을 클릭
 - ❷ [속성]에 이어
 - ❸ [고급 시스템 설정]을 선택
 - ❹ [시스템 속성] 창의 [고급] 탭에서
 - ❺ <환경 변수>를 클릭

자바 개발 환경 구축

■ 자바 설치 및 환경 설정

• 01. [환경 변수] 대화상자 열기



자바 개발 환경 구축

■ 자바 설치 및 환경 설정

• 02. 자바 환경 변수 설정하기

→ 환경 변수에는 사용자 변수와 시스템 변수가 있음

✓ 사용자 변수 : 현재 로그인한 사용자에게만 적용되는 환경 변수

✓ 시스템 변수 : 모든 사용자에게 적용되는 환경 변수

→ ❶ [시스템 변수]의 <새로 만들기>를 클릭

→ ❷ JAVA_HOME 환경 변수의 값에 JDK가 설치된 디렉터리 'C:\Program Files\Java\jdk-17.0.1'을 지정하고

→ ❸ <확인>을 클릭

→ ❹ [시스템 변수]의 Path 변수를 선택하고

→ ❺ <편집>을 클릭

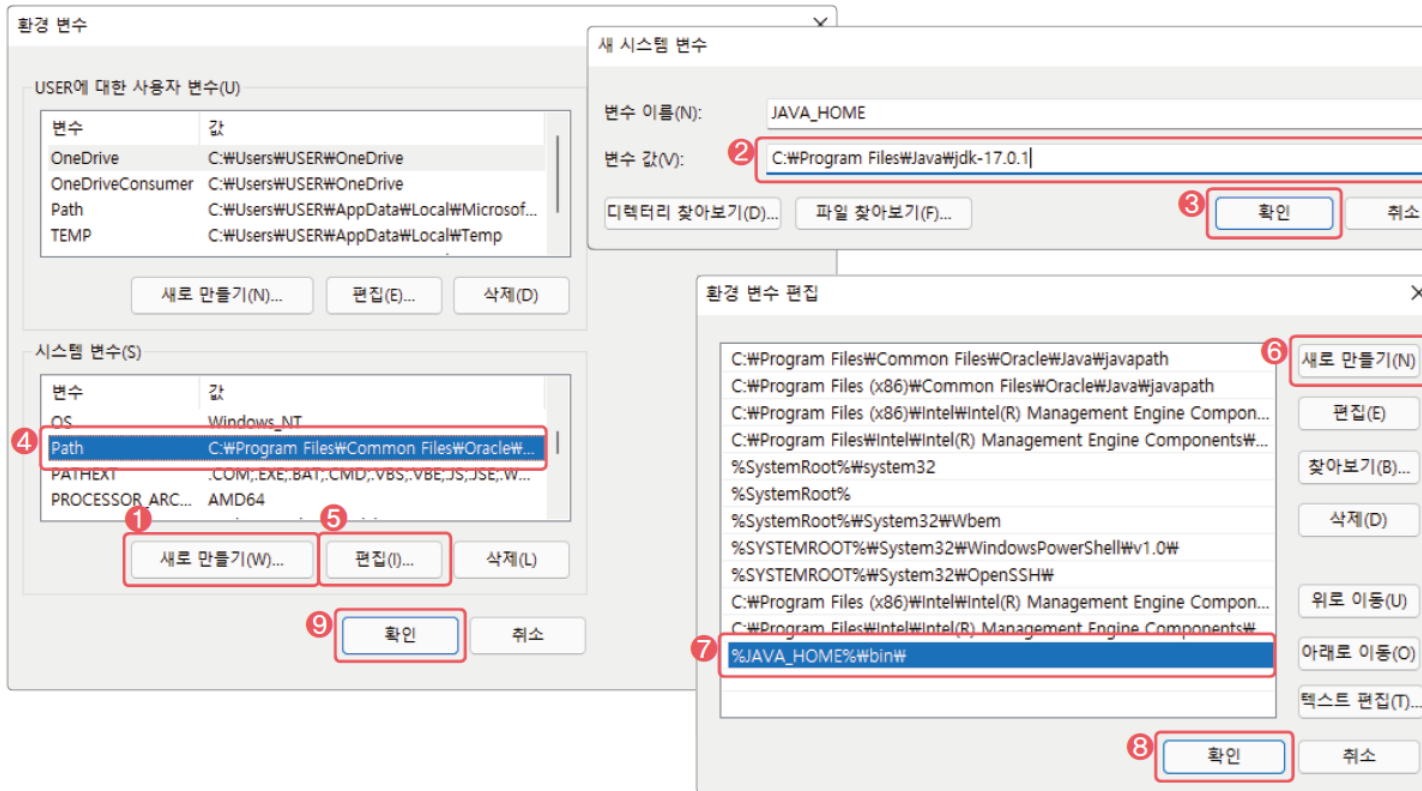
→ ❻ [환경 변수 편집] 창에서 <새로 만들기>를 클릭

→ ❼ Path 변수값의 맨 끝에 앞의 변수값과 구분하기 위한 ';'을 입력하고, JDK가 설치된 위치를 설정한 환경 변수
%JAVA_HOME%\bin을 추가하여 등록

자바 개발 환경 구축

■ 자바 설치 및 환경 설정

• 02. 자바 환경 변수 설정하기



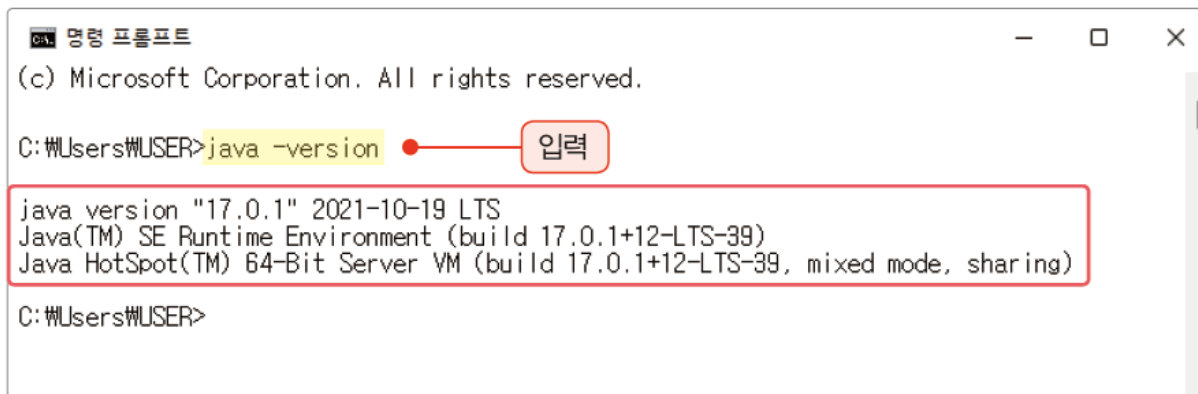
자바 개발 환경 구축

■ 자바 설치 및 환경 설정

• 03. 자바에 설정한 환경 변수 확인하기

→ 윈도우의 시작 버튼에서 'cmd'를 입력하여 명령 프롬프트 창을 띄우기

→ 'java -version'을 입력했을 때 설치된 JDK의 버전이 나오면 정상적으로 설치된 것임



```
명령 프롬프트
(c) Microsoft Corporation. All rights reserved.

C:\Users\WUSER>java -version
java version "17.0.1" 2021-10-19 LTS
Java(TM) SE Runtime Environment (build 17.0.1+12-LTS-39)
Java HotSpot(TM) 64-Bit Server VM (build 17.0.1+12-LTS-39, mixed mode, sharing)

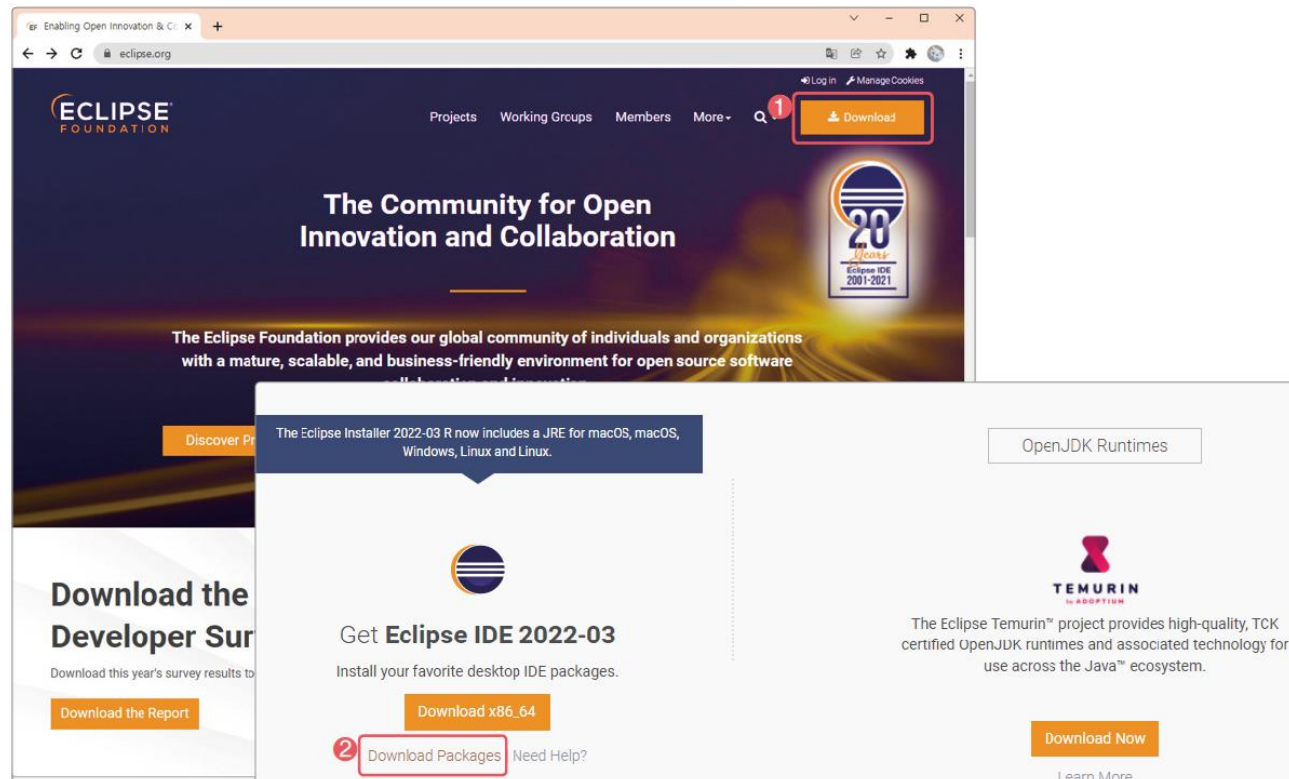
C:\Users\WUSER>
```

자바 개발 환경 구축

■ 이클립스 설치하기

• 01. 이클립스 사이트에 접속하기

→ <http://www.eclipse.org/downloads/> 에 접속하여 하단의 [Download Packages]를 클릭하기



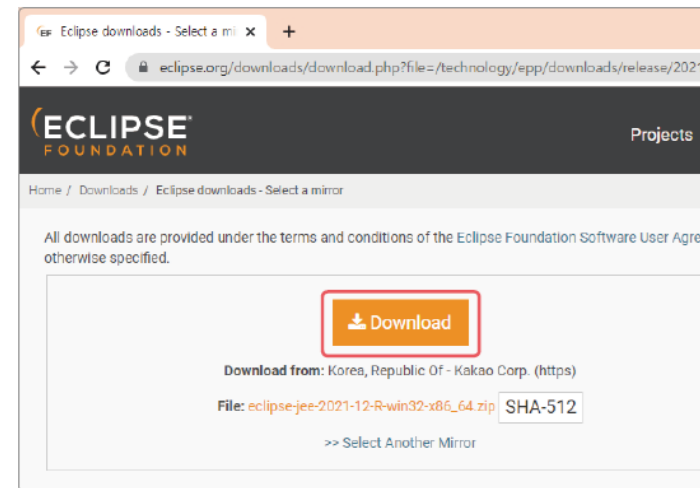
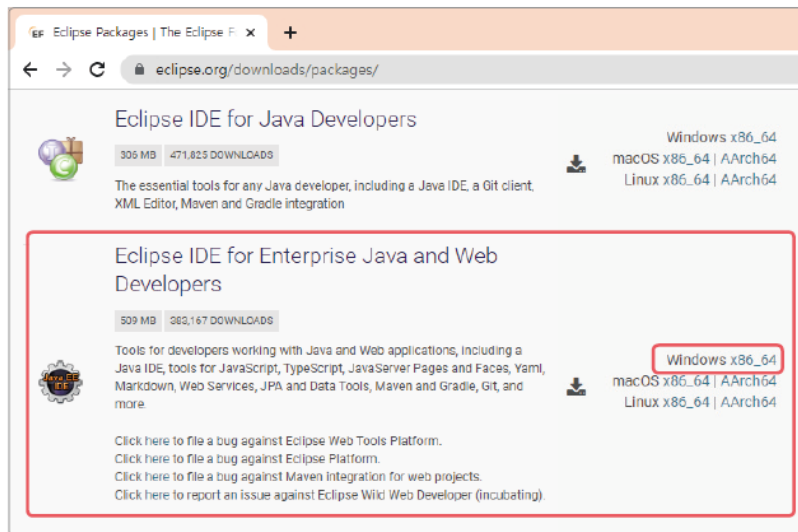
자바 개발 환경 구축

■ 통합 개발 환경 설치

• 02. 이클립스 설치 파일 다운로드하기

→ 컴퓨터의 운영체제에 맞는 이클립스 설치 파일('Eclipse IDE for Enterprise Java and Web Developers')의 'Windows x86_64'를 클릭하기

→ 이어지는 화면에서 eclipse-jee-2021-12-R-win32-x86_64.zip 파일의 <Download>를 클릭



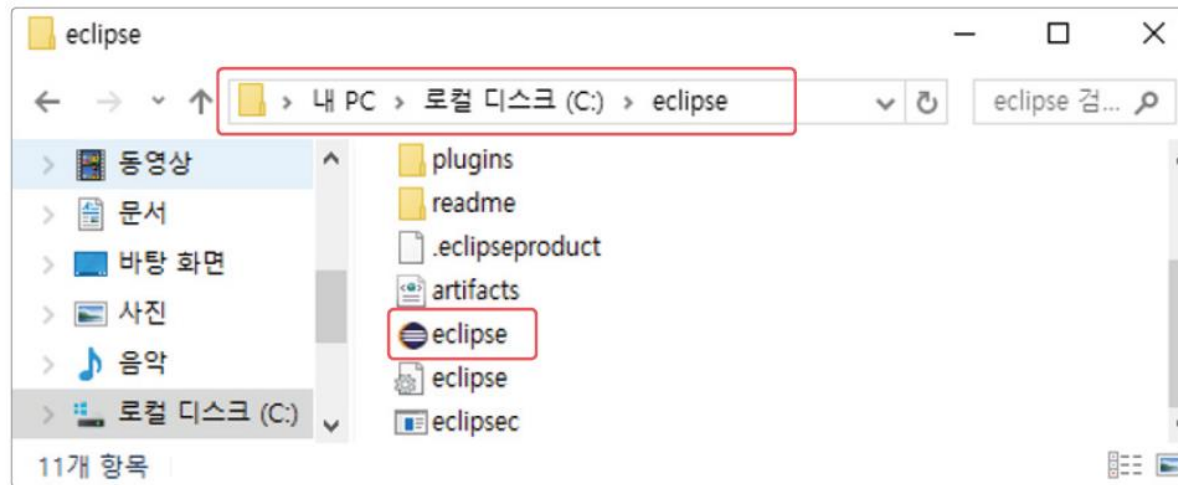
자바 개발 환경 구축

■ 통합 개발 환경 설치

• 03. 설치 완료하고 실행하기

→ 다운로드한 설치 파일의 압축을 풀고 하위에 있는 'eclipse' 폴더를 C 드라이브로 옮기기

→ C:Weclipse 폴더 안의 eclipse.exe 파일을 더블클릭하여 이클립스를 실행하기



자바 개발 환경 구축

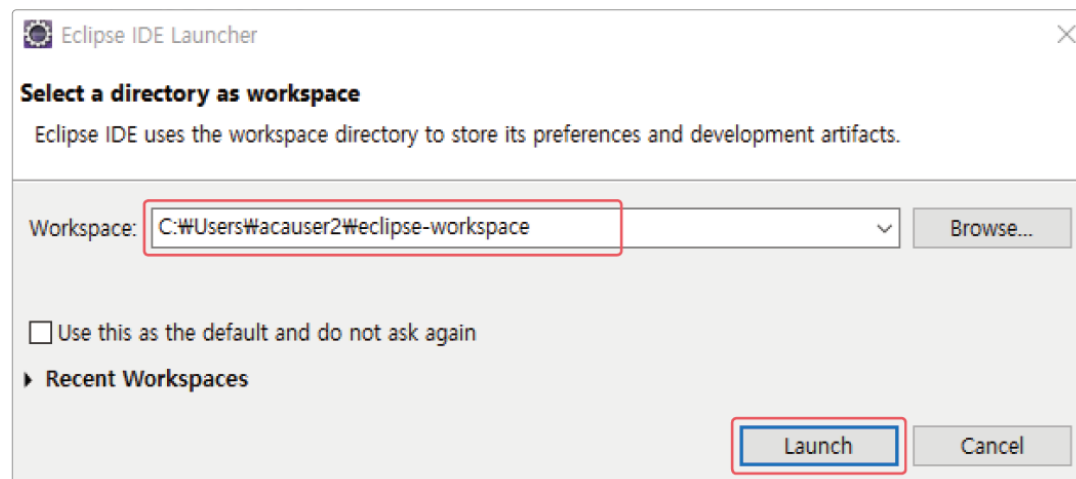
■ 통합 개발 환경 설치

• 04. 이클립스 작업 공간 설정하기

→workspace : 이클립스에서 생성한 프로젝트를 저장하는 공간

→ [Select a directory as workspace] 창에서 기본적으로 C:\Users\사용자\workspace 폴더로 지정되며 변경도 가능함

✓ 작업 공간을 변경하지 않는다면 <Launch>를 클릭

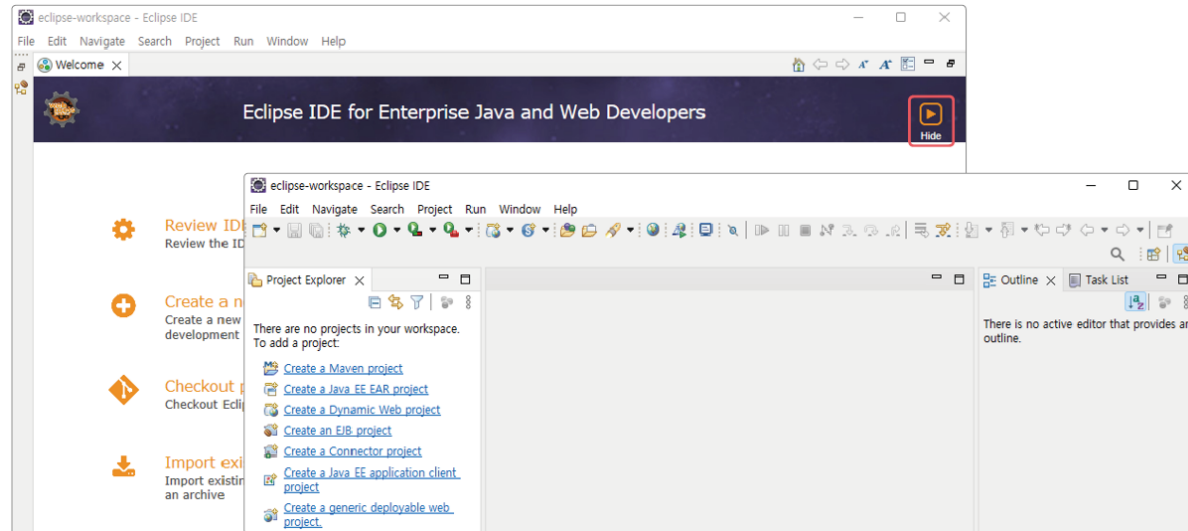


자바 개발 환경 구축

■ 통합 개발 환경 설치

• 05. 이클립스 실행 화면

→ [Welcome] 창 오른쪽 상단의 <Hide>를 클릭하면 이클립스 창이 나타남



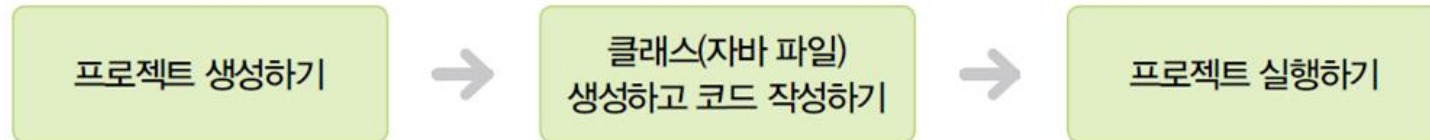
Section 04

처음 만드는 자바 프로그램

처음 만드는 자바 프로그램

■ 자바 애플리케이션 개발 과정

1. 이클립스를 이용하여 프로젝트를 생성
2. 여기에 자바 파일을 생성하고 코드를 작성
3. 프로젝트를 실행



처음 만드는 자바 프로그램

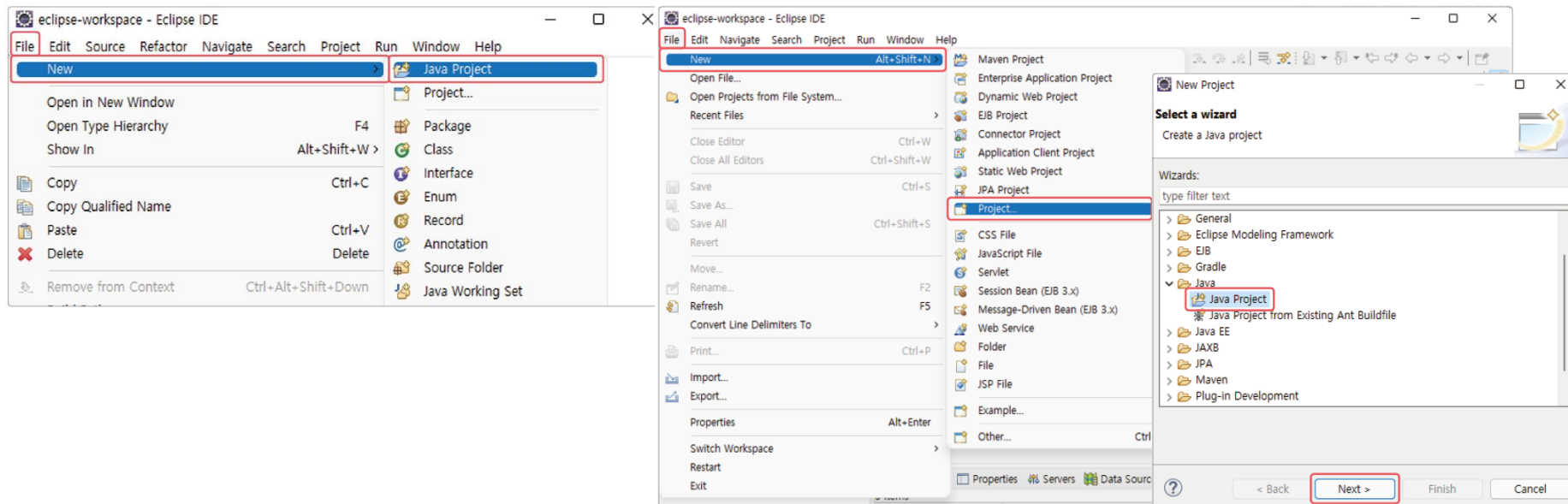
■ 프로젝트 생성하기

• 01. 자바 프로젝트 생성하기

→ 이클립스의 [File]-[New]-[Java Project]를 선택

→ [Java Project] 메뉴가 보이지 않는 경우

✓ [File]-[New]-[Project]를 선택하여 나타나는 창에서 [Java]-[Java Project]를 선택하고 <Next> 클릭

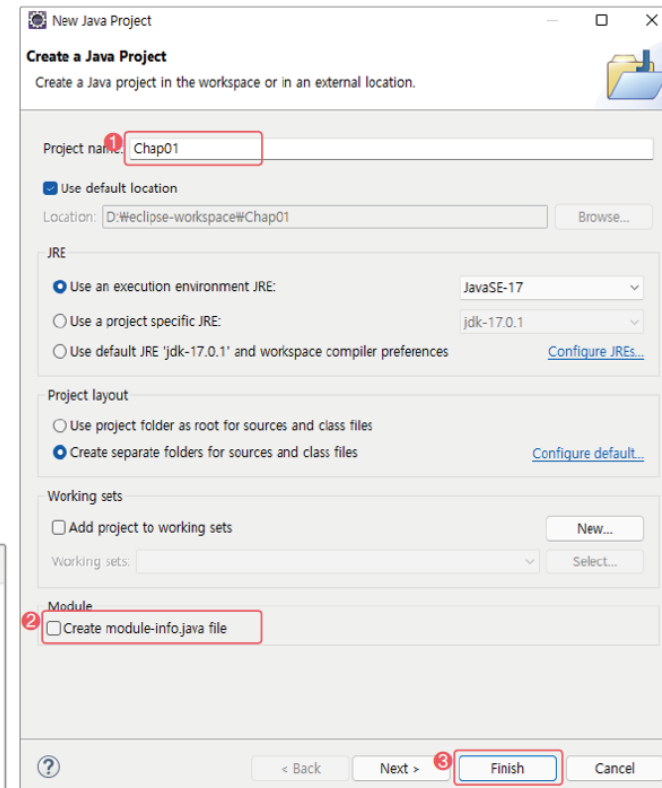
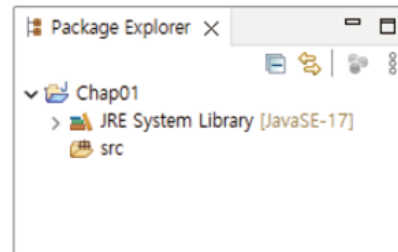


처음 만드는 자바 프로그램

■ 프로젝트 생성하기

• 02. 프로젝트명 설정하기

- ❶ [New Java Project] 창에서 Project name에 'Chap01' 입력
- ❷ Module에서 'Create module-info.java file'의 체크박스가 해제되어 있는지 확인
- ❸ <Finish> 클릭



처음 만드는 자바 프로그램

■ 클래스(자바 파일) 생성하고 코드 작성하기

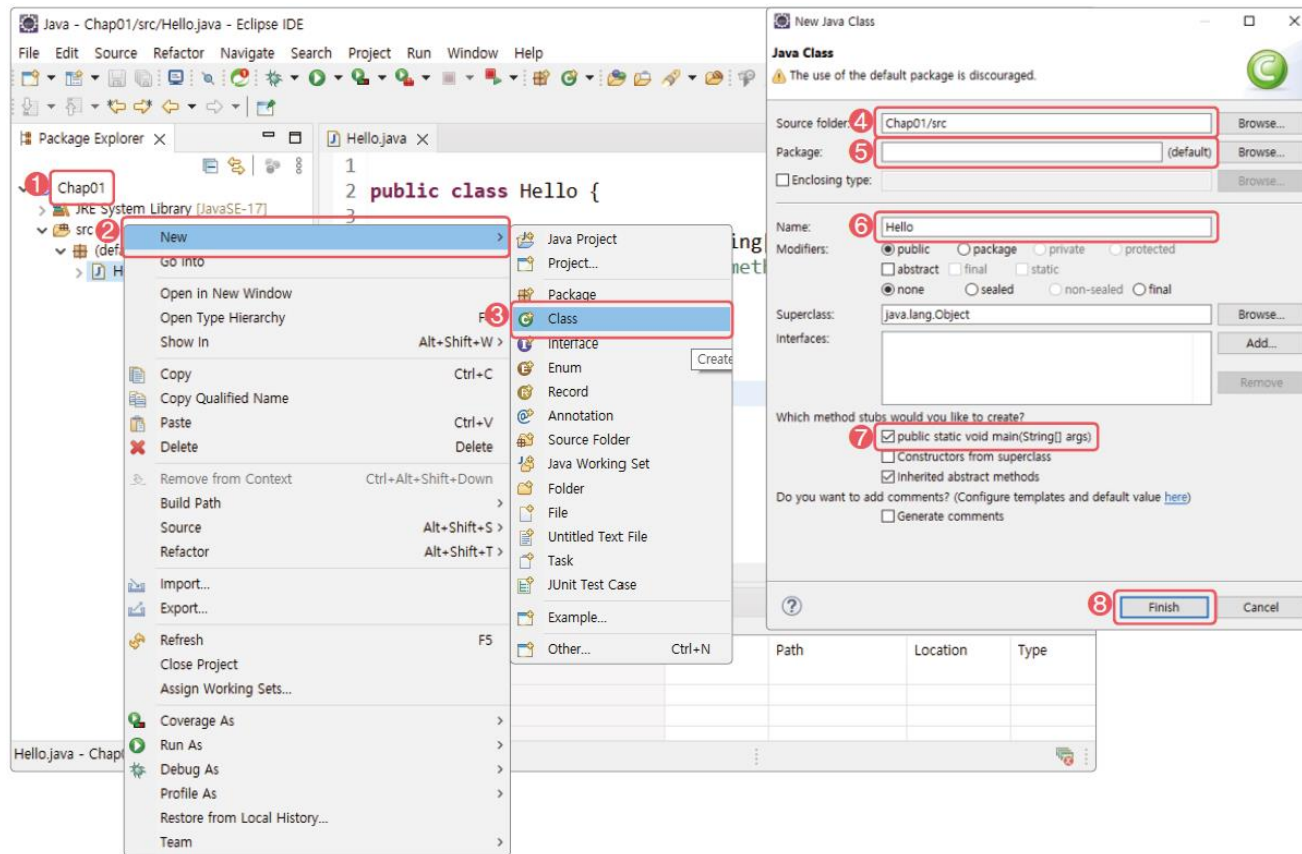
• 03. 자바 파일 생성하기

- ❶ 생성된 Chap01 프로젝트에서 마우스 오른쪽 버튼 클릭
- ❷ [New]와 ❸ [Class]를 선택하여 [New Java Class] 창에서 현재 경로를 나타내는 Source folder가
- ❹ Chap01/src인지 확인하기
- ❺ Package 입력을 생략 (10장에서 다룸)
- ❻ Name에 'Hello'를 입력하며,
- ❼ public static void main(String[] args)에 체크하기
- ❽ <Finish>를 클릭하면 프로젝트 생성이 완료됨

처음 만드는 자바 프로그램

■ 클래스(자바 파일) 생성하고 코드 작성하기

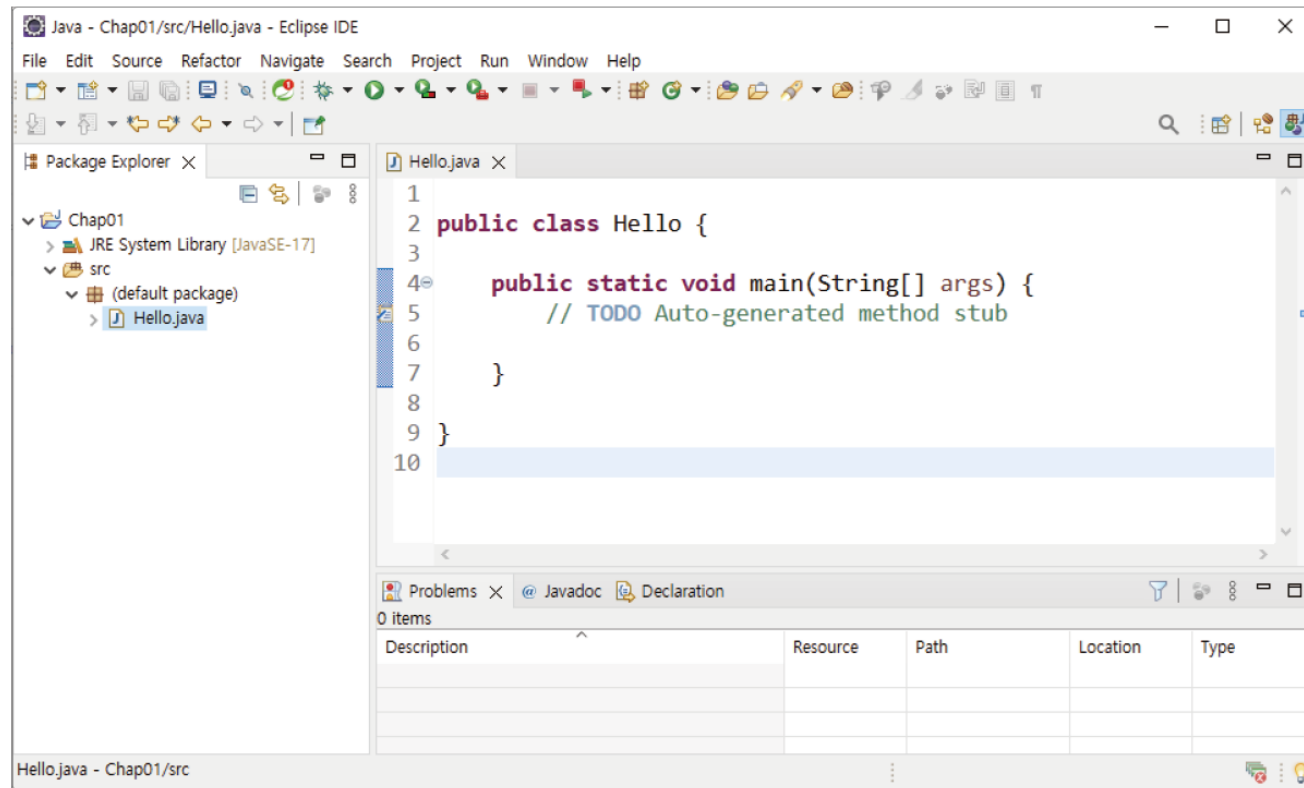
• 03. 자바 파일 생성하기



처음 만드는 자바 프로그램

■ 클래스(자바 파일) 생성하고 코드 작성하기

• 03. 자바 파일 생성하기

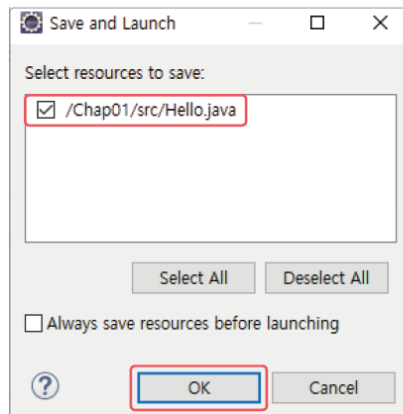
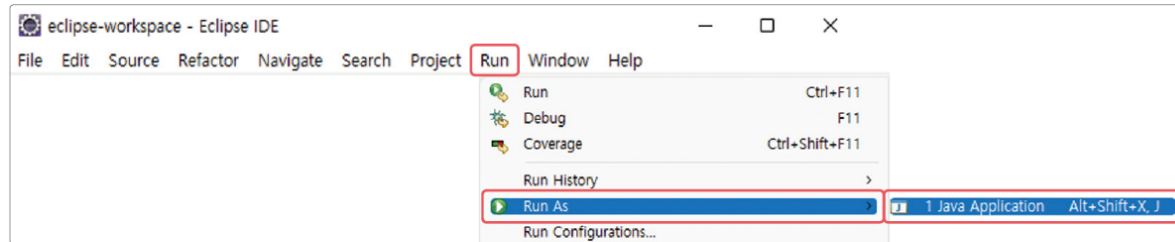


처음 만드는 자바 프로그램

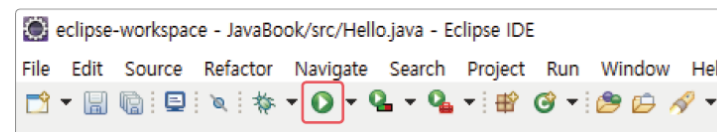
■ 프로젝트 실행하기

• 04. 자바 코드 실행하기

→ [Run]–[Run As]–[Java Application]을 선택하고 [Save and Launch] 창에서 /Chap01/src/Hello.java에 체크하고 <OK>를 클릭



TIP [Run]–[Run As]–[Java Application]을 선택하는 대신 도구 모음의 을 클릭해도 됩니다.

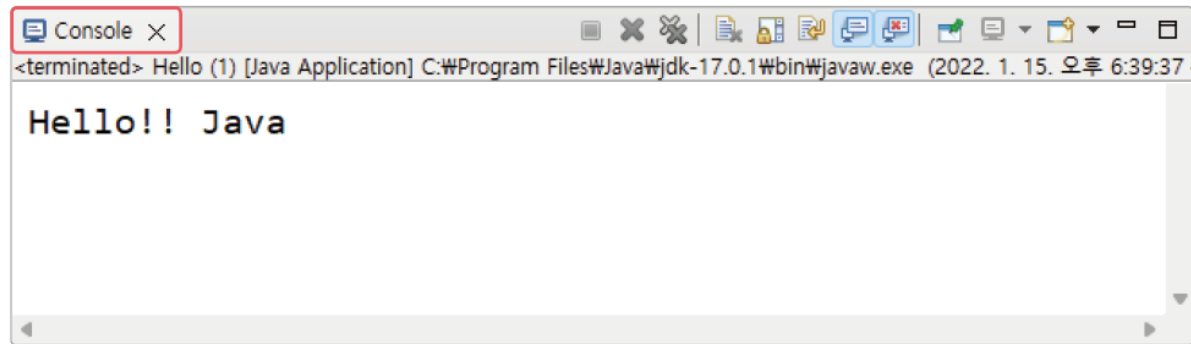


처음 만드는 자바 프로그램

■ 프로젝트 실행하기

• 05. 실행 결과 확인하기

→ 이클립스의 콘솔 창에서 실행 결과를 확인하기



Section 05

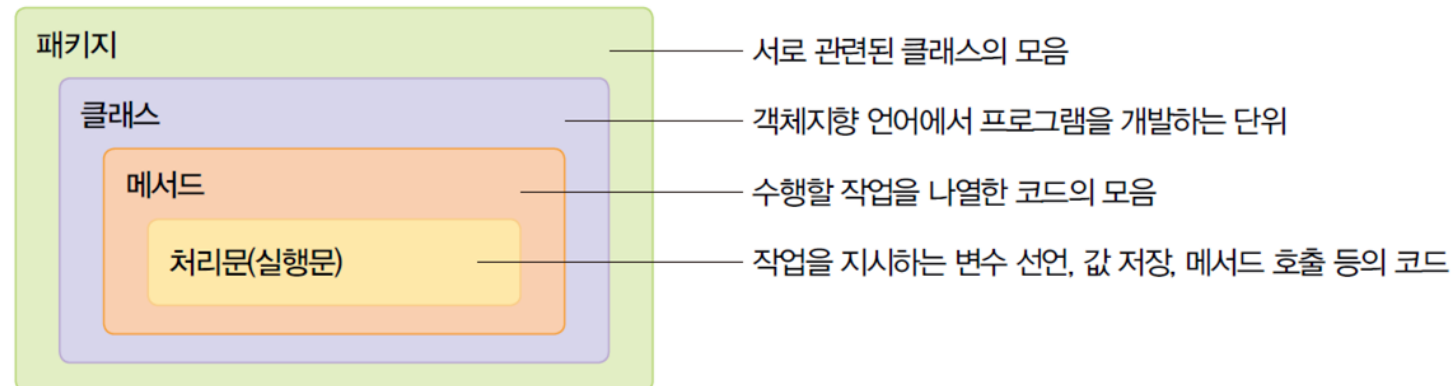
자바 프로그램의 기본 구조

자바 프로그램의 기본 구조

■ 자바 프로그램의 기본 구조

- 자바 프로그램은 하나 이상의 클래스로 구성됨
- 각 클래스의 프로그램 코드를 별도의 소스 파일에 저장하고 각 소스 파일명을 소스 파일에 정의된 클래스명과 동일하게 지정해야 함
 - 만약 이름이 다르면 컴파일 과정에서 오류 발생
 - 모든 자바 소스 파일의 확장자 : .java

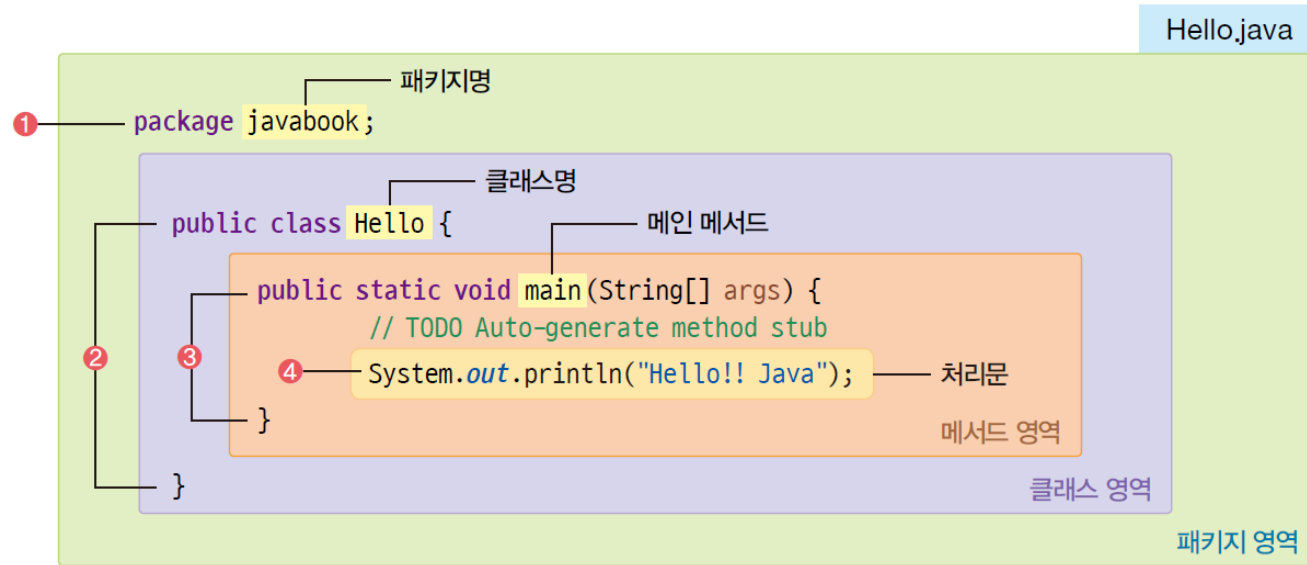
소스 파일(클래스명.java)



자바 프로그램의 기본 구조

■ 자바 프로그램의 기본 구조

- Hello.java 파일의 기본 구조



자바 프로그램의 기본 구조

■ 패키지

- 패키지는 기능을 기반으로 클래스를 구성하는 데 사용됨
- 패키지문 : 클래스가 저장되는 네임스페이스
- 패키지문을 생략하면 이름이 없는 기본 패키지에 클래스명만 선언됨

```
package 패키지명;
```

→ [예] package javabook;

- ✓ ‘프로젝트명/src’ 폴더 안에 ‘javabook’ 폴더가 생성됨
- ✓ javabook이 패키지명이며, 패키지명은 소문자로 지정하는 것이 관례임

자바 프로그램의 기본 구조

■ 클래스

- class 키워드를 사용하여 클래스를 선언함
- 클래스명은 대부분 첫 글자가 대문자로 시작함
- 전체 클래스의 내용은 중괄호({ }) 안에 포함되어야 함
- public 키워드를 사용하여 패키지 외부에서 클래스의 접근 가능성을 지정함

```
public class 클래스명 {  
  
}
```

→[예] class Hello { }

✓ 'src/javabook' 패키지 안에 'Hello.java' 파일을 저장함

→ 클래스명과 자바 파일명이 반드시 동일해야 함

✓ 클래스명이 Hello이므로 자바 파일명을 'Hello.java'로 저장해야 함

✓ 'Hello.java'와 'hello.java'는 동일하지 않음

자바 프로그램의 기본 구조

■ main() 메서드

- main() 메서드 : 모든 자바 애플리케이션의 시작점이자 진입점임
→ 즉 자바 애플리케이션이 시작될 때마다 가장 먼저 호출되는 메서드임
- 메서드명은 대부분 소문자로 시작함
- 하나의 자바 프로그램에는 main() 메서드를 가진 클래스가 반드시 존재해야 함

```
public static void main(String[] args) {  
    }  
}
```

→ 중괄호({ }) : 중괄호 내부에 main() 메서드의 내용을 담음

→ String[] args : main() 메서드가 문자열 배열을 입력받을 수 있음을 의미함

자바 프로그램의 기본 구조

■ 처리문(statement)

- 자바 프로그램의 동작을 명시하고 이러한 동작을 컴퓨터에 알려주는 데 사용되는 문장
- 자바의 모든 처리문은 반드시 세미콜론(;)으로 끝나야 함
- 프로그램을 실행하면 main() 메서드 안의 처리문이 순차적으로 실행됨

```
System.out.println( ); // 괄호 안의 내용을 출력한 후 줄바꿈  
System.out.print( );  // 괄호 안의 내용을 출력한 후 줄바꿈하지 않음
```

→[예] System.out.println("Hello!! Java");

✓ System.out.println() : 큰따옴표(" ") 안에 있는 Hello!! Java를 출력 창에 표시하는 메서드

자바 프로그램의 기본 구조

■ 주석(comment)

- 코드에 대한 이해를 돕기 위해 프로그램에 설명을 덧붙이거나 디버깅을 위해 작성하는 일종의 메모

→ 주석은 바이트코드로 컴파일되지 않음

- 줄 주석

→//로 시작하며 한 행을 주석 처리함. 줄 주석에는 끝을 나타내는 기호가 필요하지 않음

```
// 설명문
```

- 블록 주석

→/*로 시작하여 */로 끝나는 행까지 여러 행을 주석 처리함

```
/* 설명문 */
```

```
/*  
    설명문  
*/
```

프로그래밍기초

Copyright © Lee Seungwon Professor
All rights reserved.

<Q&A : lsw@kopo.ac.kr>