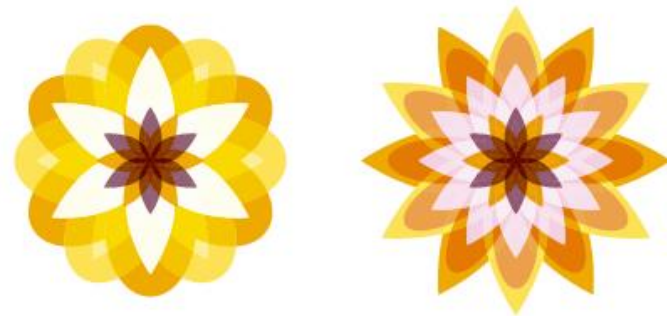


*Chapter 00*

# 자바 시작하기



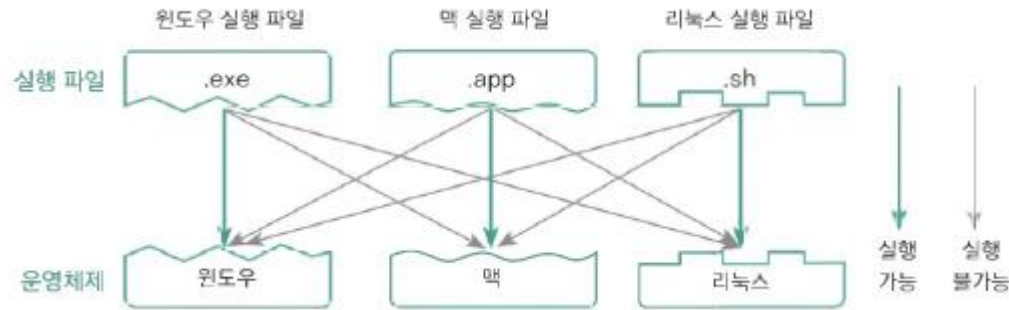
## ■ 자바의 플랫폼 독립성

- 자바는 가장 널리 쓰이는 프로그래밍 언어 중 하나다.
- 자바는 어떻게 높은 점유율을 얻으면서 전 세계 개발자들에게 사랑받게 됐을까?
- 자바는 플랫폼 독립성, 객체지향 언어, 함수형 코딩 지원, 분산처리 지원, 멀티 쓰레드 지원 등 여러 가지 특징을 지닌 프로그래밍 언어다.
- 이 중 가장 큰 특징은 '플랫폼 독립성'이다.
- 이 특징은 자바의 좌우명인 'Write Once, Run Anywhere(한 번 작성하면 어느 플랫폼에서나 실행)'와도 직결된다.
- 나머지 특징들은 나중에 알아보기로 하고, 여기서는 자바의 대표적인 특징인 플랫폼 독립성부터 알아보자.

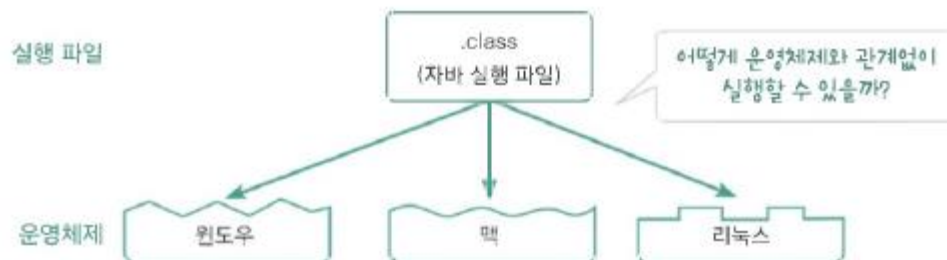
# 1. 프로그래밍 언어와 자바

## ■ 자바의 플랫폼 독립성

- 플랫폼 종속성, 플랫폼 독립성이란?



플랫폼 종속적 프로그램의 특징



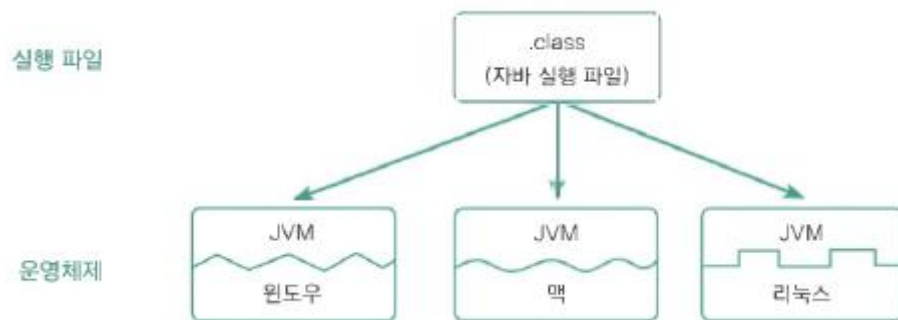
플랫폼 독립적 프로그램의 특징

# 1. 프로그래밍 언어와 자바

## ■ 자바의 플랫폼 독립성

### ■ 플랫폼 종속성, 플랫폼 독립성이란?

- 자바는 어떻게 플랫폼독립성을 지닐 수 있었을까?
- 그것은 바로 자바 가상 머신(JVM, Java Virtual Machine) 덕분이다.



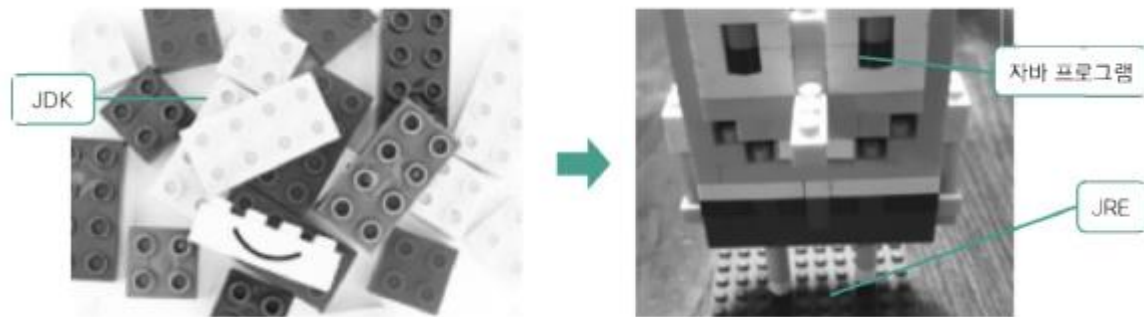
자바 가상 머신을 이용한 자바의 플랫폼 독립성

# 1. 프로그래밍 언어와 자바

## ■ 자바 개발 도구와 자바 실행 환경

### ■ 자바 개발 도구란?

- JDK는 말 그대로 자바를 이용해 프로그램을 개발하는 데 필요한 도구를 모아 둔 집합
- JRE는 완성된 프로그램을 실행하는 데 필요한 환경



자바 프로그램, 자바 개발 도구, 자바 개발 환경의 개념



## 2. 자바 개발 환경

■ 자바를 개발하기 위해서는 JDK와 이클립스(Eclipse)를 순서대로 설치해야 한다.

### ■ JDK 설치

- JDK는 다음의 오라클 사이트에서 내려받을 수 있다.

<https://www.oracle.com/java/technologies/downloads/>

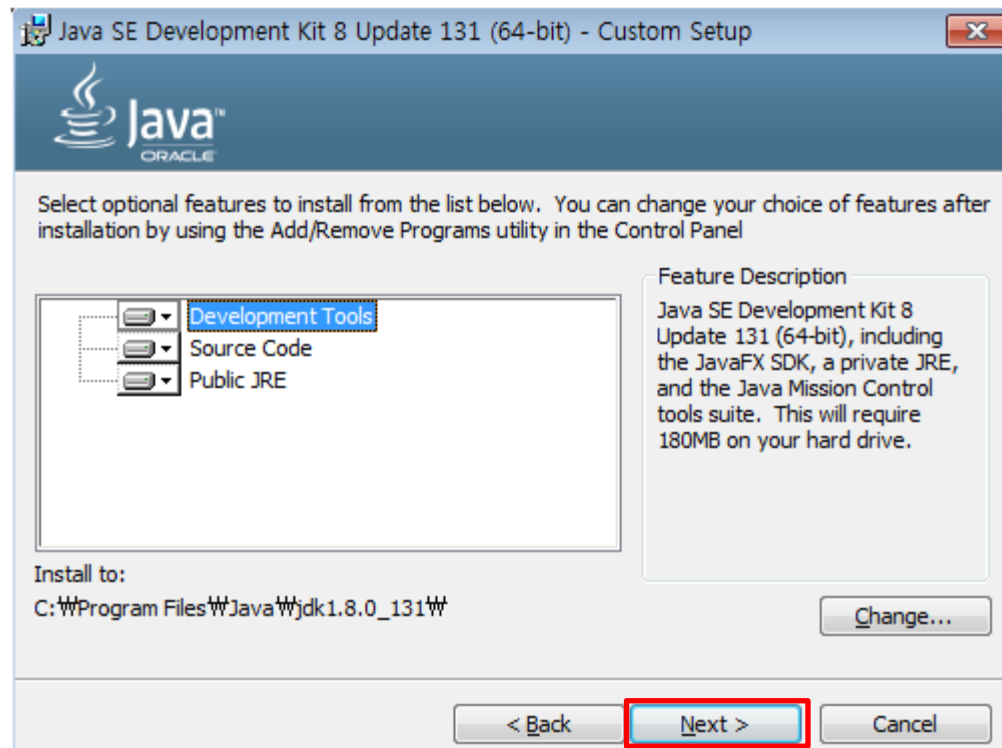
- 화면에서 windows를 선택하고, x64 Installer를 선택하여 파일을 다운로드

The screenshot shows the Oracle Java SE Development Kit 18.0.1.1 download page. The 'Windows' tab is selected, and the 'x64 Installer' download link is highlighted with a red box. The page includes a table with download links for different file types and sizes.

Product/file description	File size	Download
x64 Compressed Archive	172.8 MB	<a href="https://download.oracle.com/java/18/latest/jdk-18_windows-x64_bin.zip">https://download.oracle.com/java/18/latest/jdk-18_windows-x64_bin.zip</a> (sha256 <a href="#">🔗</a> )
x64 Installer	153.38 MB	<a href="https://download.oracle.com/java/18/latest/jdk-18_windows-x64_bin.exe">https://download.oracle.com/java/18/latest/jdk-18_windows-x64_bin.exe</a> (sha256 <a href="#">🔗</a> )
x64 MSI Installer	152.26 MB	<a href="https://download.oracle.com/java/18/latest/jdk-18_windows-x64_bin.msi">https://download.oracle.com/java/18/latest/jdk-18_windows-x64_bin.msi</a> (sha256 <a href="#">🔗</a> )

## 2. 자바 개발 환경

- 자바를 개발하기 위해서는 JDK와 이클립스(Eclipse)를 순서대로 설치해야 한다.
- JDK 설치
  - 다운받은 파일을 실행하고 디폴트 값으로 [Next]를 클릭

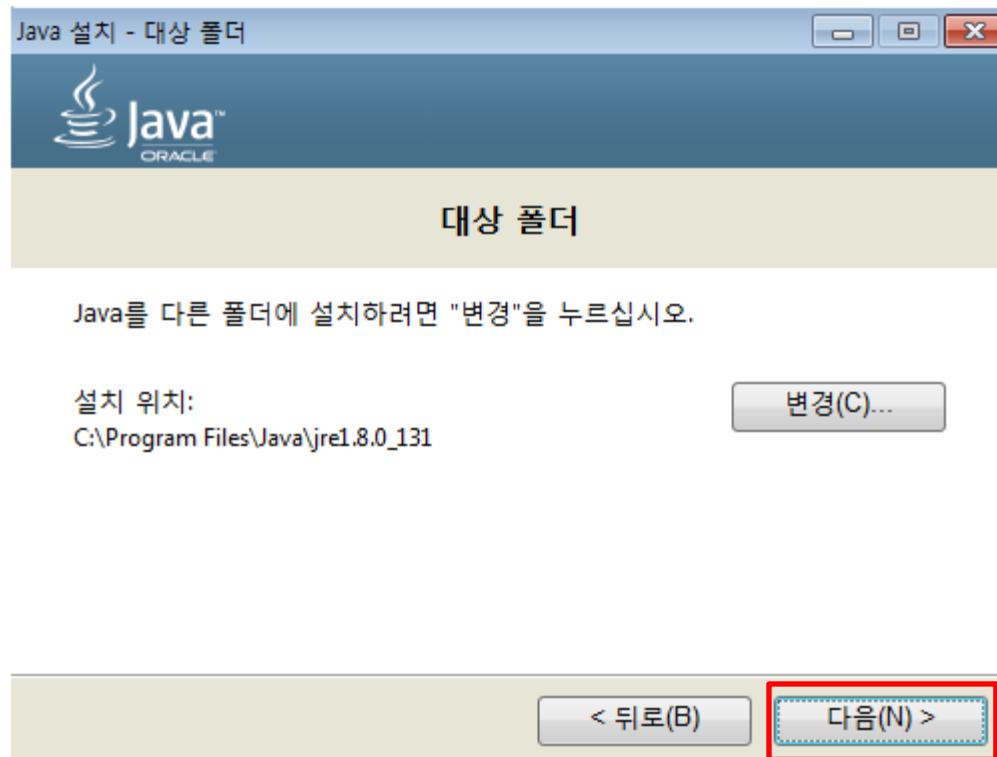


## 2. 자바 개발 환경

■ 자바를 개발하기 위해서는 JDK와 이클립스(Eclipse)를 순서대로 설치해야 한다.

■ JDK 설치

- 설치할 폴더를 선택 후 [다음(N)]을 클릭





## 2. 자바 개발 환경

■ 자바를 개발하기 위해서는 JDK와 이클립스(Eclipse)를 순서대로 설치해야 한다.

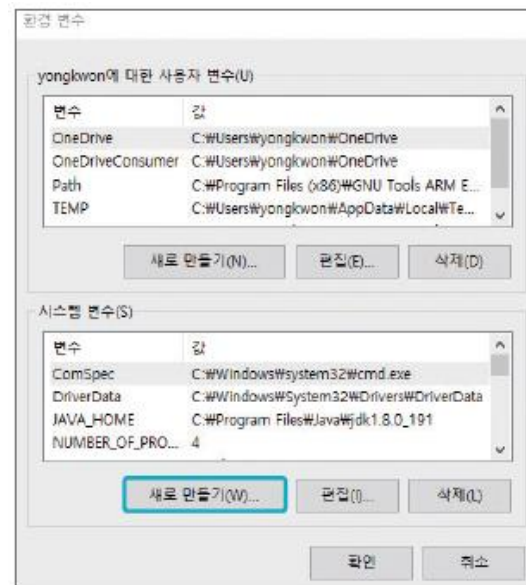
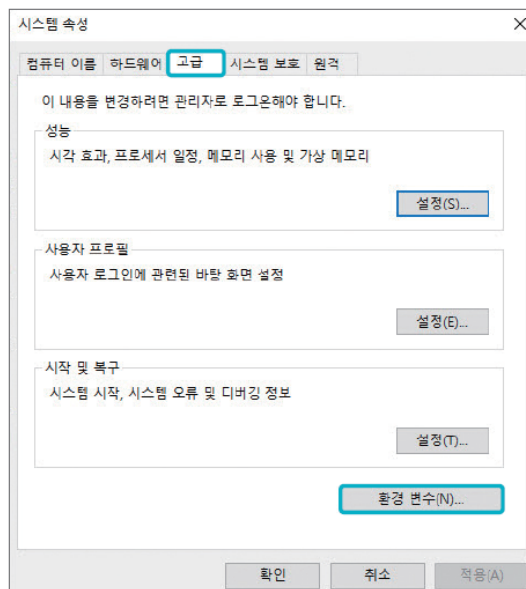
### ■ 환경변수 설정

#### ■ JAVA\_HOME

- JDK가 설치된 폴더
- JDK 위치를 찾을 때 JAVA\_HOME 환경 변수 이용 필요한 경우 있음

#### ■ JAVA\_HOME 환경 변수 만들고 JDK 설치 폴더 등록

- [시스템 속성] - [고급] - [환경 변수] - [시스템 변수] - [새로 만들기]



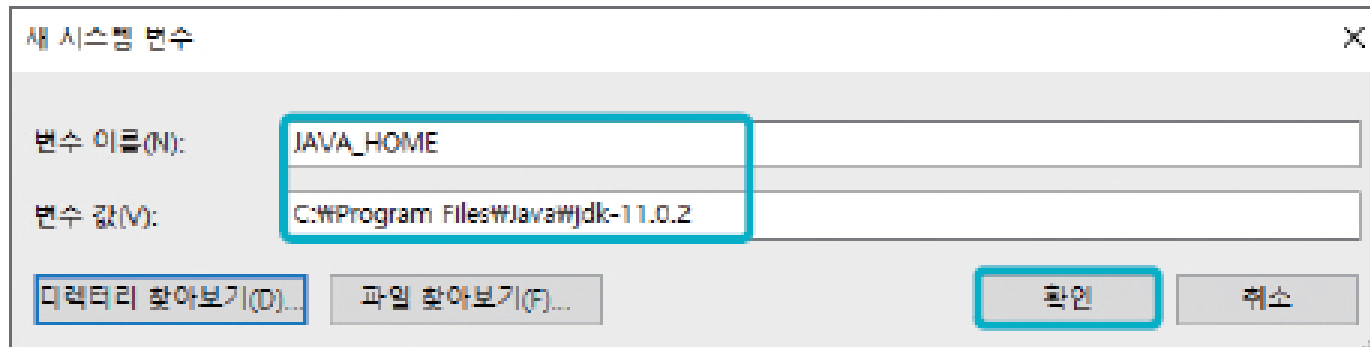
## 2. 자바 개발 환경

■ 자바를 개발하기 위해서는 JDK와 이클립스(Eclipse)를 순서대로 설치해야 한다.

### ■ 환경변수 설정

#### ■ JAVA\_HOME 환경 변수 만들고 JDK 설치 폴더 등록

- [새 시스템 변수] 대화상자
  - [변수 이름] : JAVA\_HOME
  - [변수 값] : JDK 설치 경로 입력
  - [확인]

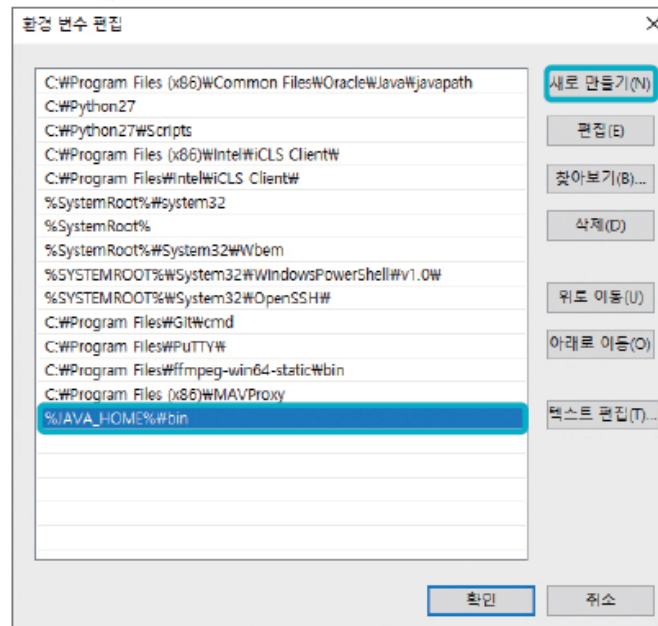
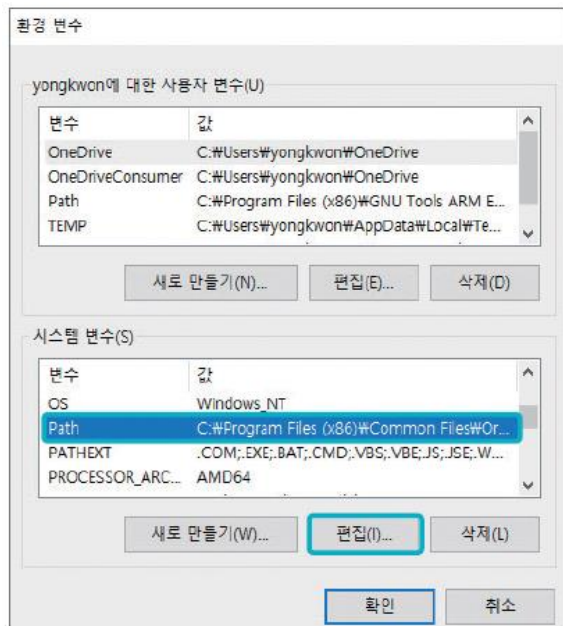
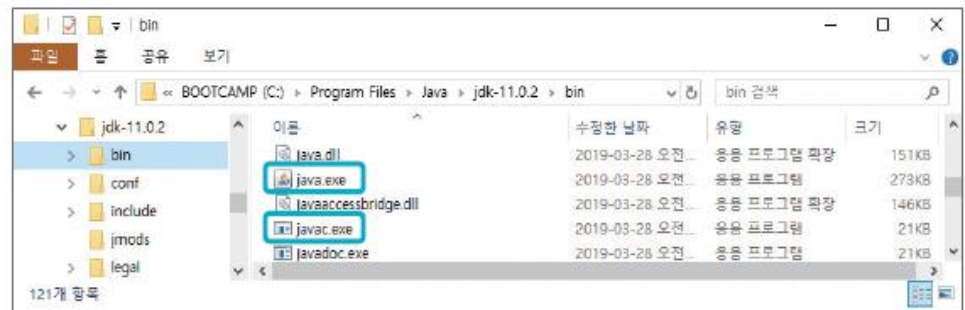


## 2. 자바 개발 환경

### ■ 환경변수 설정

#### ■ Path 환경 변수 수정

- javac 및 java 명령어를 다른 폴더에서 사용하려면 Path에 bin 폴더 등록
- [환경 변수] 대화상자
  - [시스템 변수]에서 Path 환경변수 선택 후 [편집]
- [환경 변수 편집] 대화상자
  - [새로 만들기] - %JAVA\_HOME%\bin 입력

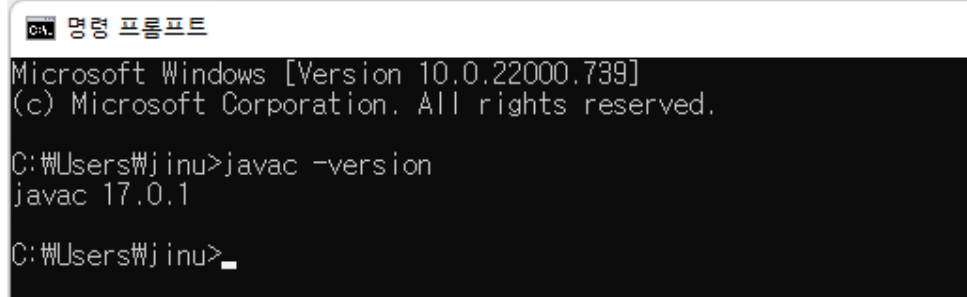


## 2. 자바 개발 환경

### ■ 환경변수 설정

#### ■ Path 환경 변수 수정

- 명령 프롬프트 실행
  - javac -version 입력 후 키보드 [Enter]
  - 그림과 같이 출력



```
C:\> 명령 프롬프트
Microsoft Windows [Version 10.0.22000.739]
(c) Microsoft Corporation. All rights reserved.

C:\Users\jinu>javac -version
javac 17.0.1

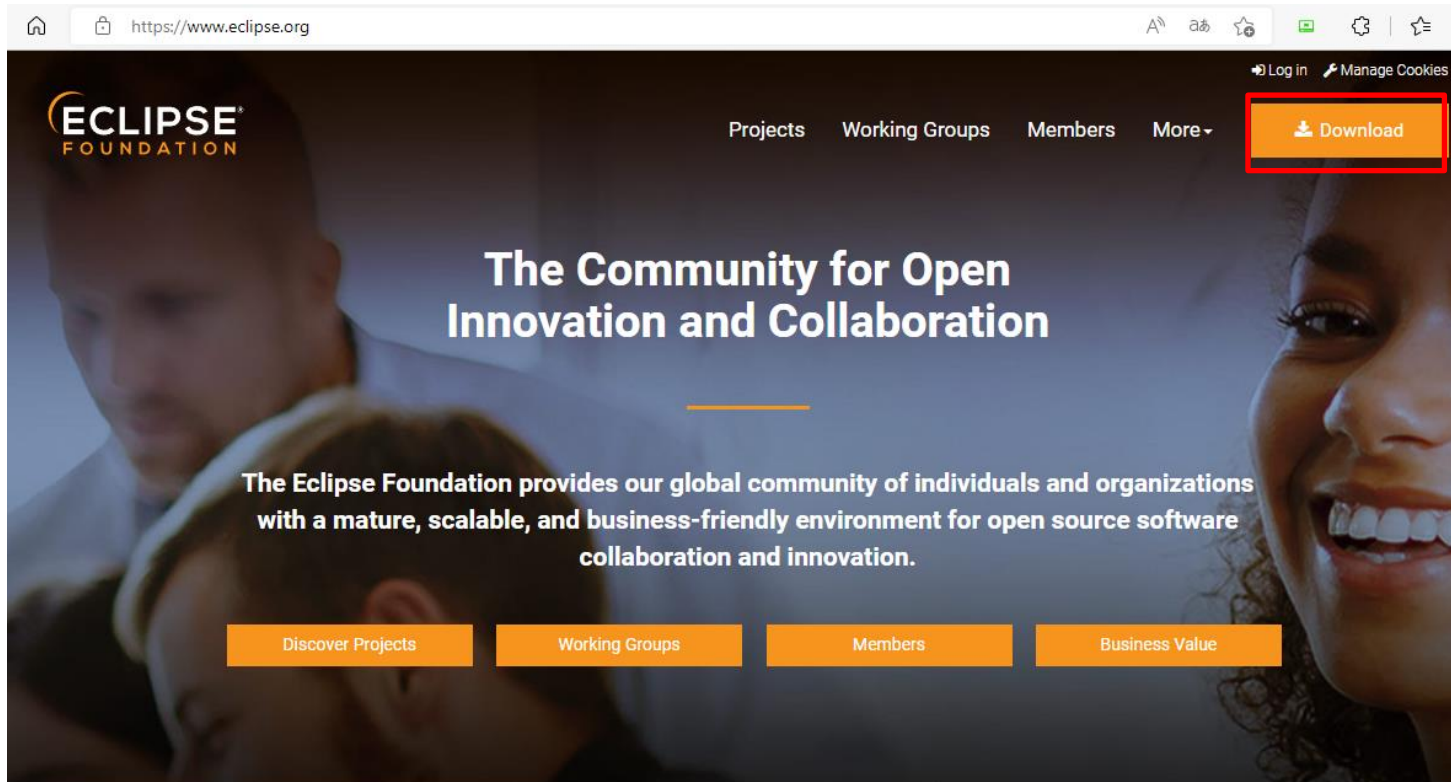
C:\Users\jinu>_
```

## 2. 자바 개발 환경

### ■ 이클립스 설치

- 이클립스(Eclipse)는 자바 개발에 가장 많이 사용되는 무료 통합 개발 환경(IDE, Integrated Development Environment) 으로, 자바뿐 아니라 C, C++ , PHP, 파이썬 등의 개발에도 사용된다.
- 다음 사이트에 접속한 후 [Download]를 클릭

<https://www.eclipse.org/>



## 2. 자바 개발 환경

### ■ 이클립스 설치

- 다운로드 페이지에서 [Download]를 클릭

The screenshot shows the Eclipse Foundation website. At the top, there's a navigation bar with the Eclipse Foundation logo and links for Projects, Working Groups, Members, and a More dropdown. Below the navigation bar, there's a main section with the text "Download Eclipse Technology that is right for you". To the right of this text is a sponsored ad for Red Hat Developer. Below the main text is another sponsored ad for SDV Contribution Day. At the bottom of the page, there are two main download sections. The left section is for "Get Eclipse IDE 2022-06" and includes a "Download x86\_64" button, which is highlighted with a red box. The right section is for "OpenJDK Runtimes" and includes a "Download Now" button. Both sections also have links for "Download Packages" and "Need Help?".

ECLIPSE FOUNDATION

Projects Working Groups Members More ▾

Download Eclipse Technology that is right for you

Sponsored Ad

Red Hat Developer

Build here. Go anywhere.

Get updates on Red Hat Developer events that help you learn and dev tools that help you lead.

Join the DevNation

Advertise Here

SDV Contribution Day

Hybrid Event | June 30, 2022  
ZF Forum | Friedrichshafen, Germany

Advertise Here

The Eclipse Installer 2022-06 R now includes a JRE for macOS, Windows and Linux.

OpenJDK Runtimes

Get **Eclipse IDE 2022-06**

Install your favorite desktop IDE packages.

**Download x86\_64**

Download Packages | Need Help?

TEMURIN

The Eclipse Temurin™ project provides high-quality, TCK certified OpenJDK runtimes and associated technology for use across the Java™ ecosystem.

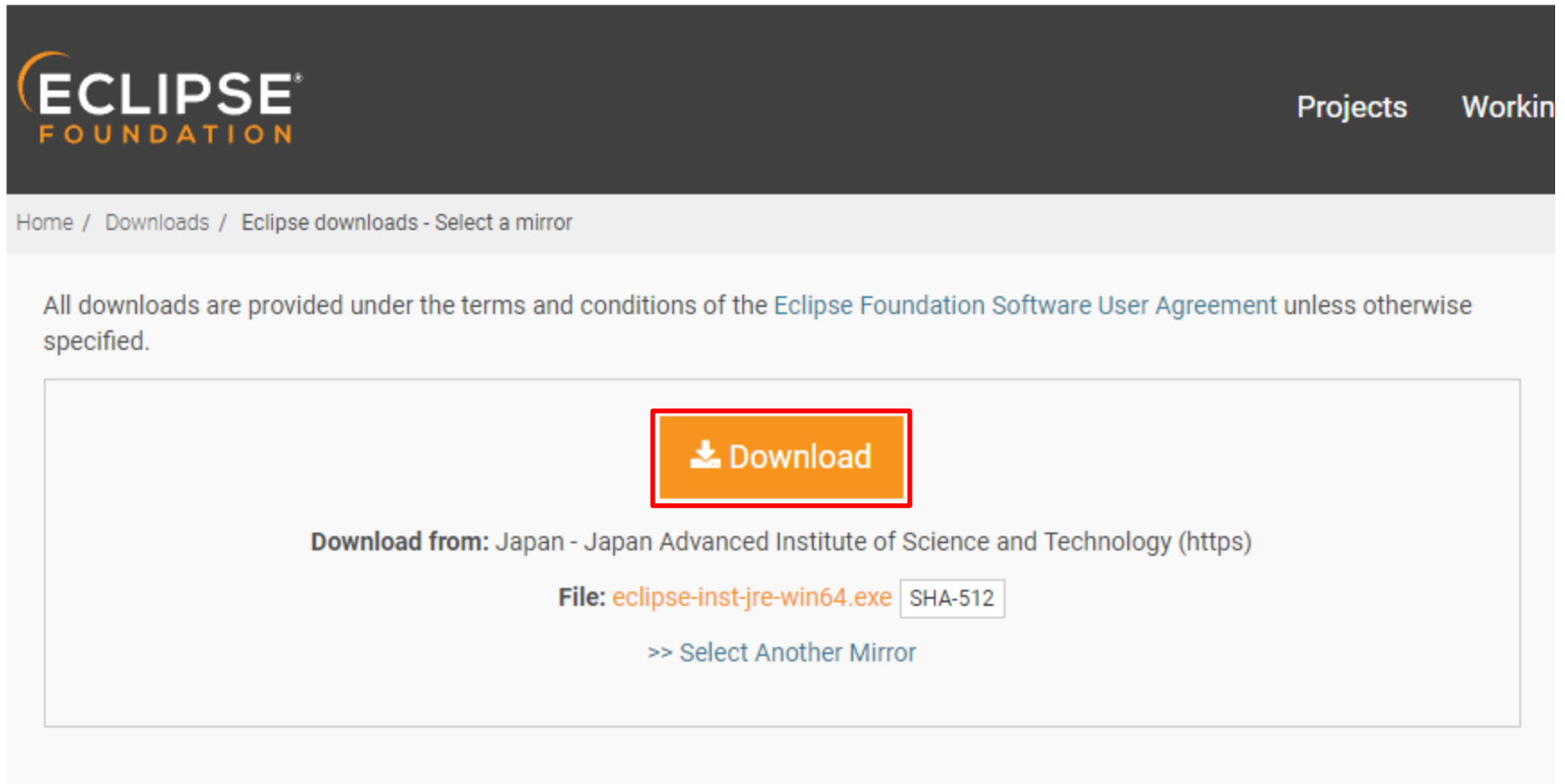
**Download Now**

Learn More

## 2. 자바 개발 환경

### ■ 이클립스 설치

- 다운로드 페이지에서 [Download]를 클릭




ECLIPSE<sup>®</sup>  
FOUNDATION

Projects Working

Home / Downloads / Eclipse downloads - Select a mirror

All downloads are provided under the terms and conditions of the [Eclipse Foundation Software User Agreement](#) unless otherwise specified.

 **Download**

Download from: Japan - Japan Advanced Institute of Science and Technology (<https>)

File: [eclipse-inst-jre-win64.exe](#) SHA-512

[>> Select Another Mirror](#)

## 2. 자바 개발 환경

### ■ 이클립스 설치

- 다음의 내용은 무시하면 된다.

**Thank you for your download!**

If the download doesn't start in a few seconds, please [click here](#) to start the download.

Eclipse technologies are 100% free and open source and used by millions of developers every day. Here are three easy ways you can contribute toward the future development of Eclipse projects and technologies.

### Donate to the Eclipse Community

Contributions from users like you help fund the operations of the Eclipse Foundation. All money donated to the Eclipse Foundation will be used to support the Eclipse Community.

\$5      \$10      \$35      \$50      \$100

35	USD	Donate
----	-----	--------

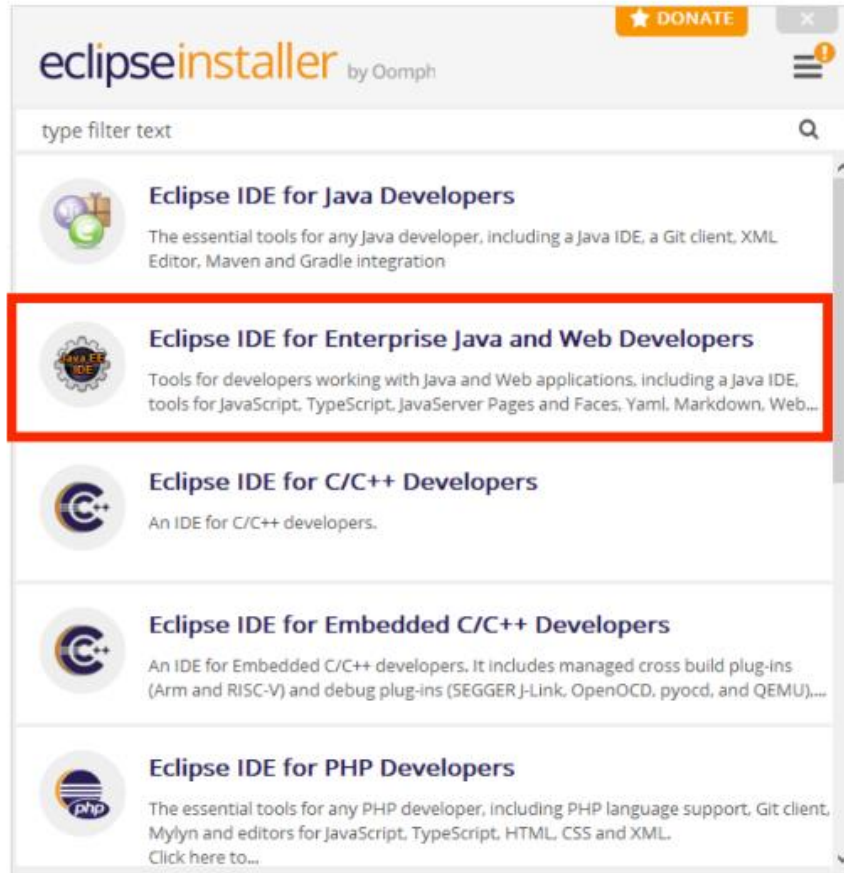
[Problems donating?](#) [Donation FAQ](#)



## 2. 자바 개발 환경

### ■ 이클립스 설치

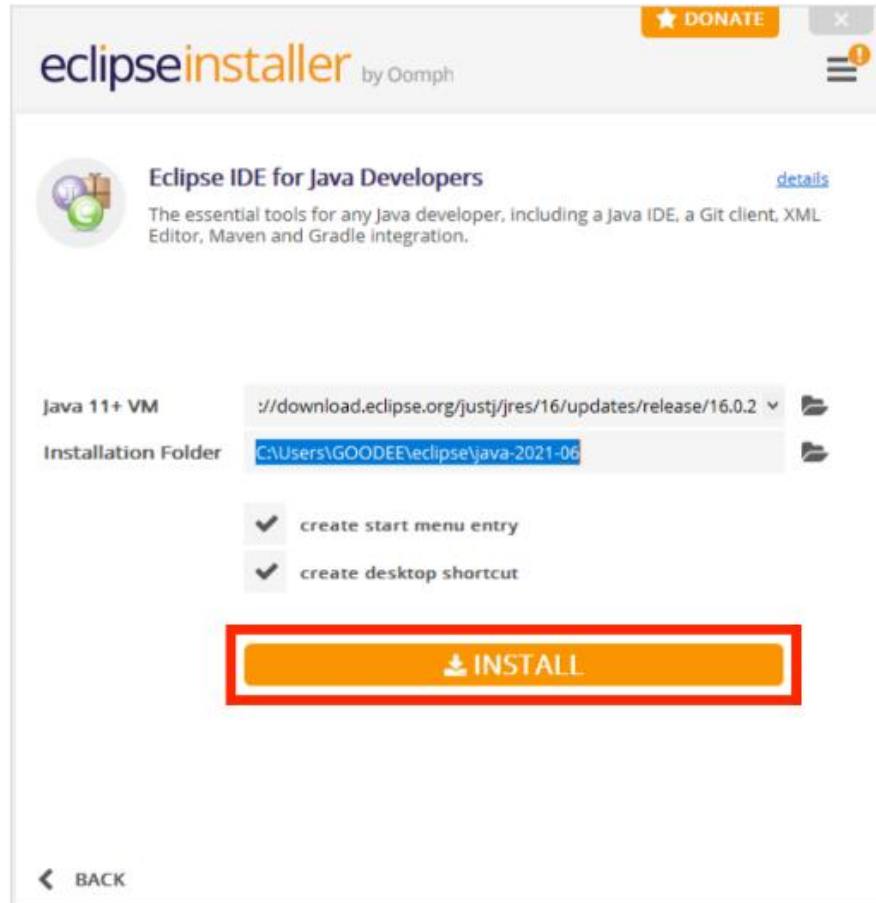
- 다운로드받은 파일을 실행하여 이클립스 패키지를 선택



## 2. 자바 개발 환경

### ■ 이클립스 설치

- [INSTALL] 버튼을 클릭



## 2. 자바 개발 환경

### ■ 이클립스 설치

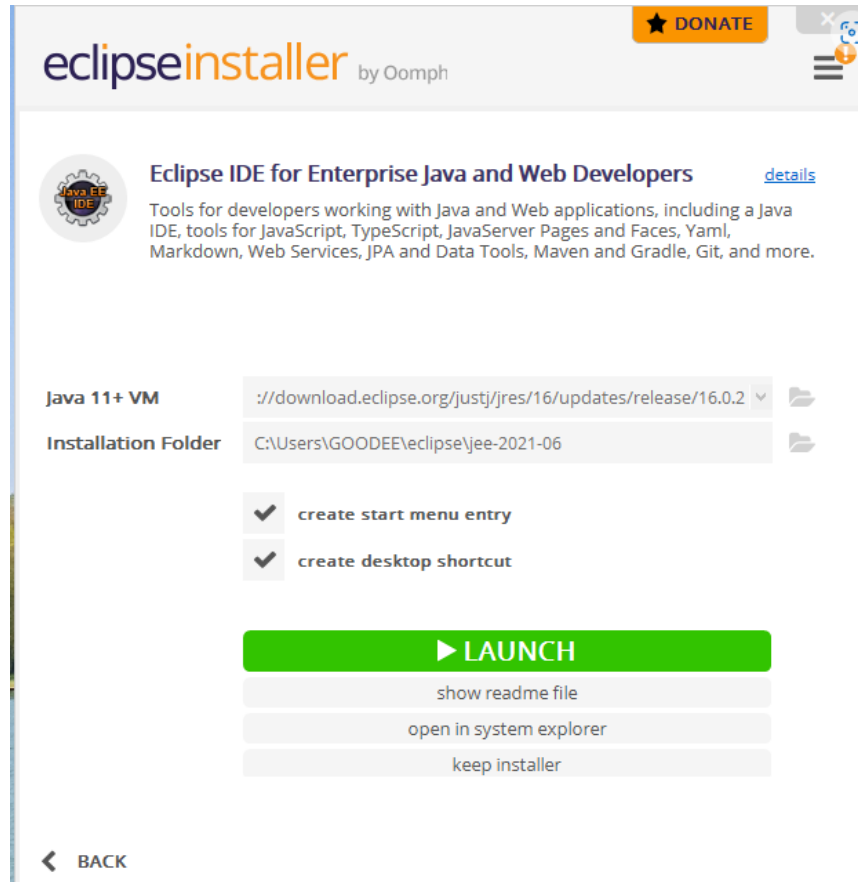
- 라이선스 동의함을 클릭



## 2. 자바 개발 환경

### ■ 이클립스 설치

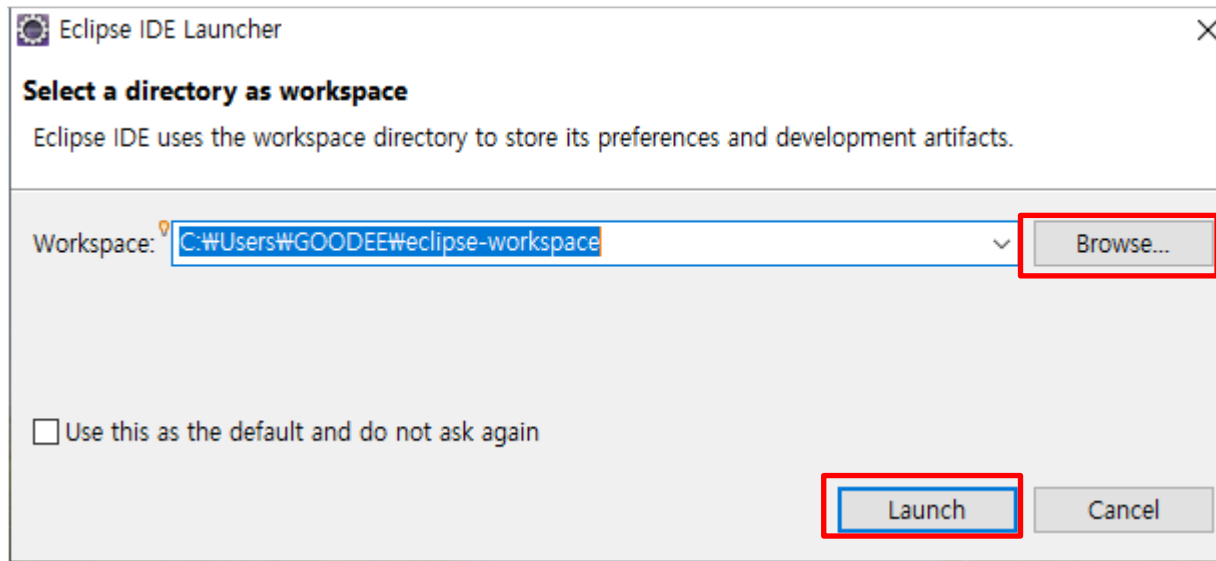
- 설치가 완료되면 [LAUNCH] 버튼을 클릭



## 2. 자바 개발 환경

### ■ 이클립스 설치

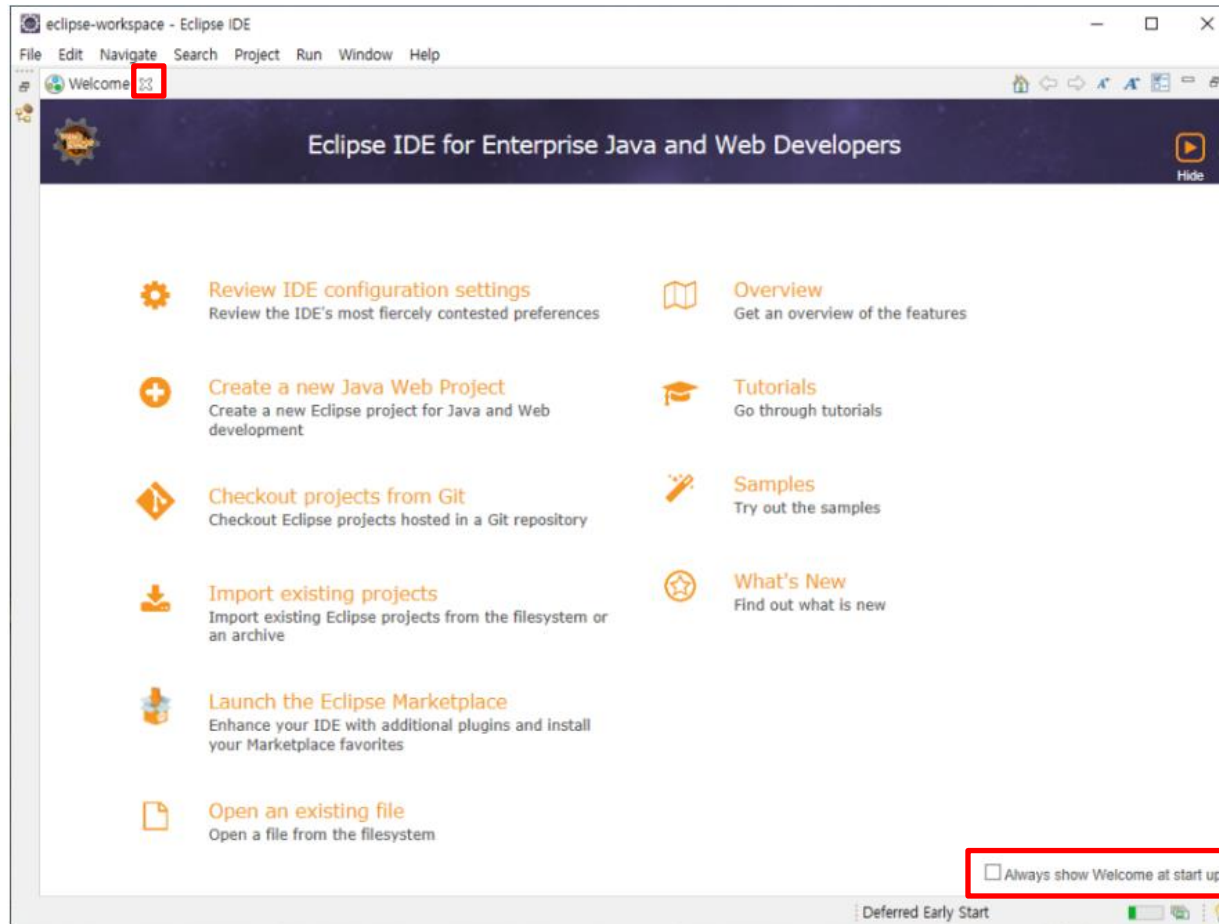
- Workspace 경로는 원하는 디렉토리를 선택



## 2. 자바 개발 환경

### ■ 이클립스 설치

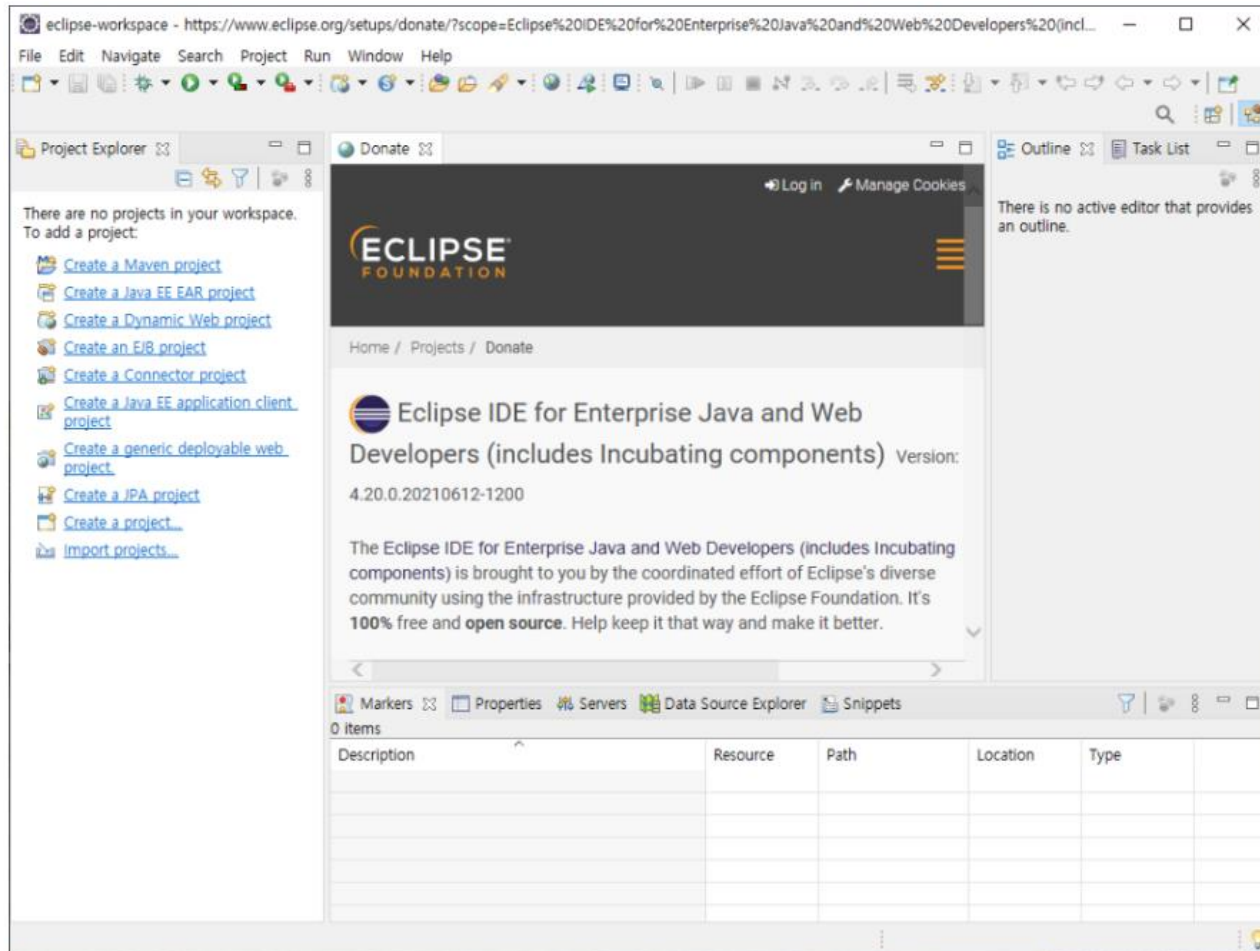
- Wellcome 창은 닫는다.
- 닫을 때, 우측 하단 체크박스를 체크하면, 이후 이클립스를 실행할 때 Wellcome 창이 뜨지 않는다.



## 2. 자바 개발 환경

### ■ 이클립스 설치

- 마찬가지로 Donate 창도 닫는다.



## 2. 자바 개발 환경

### ■ 이클립스 화면 구성

#### ■ 워크스페이스(workspace)

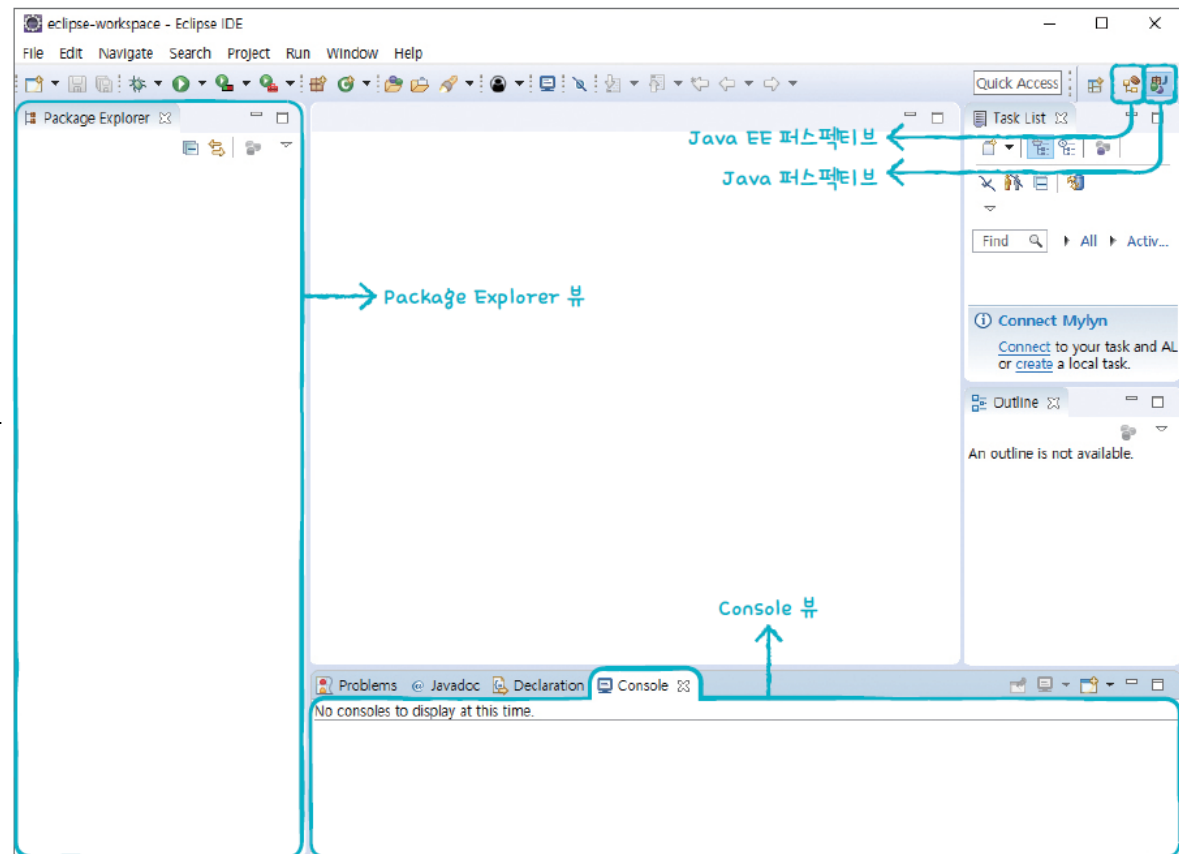
- 프로젝트 폴더가 저장
- 개발 환경 정보와 관련된 메타 데이터가 저장된 폴더(.metadata)가 저장

#### ■ 퍼스펙티브(perspective)

- 프로젝트를 개발할 때 유용하게
- 사용할 수 있는 뷰View들을
- 미리 묶어 이름을 붙여놓은 것

#### ■ 뷰(view)

- 이클립스 내부에서 사용되는 작은 창

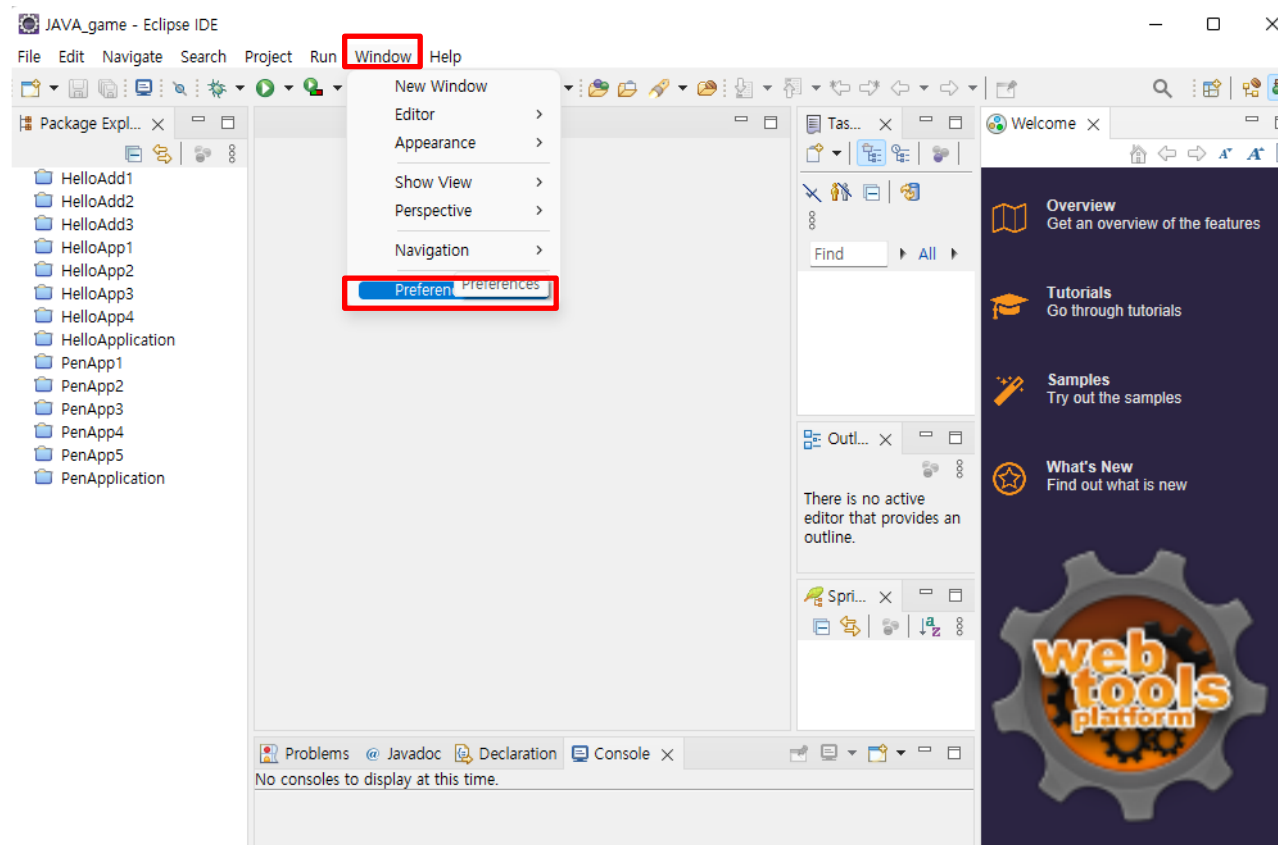




## 2. 자바 개발 환경

### ■ 이클립스 설정

- 이클립스에 기본적으로 설정돼 있는 글꼴, 문자 인코딩 방식, JRE 버전 등의 설정값을 변경하고자 할 때는 [Window — Preferences] 메뉴를 선택해 설정한다.

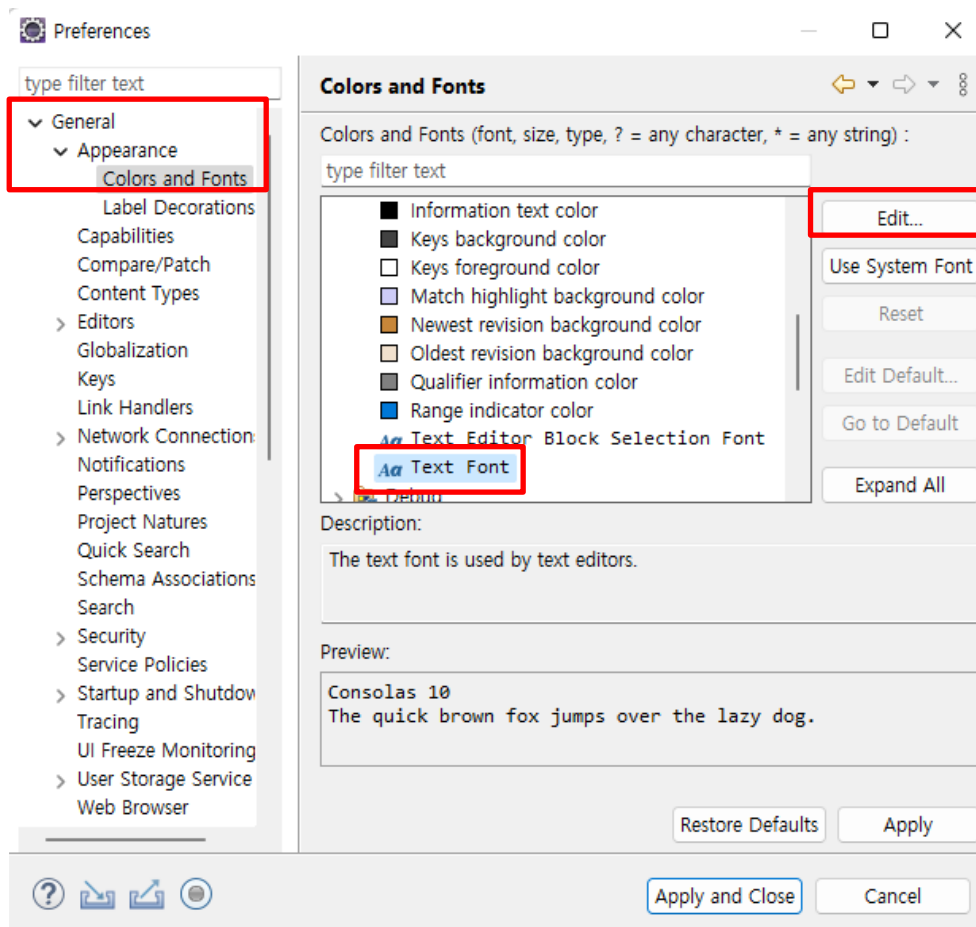


## 2. 자바 개발 환경

### ■ 이클립스 설정

#### ■ 글꼴 설정 바꾸기

- 글꼴 설정을 바꾸려 면 [Preferences] 창의 [General → Appearance → Colors and Fonts → Basic → Text Font]를 선택한다.



## 2. 자바 개발 환경

### ■ 이클립스 설정

#### ■ 텍스트 인코딩 변경하기

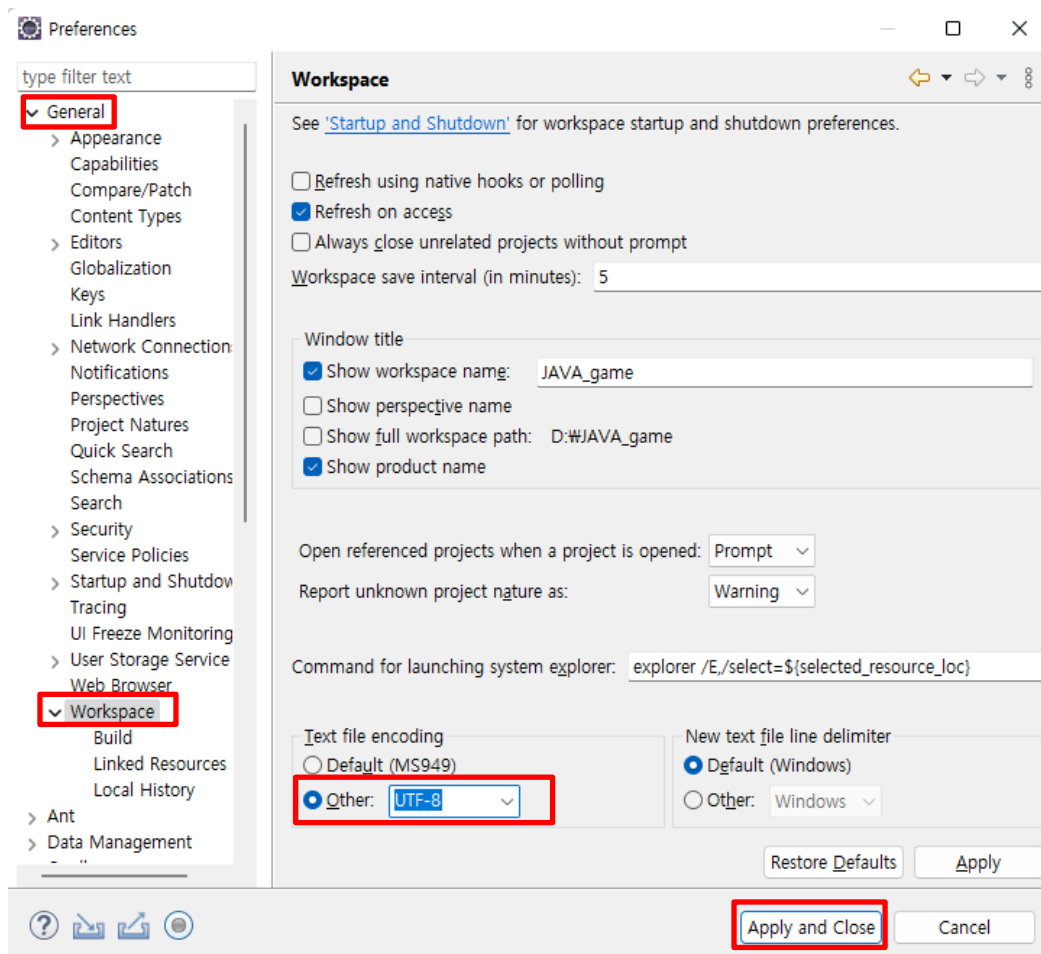
- 윈도우에서 설치한 이클립스의 기본 텍스트 인코딩 방식은 M2949
- 반면, 깃허브, 데이터베이스 등을 비롯한 여러 서버용 플랫폼은 대부분 UTF-8 방식의 텍스트 인코딩을 사용
- 따라서 이클립스에서 코드를 작성할 때 한글로 주석을 달거나 코드 자체에 한글을 사용한 프로그램을 서버에 업로드하면 한글이 모두 깨지는 문제가 발생
- 다음과 같이 [Preferences] 창에서 [General → Workspace → Text file encoding → Other: UTF-8]을 선택한 후 [Apply and Close]를 클릭

## 2. 자바 개발 환경

### ■ 이클립스 설정

#### ■ 텍스트 인코딩 변경하기

- 다음과 같이 [Preferences] 창에서 [General → Workspace → Text file encoding → Other: UTF-8]을 선택한 후 [Apply and Close]를 클릭



## 2. 자바 개발 환경

### ■ 이클립스 단축키

주요 평선키 (Function Key)	<b>F2</b>	패키지/클래스명 바꾸기(rename)
	<b>F3</b>	자바 API 클래스 및 함수는 class source 연결 필요 (클래스 및 함수 정의로 이동)
	<b>F4</b>	클래스 정의 확인(상속 관계 포함)
기능 관련 단축키	<b>Ctrl + Shift + O</b>	자동 임포트
	<b>Ctrl + /</b>	1줄 주석 설정 및 해제
	<b>Ctrl + Shift + /</b> <b>Ctrl + Shift + \</b>	블록 주석 설정 및 해제
	(클래스명 선택 후) <b>Ctrl + T</b>	상속 관계 표현(한 번 더 <b>Ctrl + T</b> 입력 시 슈퍼 타입 확인)
편집 관련 단축키	<b>Ctrl + Shift + F</b>	자동 정렬
	<b>Alt + +</b> <b>Alt + -</b>	블록을 선택했을 때: 블록 전체를 위아래로 이동 블록을 선택하지 않았을 때: 커서 위치 라인을 위아래로 이동
	(한글 ㄱ~ㅎ을 입력한 후) 한자	특수키 입력
편리하게 보기 위한 단축키	<b>Ctrl + +</b> <b>Ctrl + -</b>	폰트 확대 및 축소
	<b>Ctrl + Shift + -</b> <b>Ctrl + Shift + [</b>	화면 가로 및 세로 나누기
	<b>Alt + -</b> <b>Alt + -</b>	이전 자바 파일 히스토리로 이동(자바 파일)

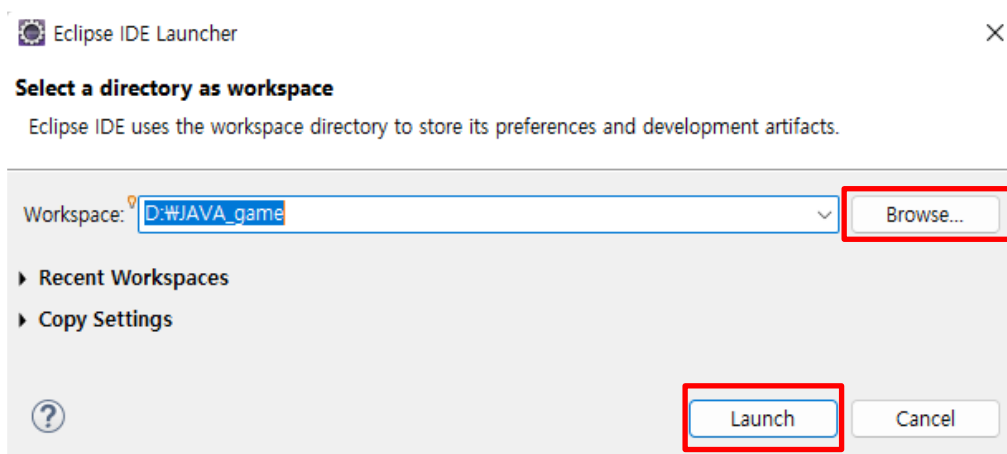
### 3. 자바 프로젝트

#### ■ 자바 프로젝트 생성 및 실행하기



#### ■ 자바 프로젝트 생성

- 이클립스를 실행하면 다음과 같이 워크스페이스를 설정하는 창이 나타난다

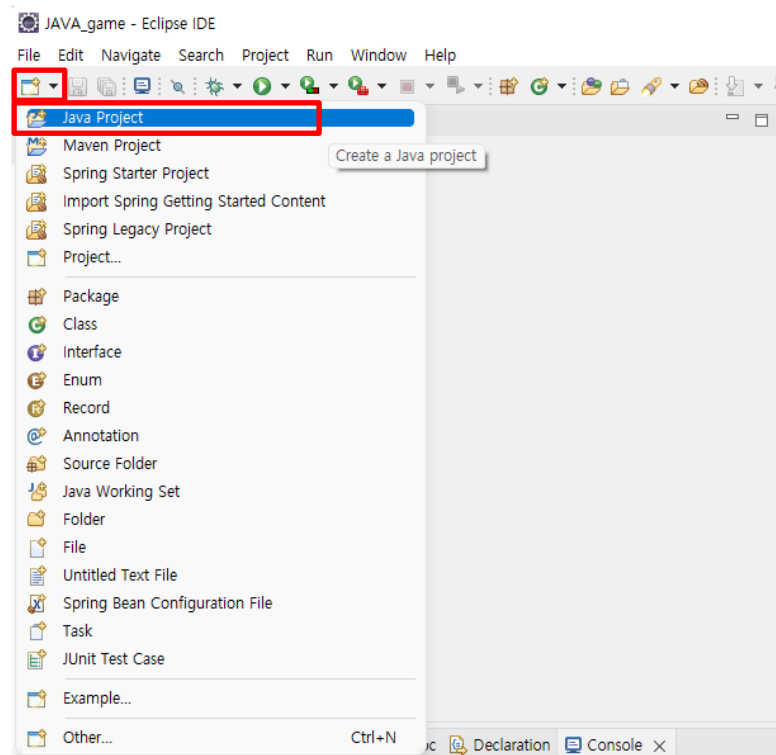
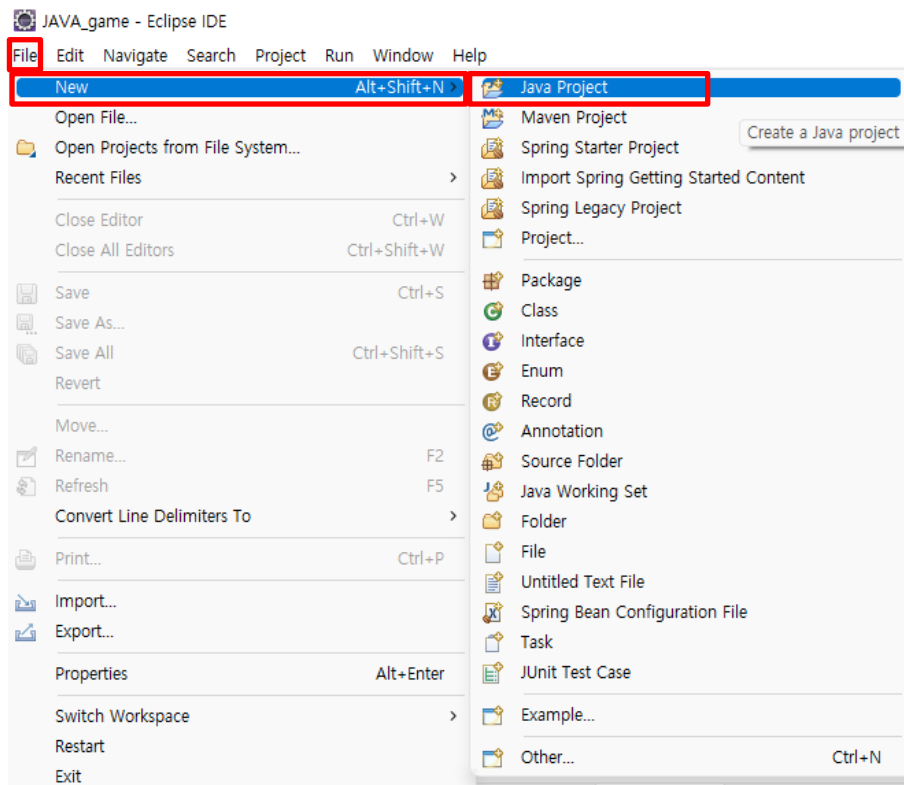


# 3. 자바 프로젝트

## ■ 자바 프로젝트 생성 및 실행하기

### ■ 자바 프로젝트 생성

- 첫 번째는 자바 프로젝트를 생성하는 단계로, 메뉴에서 [File → New → Java Project]를 생성하거나, 좌측 상단 아이콘 옆의 삼각형을 클릭해 [JavaProject]를 직접 선택할 수도 있다.



### 3. 자바 프로젝트

#### ■ 자바 프로젝트 생성 및 실행하기

##### ■ 자바 프로젝트 생성

- [프로젝트 설정] 창이 나타나면 프로젝트명을 입력한 후 JRE 버전을 확인하고 [Next] 버튼을 클릭

New Java Project

Create a Java Project

Create a Java project in the workspace or in an external location.

Project name: Test

☒ Use default location

Location: D:\JAVA\_game\Test Browse...

JRE

☒ Use an execution environment JRE: JavaSE-16

☐ Use a project specific JRE: jre

☐ Use default JRE 'jre' and workspace compiler preferences [Configure JREs...](#)

Project layout

☐ Use project folder as root for sources and class files

☒ Create separate folders for sources and class files [Configure default...](#)

Working sets

☐ Add project to working sets New...

Working sets: Select...

Module

☐ Create module-info.java file

< Back Next > Finish Cancel

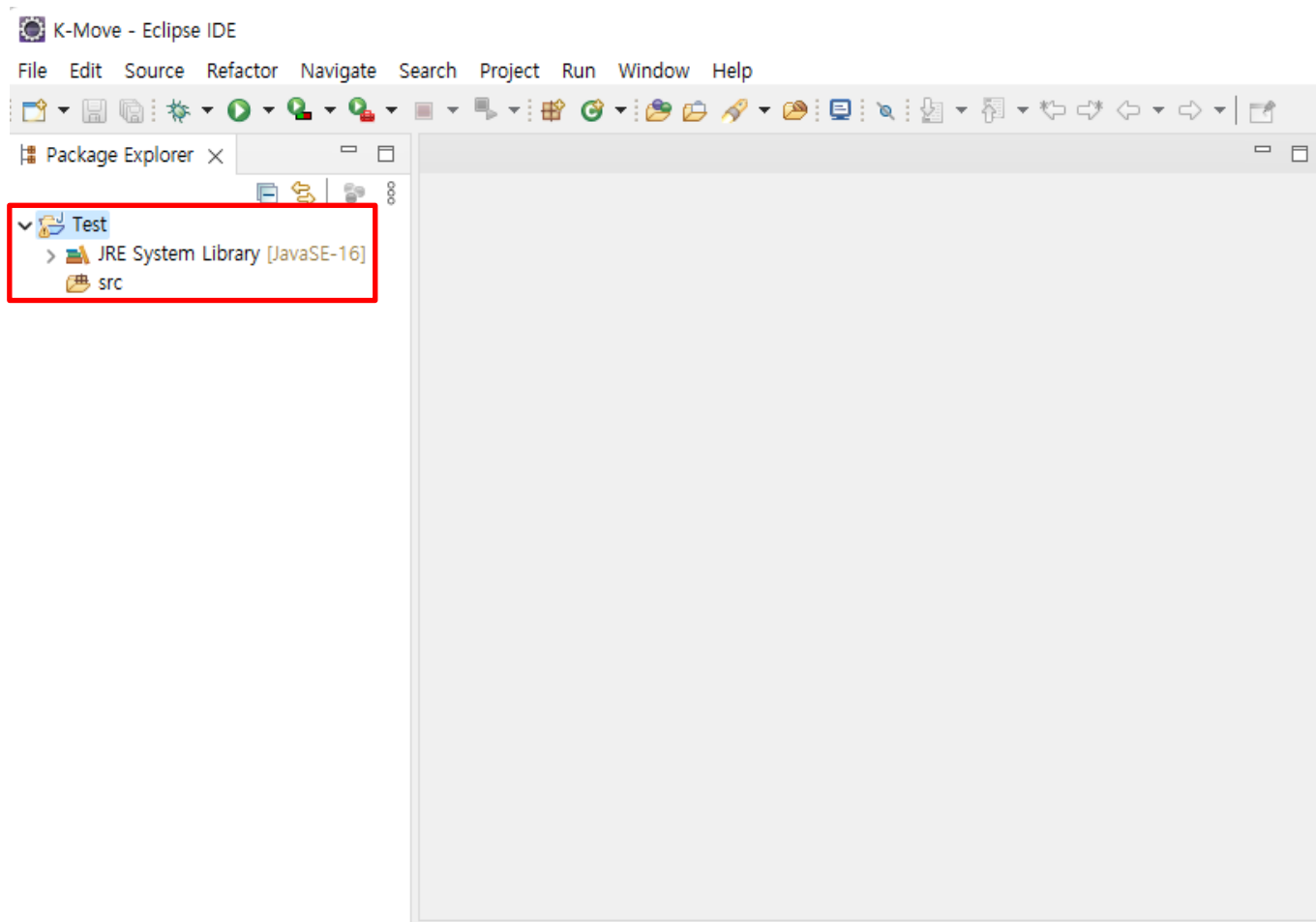


### 3. 자바 프로젝트

#### ■ 자바 프로젝트 생성 및 실행하기

##### ■ 자바 프로젝트 생성

- 이상의 과정을 거치면 이클립스를 시작할 때 설정했던 워크스페이스 안에 프로젝트 단위의 폴더가 생성

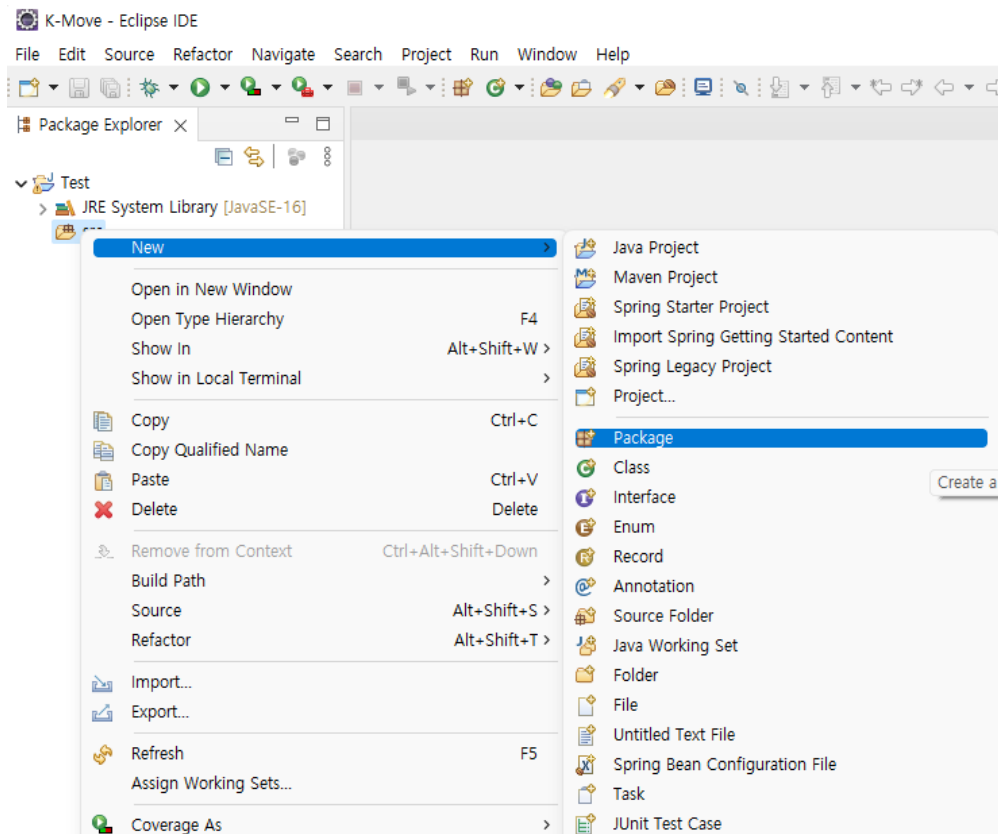


# 3. 자바 프로젝트

## ■ 자바 프로젝트 생성 및 실행하기

### ■ 패키지 생성

- 두 번째는 패키지를 생성하는 단계로, 소스 파일이 저장되는 src 폴더와 바이트 코드가 저장되는 bin 폴더 내에 하위 폴더를 생성.
- 프로젝트 폴더를 선택한 후 마우스 오른쪽 버튼으로 클릭하면 나타나는 팝업 창에서 [New → Package]를 순서대로 클릭.

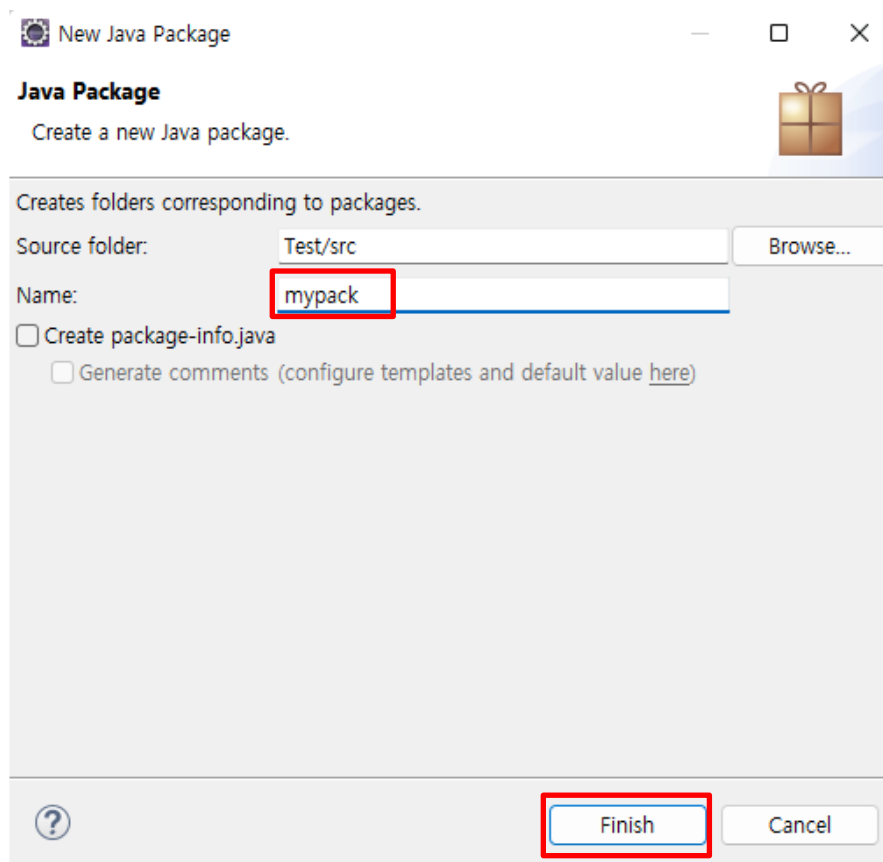


## 3. 자바 프로젝트

### ■ 자바 프로젝트 생성 및 실행하기

#### ■ 패키지 생성

- 패키지 생성 창이 나타나면 패키지명을 입력한 후 [Finish] 버튼을 클릭

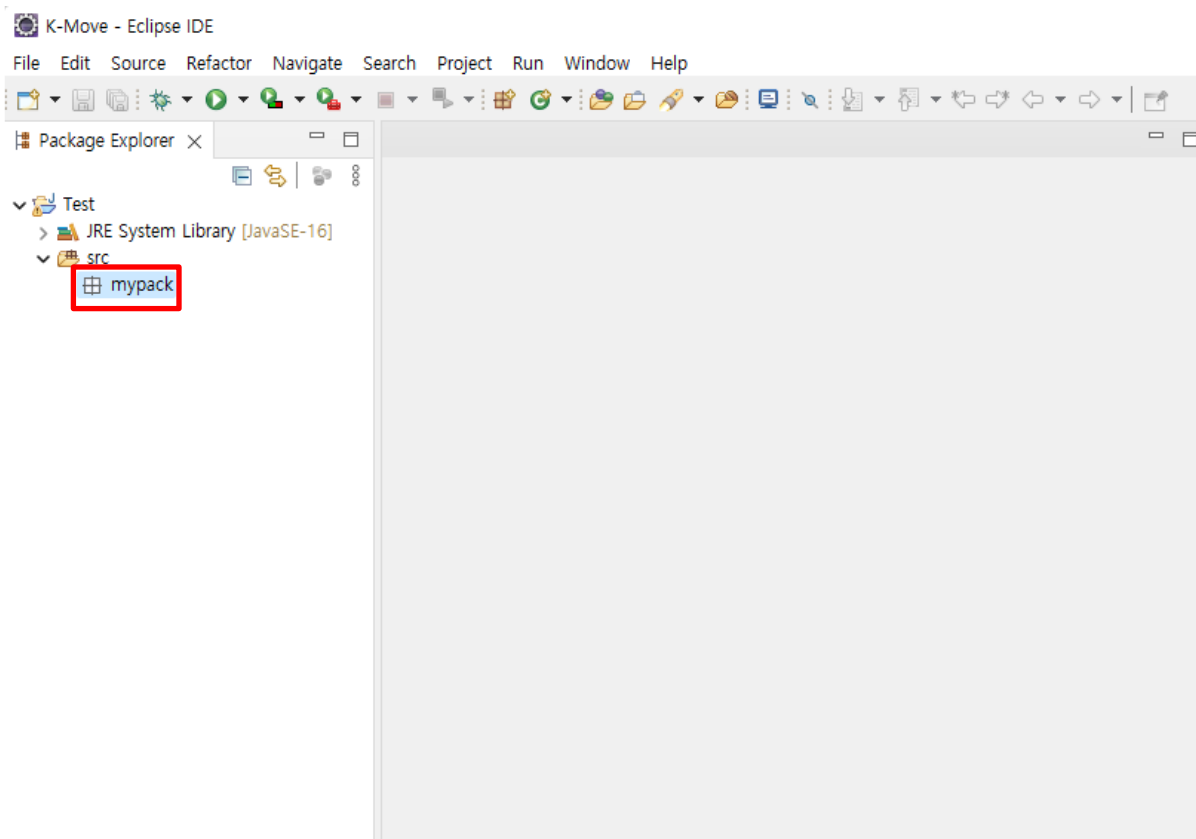


## 3. 자바 프로젝트

### ■ 자바 프로젝트 생성 및 실행하기

#### ■ 패키지 생성

- src 폴더 아래에 앞에서 입력한 패키지명으로 폴더가 생성된 것을 확인할 수 있다.

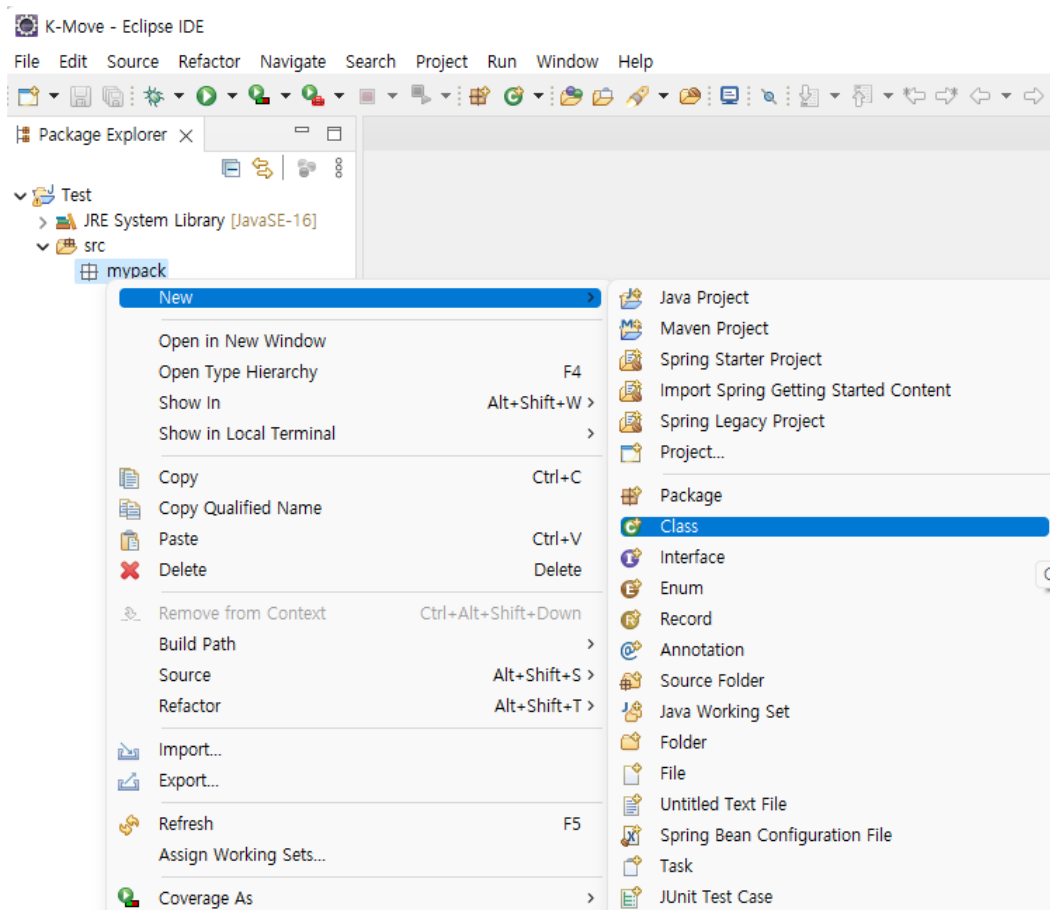


# 3. 자바 프로젝트

## ■ 자바 프로젝트 생성 및 실행하기

### ■ 자바 소스 파일 생성

- 세 번째는 자바 소스 파일을 생성하는 단계로, 프로젝트 폴더를 선택한 후 마우스 오른쪽 버튼 클릭하면 나타나는 팝업 창에서 [New → Class] 메뉴를 클릭



### 3. 자바 프로젝트

#### ■ 자바 프로젝트 생성 및 실행하기

##### ■ 자바 소스 파일 생성

- 클래스 생성 창이 나타나면 클래스명을 입력하고 [Finish]를 클릭

New Java Class

Java Class

Create a new Java class.

Source folder: Test/src Browse...

Package: mypack Browse...

☐ Enclosing type: Browse...

Name: Test

Modifiers: ☒ public ☐ package ☐ private ☐ protected  
☐ abstract ☐ final ☐ static

Superclass: java.lang.Object Browse...

Interfaces: Add... Remove

Which method stubs would you like to create?

☒ public static void main(String[] args)

☐ Constructors from superclass

☒ Inherited abstract methods

Do you want to add comments? (Configure templates and default value [here](#))

☐ Generate comments

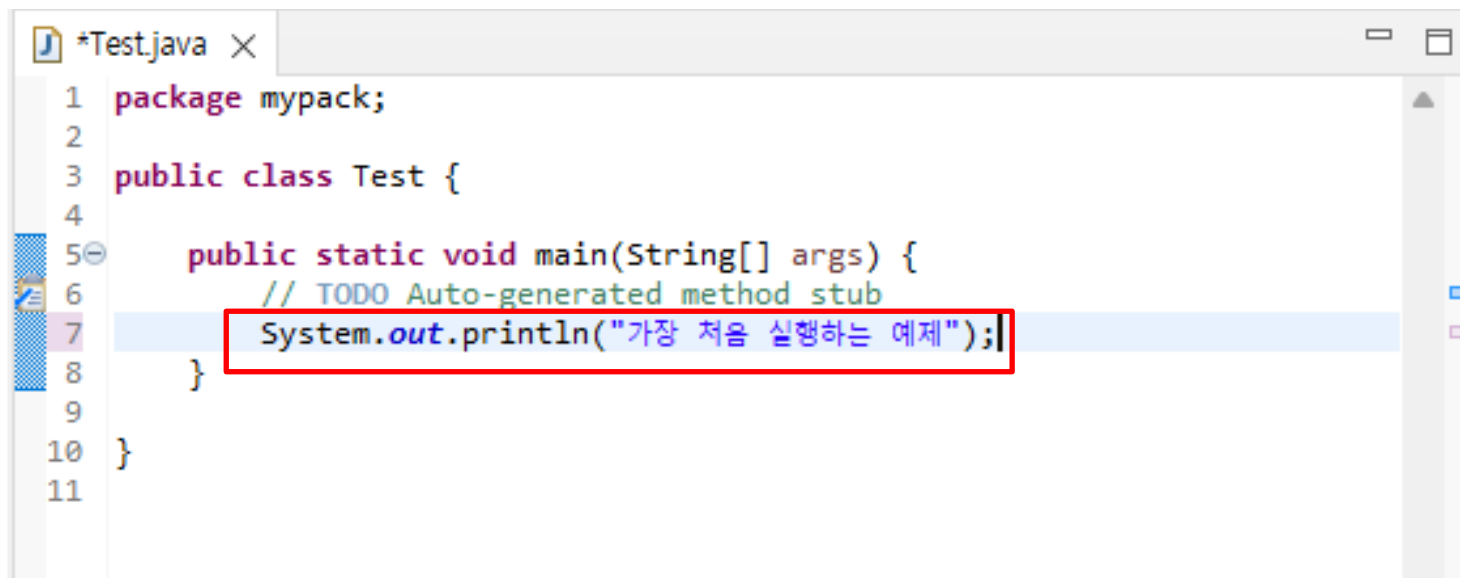
Finish Cancel

## 3. 자바 프로젝트

### ■ 자바 프로젝트 생성 및 실행하기

#### ■ 자바 소스 파일 생성

- 이후 실행 과정에서 간단한 출력을 할 수 있도록 다음과 같이 클래스 내부에 3줄의 코드를 작성



```
1 package mypack;
2
3 public class Test {
4
5     public static void main(String[] args) {
6         // TODO Auto-generated method stub
7         System.out.println("가장 처음 실행하는 예제");
8     }
9
10 }
11
```

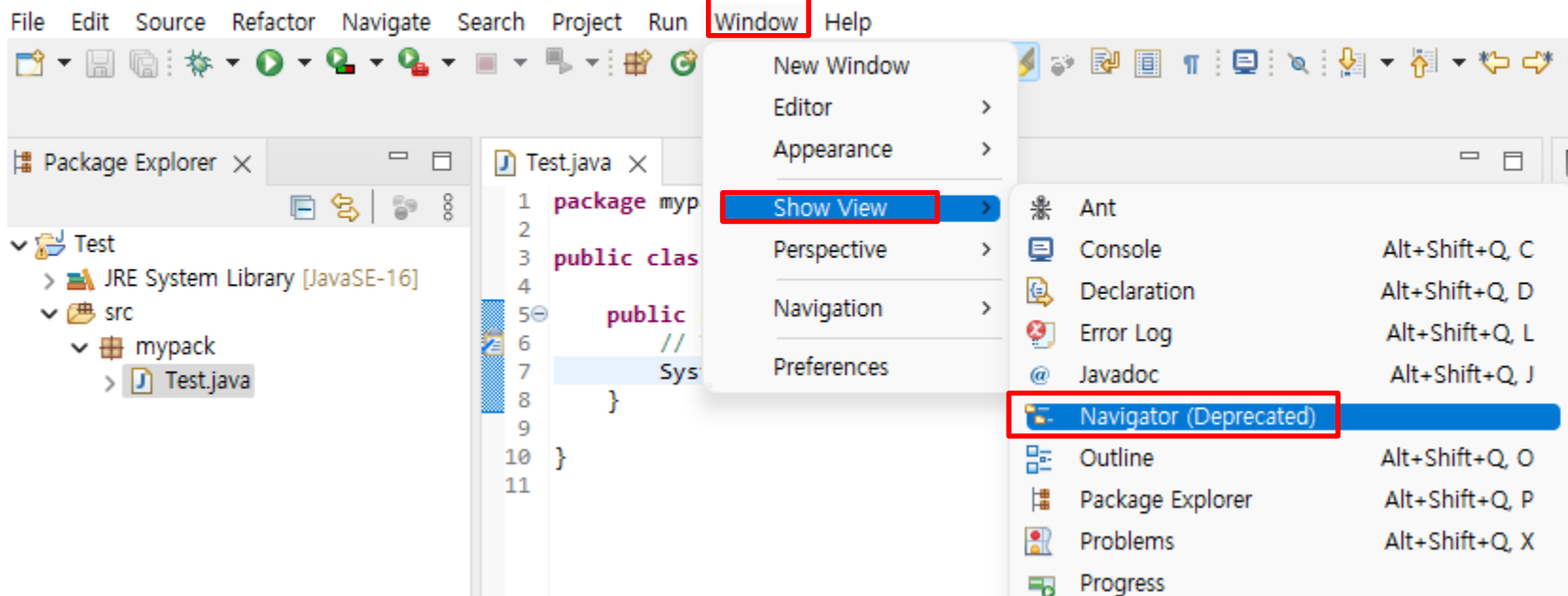
# 3. 자바 프로젝트

## ■ 자바 프로젝트 생성 및 실행하기

### ■ 컴파일 및 바이트 코드 생성

- 네 번째는 컴파일 및 바이트 코드를 생성하는 단계로, 자바 소스 파일을 컴파일하면 자바 가상 머신에서 실행할 수 있는 바이트 코드(.class)가 생성
- 컴파일은 명령 프롬프트상에서 javac 명령을 이용해 명시적으로 실행해야 하지만, 이클립스를 사용해 개발할 때는 소스 파일의 저장과 동시에 자동 컴파일이 수행되므로 컴파일을 하기 위해 별도의 작업을 할 필요가 없다
- 내비게이터 창을 이용하면 bin 폴더의 내용을 이클립스에서도 확인할 수 있다.
- 내비게이터 창을 추가하려면 [Window → Show View → Navigator] 메뉴를 선택한다.

K-Move - Test/src/mypack/Test.java - Eclipse IDE



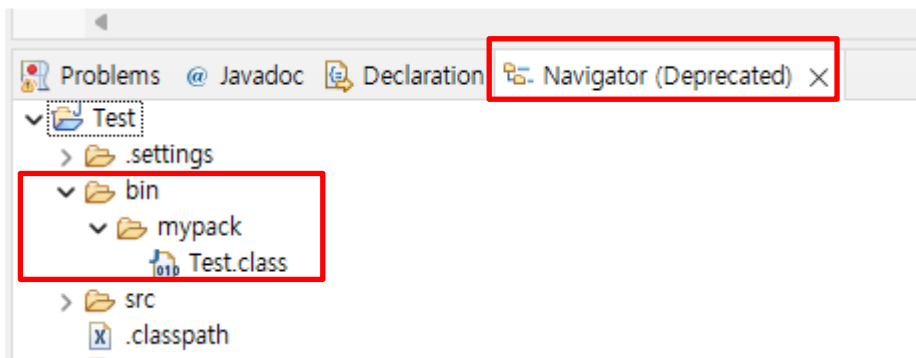


## 3. 자바 프로젝트

### ■ 자바 프로젝트 생성 및 실행하기

#### ■ 컴파일 및 바이트 코드 생성

- 이렇게 추가된 내비게이터 창에서 [프로젝트 폴더 → bin → 패키지 폴더]를 선택하면 윈도우 탐색기에서와 마찬가지로 컴파일된 자바 바이트 코드(.class) 파일을 확인할 수 있다.

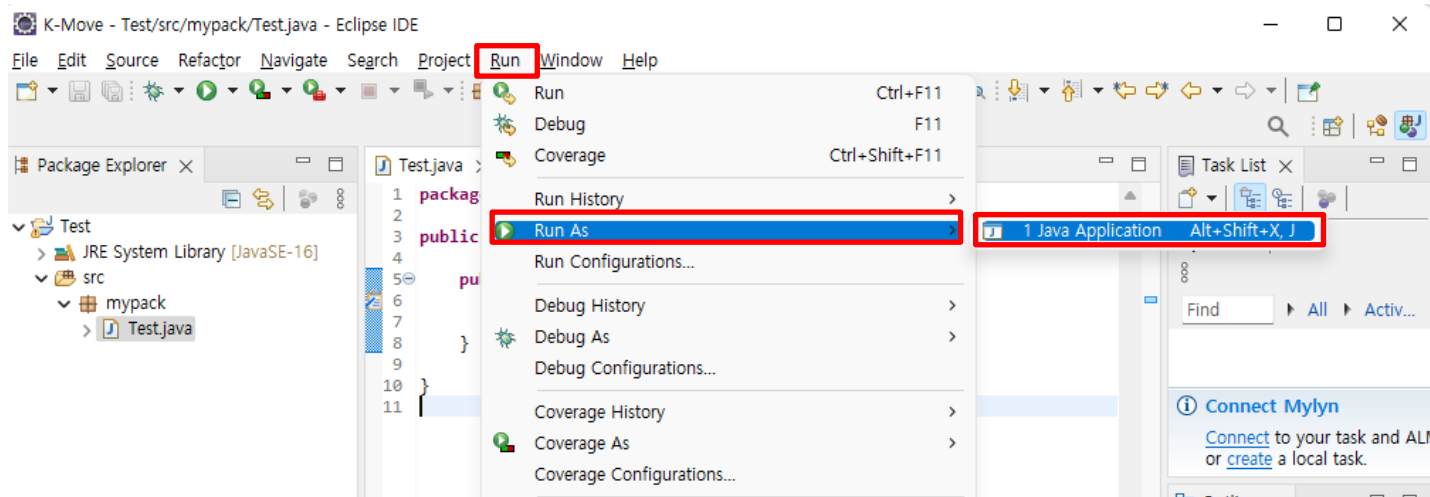


### 3. 자바 프로젝트

#### ■ 자바 프로젝트 생성 및 실행하기

##### ■ 실행하기

- 다섯 번째는 앞의 네 단계를 거쳐 생성된 바이트 코드 (.class)를 실행하는 단계다.
- [Run → Run As → Java Application] 메뉴를 선택하면 실행된다

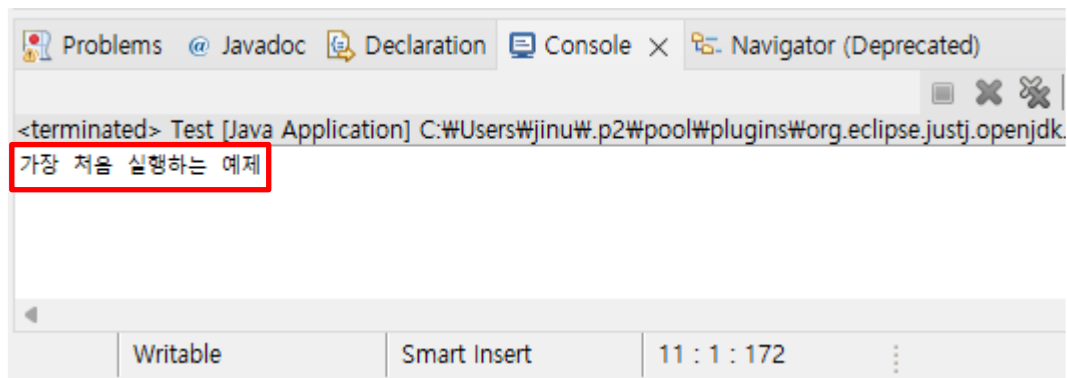


### 3. 자바 프로젝트

#### ■ 자바 프로젝트 생성 및 실행하기

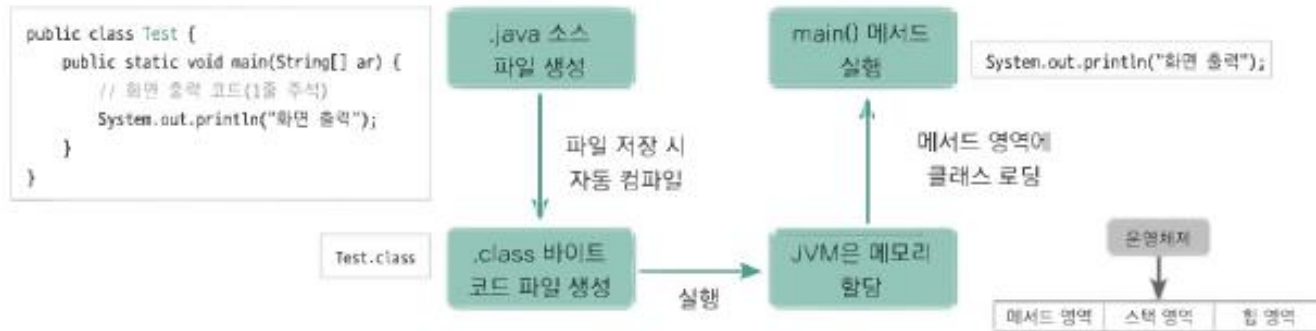
##### ■ 실행하기

- 만약 구문에 화면에 값을 출력하는 내용이 있으면 콘솔 창에서 출력 결과를 확인할 수 있다.
- 다음은 앞에서 Test 클래스에 추가한 실행 결과로 출력된 값이다.



## 4. 자바 프로그램의 기본 구조

### ■ 자바 소스 코드의 실행 과정



## 4. 자바 프로그램의 기본 구조

### ■ 소스 코드의 기본 구조 분석

```
/*  
처음 만든 클래스  
*/  
package exam01;  
  
public class Test {  
    public static void main(String[] args) {  
        // 화면 출력 코드(1줄 주석)  
        System.out.println("콘솔 화면 출력");  
    }  
}
```

#### ■ 주석

- 먼저 프로그램을 작성하는 과정에서 일종의 메모의 기능을 수행하는 주석(comment)은 협업을 할 때 프로그램의 가독성을 높이는 데 필요.
- 주석을 1줄만 쓸 때 '// 주석 내용', 주석을 2줄 이상 쓸 때 '/\* 주석 내용 \*/'과 같이 표기.
- 주석이 1줄일 때는 '/'가 표기된 위치부터 해당 줄의 끝까지가 주석으로 처리.
- 주석 처리된 문장은 컴파일할 때 제외되므로 실행 자체에는 아무런 영향을 미치지 않는다.

줄 수	형식	단축키
1줄	// 주석 내용	Ctrl + /
2줄 이상	/* 주석 내용 */	Ctrl + Shift + /

## 4. 자바 프로그램의 기본 구조

### ■ 소스 코드의 기본 구조 분석

#### ■ 패키지 선언부

- 패키지를 지정하면 주석을 제외한 첫 줄에 반드시 패키지의 선언이 와야 한다.
- 위 예제처럼 첫 줄에 `package exam01` 이 정의돼 있다면, 이는 소스 파일이 `exam01` 패키지 안에 위치하고 있다는 의미다.
- 앞에서 언급한 것처럼 패키지를 지정하지 않았을 때, 즉 디폴트 패키지를 사용할 때는 패키지 선언이 생략된다.

#### ■ 클래스 선언부

```
public class Test {  
  
}  
  
}
```

클래스 이름

클래스라는 것을 알려 주는 자바 키워드

- 먼저 `public`은 이 클래스를 다른 패키지에서도 사용할 수 있다는 의미를 지닌 접근 지정자
- 1 개의 소스 파일에는 여러 개의 클래스가 존재할 수 있는데, 몇 개의 클래스가 존재하든 최대 1 개의 클래스만 `public`을 포함할 수 있다
- `public` 클래스 이름은 반드시 소스파일명과 일치해야 한다

## 4. 자바 프로그램의 기본 구조

### ■ 소스 코드의 기본 구조 분석

#### ■ main() 메서드

- 메서드의 구조는 '리턴 타입 메서드명(...) {}'의 형태를 띈다.
- 메서드 원형 앞에 위치한 public은 접근 지정자, static은 정적 메서드를 나타내는 키워드이다.
- 바이트 코드(.class)가 메서드 영역에 로딩되면 자바 가상 머신은 main() 메서드부터 찾는다.
- 따라서 실행 이후 가장 먼저 실행되는 메서드가 main() 메서드라는 것을 기억하자.

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

```
}
```

리턴 타입

메서드 이름

## 4. 자바 프로그램의 기본 구조

### ■ 소스 코드의 기본 구조 분석

#### ■ 실습. 생성 프로젝트의 기본 구조

JavaBasicStructure

```
1 package sec01_basicsyntax.EX01_JavaBasicStructure;
2 /* Ctrl+Shift+'/'
3  * 처음 만든 클래스
4  * (여러 줄 주석)
5  */
6
7 public class JavaBasicStructure {
8     public static void main(String[] args) {
9         // 1줄 주석: 화면 출력 코드 Ctrl+'/'
10        System.out.println("콘솔 화면에 출력");
11    }
12 }
```



## 4. 자바 프로그램의 기본 구조

### ■ 소스 파일 컴파일과 바이트 코드 생성

- 1개의 소스 파일에는 여러 개의 클래스 파일이 포함될 수 있다.
- 그렇다면 여기서 2가지 의문이 생길 수 있다.
- 소스 파일명은 반드시 클래스명과 같아야 한다고 했다.
- 그런데 하나의 자바 소스 파일에 클래스가 여러 개면 그 많은 클래스 중 어떤 클래스와 이름을 일치시켜야 할까?
- 정답은 `public`이 붙은 클래스다.
- 이것이 바로 하나의 소스 파일에 최대 1개의 `public` 클래스만 존재해야 하는 이유다.
- 그럼 이제 두 번째 의문을 풀어 보자.
- 다음 소스 파일을 컴파일하면 바이트 코드(`.class`)가 몇 개 생길까?
- 즉, 소스 파일(`.java`)당 바이트 코드(`.class`)가 1개일까, 정의된 클래스(`class`)당 바이트 코드(`.class`)가 1개일까?
- 바이트 코드(`.class`)의 확장명에서 힌트를 얻을 수 있는 것처럼 자바의 바이트 코드(`.class`)는 클래스당 하나씩 생성된다.
- 하나의 소스 파일에 100개의 클래스가 정의돼 있으면 100개의 바이트 코드(`.class`)가 생기는 것이다.

## 4. 자바 프로그램의 기본 구조

### ■ 소스 파일 컴파일과 바이트 코드 생성

- 다음 예를 살펴보자.

```
public class A {  
    // ...  
}  
class B {  
    // ...  
}  
class C {  
    class D {  
        // ...  
    }  
}
```

1 개의 소스 파일(.java)에는 최대 1개의 public class만 선언할 수 있음.

- 외부에 있는 클래스는 '클래스명.class'와 같이 생성된다(class A{} → A.class, class B{} → B.class, class C{} → C.class).
- 반면 클래스 내부에 포함된 이너 클래스는 반드시 자신을 감싸고 있는 클래스부터 표현해야 한다.
- 따라서 생성되는 바이트 코드도 '아우터 클래스\$이너 클래스.class'와 같이 생성된다(C\$D.class).

## 4. 자바 프로그램의 기본 구조

### ■ 소스 파일 컴파일과 바이트 코드 생성

- 실습. 컴파일 후 생성되는 바이트 코드(.class)

ByteCodeFiles.java

```
1 package sec01_basicsyntax.EX02_ByteCodeFiles;
2
3 class A{
4 }
5 class B{
6 }
7 class C{
8     class D{
9     }
10 }
11 public class ByteCodeFiles {
12     public static void main(String[] args) {
13     }
14 }
```

## 4. 자바 프로그램의 기본 구조

### ■ 콘솔 출력 메서드와 문자열 출력

#### ■ 문자열 표현하기

- 먼저 문자열을 알아보자.
- 문자열은 String 자료형으로 저장되며, 값의 표현은 "안녕", "반가워" 등과 같이 반드시 큰따옴표("") 안에 표기해야 한다.
- 문자열과 문자열을 더하거나 문자열과 기본 자료형을 더하면 다시 문자열이 되며, 연산 결과는 문자열을 연결한 형태가 된다.
- 예를 들어 "안녕" + "반가워" = "안녕반가워", "안녕" + 3 = "안녕3"이 되는 것이다.
- 이 단계에서 기본 자료형은 단순히 숫자(정수 또는 실수)로만 생각하자.

#### ■ 줄 바꾸면서 출력하기 - `System.out.println()`

- 이제 콘솔에 출력하는 방법을 알아보자.
- 콘솔에 값을 출력하기 위해서는 `System.out.println()`, `System.out.print()` 또는 `System.out.printf()` 메서드를 활용해야 한다.
- 먼저 `System.out.println()` 메서드는 소괄호 안의 내용을 출력하고 줄을 바꾼다.
- 그래서 line을 축약한 `ln`이 `print` 뒤에 들어갔다.

## 4. 자바 프로그램의 기본 구조

### ■ 콘솔 출력 메서드와 문자열 출력

#### ■ 변수에 넣어 출력하기

- 소괄호 안에 값을 직접 입력해도 되지만, 변수라는 저장 공간에 값을 담은 후 변수를 넘겨 줘도 동일한 결과를 얻을 수 있다.
- 다음 예를 살펴보자.

```
int a = 3;  
String b = "화면";  
System.out.println(a);  
System.out.println(b);  
System.out.println(b + "출력");  
System.out.println(a + b + "출력");
```

## 4. 자바 프로그램의 기본 구조

### ■ 콘솔 출력 메서드와 문자열 출력

#### ■ 1 줄로 출력하기 - `System.out.print()`

- 이제 `System.out.print()` 메서드를 알아보자.
- `Print()` 메서드는 출력 이후 개행하지 않는다는 점을 제외하면 `println()` 메서드와 동일하다.
- 즉, 모든 출력을 연속적으로 1줄로 출력한다.
- `System.out.print("화면")`을 실행하면 콘솔에는 "화면"이 출력된다.
- 이때 커서는 "화면" 다음에 위치하며, 이후 출력을 대기하고 있다.
- 따라서 이어 출력되는 "출력"은 화면 옆에 나란히 출력된다.

```
System.out.print(" 화면");  
System.out.print(" 출력");  
System.out.print(3);
```

- `\n`을 출력하면 개행, 즉 줄바꿈을 실행된다.
- 따라서 다음 두 코드는 동일한 기능을 수행한다.

```
System.out.println(" 출력");  
System.out.print(" 출력\n");
```

## 4. 자바 프로그램의 기본 구조

### ■ 콘솔 출력 메서드와 문자열 출력

#### ■ 형식대로 출력하기 - System.out.printf()

- 이제 마지막으로 System.out.printf() 메서드다.
- 이 메서드는 C 언어의 printf()와 동일한 동작을 수행하며, 기본 형식은 System.out.printf("출력 포맷", 인자, 인자, ...)의 형태를 띈다.
- 말 그대로 출력 포맷을 지정하는 메서드로, 큰따옴표("") 안에 출력하고자 하는 형식을 지정한다.
- 출력 포맷 내에 %로 시작하는 위치는 인자로 값이 넘어 오는 위치이며, 출력 타입은 % 다음에 나오는 문자에 따라 결정된다.

```
System.out.printf("%d\n", 30);           // 30(10진수)
System.out.printf("%o\n", 30);           // 36(8진수)
System.out.printf("%x\n", 30);           // 1e(16진수)
System.out.printf("%s\n", "출력");       // 출력
System.out.printf(" %f\n", 5.8);         // 5.800000
System.out.printf("%4.2f\n", 5.8);       // 5.80
System.out.printf("%d와 %4.2f\n", 4, 5.8); // 4와 5.80
```

## 4. 자바 프로그램의 기본 구조

### ■ 콘솔 출력 메서드와 문자열 출력

#### ■ 실습. 기본적인 콘솔 출력 방법

ConsoleOutput.java

```
1 package sec01_basicsyntax.EX03_ConsoleOutput;
2
3 public class ConsoleOutput {
4     public static void main(String[] args) {
5         // 1. System.out.println()
6         System.out.println("안녕하세요");
7         System.out.println("안녕" + "하세요");
8         System.out.println(2 + 4);
9         System.out.println(4.6);
10        System.out.println("문자" + 1);
11        System.out.println("문자" + 1 + 2);
12        System.out.println(1 + 2 + "문자");
13        System.out.println();
14        int a = 5;
```



## 4. 자바 프로그램의 기본 구조

### ■ 콘솔 출력 메서드와 문자열 출력

#### ■ 실습. 기본적인 콘솔 출력 방법

ConsoleOutput.java

```
15      String b = "하세요";
16      System.out.println(a);
17      System.out.println(b);
18      System.out.println("안녕" + b);
19      System.out.println(a + "안녕" + b);
20      System.out.println();
21      // 2. System.out.print()
22      System.out.print("반갑");
23      System.out.print("습니다");
24      System.out.print("7");
25      System.out.print("\n");
26      System.out.print("\n");
27      // 3. System.out.printf()
28      System.out.printf("%d\n", 10);
```

## 4. 자바 프로그램의 기본 구조

### ■ 콘솔 출력 메서드와 문자열 출력

#### ▣ 실습. 기본적인 콘솔 출력 방법

ConsoleOutput.java

```
29      System.out.printf("%o\n", 10);
30      System.out.printf("%x\n", 10);
31      System.out.printf("%s\n", "문자열 출력");
32      System.out.printf("%f\n", 3.2582);
33      System.out.printf("%4.2f\n", 3.2582);
34      System.out.printf("%d와 %4.2f\n", 10, 3.2582);
35  }
36 }
```



**Thank You**

---