



### ● 자바 응용 프로그램

- C 프로그램과 같이 일반적인 응용 프로그램을 의미

### ● 자바 애플릿

- 웹 검색기상에서 작동하는 프로그램

### ● 자바 서블릿(Servlet)

- 웹 환경에서 실행되는 자바 프로그램

### ● JSP(Java Server Page)

- HTML 속에 자바 코드를 삽입하여 사용하는 형태

### ● 자바 빈스(Beans)

- 자바로 작성한 프로그램들을 부품처럼 사용하여 프로그래밍하는 방법

### ● 안드로이드 애플리케이션

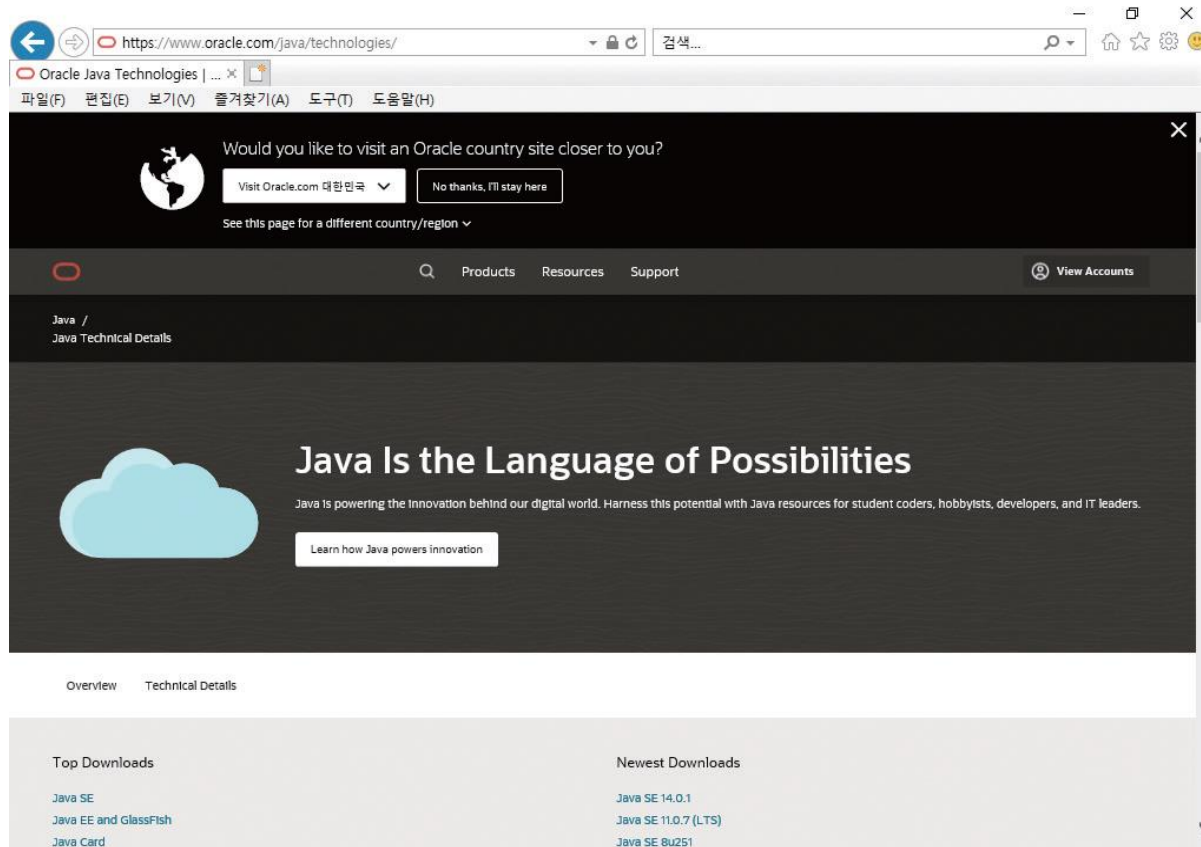
- 스마트폰에서 실행되는 대부분의 응용 프로그램(앱)은 자바로 개발
- 자바 앱은 스마트폰에 최적화된 달빅Dalvik 가상 머신에서 실행



- 자바 프로그램을 배우기 위해서는 선(Sun)사에서 제공되는 개발 환경과 개발 환경을 기반으로 프로그램을 작성할 수 있게 해주는 이클립스(Eclipse) 소프트웨어의 사용이 필수적
- 두 개의 소프트웨어는 모두 인터넷에서 무료로 공개되는 소프트웨어

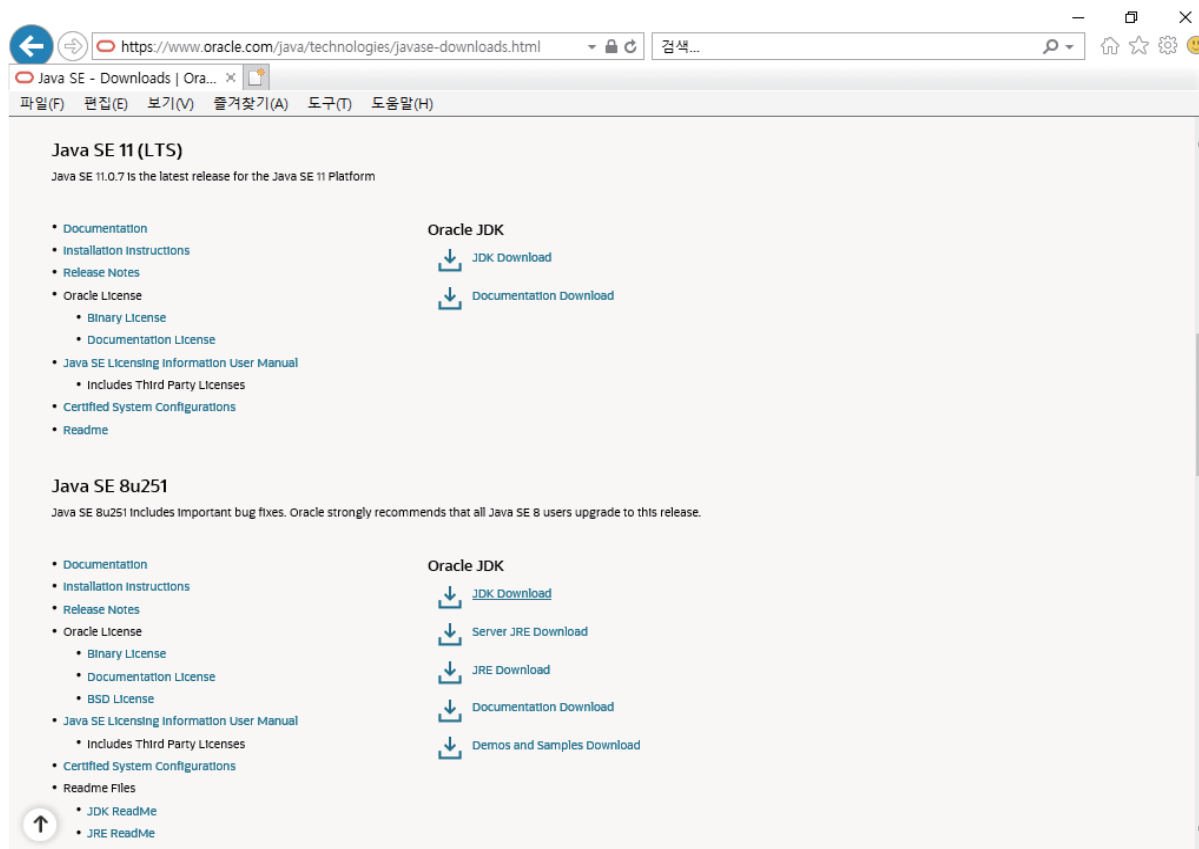
#### ● Step 1 : 선사의 홈페이지에서 JDK를 다운받는다.

- 오라클사 : <https://www.oracle.com/java/technologies/>
- 홈페이지 왼쪽 "Top Downloads"에서 "Java SE"를 선택한 다음 자바8 버전을 다운



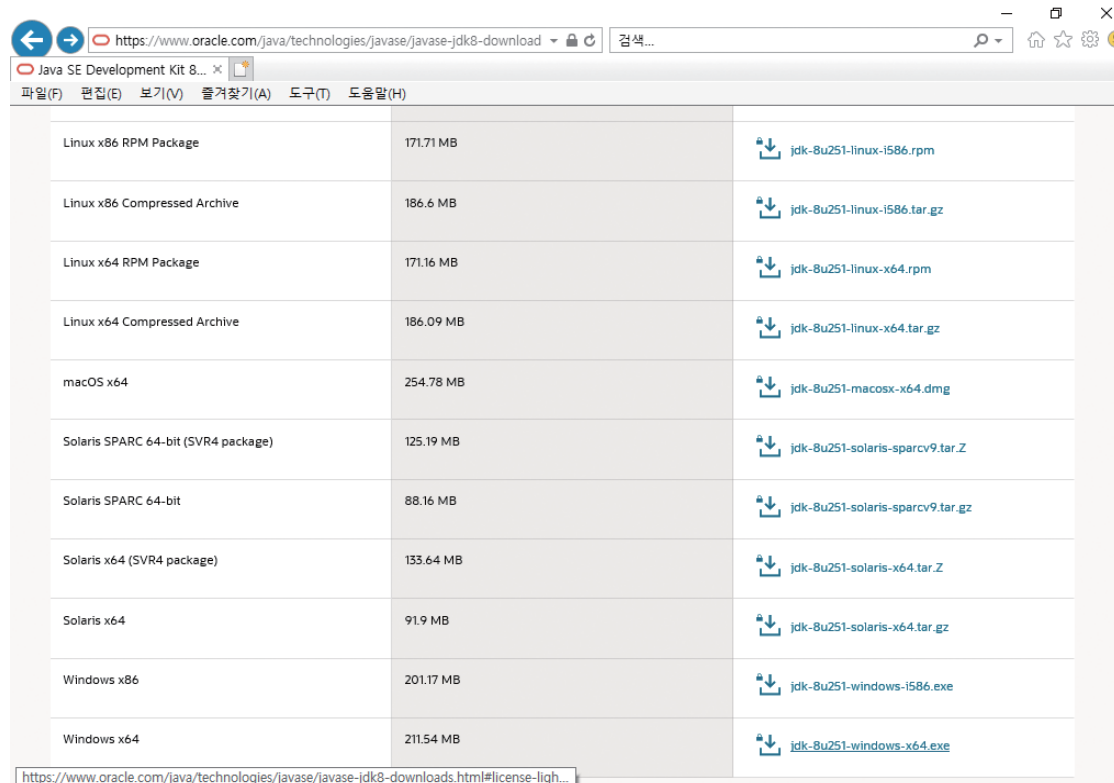
#### ● Step 1 : 선사의 홈페이지에서 JDK를 다운받는다.

- 자바8 버전을 찾아 "JDK Download"를 선택



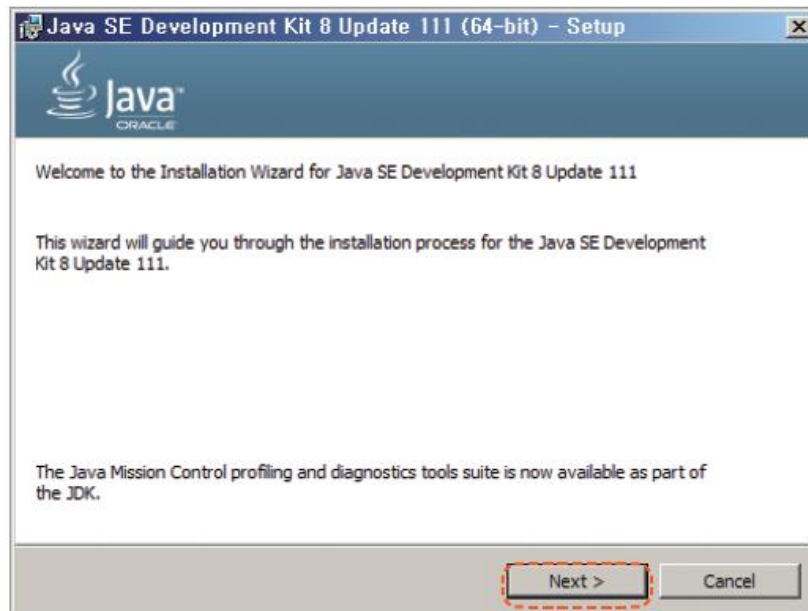
#### ● Step 1 : 선사의 홈페이지에서 JDK를 다운받는다.

- 다음 화면에서 Platform을 자신의 컴퓨터에 적합한 운영체제를 선택
- 다음 화면의 아랫부분에서 제공되는 파일(jdk-8u251-windows-i586 : 버전에 따라 이름이 다를 수 있음)을 다운



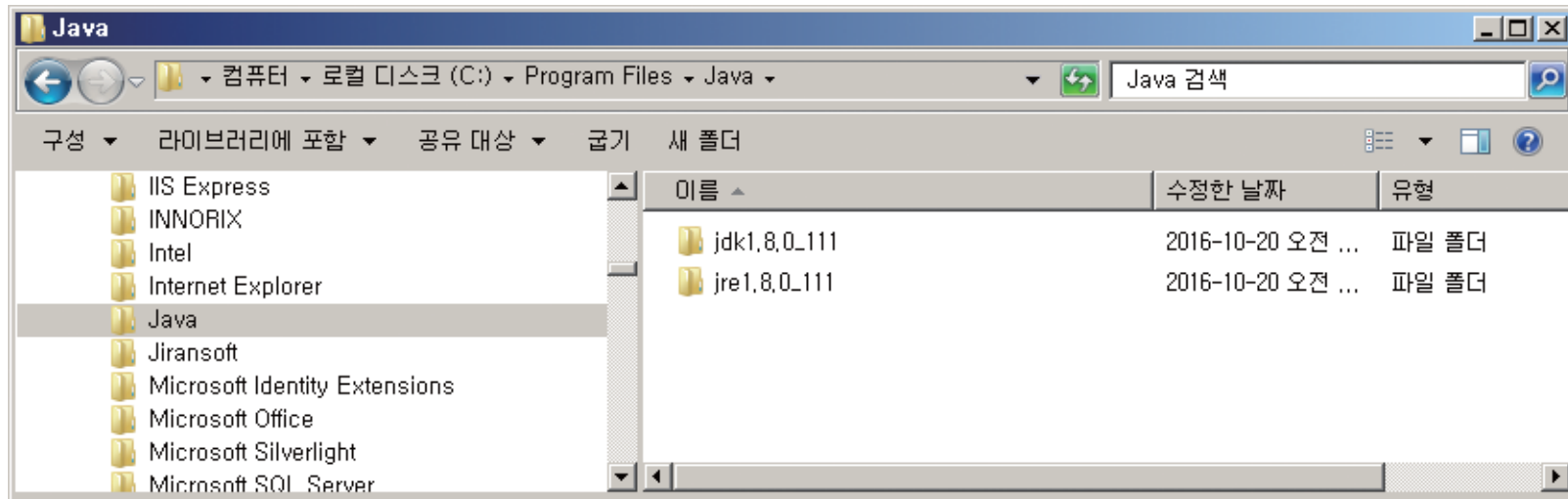
- Step 2 : 다운받은 소프트웨어를 실행한다.

- 다운받은 소프트웨어를 더블 클릭하여 설치를 완료한다.



#### ● Step 2 : 다운받은 소프트웨어를 실행한다.

- 완료되면 일반적으로 "C:\Program Files\Java" 디렉터리에 두 개의 디렉터리가 생성





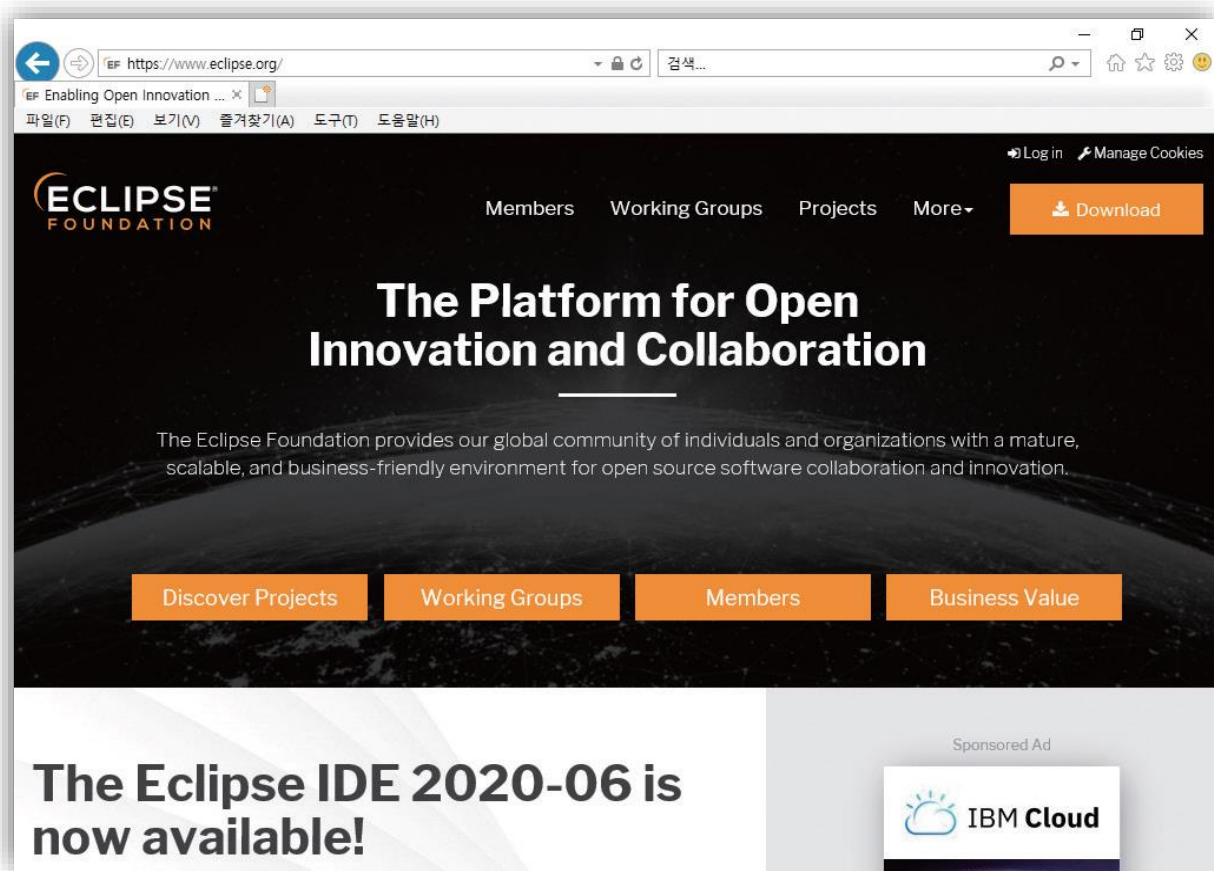
## 2-2 자바 통합 환경 :이클립스의 설치

- 이클립스(Eclipse) : 자바 프로그램을 개발하기 위한 통합 개발 환경 (Integrated Development Environment)
- 자바 프로그램을 개발하기 위한 다양한 도구들이 존재
- 무료로 제공되는 이클립스를 가장 많이 사용

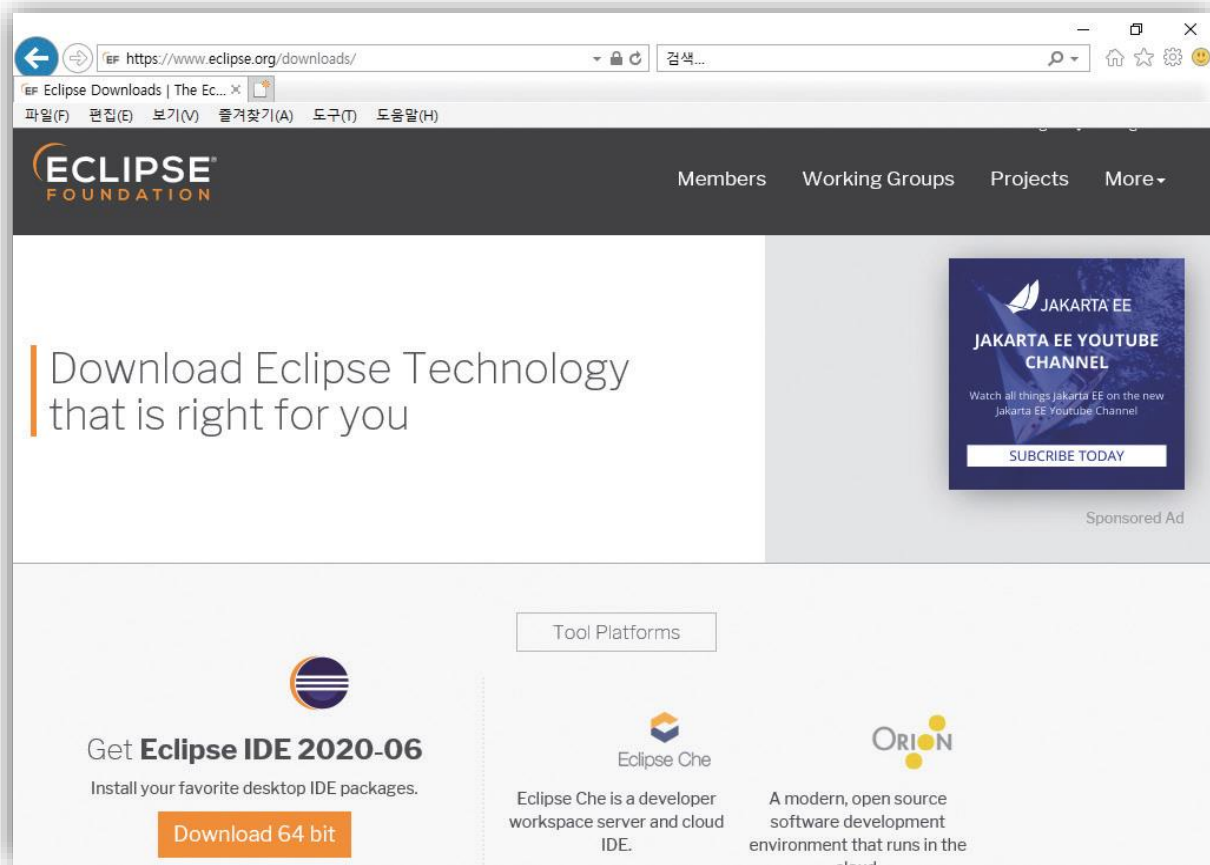


● Step 1 : 이클립스 홈페이지([www.eclipse.org](http://www.eclipse.org))를 방문

- 오른쪽 상단에 있는 "Download Eclipse"를 선택

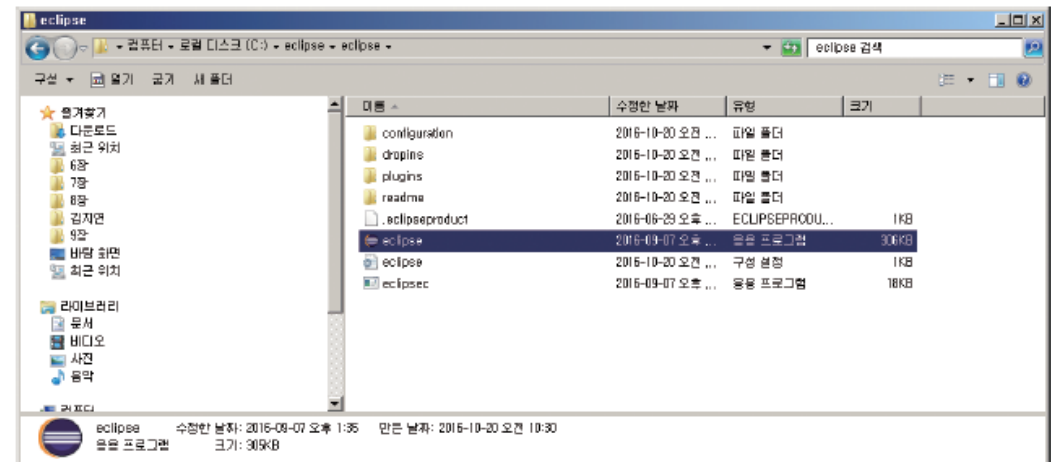


- **Step 1 : 이클립스 홈페이지([www.eclipse.org](http://www.eclipse.org))를 방문**
- 다운로드 페이지에서 자신의 컴퓨터에 적합한(32비트 또는 64비트) "Eclipse Standard"를 선택하여 다운로드



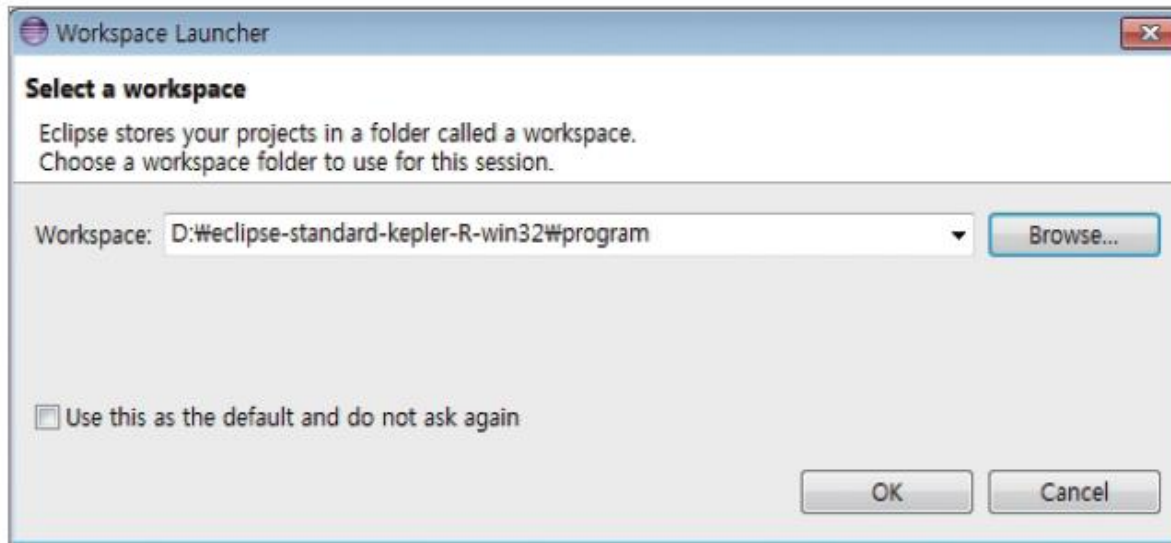
#### ● Step 2 : 이클립스를 설치 (31p.)

- 다운받은 소프트웨어를 더블 클릭하여 적당한 위치에 압축을 풀면 설치가 완료



#### ● 이클립스의 실행

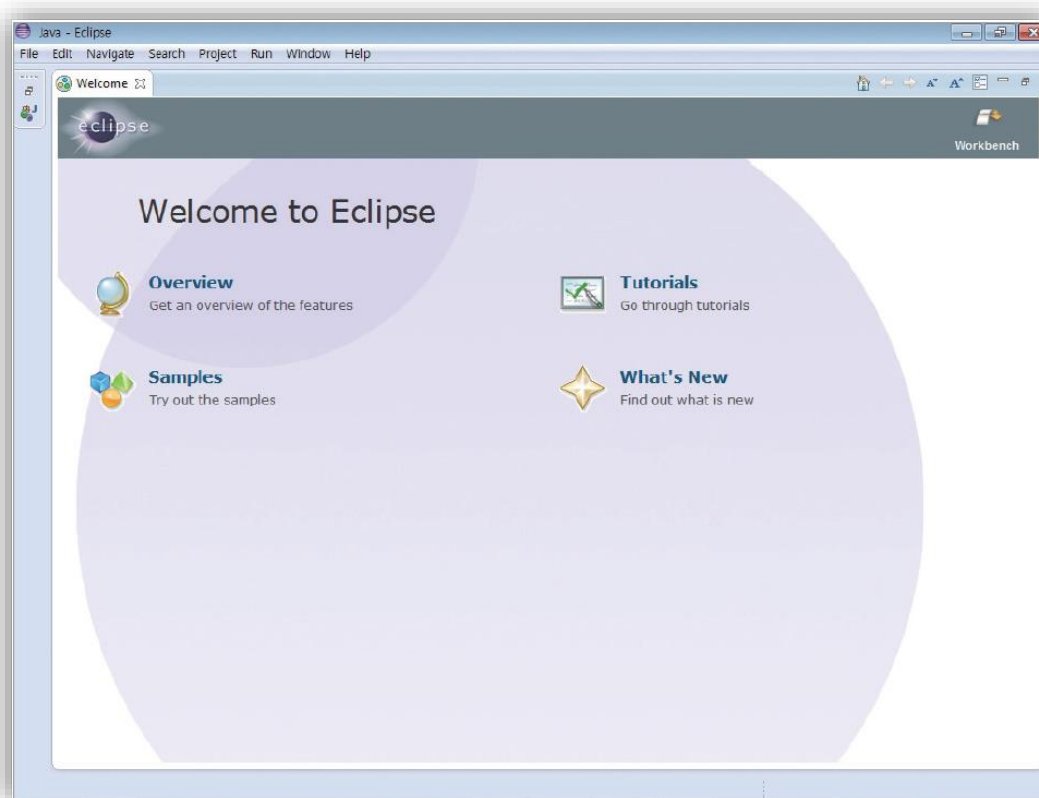
- eclipse 폴더에 있는 eclipse.exe 파일을 실행 : workspace를 묻는 화면이 나타난다
  - workspace : 작성된 프로그램이 저장될 공간



- 공간을 지정하고 "OK" 버튼을 선택

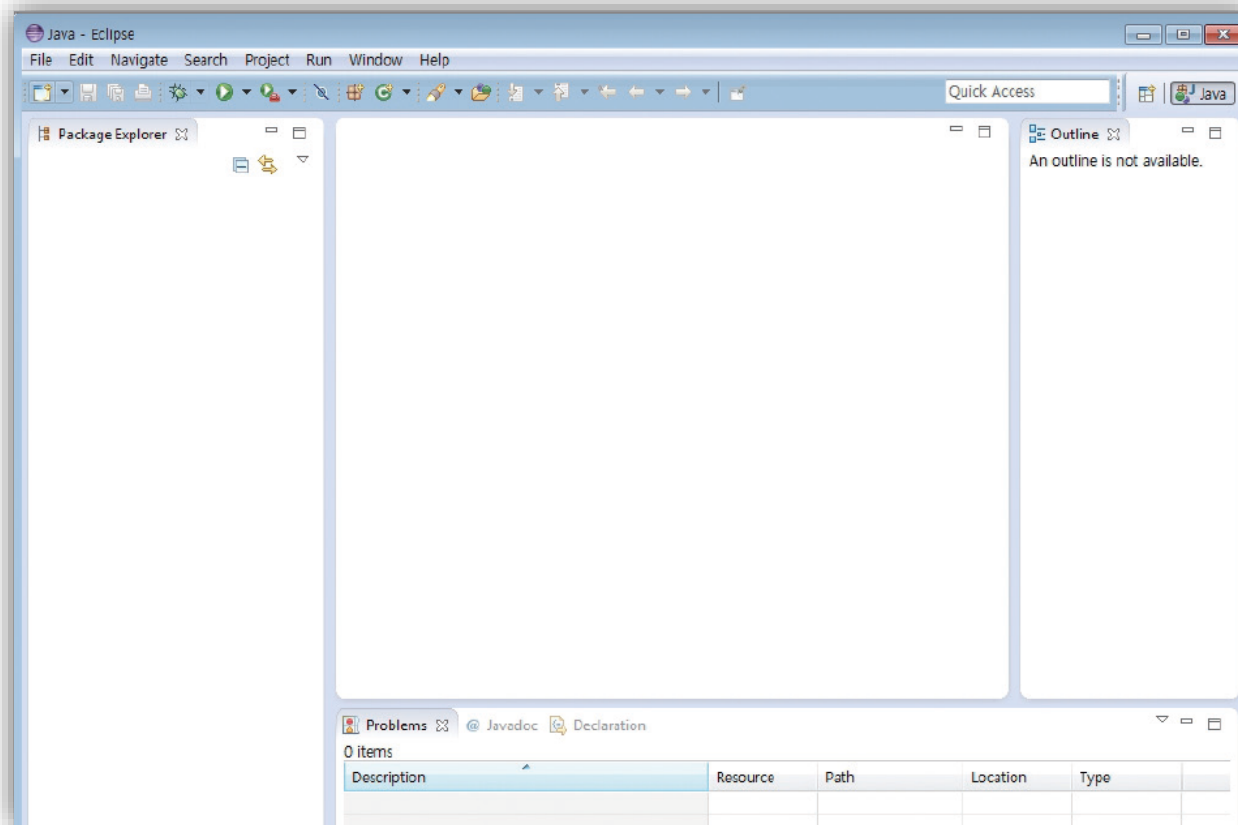
#### ● 이클립스의 실행

- Welcome 화면이 나타남 : 처음 한번만 나타나고 다음 실행부터 나타나지 않는다

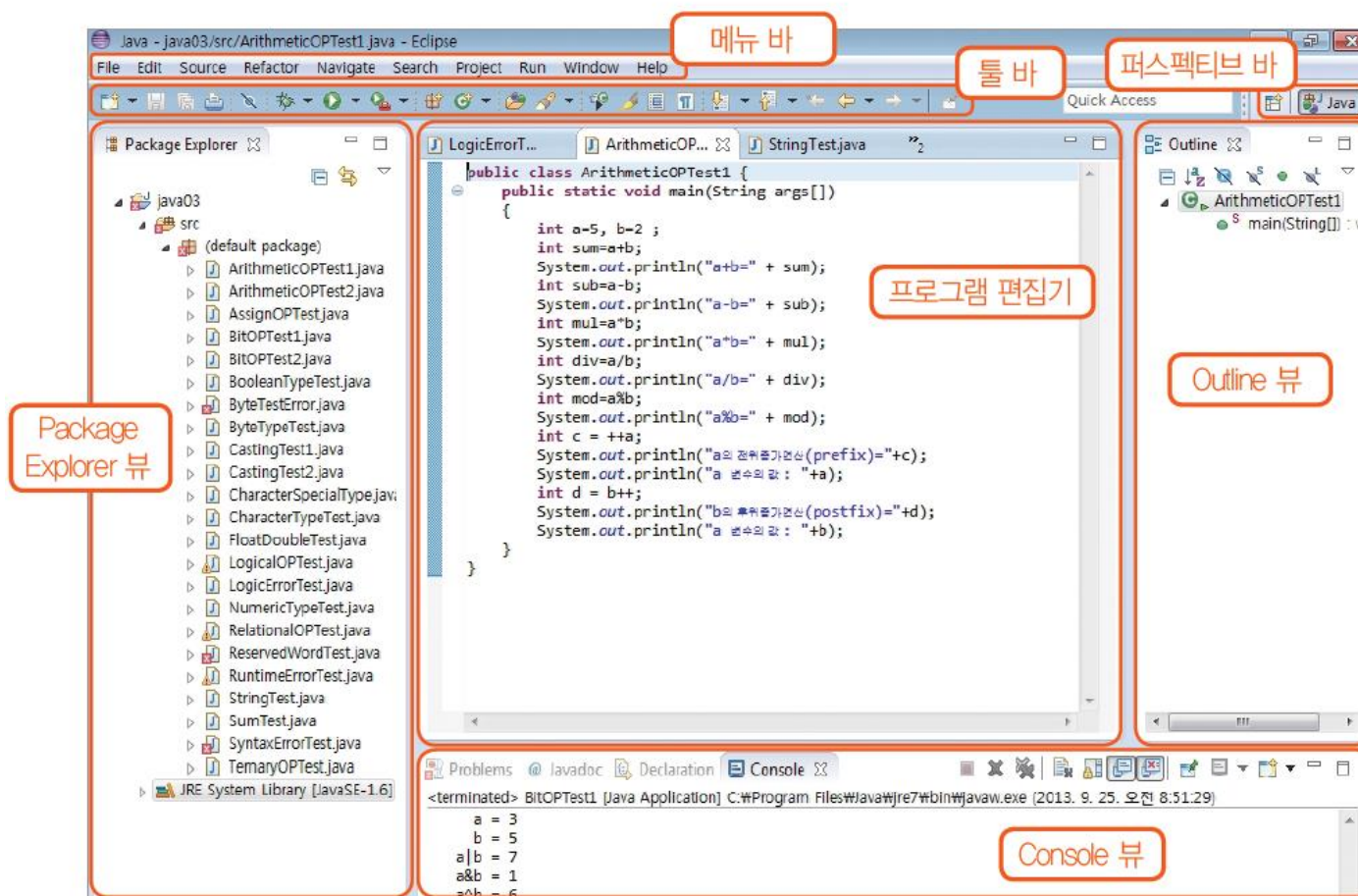


#### ● 이클립스의 실행

- Welcome 화면을 닫으면 이클립스 초기화면이 나타남



#### ● 이클립스의 기본 구조





#### ● 메뉴 바

- 메뉴를 통하여 이클립스의 모든 기능들을 사용할 수 있습니다.

#### ● 툴 바

- 자주 쓰이는 기능들을 편리하게 사용할 수 있도록 버튼으로 제공하는 바입니다. 다양한 형태의 버튼으로 제공되고 있습니다.

#### ● Package Explorer 뷰

- 프로젝트를 중심으로 패키지과 클래스 파일, 라이브러리를 관리하는 뷰입니다. 프로젝트에 소속된 클래스와 패키지, 라이브러리 등을 볼 수 있는 창입니다.

#### ● 프로그램 편집기

- 자바 프로그램을 작성할 수 있는 편집기 창입니다. 자바 프로그램의 편집기는 자동 완성 기능을 포함한 다양한 기능들이 제공되고 있습니다.





#### ● Outline 뷰

- 현재 편집되고 있는 프로그램 코드의 개요와 트리 구조를 나타냅니다.

#### ● Console 뷰

- 프로그램의 실행 결과나 프로그램에 오류가 있을 경우 오류를 나타내는 창입니다.

#### ● 퍼스펙티브 바

- 퍼스펙티브는 이클립스를 사용할 때의 뷰나 창의 배열 형태를 의미합니다. 이클립스에서는 다양한 형태의 퍼스펙티브를 제공하고 있고[그림 2-13], 사용자가 자신만의 퍼스펙티브를 만들어 저장한 다음 사용할 수도 있습니다.

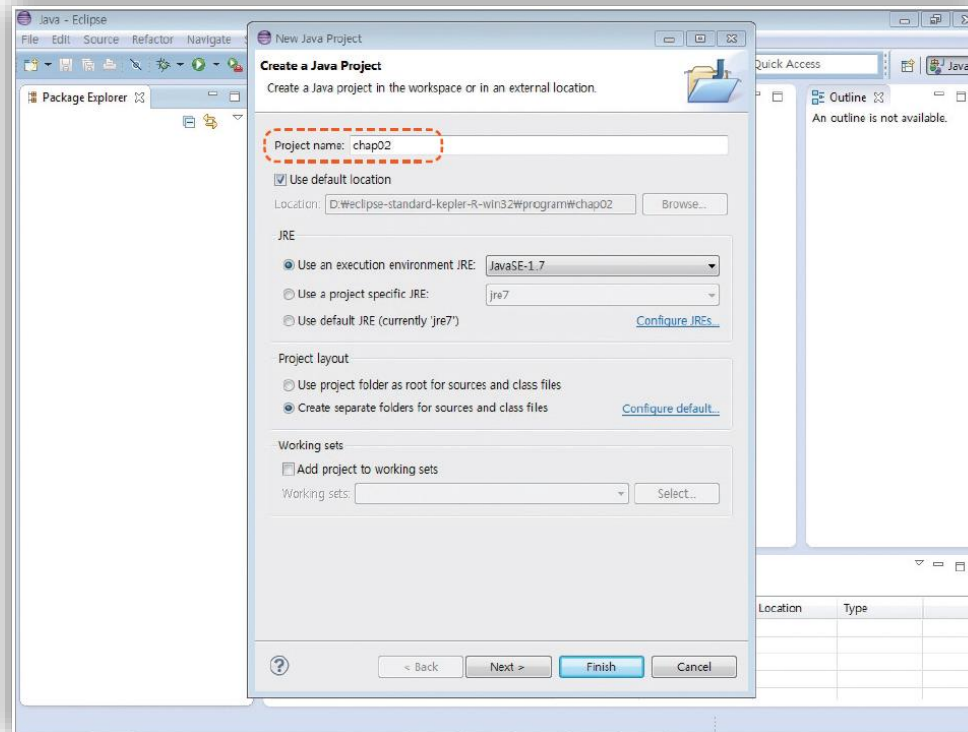


### ● 이클립스를 이용한 프로그램 작성 순서

- Step 1 : 프로젝트 선정
- Step 2 : 프로그램(클래스) 작성
- Step 3 : 프로그램 실행

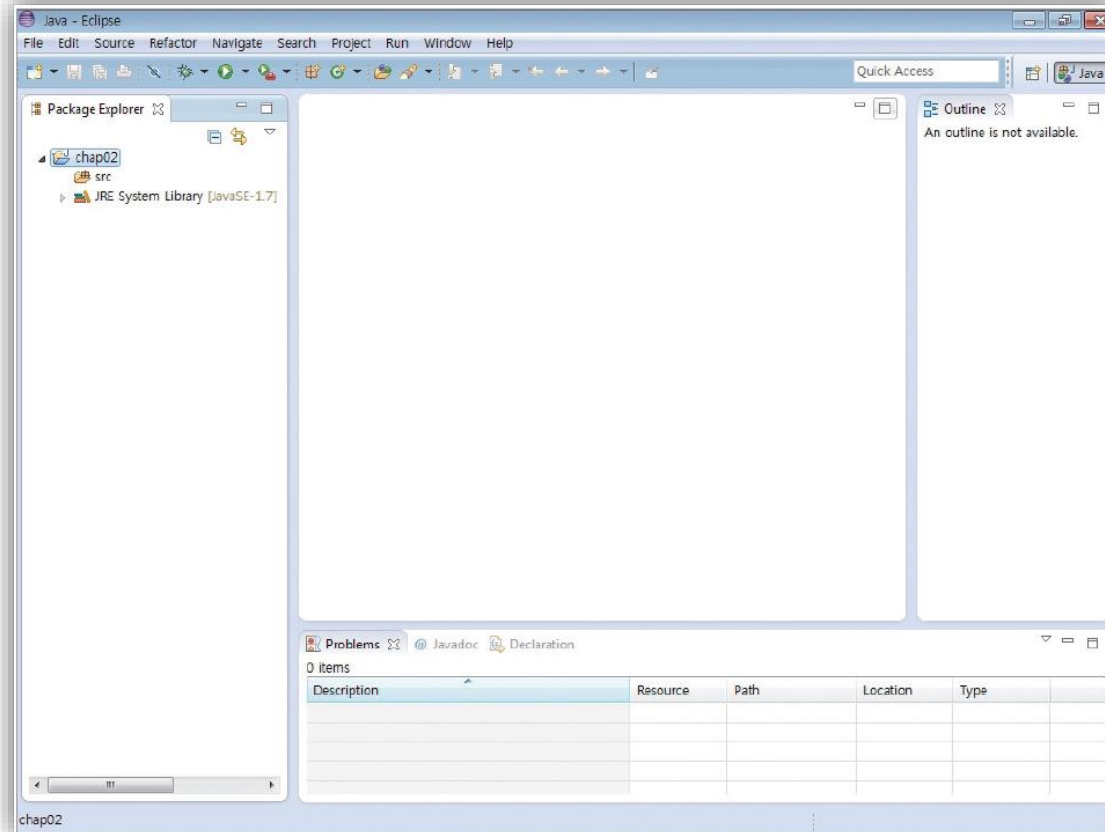
#### ● Step 1 : 프로젝트 만들기

- 이클립스는 프로젝트 단위로 작업이 진행
- 메뉴에서 [File] → [New] → [Java Project]를 선택
- 프로젝트를 chap02로 지정하고 하단의 "Finish" 버튼 선택



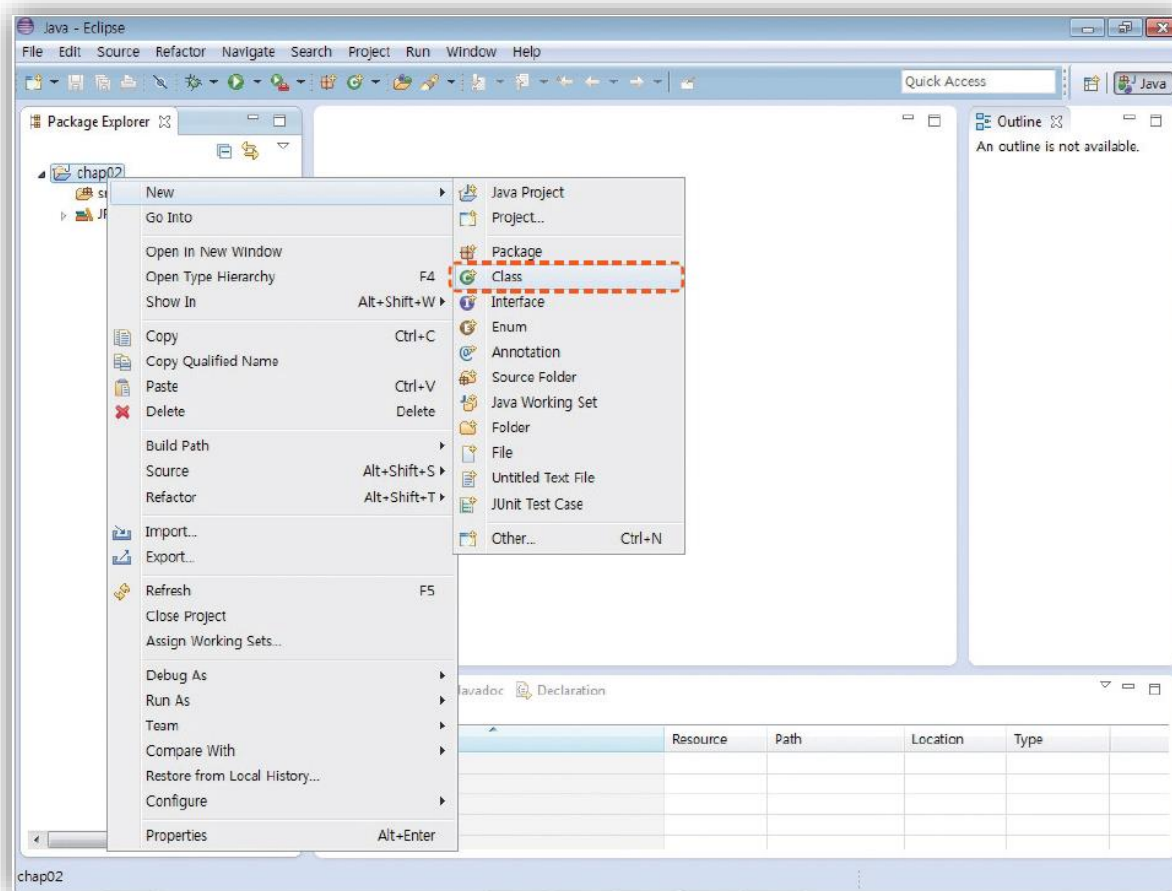
#### ● Step 1 : 프로젝트 만들기

- Package Explorer 뷰에 패키지가 표시



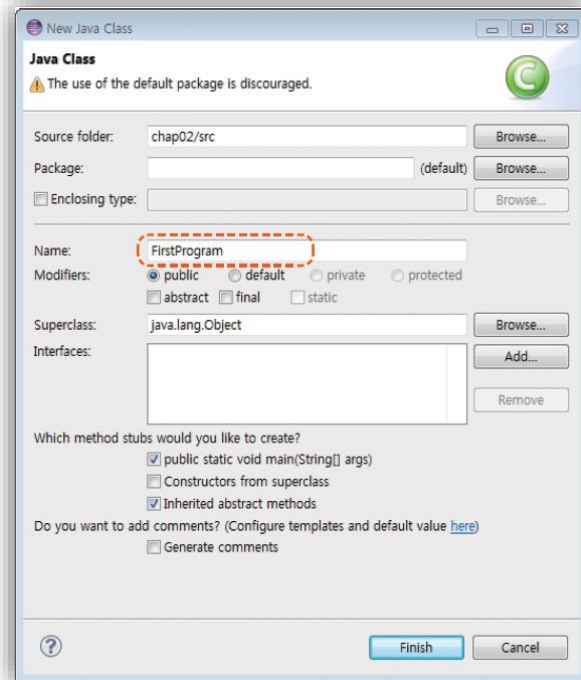
#### ● Step 2 : 프로그램(클래스) 작성

- 프로젝트 명을 마우스 오른쪽 버튼으로 선택한 다음 [New] → [Class]를 선택



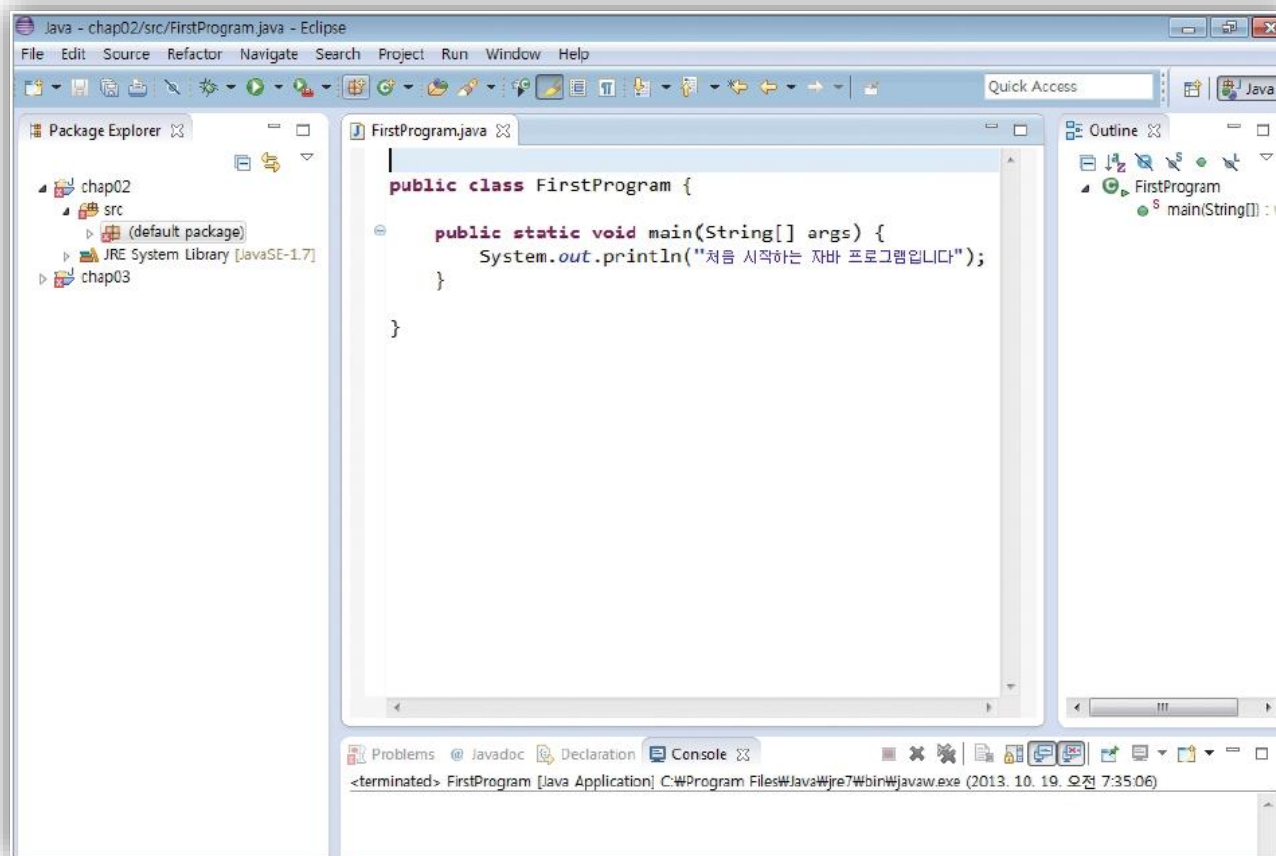
#### ● Step 2 : 프로그램(클래스) 작성

- 클래스 생성을 위한 팝업 창이 나타납니다.
- Package 필드에 "chap02"가 나타납니다. 그 글자를 지우면 아래의 화면처럼 default가 표시 됩니다.
- Name 필드에 생성될 클래스 이름을 지정(FirstProgram) 합니다.



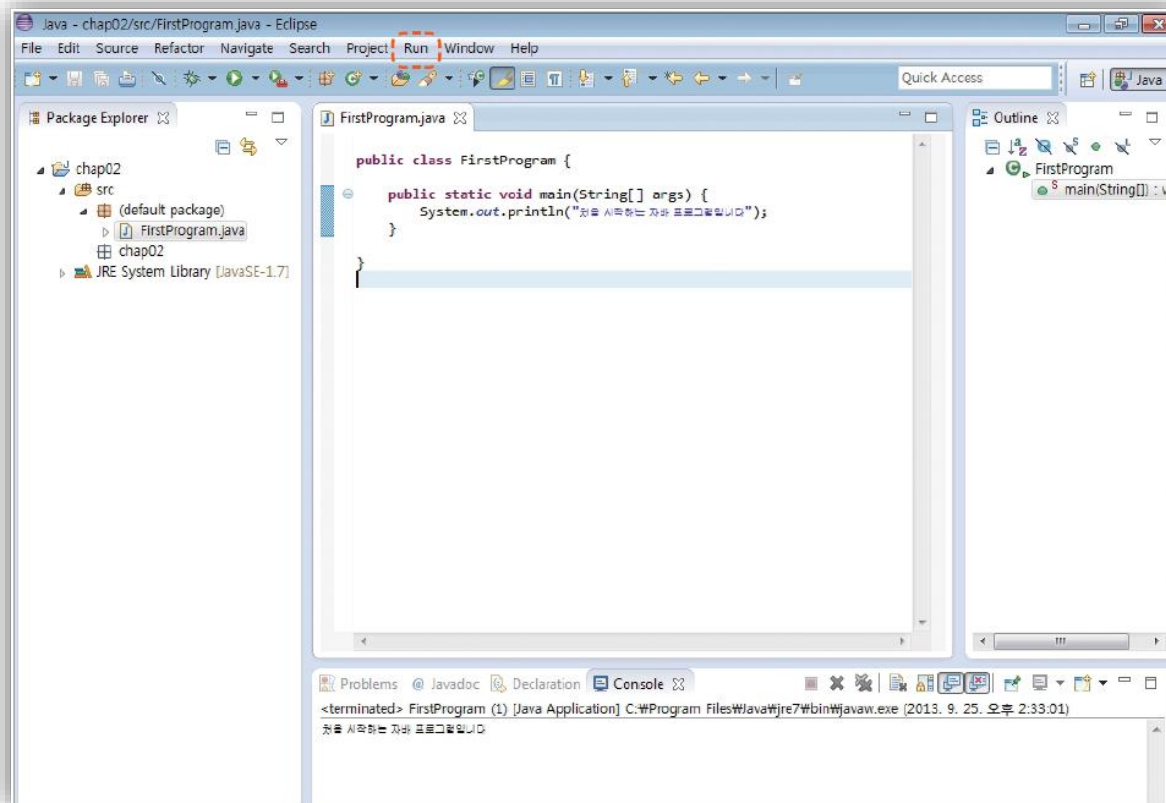
#### ● Step 2 : 프로그램(클래스) 작성

- 편집기 창에서 프로그램을 입력



#### ● Step 3 : 프로그램 실행.

- 메뉴에서 [Run] → [Run]을 실행
- 프로그램의 실행 결과가 아래의 Console 창에 나타난다







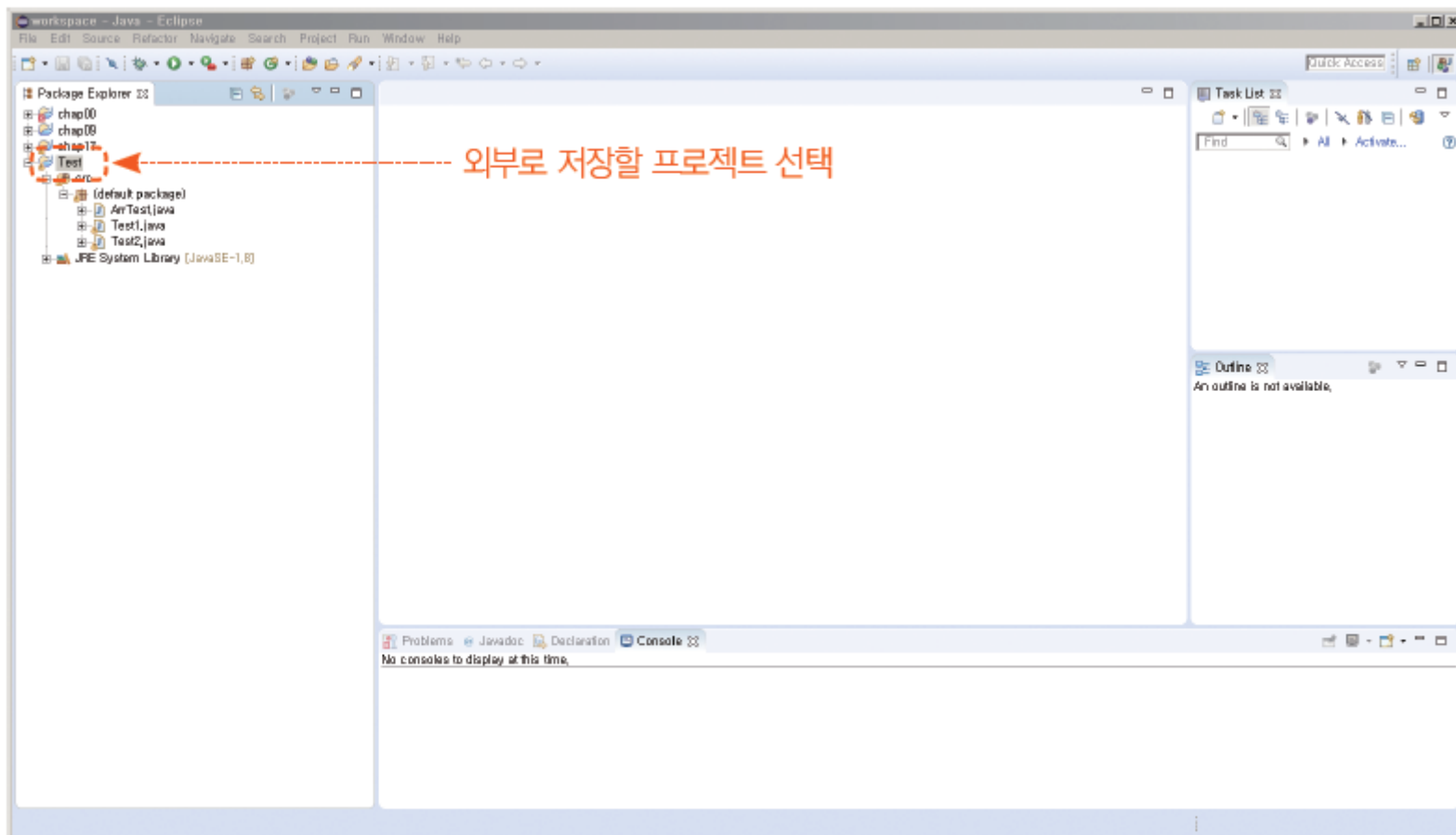
- 이클립스에서는 프로젝트 단위로 개발 환경을 제공
- 프로젝트는 소프트웨어를 개발하는 단위(규모)
- Import-Export 기능을 이용하여 특정 컴퓨터에서 작성된 프로젝트를 다른 컴퓨터에서 그대로 사용할 수 있는 편리한 개발환경을 제공.



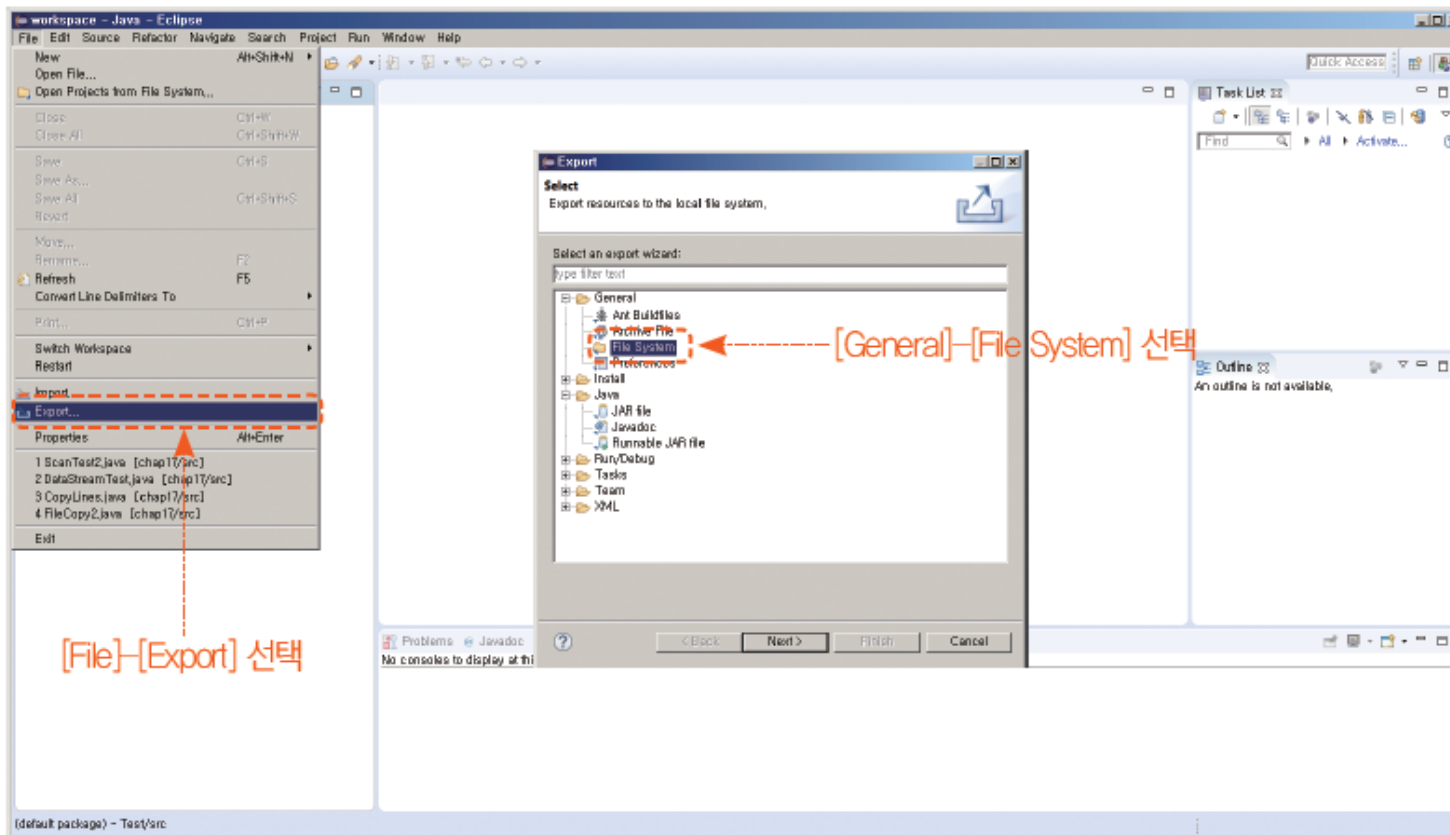
- 이클립스에서 프로젝트 출력

- 개발된 프로젝트를 외부 컴퓨터에서 사용하기 위해 프로젝트 단위로 특정 폴더에 저장하는 기능

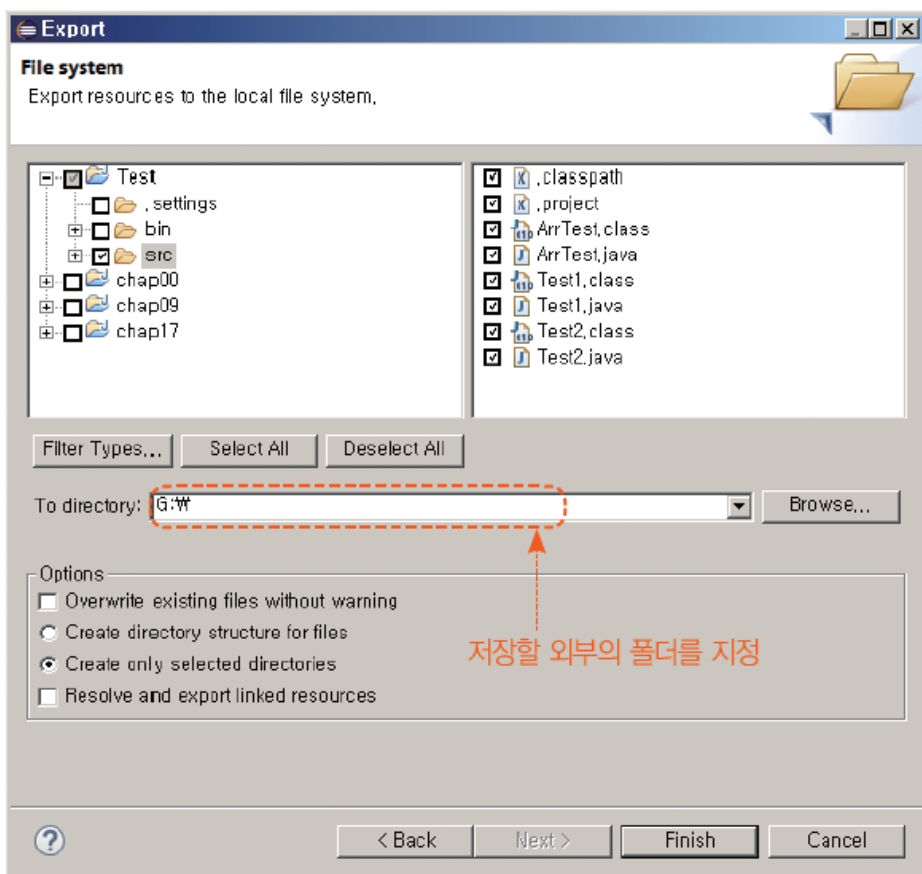
#### ● Step 1 : 프로젝트를 선택



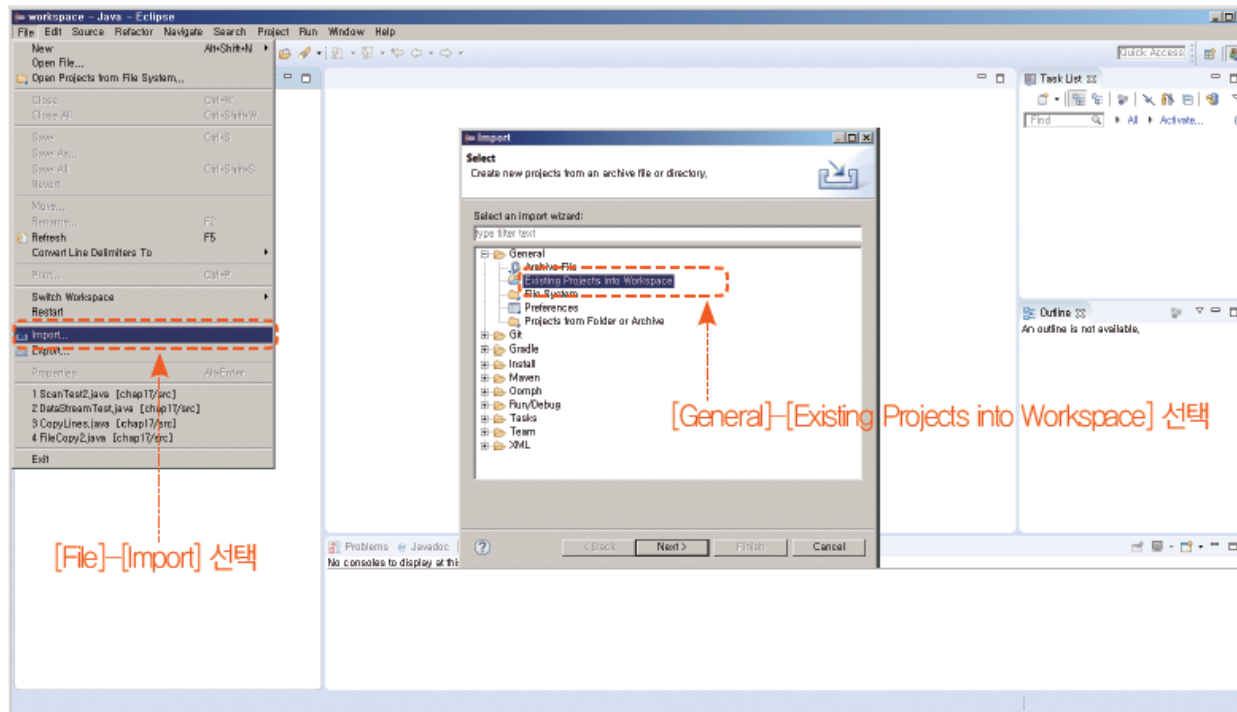
- Step 2 : [File]->[Export] 메뉴를 선택하고 팝업창에서 [General]->[File System]을 선택하고 Next 버튼을 선택



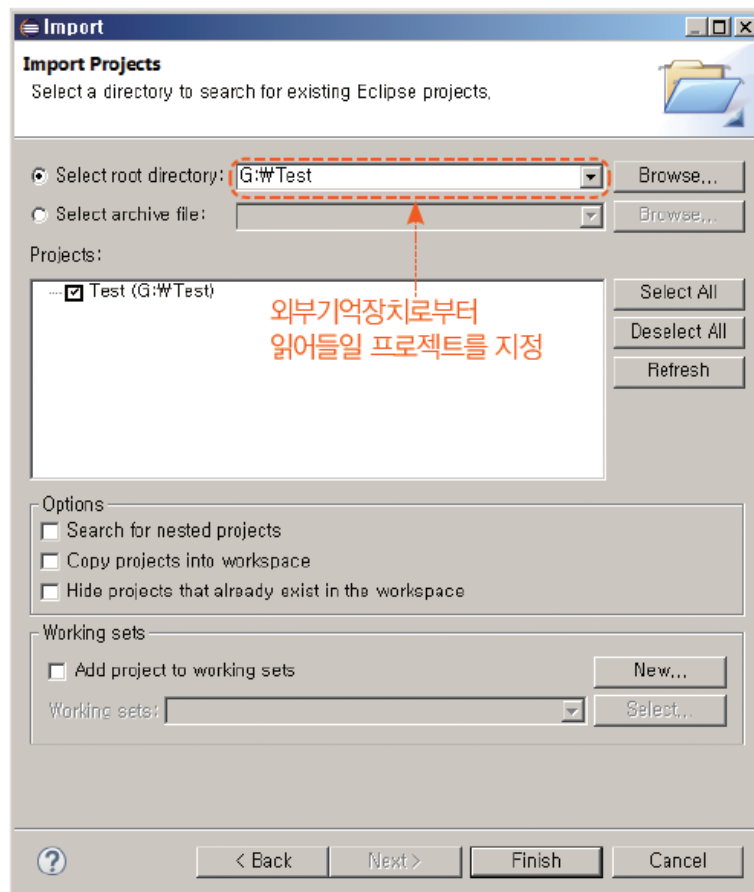
- Step 3 : 나타난 팝업창에서 저장할 디렉토리를 지정하고 Finish 버튼을 선택



- 이클립스에서 프로젝트 입력
- **Step 1** : [File]->[Import]를 선택하고 팝업창에서 [General]->[Existing Projects into Workspace]를 선택



#### ● Step 2 : 팝업창에서 외부장치로부터 읽어 들일 폴더를 선택



## 2-6 자바 API 참조하기

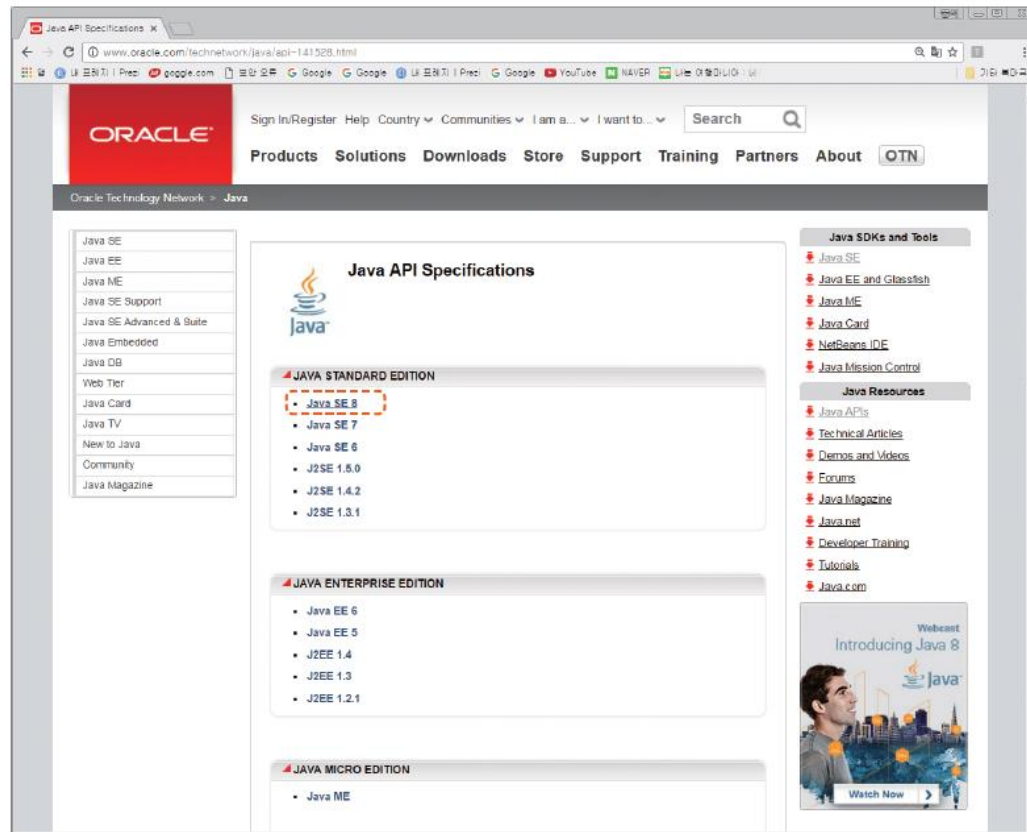
- 자바에서 제공되는 클래스 라이브러리(API)
  - Java 8 버전 : 약 5천 개
- 자바 프로그램에서 클래스 라이브러리의 사용은 필수적
- 프로그래머는 라이브러리를 사용하기 위해서는 클래스의 속성과 기능을 숙지
- 자바 프로그램에서 클래스 라이브러리를 사용하지 않고 작성한다는 것은 불가능
- [java.sun.com](http://java.sun.com)에서 클래스 라이브러리의 속성과 기능을 제공



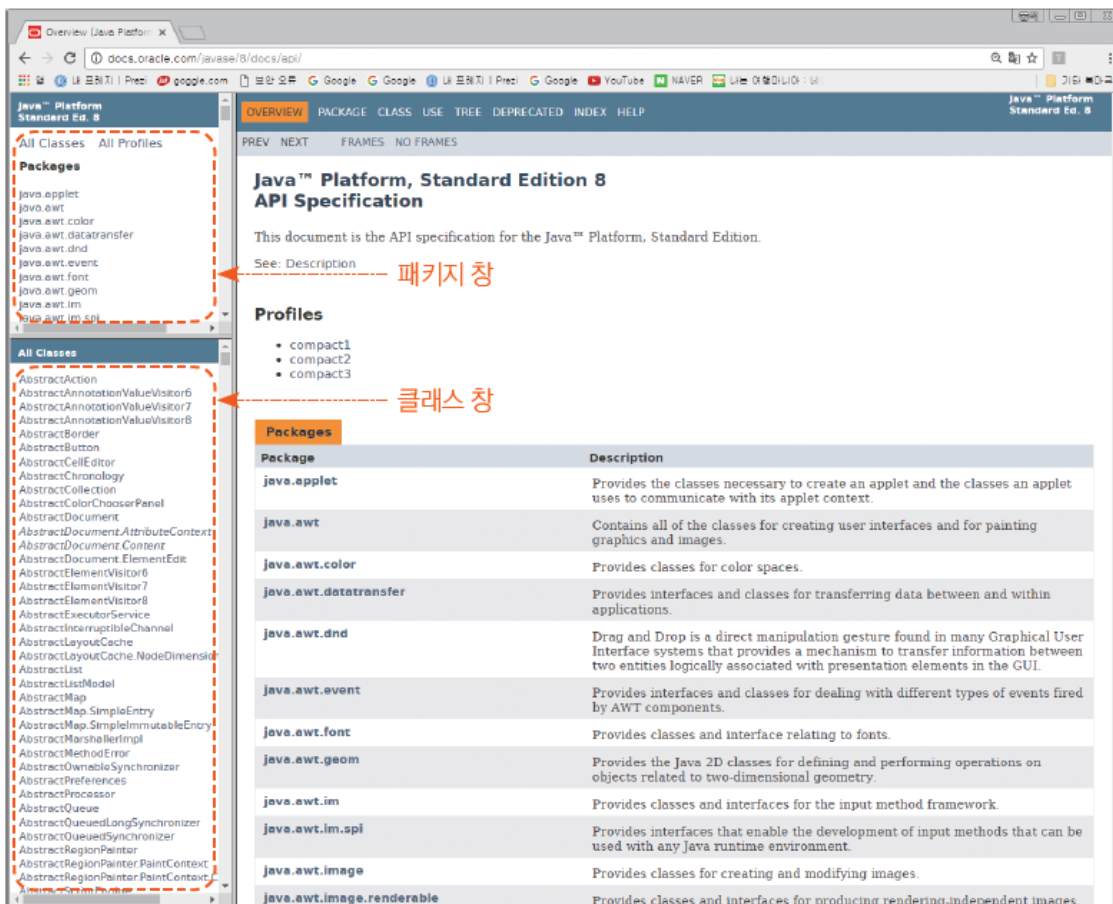
#### ● Step 1 : java.sun.com에 접근하여 Java APIs를 선택



### ● Step 2 : Java SE 8 최근 버전을 선택



### ● Step 2 : Java SE 8 최근 버전을 선택



The screenshot shows the Java Platform Standard Edition 8 API Specification website. The left sidebar contains a list of packages and classes. The main content area shows the 'Overview' page for Java Platform, Standard Edition 8 API Specification. Red dashed boxes and arrows highlight the 'Packages' and 'All Classes' sections in the sidebar, with labels '패키지 창' (Package Window) and '클래스 창' (Class Window) respectively.

**패키지 창 (Package Window):**

- java.applet
- java.awt
- java.awt.color
- java.awt.datatransfer
- java.awt.dnd
- java.awt.event
- java.awt.font
- java.awt.geom
- java.awt.im
- java.awt.im.spi

**클래스 창 (Class Window):**

- AbstractAction
- AbstractAnnotationValueVisitor6
- AbstractAnnotationValueVisitor7
- AbstractAnnotationValueVisitor8
- AbstractBorder
- AbstractButton
- AbstractCellEditor
- AbstractChronology
- AbstractCollection
- AbstractColorChooserPanel
- AbstractDocument
- AbstractDocument.AttributeContext
- AbstractDocument.Content
- AbstractDocument.ElementEdit
- AbstractElementVisitor6
- AbstractElementVisitor7
- AbstractElementVisitor8
- AbstractExecutorService
- AbstractInputStreamChannel
- AbstractLayoutCache
- AbstractLayoutCache.NodeDimension
- AbstractList
- AbstractListModel
- AbstractMap
- AbstractMap.SimpleEntry
- AbstractMap.SimpleImmutableEntry
- AbstractMarshalerImpl
- AbstractMethodError
- AbstractObservableSynchronizer
- AbstractPreferences
- AbstractProcessor
- AbstractQueue
- AbstractQueuedLongSynchronizer
- AbstractQueuedSynchronizer
- AbstractRegionPainter
- AbstractRegionPainter.PaintContext
- AbstractRegionPainter.PaintContext

**Overview Page Content:**

**Java™ Platform, Standard Edition 8 API Specification**

This document is the API specification for the Java™ Platform, Standard Edition.

See: Description

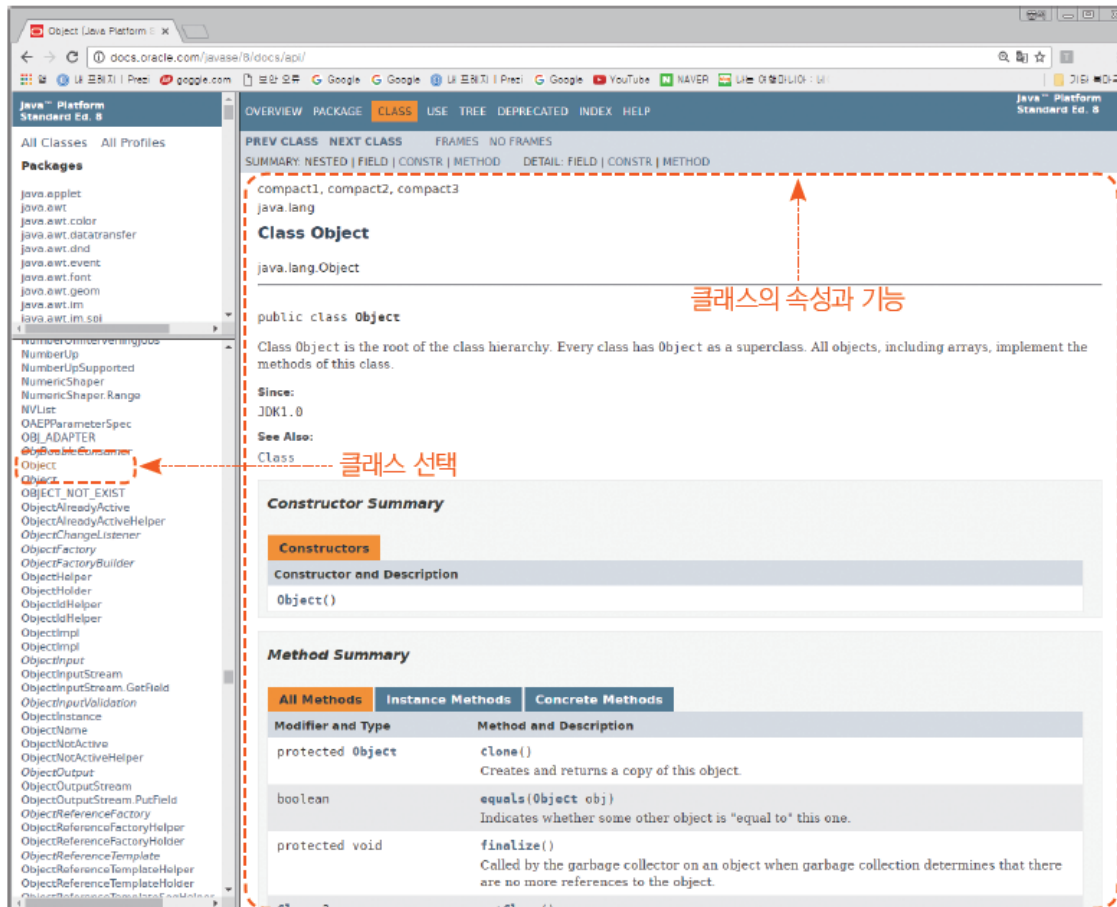
**Profiles**

- compact1
- compact2
- compact3

**Packages**

Package	Description
java.applet	Provides the classes necessary to create an applet and the classes an applet uses to communicate with its applet context.
java.awt	Contains all of the classes for creating user interfaces and for painting graphics and images.
java.awt.color	Provides classes for color spaces.
java.awt.datatransfer	Provides interfaces and classes for transferring data between and within applications.
java.awt.dnd	Drag and Drop is a direct manipulation gesture found in many Graphical User Interface systems that provides a mechanism to transfer information between two entities logically associated with presentation elements in the GUI.
java.awt.event	Provides interfaces and classes for dealing with different types of events fired by AWT components.
java.awt.font	Provides classes and interface relating to fonts.
java.awt.geom	Provides the Java 2D classes for defining and performing operations on objects related to two-dimensional geometry.
java.awt.im	Provides classes and interfaces for the input method framework.
java.awt.im.spi	Provides interfaces that enable the development of input methods that can be used with any Java runtime environment.
java.awt.image	Provides classes for creating and modifying images.
java.awt.image.renderable	Provides classes and interfaces for producing rendering-independent images.

#### ● Step 3 : 원하는 클래스를 선택



The screenshot shows the Java Platform Standard Ed. 8 API documentation website. The left sidebar lists various packages and classes, with 'Object' selected. The main content area displays the 'Class Object' page, which includes a description of the class, its constructors, and a table of methods. Red dashed boxes and arrows highlight the 'Object' class in the sidebar and the 'Class Object' page content, with labels '클래스 선택' (Class Selection) and '클래스의 속성과 기능' (Class Properties and Functions).

**클래스 선택**

**클래스의 속성과 기능**

**Constructor Summary**

**Constructors**

**Constructor and Description**

**Object()**

**Method Summary**

Modifier and Type	Method and Description
protected Object	<b>clone()</b> Creates and returns a copy of this object.
boolean	<b>equals(Object obj)</b> Indicates whether some other object is "equal to" this one.
protected void	<b>finalize()</b> Called by the garbage collector on an object when garbage collection determines that there are no more references to the object.

- **자바의 기본 구조를 익히기 위해 자바의 입출력 방법이나 오류를 미리 익혀야 한다.**
  - 자바의 입출력은 라이브러리 클래스를 이용합니다.
  - 현 시점에서 입출력과 관련된 라이브러리를 이해할 필요는 없습니다.
  - 입출력 방법을 상용 구문처럼 외워서 사용하면 됩니다.
  - 프로그램을 설명하는 주석문에 대해 학습합니다.
  - 프로그램 작성시 발생하는 오류에 대해 학습합니다.
- **입출력과 오류에 관한 자세한 내용들은 해당 부분(13장)에서 설명합니다**

## 3-1 간단한 자바 프로그램과 문자열의 출력

### ● 간단한 자바 프로그램의 구조

- 하나의 클래스에 하나의 메소드만 가진 간단한 클래스
- 6장까지는 이러한 구조의 클래스만 사용(public, static 등의 의미는 7장 이후에 설명함, 그 전까지는 상용구문처럼 외워서 사용)

예제 2.1

FirstProgram.java

```
01: public class FirstProgram {  
02:  
03:     public static void main(String[] args) {  
04:         System.out.println("처음 시작하는 자바 프로그램입니다");  
05:     }  
06: }
```

실행 결과

처음 시작하는 자바 프로그램입니다



### ● 문자열의 출력

- 자바 언어는 표준 출력문으로 System.out.println() 문장을 제공

```
System.out.println("처음 시작하는 자바 프로그램입니다"); ← 출력 후 줄을 바꿉니다  
System.out.print("처음 시작하는 자바 프로그램입니다"); ← 출력 후 줄을 바꾸지 않습니다
```

## ● 실습 예제 2.2

예제 2.2

SecondProgram.java

```
01: public class SecondProgram {  
01: public static void main(String[] args) {  
02:     System.out.print("두 번째로 작성해 보는 ");    ◀--내용을 출력한 다음 줄을 바꾸지 않는다  
03:     System.out.println("자바 프로그램입니다 ");  
04:     System.out.println("처음 시작하는 " + "자바 프로그램입니다");  
05: }  
06: }
```

## 실행 결과

두 번째로 작성해 보는 자바 프로그램입니다  
처음 시작하는 자바 프로그램입니다



## 3-2 프로그램에 값을 입력하기 위한 방법

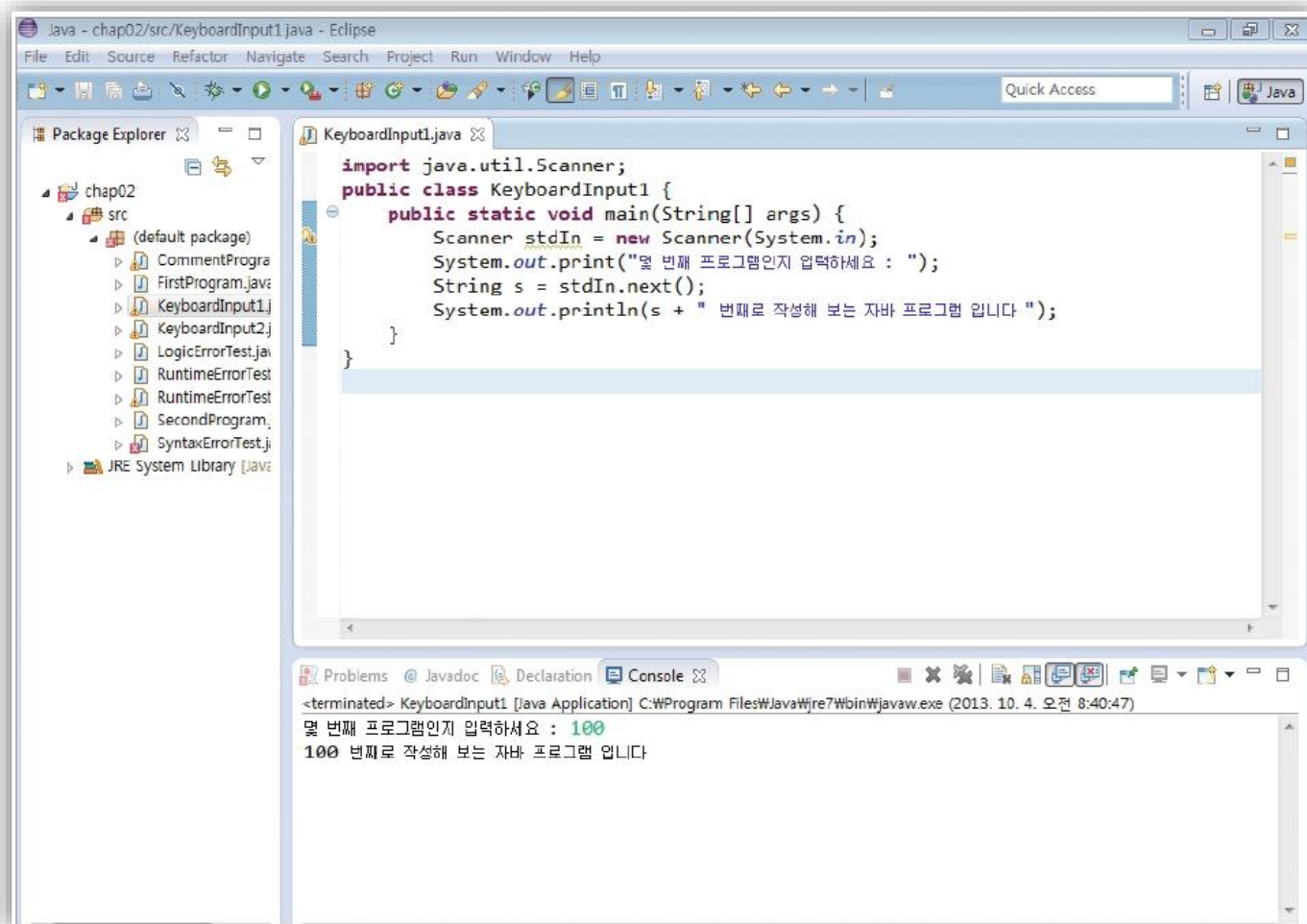
- 프로그램 실행 시 값을 입력할 수 있는 방법 중에서 Scanner 라이브러리 클래스를 이용한다
- 실습 예제 2.3

## 예제 2.3

## KeyboardInput1.java

```
01: import java.util.Scanner; ← 라이브러리 클래스 포함.
02: public class KeyboardInput1 {
03:     public static void main(String[] args) {
04:         Scanner stdIn = new Scanner(System.in); ← 키보드를 입력을 위해 Scanner 객체 생성
05:         System.out.print("몇 번째 프로그램인지 입력하세요 : ");
06:         String s = stdIn.next(); ← 사용자의 입력을 문자열로 받는다
07:         System.out.println(s + " 번째로 작성해 보는 자바 프로그램 입니다 ");
08:     }
09: }
```

- 프로그램을 실행시키고 Console 창에서 데이터를 입력한다





### 3 자바 학습을 위한 준비

## 3-2 프로그램에 값을 입력하기 위한 방법

9th edition

### ● 실습 예제 2.4 : Scanner 클래스의 nextInt(), nextDouble() 메소드 사용

#### 예제 2.4

KeyboardInput2.java

```
01: import java.util.Scanner;
02: public class KeyboardInput2 {
03:     public static void main(String[] args) {
04:         Scanner stdIn = new Scanner(System.in);
05:         System.out.print("이름과 나이, 몸무게를 공간(space)으로 구분하여 입력 : ");
06:         String name = stdIn.next(); <----- 키보드로부터 이름을 문자열로 입력
07:         int i = stdIn.nextInt(); <----- 키보드로부터 나이를 정수로 입력
08:         double d = stdIn.nextDouble(); <----- 키보드로부터 몸무게를 실수로 입력
09:         System.out.println(name + "씨의 나이는 " + i + "세이고");
10:         System.out.println("몸무게는 " + d + "Kg 입니다");
11:     }
12: }
```

### ● 실습 예제 2.4 : 실행 결과

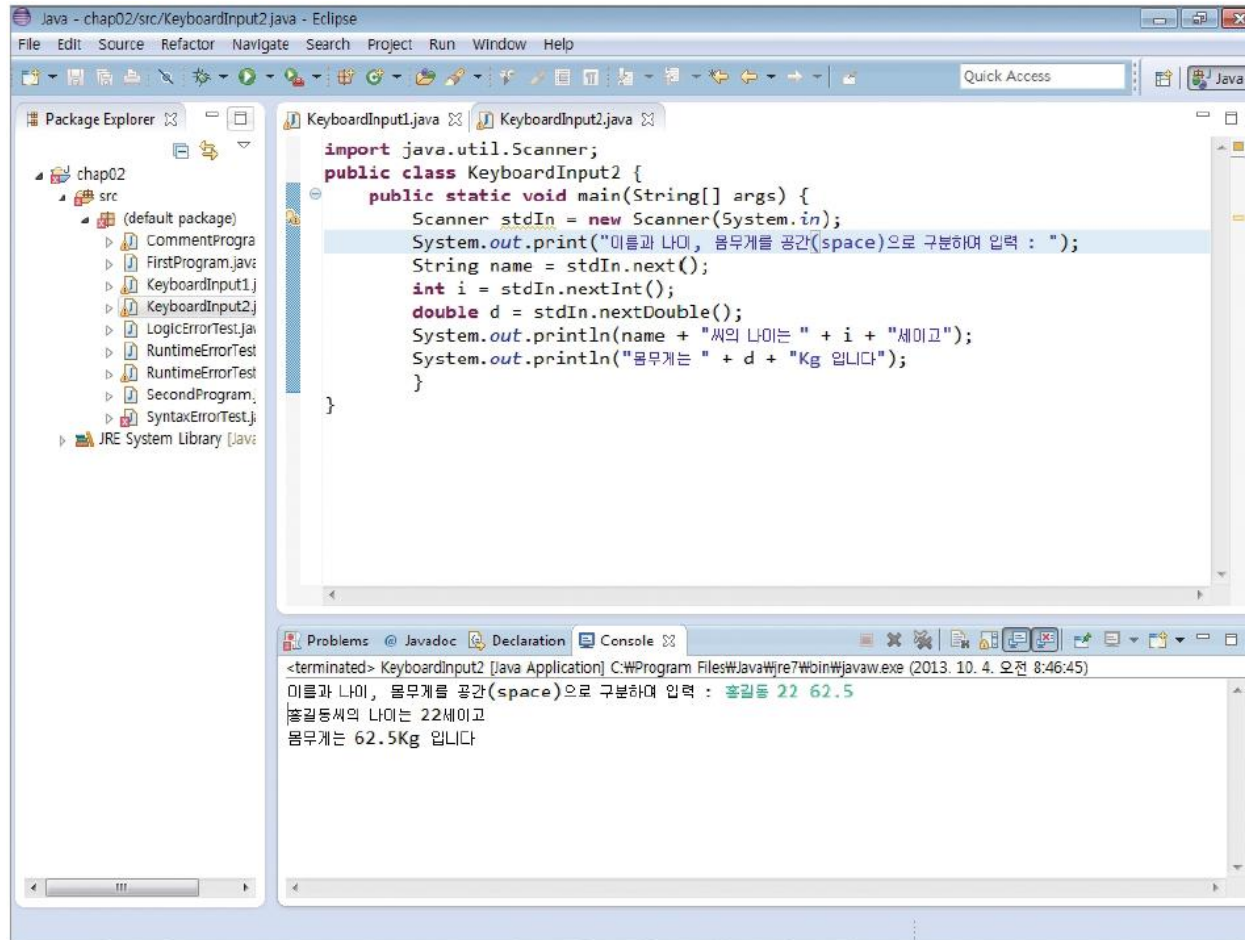
실행 결과

입력 값으로 홍길동과 22, 62.5를 입력

홍길동 씨의 나이는 22세이고

몸무게는 62.5Kg 입니다

### ● 실습 예제 2.4 : 실행 시 데이터 입력은 공간으로 구분한다



The screenshot shows the Eclipse IDE with the following components:

- Package Explorer:** Shows a project named 'chap02' with a source folder 'src' containing several Java files, including 'KeyboardInput2.java'.
- Editor:** Displays the code for 'KeyboardInput2.java'. The code uses the `Scanner` class to read input from the user. It prompts for a name and age, then calculates and prints the age in years and the weight in kilograms.
- Console:** Shows the output of the program. It displays the prompt '이름과 나이, 몸무게를 공간(space)으로 구분하여 입력 : ' followed by the user input '홍길동 22 62.5'. The program then prints '홍길동씨의 나이는 22세이고' and '몸무게는 62.5Kg 입니다'.

```
import java.util.Scanner;
public class KeyboardInput2 {
    public static void main(String[] args) {
        Scanner stdIn = new Scanner(System.in);
        System.out.print("이름과 나이, 몸무게를 공간(space)으로 구분하여 입력 : ");
        String name = stdIn.next();
        int i = stdIn.nextInt();
        double d = stdIn.nextDouble();
        System.out.println(name + "씨의 나이는 " + i + "세이고");
        System.out.println("몸무게는 " + d + "Kg 입니다");
    }
}
```

<terminated> KeyboardInput2 [Java Application] C:\Program Files\Java\jre7\bin\javaw.exe (2013. 10. 4. 오전 8:46:45)  
이름과 나이, 몸무게를 공간(space)으로 구분하여 입력 : 홍길동 22 62.5  
홍길동씨의 나이는 22세이고  
몸무게는 62.5Kg 입니다

## ● 주석문

- 프로그램의 구조와 동작 방법을 설명하기 위한 부분
- 프로그램의 실행에는 영향을 미치지 않는다
- 주석문은 향후에 다른 개발자가 프로그램을 수정하거나 보완할 때 매우 유용하다

자바의 주석문은 다음과 같이 3가지 형태로 많이 사용됩니다.

① `/* ..... */` : 여러 줄의 문장을 주석으로 처리할 수 있습니다.

② `// ..... :` 한 줄의 문장을 주석으로 처리합니다.

③ `/** ..... */` : 여러 줄의 문장을 주석으로 처리하고, javadoc 도구에 의해 사양서를 만들 수 있는 주석입니다.

### ● 예제 2.5 : 주석문을 사용한 프로그램

예제 2.5

CommentProgram.java

```
01: /**
02:  * 이 클래스는 세 개의 값을 입력받아 첫 번째 값은 문자열값으로, 두 번째 값은
    정수값으로
03:  * 세 번째 값은 실수값으로 출력하는 프로그램입니다.
04:  * @author 김충석
05:  * @see KeyboardInput2.java
06:  */
07: import java.util.Scanner;    // Scanner 클래스를 포함
08: public class CommentProgram {
09:     public static void main(String[] args) {
10:         Scanner stdIn = new Scanner(System.in);
11:         // 다음 문장은 콘솔 창에서 입력을 요청하는 문장입니다
12:         System.out.print("이름과 나이, 몸무게를 공간(space)으로 구분하여 입력 : ");
13:         /* 표준 입력으로 문자열과 정수,
14:          * 실수를 입력받아 적합한 형의 변수에 값을 저장 */
15:         String name = stdIn.next();
16:         int i = stdIn.nextInt();
17:         double d = stdIn.nextDouble();
18:         System.out.println(name + "씨의 나이는 " + i + "세이고");
19:         System.out.println("몸무게는 " + d + "Kg 입니다"); // 저장된 값을 출력
20:     }
21: }
```



### ● 프로그래밍 오류

- 능숙한 프로그래머도 오류를 피해갈 수는 없다

### ● 오류의 종류

- 컴파일 시간에 발생하는 **구문(Syntax) 오류** : 이클립스에 의해 오류가 자동 표시된다
- 실행 시 발생하는 **실행 시간(Runtime) 오류** : 프로그램 실행 시 JVM에 의해 오류를 발생시킨다
- 논리적 문제로 발생하는 **논리(Logic) 오류** : 프로그래머에 의해 발견되어야 수정할 수 있다



### ● 프로그래밍 오류 : 구문 오류 – 예제 2.6

예제 2.6

SyntaxErrorTest.java

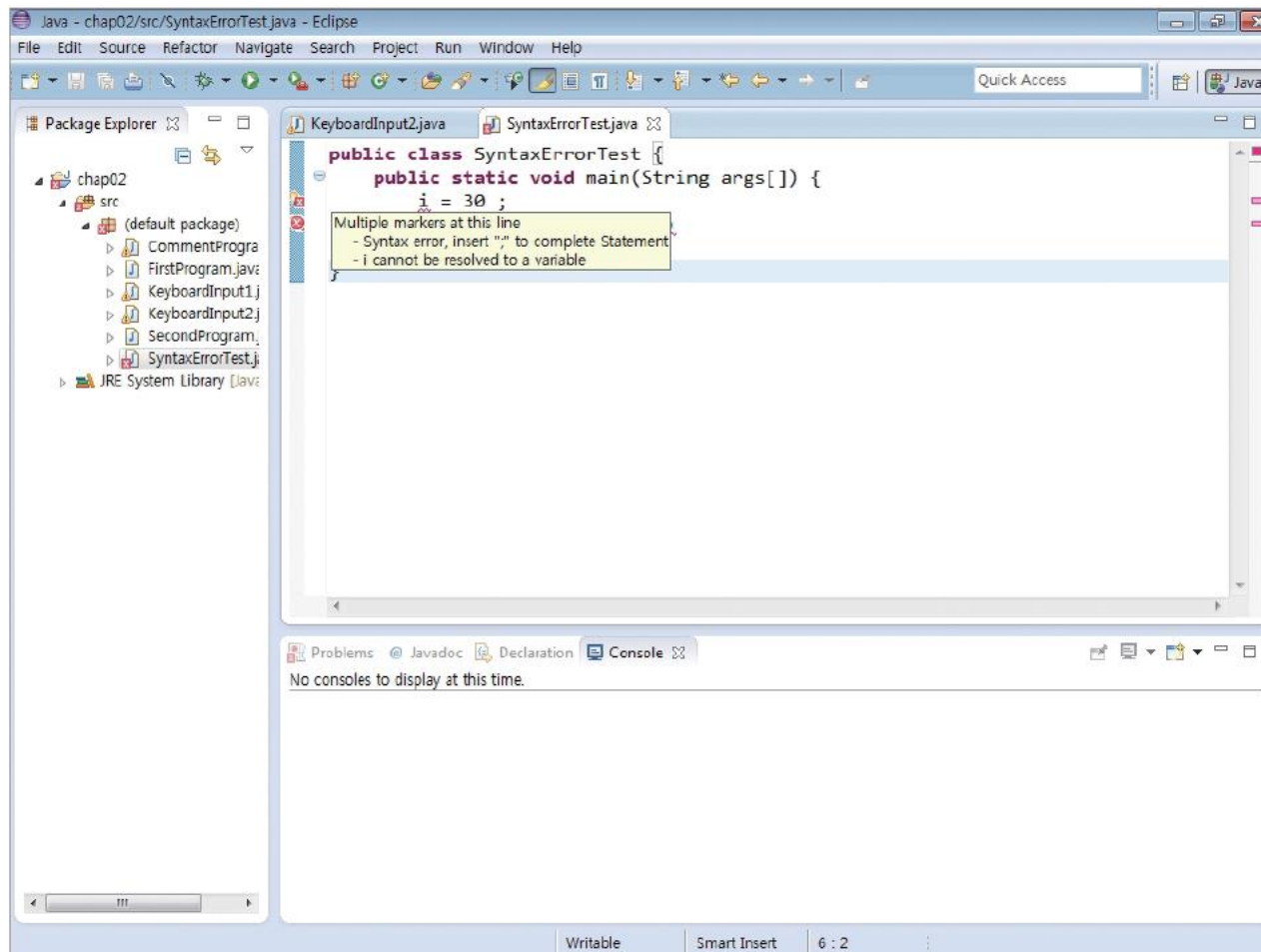
```
01: public class SyntaxErrorTest {
02:     public static void main(String args[]) {
03:         i = 30 ; ← 변수를 선언하지 않고 사용
04:         System.out.println( i ) ← 문장 종료 기호가 없음
05:     }
06: }
```

#### 실행 결과 오류 메시지 출력

```
Exception in thread "main" java.lang.Error: Unresolved compilation problems:
    i cannot be resolved to a variable
    i cannot be resolved to a variable
    Syntax error, insert ";" to complete Statement

    at SyntaxErrorTest.main(SyntaxErrorTest.java:3)
```

- 프로그래밍 오류 : 구문 오류(구문 오류는 이클립스에 적색으로 표시된다)



## ● 프로그래밍 오류

- 0으로 나누거나, 배열의 첨자가 벗어났거나, 사용자의 입력 값이 잘못 입력 되었거나 등등
- 실행 시간 오류는 프로그램을 비 정상적으로 종료 시킨다

### 예제 2.7

RuntimeErrorTest.java

```
01: public class RuntimeErrorTest {  
02:     public static void main(String args[]) {  
03:         int i = 3 / 0 ;  
04:         int j = 5 / 0 ;  
05:         System.out.println(i + j);  
06:     }  
07: }
```

## 3-3 주석문과 프로그램 오류

### ● 실행 시간 오류 메시지 출력

- 3번 라인에서 실행 시간 오류 발생하여 프로그램 종료되고 아래와 같은 메시지 출력

실행 결과 실행 시간 오류 메시지 출력

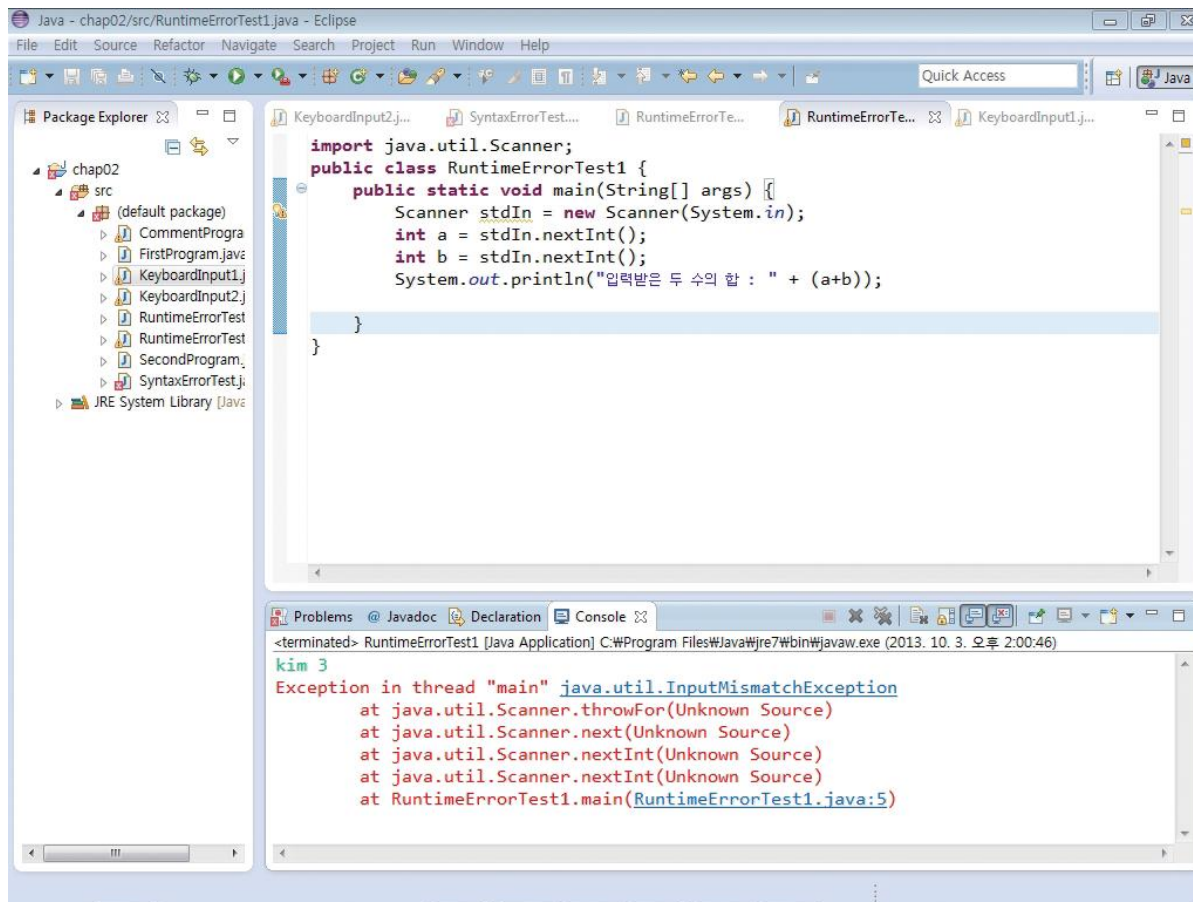
```
Exception in thread "main" java.lang.ArithmeticException: / by zero  
at RuntimeErrorTest.main(RuntimeErrorTest.java:3)
```

- 3번 라인을 수정한 다음 다시 실행시키면 다시 4번 라인에서 실행 시간 오류 메시지 출력

실행 결과 3번 라인의 오류를 수정한 다음 실행

```
Exception in thread "main" java.lang.ArithmeticException: / by zero  
at RuntimeErrorTest.main(RuntimeErrorTest.java:4)
```

#### ● 실행 시간 오류 : 이클립스의 실행 시간 오류





### ● 논리 오류

- 프로그램 작성자의 의도와 다른 결과를 나타내는 오류
- 프로그램의 실행이 종료되어도 나타나지 않기 때문에 오류로 취급되지 않는 경우도 있지만, 프로그램을 작성한 사람이 의도한 결과를 출력하지 않았기 때문에 오류로 취급
- 논리 오류는 프로그램 작성자나 수정하는 사람에 의해 발견되어야 한다

## ● 논리 오류의 예 : 예제 2.8

예제 2.8

LogicError.java

```
01: public class LogicErrorTest {  
02:     public static void main(String args[]) {  
03:         int i = 300 ;  
04:         int j = 500 ;  
05:         j += i + j ;  
06:         System.out.println("합계는 = " + j) ;  
07:     }  
08: }
```

### » 설명

05      작성자의 의도는  $j = i + j$  인데 잘못 기술됨

### 실행 결과

합계는 = 1300

### ● 자바 프로그램의 형태

- ① 자바 응용 프로그램
- ② 자바 애플릿
- ③ 자바 서블릿(Servlet)
- ④ JSP(Java Server Page)
- ⑤ 자바 빈스(Beans)

### ● 학습을 위한 준비

#### ① 표준 출력

- `System.out.print()` : 내용을 모니터로 출력하고 줄을 바꾸지 않습니다.
- `System.out.println()` : 내용을 모니터로 출력하고 줄을 바꿉니다.

#### ② 키보드로부터의 입력

- `java.util.Scanner` 클래스를 이용하여 키보드로부터 입력을 받습니다. 한 줄에 여러 개의 데이터를 입력받기 위해서는 공간을 구분자로 사용합니다.



- next() : 문자열을 입력받는 메소드
- nextInt() : 정수를 입력받는 메소드
- nextDouble() : 실수를 입력받는 메소드

### ③ 자바의 주석문 : 자바에서는 3가지 형태의 주석문이 제공되고 있습니다.

- /\* ..... \*/ : 여러 줄의 문장을 주석으로 처리할 수 있습니다.
- // ..... : 한 줄의 문장을 주석으로 처리합니다.
- /\*\* ..... \*/ : 여러 줄의 문장을 주석으로 처리하고, javadoc 도구에 의해 사양서를 만들 수 있는 주석입니다.

### ④ 자바의 오류

- 구문 오류 : 프로그램 구조를 잘못 사용할 때 발생하는 오류로서 쉽게 발견될 수 있습니다.
- 실행 시간 오류 : 실행 시간에 발생하는 오류로서 프로그램이 비정상적으로 종료될 수 있는 오류입니다.
- 논리 오류 : 프로그램의 컴파일과 실행이 오류 없이 수행되지만, 실행 결과가 작성자의 의도와는 다른 결과를 출력하는 오류입니다. 논리 오류는 발견과 수정이 쉽지 않은 오류입니다.