한국방송통신대학교 프라임칼리지

Java 프로그래밍 기초

김희천 교수

1차시. Java 언어 개요

들어가기

■ 학습개요

- 본 장에서는 Java 언어를 소개한다. Java 언어의 발전 과정을 알아보고 Java 언어의 특징을 살펴본다. 바이트코드, Java 플랫폼, 자바 가상기계 등의 새로운 개념을 이해하도록 한다. Java 프로그래밍을 위한 준비 작업으로 개발 환경을 설치하고 간단한 예제 프로그램을 통해 실행 방법을 확인하도록 한다.

■ 학습목표

1	Java 언어의 특징을 나열하고 설명할 수 있다.
2	Java 플랫폼의 구성을 설명할 수 있다.
3	Java 개발 환경을 설치할 수 있다.
4	프로그래밍 작업을 수행할 수 있다.

■ 주요용어

	주요용어	의미
1	바이트코드	Java 소스 코드를 컴파일하면 확장자가 '.class'인 파일이 만들어지며 이것을 '바이트코드' 또는 '클래스 파일'이라한다. Java VM의 기계어라고 할 수 있다.
2	Java 플랫폼	Java VM과 Java API로 구성된다. Java VM은 실행환경을 제공하는 가상 기계로 여러 다른 하드웨어에 설치될 수 있다. Java API는 프로그램 작성을 위해 제공되는 소프트웨어 요소이다.
3	Eclipse	공개 소프트웨어로 Java 언어를 포함하여 다양한 언어를 지원하는 통합 개발 환경이다.

학습하기

1. Java 언어 소개

1.1. Java 언어의 역사

- 창시자는 Sun Microsystems의 제임스 고슬링
- 1990년 그린 프로젝트와 Oak 언어
- 1995년 Java와 HotJava 발표(웹의 확산)
- 1996년 1월 JDK1.0 발표
- 2009년 Oracle이 Sun을 인수
- 2021년 Java SE 16 (JDK 16)

캘리포니아에 있는 선 마이크로시스템즈사에서 시작한 자바의 개발은 1991년 시작된 월드 와이드 웹의 개발과 관련이 있다. 자바의 개발자인 제임스 고슬링(James Gosling)이 포함된 1990년 "그린 프로젝트" 팀의 초기 개발 목적은 소비자용 전자 제품의 제어박스를 개발하는 것이었다. 여기서 필요한 것은 플랫폼과 무관한 코드를 작성하고 모든 CPU에서 실행될 수 있는 소프트웨어를 제작하는 것이었다. 다양한 하드웨어 장비들에 대하여 작고, 빠르고, 효과적이고, 이식성이 좋은 언어를 만들어야 했고 그리하여 C++를 포기하고 Oak (나중에 Java로 개명됨)라는 언어를 개발하였다.

이때 웹이라는 새로운 네트워크 환경이 주목받았는데, 다양한 컴퓨터를 연결하고 있는 웹에는 플랫폼에 독립적인 자바와 같은 언어가 이상적이라고 생각되었던 것이다. 이러한 개념을 바탕으로 자바 언어를 웹 환경에서 실행 가능한 프로그램들을 전달하기 위한 이상적인 언어로 만들었고, 다른 시스템 환경 상에서도 쉽게 사용할 수 있고, 이식성이 뛰어난 프로그램을 제작하기 위한 범용 개발 언어로 발전되었다

1.2. Java 언어의 특징

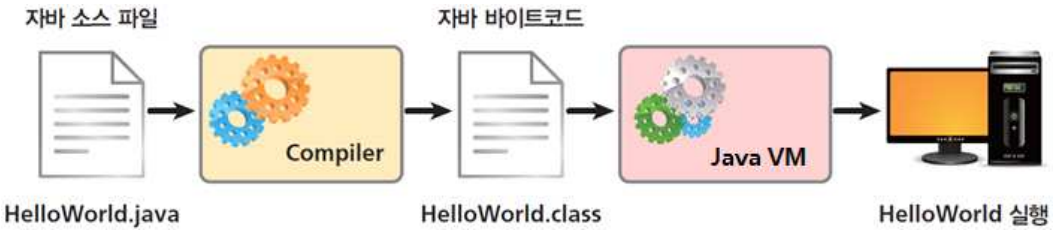
- C/C++ 언어와 유사하나 단순함
- 문법적으로 C/C++과 유사하나 배우기 쉽다. C/C++ 언어와 다르게 포인터와 전처리기를 사용하지 않으며 메모리 할당 후의 제거 작업이 필요하지 않다.
- 플랫폼에 독립적인 언어
- 바이트코드는 아키텍쳐 중립적인 이진 파일이다. Java VM(가상 기계)이 설치된 다양한 하드웨어와 운영체제에서 똑같은 바이트코드를 실행시킬 수 있다.
- 완전한 객체지향 언어
- 완전한 객체지향 언어로 객체지향 개념의 장점을 모두 포함한다.
- 분산 처리 기능(웹/네트워크 프로그래밍이 용이)
- 웹이나 네트워크 기반의 프로그래밍 또는 분산 처리를 위한 다양한 기능을 제공한다. 예를 들어 인터넷 프로토콜 TCP/IP, 웹 서비스에 사용되는 HTTP와 관련된 클래스 라이브러리를 제공한다.
- 멀티 스레딩 지원
- 하나의 프로그램에서 여러 스레드가 실행될 수 있다. 각 스레드는 마치 독립적 프로그램처럼 수행되어 사용자와의 상호작용을 극대화시킨다.
- 동적 실행
- 실행 시간에 동적으로 필요한 클래스를 로드할 수 있다. 동적 클래스를 로드하기 위해 네트워크를 사용할 수 있다.
- 강건함(엄격한 자료형의 검사, 예외 처리 기능 제공)
- 컴파일 시간에 강력한 검사 기능을 제공하여 신뢰성 높은 소프트웨어를 만들 수 있다. 포인터 타입과 포인터 연산이 없고 메모리의 자동 수집 기능을 제공함으로써 오류를 줄인다. 예외처리 기능을 통해 실행 중 발생하는 오류를 처리할 수 있다.
- 보안성
- 바이트코드는 Java VM에서 실행되므로 문제가 있더라도 운영체제나 컴퓨터 자원에 직접적인 영향을 주지 않도록 할 수 있다.

1.3. Java 프로그램의 종류

- 데스크탑 어플리케이션
- 웹 어플리케이션
- 기업체 어플리케이션
- 모바일 어플리케이션

2. Java 플랫폼과 프로그램의 실행

2.1. Java 프로그램의 실행



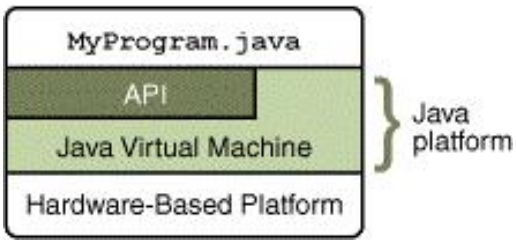
- 소스 프로그램의 확장자는 .java
- 바이트코드의 확장자는 .class
- 바이트코드는 Java VM에 의해 실행됨

2.2. Java 플랫폼

- 일반적 플랫폼

- 플랫폼이란 프로그램이 실행되는 하드웨어 또는 소프트웨어 환경을 말한다.
- 대부분의 플랫폼은 기반이 되는 하나의 하드웨어와 거기에 설치된 운영체제를 합친 조합을 말한다.

■ Java 플랫폼



- Java 플랫폼은 소프트웨어 플랫폼을 말하며 여러 다른 하드웨어에 설치될 수 있다.
- Java 플랫폼은 Java VM과 Java API로 구성된다.
- Java VM은 실행 환경을 제공하는 가상의 기계로 Java 플랫폼의 기초가 되며 여러 하드웨어 플랫폼에 설치될 수 있다.
- Java API는 프로그램에 사용되도록 이미 만들어져 제공되는 것으로 유용한 기능을 제공하는 소프트웨어 컴포넌트들이다.

자바로 작성된 후 컴파일된 프로그램(클래스 파일, 바이트코드)의 경우에는 운영체제가 실행환경이 되는 것이 아니라 별도의 자바 플랫폼이 자바 프로그램의 실행 환경이 된다. 다시 말해서 자바 프로그램이 구동되기 위해서는 운영체제 외에 해당 운영체제에 적합한 별도의 자바 플랫폼이 설치되어 있어야만 한다는 것이다. 예를 들어 Windows 환경에서 자바 프로그램이 동작하기 위해서는 Windows 환경을 위해 만들어진 자바 플랫폼이 존재해야 하고, LINUX 환경에서 자바 프로그램이 동작하기 위해서는 LINUX 환경을 위한 자바 플랫폼이 존재해야 한다.

■ Java 플랫폼의 종류

- Java SE(데스크탑 응용)

- 자바 응용 프로그램의 개발과 구현 기술
▪ Java EE(기업체 응용, 웹 응용)
- 기업체의 대규모 응용 프로그램을 개발하기 위한 표준 플랫폼
▪ Java ME(모바일이나 임베디드 응용)
- 모바일 전화기 및 PDA, TV 셋탑박스, 이동 차량에 부착된 각종 장치 및 여러 임베디드 장치를 위한 자바 플랫폼

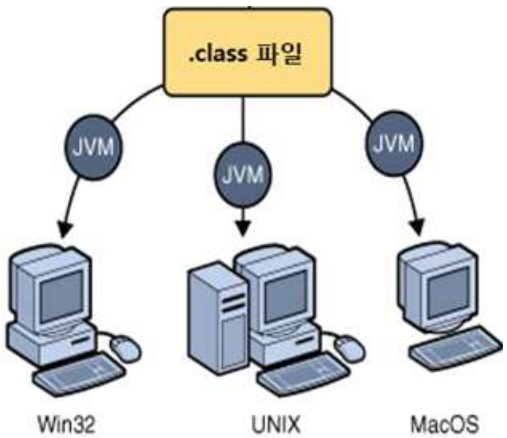
현재 자바에서는 일반적인 개인용 컴퓨터 시스템, 서버 컴퓨터 시스템, 모바일 단말기, 웹 브라우저 등과 같은 거의 모든 컴퓨팅 환경 및 운영체제를 위한 자바 플랫폼을 제공하고 있다. 자바 플랫폼을 표기할 때는 자바의 주요 버전 번호와 함께 표기한다.

▪ Java Card
- 자바로 동작하는 IC 카드에 적용되는 플랫폼

■ Java VM(JVM: Java Virtual Machine)
▪ Java 프로그램의 실행 환경을 제공하는 가상 기계
▪ 프로그램의 실행에 필요한 모든 사항을 관리
▪ new 연산자에 의해 할당되었던 메모리를 자동으로 수집한다(가비지 컬렉션).

■ Java API
▪ 프로그램의 개발에 필요한 클래스 라이브러리
▪ Java API는 관련이 있는 클래스와 인터페이스들을 묶은 패키지들로 구성된다.
▪ 파일 관리를 위해 폴더를 사용하는 것처럼 클래스를 관리하기 위해 패키지가 존재한다.
▪ 패키지들은 계층 구조로 분류되며 상위 패키지와 하위 패키지를 구분하기 위해 표기법을 사용한다.
▪ 패키지 java.lang은 최상위 패키지 java에 포함되어 있는 서브 패키지 lang을 의미한다.
▪ java.util.Vector는 java.util 패키지에 포함되어 있는 클래스 Vector를 의미한다.

2.3. 바이트코드



▪ 소스 프로그램을 컴파일한 결과물
▪ 확장자가 .class이며 클래스 파일이라고도 함

- Java 플랫폼에서 실행 가능한 중간 코드
- "write once, run anywhere"
- 한번 작성된 바이트코드는 어떤 자바 플랫폼에서도 실행 가능함

확장자가 .java인 자바 소스를 컴파일하면, 확장자가 .class인 새로운 파일이 만들어지며 이것을 클래스 파일 또는 바이트 코드라고 한다. "write once, run anywhere."라는 표현이 있는데, 이것은 자바 소스를 컴파일해서 나오는 코드가 특정 플랫폼에만 적용되는 것이 아니고 어디서든 실행될 수 있다는 의미이다.

바이트 코드는 자바 플랫폼에서 실행 가능한 자바 플랫폼의 기계어 코드라 할 수 있다. 윈도우, 유닉스, 맥OS 또는 인터넷 브라우저에서 자바 플랫폼만 있다면 바이트코드를 해석할 수 있다.

3. Java 개발 환경 준비

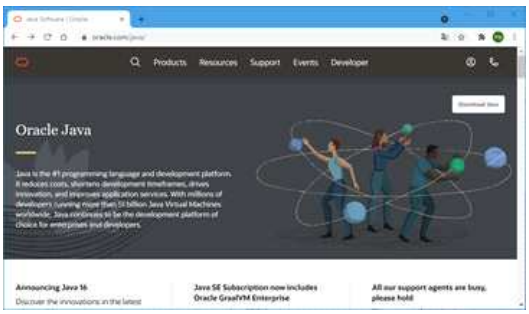
3.1. Java 프로그래밍을 위한 준비 작업

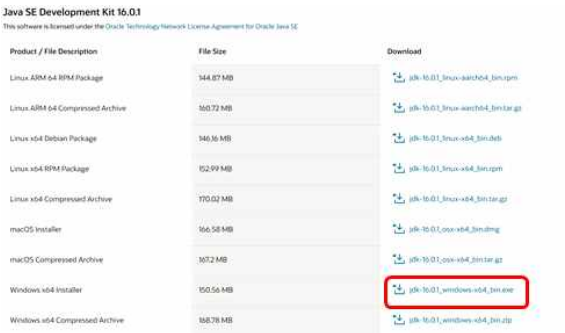
- JDK의 설치
- 컴파일/실행/디버깅 도구, JVM, API를 포함
- '오라클 JDK'와 '오픈-JDK'가 있음
- https://www.oracle.com/java와 http://jdk.java.net
- Eclipse의 설치
- 개발 도구를 사용하는 것이 편리함

자바 소스 코드의 컴파일 및 실행에 필요한 자바 컴파일러, 자바 VM, 자바 기본 API들을 한데 묶어서 Java Software Development Kit라고 하며 줄여서 간단히 JDK라고도 하는데 JDK는 자바 공식 홈페이지인 http://java.sun.com에서 무료로 다운로드 받을 수 있으며 컴퓨팅 환경 및 각종 운영체제별 버전이 존재한다. 여기서는 Java SE 16에 포함되어 있는 Windows용 JDK의 설치 방법을 살펴본다.

3.2. JDK의 설치

- Java 홈페이지에 접속
- https://www.oracle.com/java/
- Java SE 설치파일을 다운받음





- Java Standard Edition의 Development Kit를 다운받아 설치한다.
- Java 프로그램의 개발 도구(javac.exe와 java.exe 등), Java VM 및 Java의 기본 API를 묶어서 Java SDK(Software Development Kit)라고 하며 간단히 JDK라고도 한다.
- 자신이 사용하는 운영체제와 맞는 것을 선택
- https://www.oracle.com/java/technologies/javase-jdk16-downloads.html

▪ 설치 파일을 실행

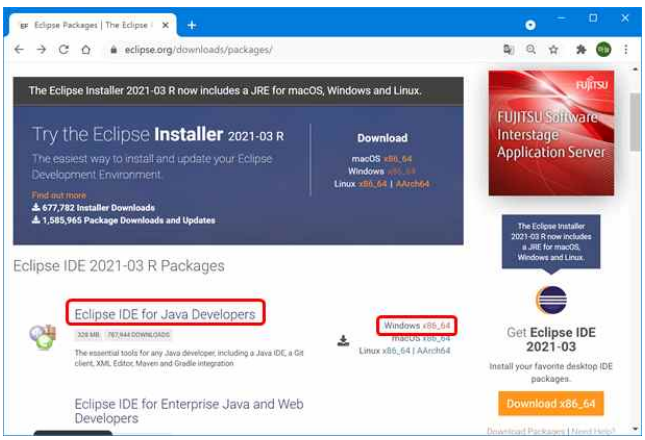
- jdk-16.0.1_windows-x64_bin.exe

3.3. Eclipse의 설치

▪ Eclipse 홈페이지에 접속

- https://www.eclipse.org

▪ "Eclipse IDE for Java Developers" 압축파일을 다운받음



- https://www.eclipse.org/downloads/packages/
- eclipse-java-2021-03-R-win32-x86_64.zip
- 파일 이름에 있는 mar를 코드명이라 하면 버전을 대신한다.

▪ 적당한 곳에 압축파일을 풀어줌

- 압축 파일에 있는 모든 내용은 이미 eclipse라는 폴더 안에 포함되어 있다.
- 따라서 D:₩에 압축을 풀면 D:₩eclipse 폴더가 생성된다.

4. HelloWorld 프로그램

4.1. 예제 프로그램

▪ 대소문자는 구분됨

▪ public class A가 존재하면 파일 이름은 A.java

- 하나의 소스 파일에 여러 클래스 정의가 포함될 수 있으나, public 클래스는 많아야 하나가 존재할 수 있으며, 파일의 이름은 public 클래스의 이름과 같아야 한다.
- 아래 예제의 파일 이름은 HelloWorld.java이다.

▪ System.out.println(" ... ")는 따옴표로 둘러싸여 있는 문자열을 화면으로 출력한다.

- main 함수에서 프로그램이 시작됨
- main 함수의 형식을 기억해야 함
- main 함수는 반드시 public static void이어야 하며 String 배열을 인자로 가져야 한다.

```
public class HelloWorld {

	public static void main(String[ ] args) {
		// TODO Auto-generated method stub
		System.out.println("HelloWorld!");
	}

}
```

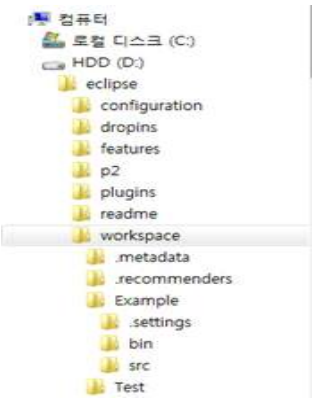
■ 프로그램 작성 시 나타나는 흔한 오류
- 괄호나 중괄호 또는 큰따옴표 등이 빠지면 오류가 발생
- 키워드 static void 등이 잘못 입력되거나 빠지면 오류가 발생
- String 또는 System 등에서 S를 소문자로 잘못 입력한 경우 오류가 발생
- 문장 끝에 세미콜론 ; 이 빠진 경우 오류가 발생
- public class에서 순서가 바뀌면 오류가 발생. class 다음에 이름이 나와야 함.

4.2. Eclipse 사용하기

- eclipse.exe 실행
- 작업공간(workspace) 지정
- 자바 프로젝트(Java Project) 생성
- 클래스 생성
- 프로그램 작성
- 프로그램 실행

4.3. 작업공간 지정과 프로젝트 생성

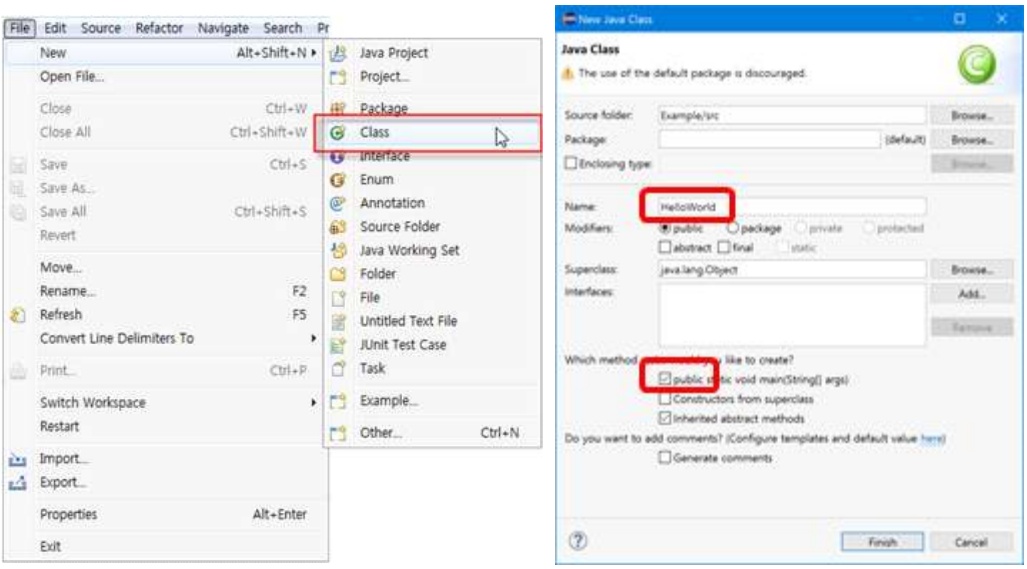
- 작업공간
- 개발 과정에서 만들어지는 여러 파일이 저장되는 장소
- 아래에서 작업공간은 D:₩eclipse₩workspace



▪ 프로젝트 생성
- 메뉴 [File/New/Java Project] 선택
- 프로젝트 이름을 입력하고 [Finish], module-info.java를 [Don't create]할 것

4.4. 클래스 생성

▪ 클래스를 작성하면 파일로 만들어짐
- 메뉴 [File/New/Class]를 선택
- 클래스 이름을 HelloWorld로 입력하고
- 체크박스 [public static void main(String[ ] args)]를 체크
- [Finish] 버튼을 선택

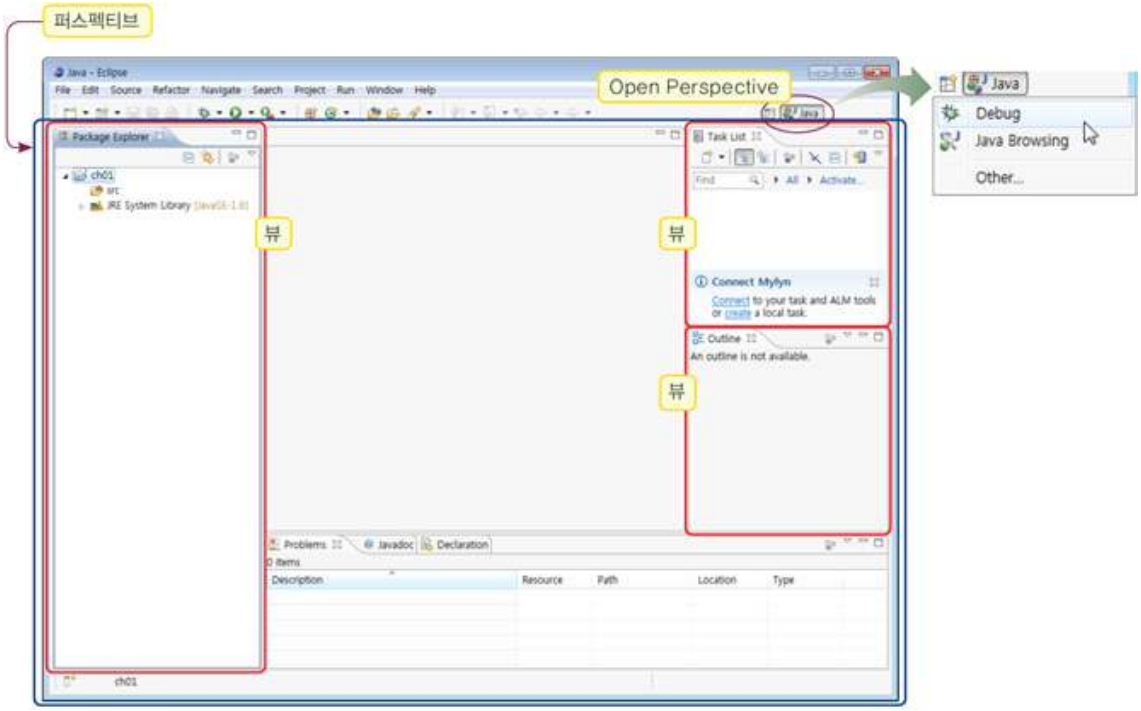


4.5. 프로그램 작성과 실행

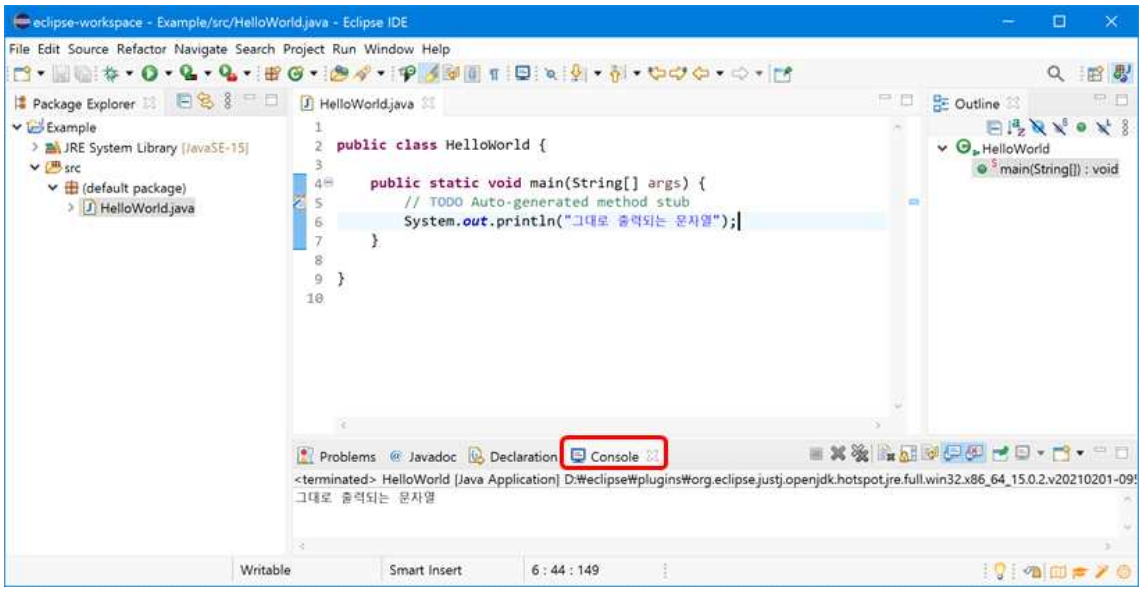
▪ 자동 생성되는 프로그램에 필요한 부분을 추가함
- TODO가 나오는 주석(//) 부분에 아래를 추가함
- System.out.println("그대로 출력되는 문자열");

▪ 실행하려면 메뉴 [Run/Run As/Java Application]를 선택
- 단축 아이콘 ▶ 또는 [ctrl]+[F11]으로도 실행 가능
- 오류가 없으면 맨 아래 "Console View"에 결과가 출력된다.

▪ "Eclipse IDE for Java Developers"을 다운받아 설치했으므로 전체 화면은 "Java Perspective"를 보여준다. 퍼스펙티브를 구성하는 작은 창을 뷰(View)라고 한다.



- 사각형 모양의 구역을 "뷰"라 하며 가장 왼쪽 뷰는 "Package Explorer View"이다.
- 아래는 "Java Perspective"를 보여주는 Eclipse 화면이다.



■ 이클립스 단축키

- [Ctrl]+[Shift]+[L]
  - 다양한 단축 키 참조 방법 나열
  - 주 메뉴에서 [Help/Key Assist...]의 선택으로도 가능
- [Ctrl] + [Space] 키 활용
  - 도움 코드를 표시
  - 표준출력을 위하여 간단히 sysout을 입력한 후 [Ctrl]+[Space] 단축 키를 누르면 문장

System.out.println();이 완성됨

- [Ctrl]+[Shift]+[F] 단축 키
- 소스의 들여쓰기를 적용

연습문제

※ 선다형, O.X형을 혼합하여 문항 별 출제 문제와 보기, 정답, 부연설명을 기입해 주십시오(선다형2문제, 선다형1문제와 O.X형1문제, O.X 2문제 중 택). **선다형 보기 개수는 3~5개 범위에서 자유롭게 기술이 가능하십니다.**

(문제1) Java 언어의 특징을 모두 고르시오.
(보기)
a.플랫폼에 독립적
b.예외처리 기능
c.멀티스레딩 지원
d.포인터의 사용
e.자료형에 대한 유연한 검사
(정답) a, b, c
(부연설명) Java 언어는 포인터 타입을 포함하지 않으며 또한 컴파일 시간에 자료형에 대한 엄격한 검사를 수행한다.

(문제2) Java 애플리케이션을 실행하기 위해 필요한 main 함수의 형식은 무엇인가?
(보기)
a. public void main(String args[ ])
b. static void main(String args[ ])
c. public static void main(String args)
d. public static void main(String args[ ])
(정답) d
(부연설명) 애플리케이션 실행의 시작을 의미하는 main 함수의 형식은 보기 d와 같아야 한다. 또는 public static void main(String[ ] xxx)와 같이 작성할 수도 있다.

정리하기

1	Java는 플랫폼에 독립적인 객체지향 프로그래밍 언어이다.
2	Java에서 자료형에 대한 검사가 엄격하다.
3	Java는 예외 처리와 멀티 스레딩을 지원한다.
4	Java 소스를 컴파일하면 클래스별로 확장자가 .class인 파일이 만들어지며 이 파일을 바이트코드라 한다.
5	Java 가상기계는 바이트코드를 해석하여 실행시킬 수 있다.
6	Java 프로그래밍을 위해 JDK를 설치해야 한다.
7	Eclipse는 다양한 언어를 지원하는 통합 개발 환경이다.

※ 참고자료

김희천, 정재헌 저, Java 프로그래밍, 방송대 출판문화원, 2017
강환수, 조진형 저, 절대 JAVA, 인피니티북스, 2014
https://docs.oracle.com/javase/tutorial/ (Java 튜토리얼 페이지)

※ 다음 차시에 대한 안내

다음 시간에는

2차시 Java 기본 문법(1)에 관해 학습하겠습니다.

다음 시간에 배울 내용을 미리 생각해 봅시다.

-Java 프로그램의 기본 구성단위는 무엇인가?

-프로그램에서 문장, 블록의 기능은 무엇인가?

-데이터의 종류로는 어떤 것이 있는가?