

자바 시작

01

컴퓨터, 소프트웨어,
프로그래밍 언어의 이해

02

자바의 출현배경과
플랫폼 독립성 개념 이해

03

자바 응용프로그램의
종류와 특징 이해



컴퓨터와 소프트웨어



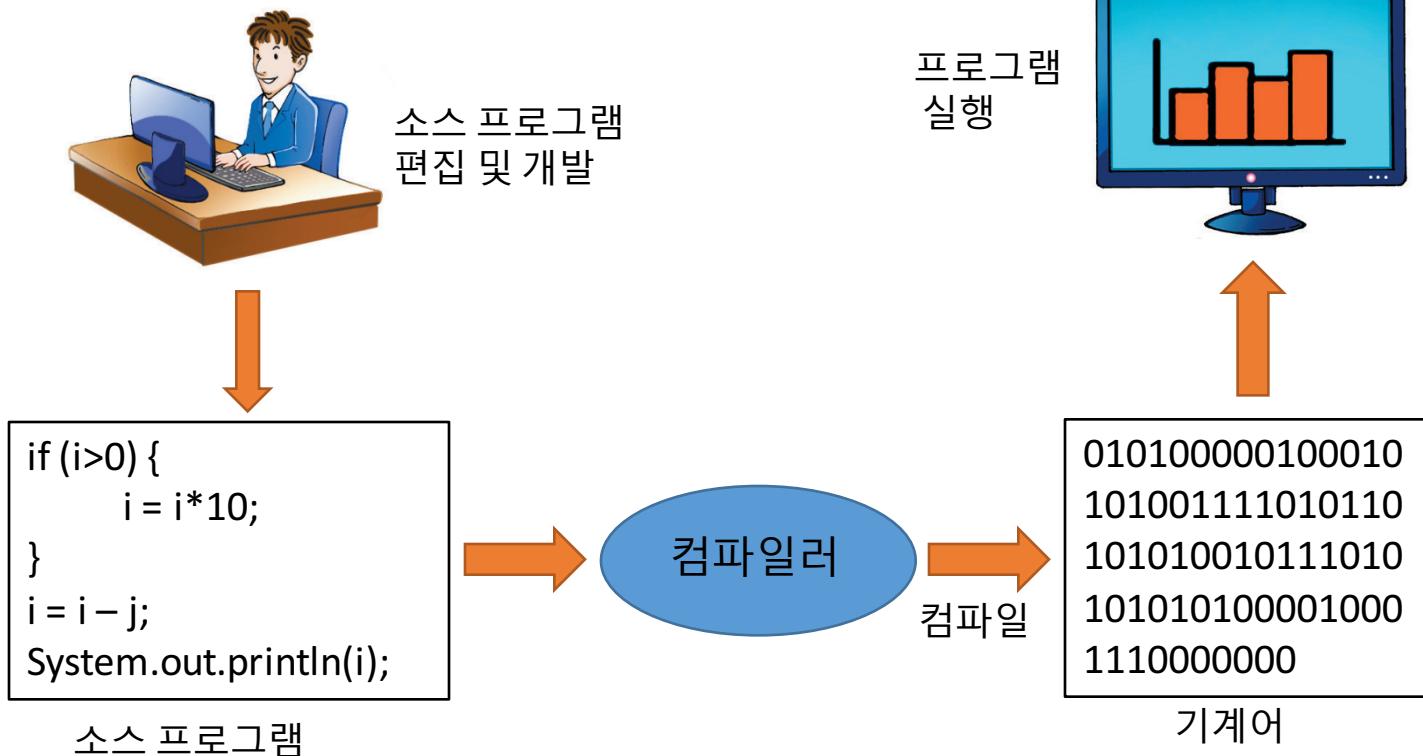
컴퓨터와 프로그래머, 소프트웨어의 관계는
만능 요리 기계, 요리설계사와, 요리순서와 같다.

프로그래밍 언어

- **프로그래밍 언어**: 프로그램 작성 언어
 - **기계어(machine language)**
 - 0, 1의 이진수로 구성된 언어
 - 컴퓨터의 CPU는 기계어만 이해하고 처리 가능
 - **어셈블리어**
 - 기계어 명령을 ADD, SUB, MOVE 등과 같은 표현하기 쉬운 상징적인 단어인 니모닉 기호(mnemonic symbol)로 일대일 대응시킨 언어
 - **고급언어**
 - 사람이 이해하기 쉽고, 복잡한 작업, 자료 구조, 알고리즘을 표현하기 위해 고안된 언어
 - Pascal, Basic, C/C++, Java, C#

프로그래밍과 컴파일

- 소스 : 프로그래밍 언어로 작성된 텍스트 파일
- 컴파일 : 소스 파일을 컴퓨터가 이해할 수 있는 기계어로 만드는 과정
 - 자바 : .java -> .class
 - C : .c -> .obj-> .exe
 - C++ : .cpp -> .obj -> .exe



자바 시작

01

컴퓨터, 소프트웨어,
프로그래밍 언어의 이해

02

자바의 출현배경과
플랫폼 독립성 개념 이해

03

자바 응용프로그램의
종류와 특징 이해



자바의 태동

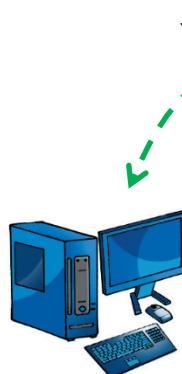
- 자바 역사
 - 1991년 그린 프로젝트(Green Project)
 - 선마이크로시스템즈의 제임스 고슬링(James Gosling)에 의해 시작
 - 가전 제품(interactive tv)에 들어갈 소프트웨어를 위해 개발
 - 1995년에 자바 발표
 - 2009년에 선마이크로시스템즈를 오라클이 인수
 - 테스크톱 PC, 웹, 임베디드, 안드로이드 애플리케이션 개발에 사용 됨
- 개발 배경 및 목적
 - 기존 언어의 플랫폼 호환성 문제
 - 기존 언어로 작성된 프로그램은 PC, 유닉스, 메인 프레임 등 플랫폼 간에 호환성 없음
 - 플랫폼 독립적인 언어 개발
 - 모든 플랫폼에서 호환성을 갖는 프로그래밍 언어 필요

기존 언어의 플랫폼 종속성



인텔 CPU를 가진
리눅스 환경에서
개발

C/C++
응용 프로그램



Intel CPU + 리눅스

실행되지
않음



Apple 사의 MAC PC



플랫폼 = 하드웨어 플랫폼 + 운영체제 플랫폼

프로그램의 플랫폼 호환성 없는 이유

- 기계어가 CPU마다 다름
- 운영체제마다 API 다름
- 운영체제마다 실행파일 형식 다름

실행
되
지
않
음



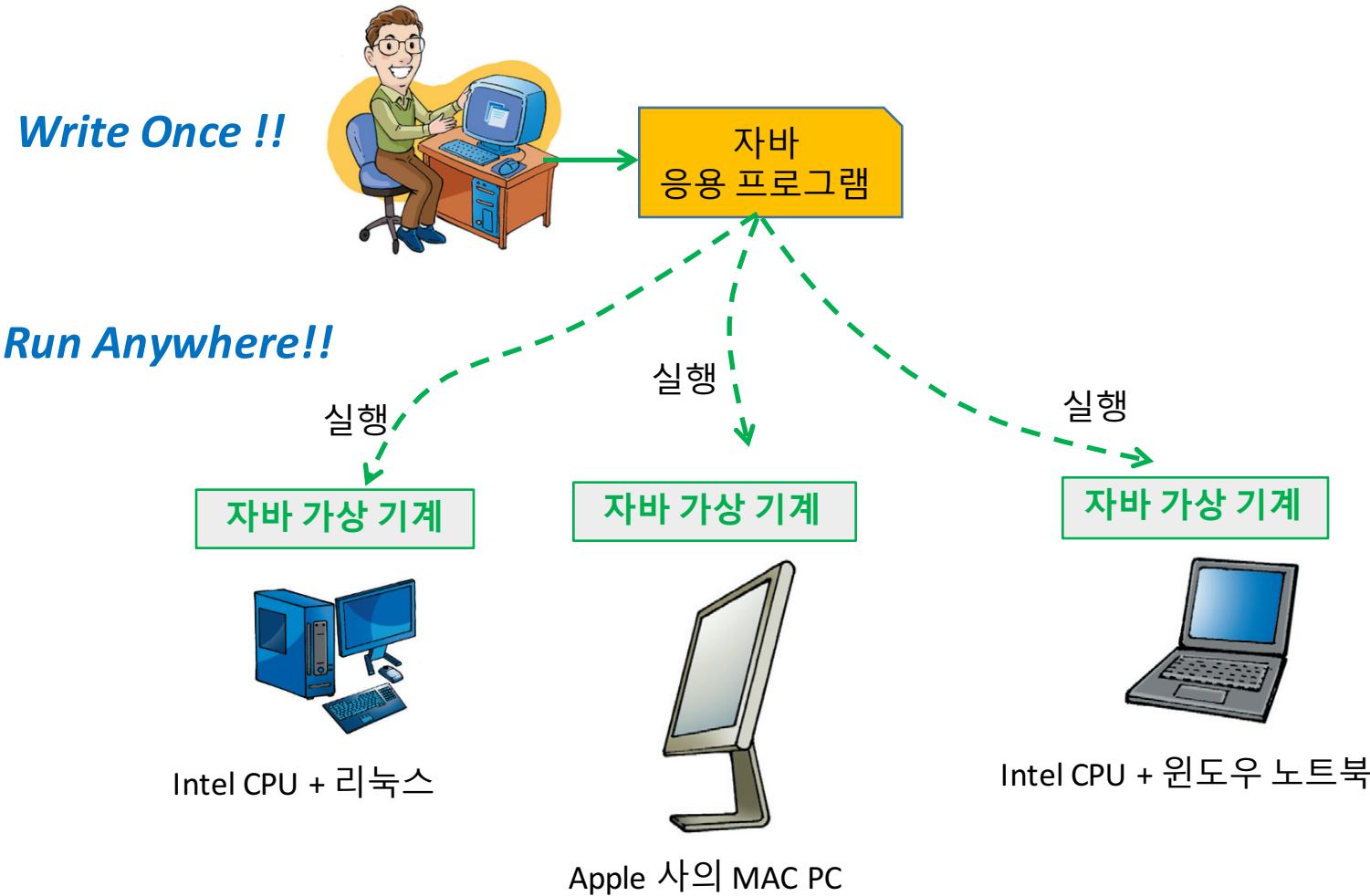
Intel CPU + 윈도우 노트북



자바의 플랫폼 독립성, WORA

- WORA(Write Once Run Anywhere)
 - 한번 작성된 코드는 모든 플랫폼에서 바로 실행되는 자바의 특징
 - C/C++ 등 기존 언어가 가진 플랫폼 종속성 극복
 - OS, H/W에 상관없이 자바 프로그램이 동일하게 실행
- WORA를 가능하게 하는 자바의 특징
 - 바이트 코드(byte code)
 - 자바 소스를 컴파일한 목적 코드
 - 특정 CPU에 종속적이지 않은 중립적인 코드
 - JVM에 의해 해석되고 실행됨
 - 자바가상기계-JVM(Java Virtual Machine)
 - 자바 바이트코드를 실행하는 가상의 기계
 - 자바 바이트 코드를 해당 플랫폼에 맞는 기계어 코드로 변환하여 실행하는 소프트웨어

자바의 플랫폼 독립성



자바 응용프로그램의 실행

* 자바는 링크 과정 없음



자바 프로그래밍

Draw.java

Hello.java

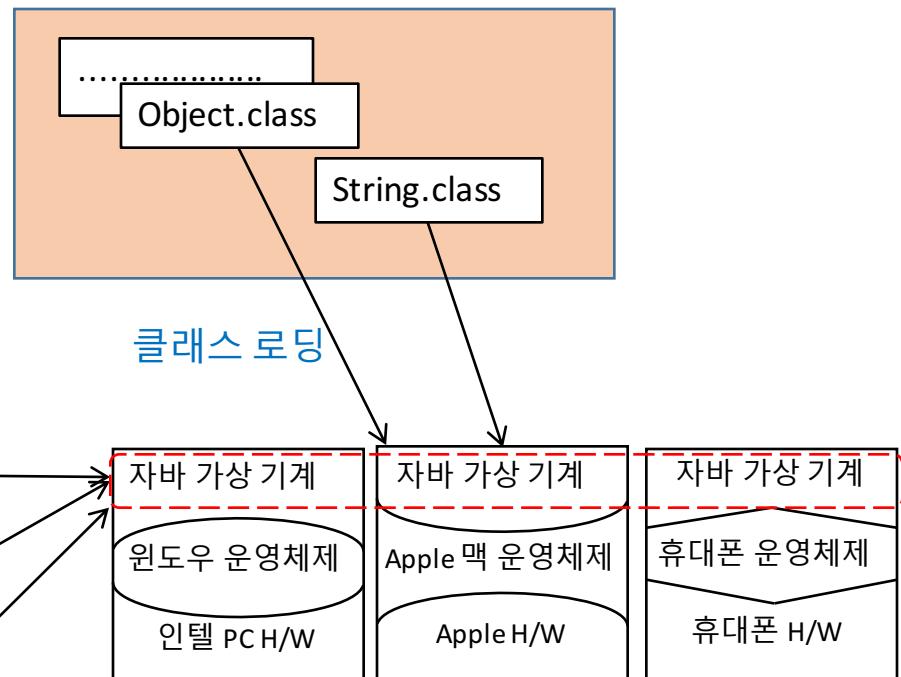
Shape.java

(소스 코드)

Draw.class
Hello.class
Shape.class
(바이트 코드)

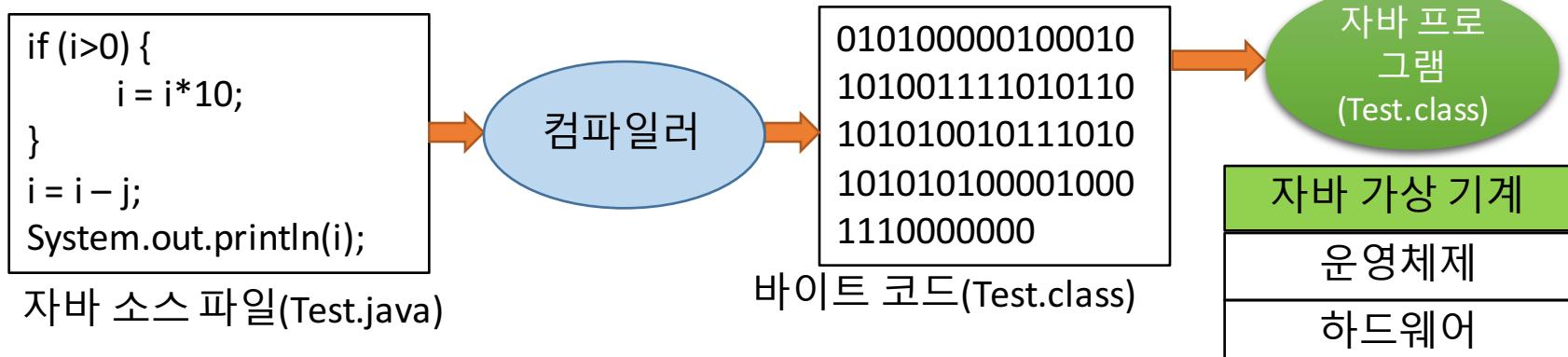
자바 컴파일러

실행에 필요한 자바 클래스 라이브러리(JDK APIs)

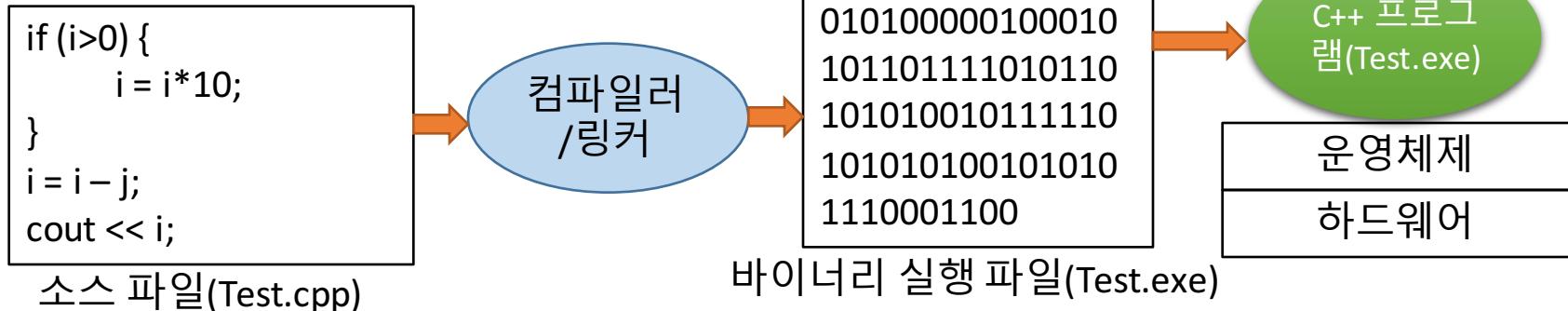


자바와 타언어(C/C++)의 실행 차이

- 자바



- C/C++



자바 시작

01

컴퓨터, 소프트웨어,
프로그래밍 언어의 이해

02

자바의 출현배경과
플랫폼 독립성 개념 이해

03

자바 응용프로그램의
종류와 특징 이해

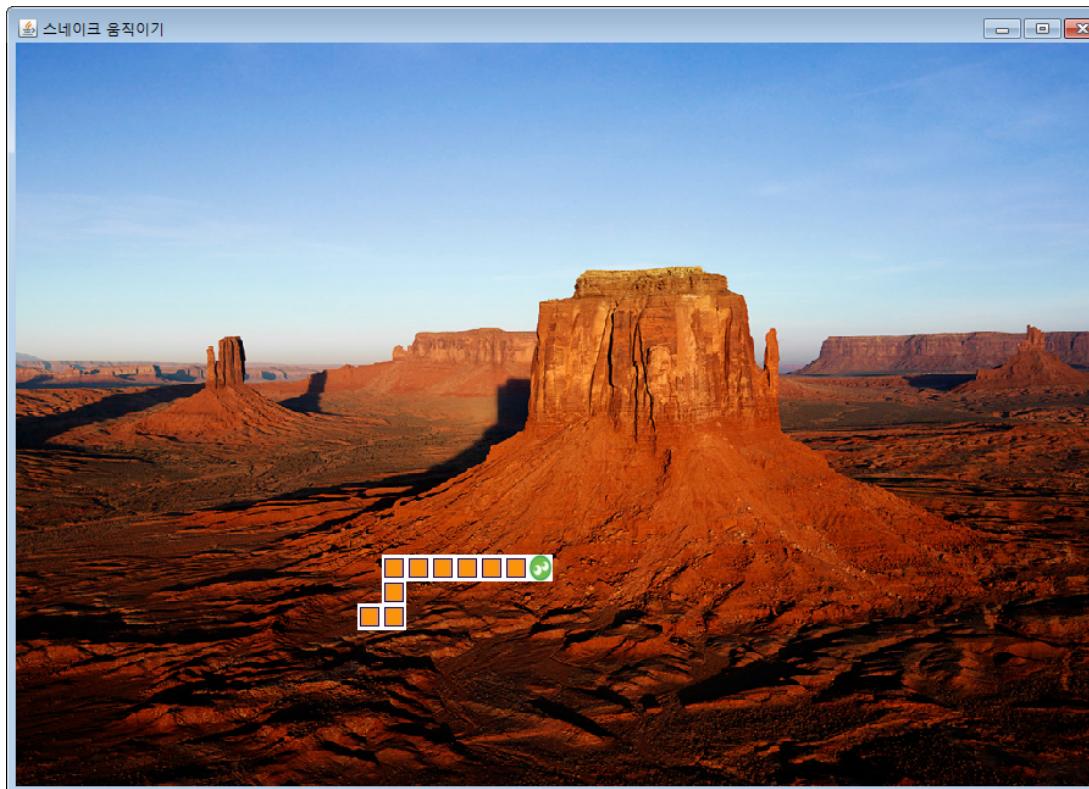


자바의 배포판

- 오라클은 개발 환경에 따라 다양한 자바 개발 배포판 제공
- Java SE
 - 자바 표준 배포판(Standard Edition)
 - 데스크탑과 서버 응용 개발 플랫폼
- Java ME
 - 자바 마이크로 배포판
 - 휴대 전화나 PDA, 셋톱박스 등 제한된 리소스를 갖는 하드웨어에서 응용 개발을 위한 플랫폼
 - Java SE의 서브셋
 - 임베디드 및 가전 제품을 위한 API 정의
- Java EE
 - 자바 기업용 배포판
 - 자바를 이용한 다중 사용자, 기업용 응용 개발을 위한 플랫폼
 - Java SE + 인터넷 기반의 서버사이드 컴퓨팅 관련 API 추가

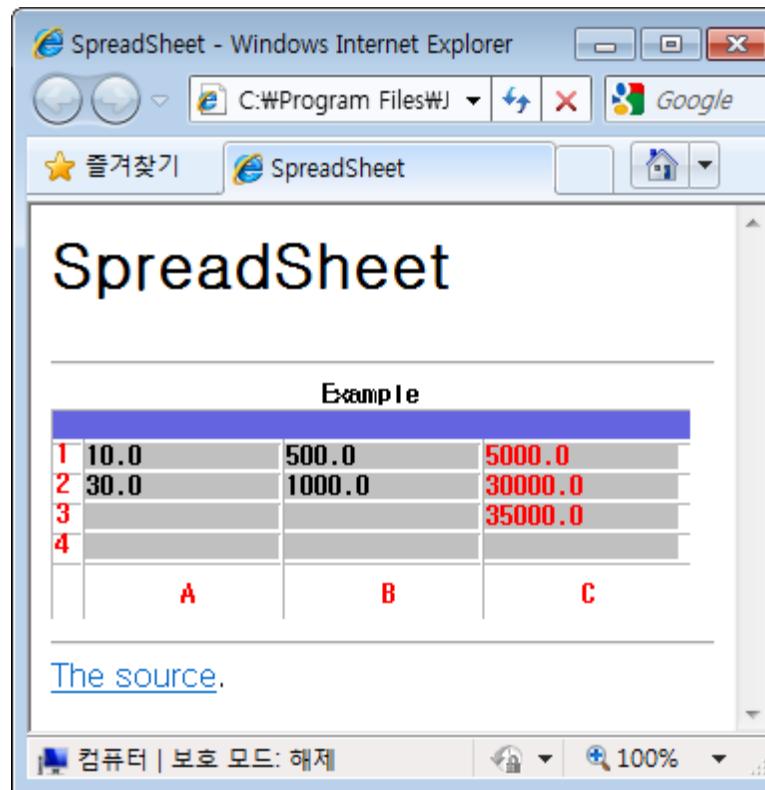
자바 응용의 종류 : 데스크톱 응용프로그램

- 가장 전형적인 자바 응용프로그램
 - PC 등의 데스크톱 컴퓨터에 설치되어 실행
 - JRE가 설치된 어떤 환경에서도 실행
 - 다른 응용프로그램의 도움이 필요 없이 단독으로 실행



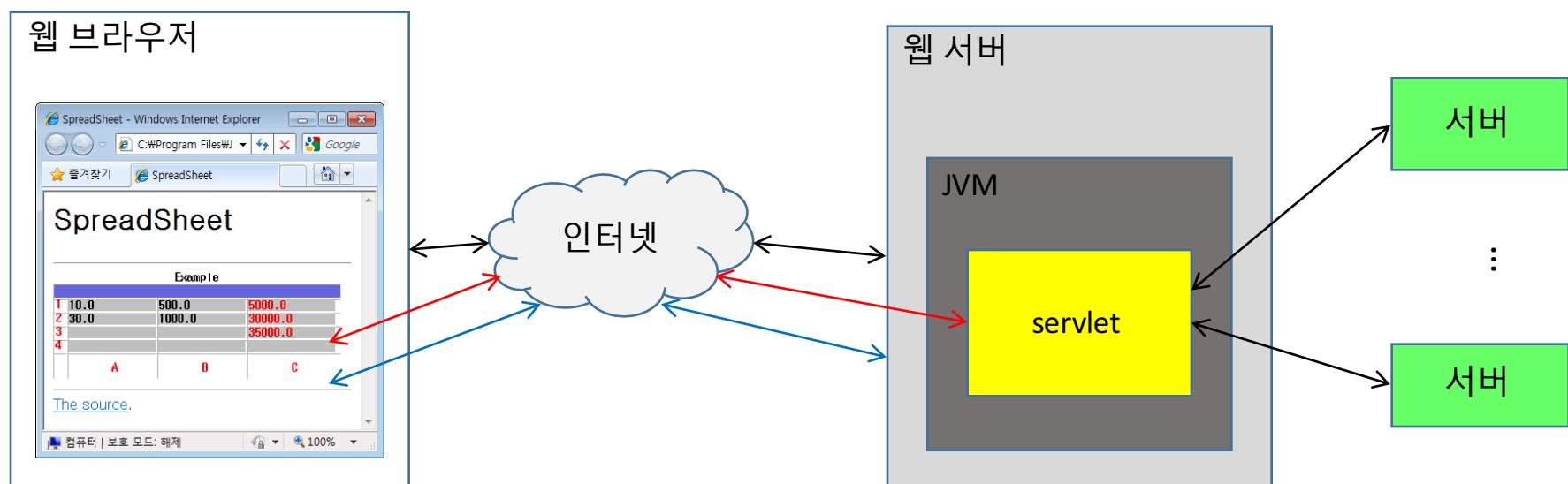
자바 응용의 종류 : 애플릿 응용프로그램

- 애플릿(applet)
 - 웹 브라우저에 의해 구동되고 실행이 제어되는 자바 프로그램
 - 애플릿은 사용할 수 있는 자원 접근에 제약 있음



자바 응용의 종류 : 서블릿 응용프로그램

- 서블릿(servlet)
 - 애플릿과 반대로 **서버에서 실행되는 자바 프로그램**
 - 서버 클라이언트 모델에서 서블릿과 애플릿이 각각 통신하면서 실행
 - **데이터베이스 서버 및 기타 서버와 연동하는 복잡한 기능 구현 시 사용**
 - 사용자 인터페이스가 필요 없는 응용
 - 웹 서버에 의해 실행 통제 받음



자바 응용의 종류 : JavaME 응용프로그램

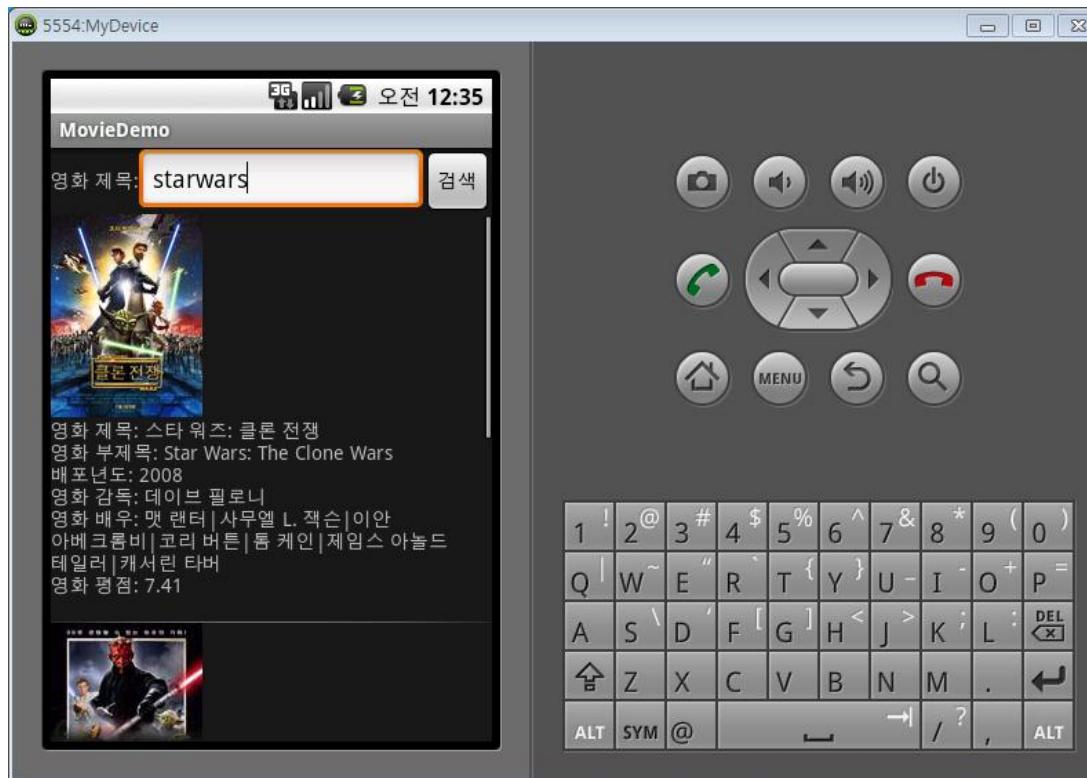
- Java ME 응용프로그램
 - Java ME : **임베디드, 모바일 기기를 위한 자바 배포판**
 - 오픈소스 HW (예, 라즈베리파이)등 임베디드 기기에 탑재
 - 유럽, 미국 시장에 출시되는 모바일 단말기에 탑재



자바 모바일 응용 : 안드로이드 앱

- 안드로이드

- 구글의 주도로 여러 모바일 회사가 모여 구성한 OHA(Open Handset Alliance)에서 만든 무료 모바일 플랫폼
- 개발 언어는 자바를 사용하나 JVM에 해당하는 Dalvik은 기존 바이트 코드와 호환성이 없어 변환 필요



요약

- 프로그래, 프로그래밍 언어, 컴파일
 - 프로그램: 컴퓨터가 수행할 작업 지시서
 - 프로그래밍 언어: 프로그램을 작성할 때 사용하는 언어
 - 컴파일: 프로그램 소스를 컴퓨터가 이해할 수 있는 언어로 변역
- 플랫폼 독립성을 지원하는 자바의 방법
 - 바이트코드: 특정 CPU에 종속적이지 않은 중립적인 코드
 - 자바가상기계: 자바 바이트코드를 해석하고 실행하는 소프트웨어
- 자바 응용프로그램의 종류
 - 테스크톱, 애플릿, 서블릿, JavaME, 안드로이드 애플리케이션 등

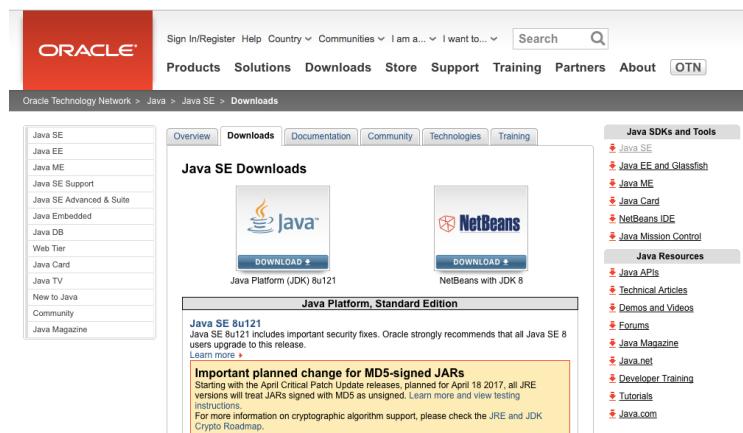
자바 프로그램 작성 및 실행

1. JAVA 개발환경 설치

A. JDK 다운로드 및 설치

i. 아래 링크를 통해서 접속

<http://www.oracle.com/technetwork/java/javase/downloads/index-jsp-138363.html>



ii. Java DOWNLOAD 클릭후 다음하면서, [Accept License Agreement]

선택하고, 자신의 컴퓨터에 맞는 파일을 선택하여 다운로드

(가령, Windows 컴퓨터인 경우, Windows 운영체제가 32 비트이면
Window x86 을 64 비트이면 Windows x64 를 선택해야 함)

Java SE Development Kit 8u121		
You must accept the Oracle Binary Code License Agreement for Java SE to download this software.		
<input type="radio"/> Accept License Agreement	<input checked="" type="radio"/> Decline License Agreement	
Product / File Description	File Size	Download
Linux ARM 32 Hard Float ABI	77.86 MB	jdk-8u121-linux-arm32-vfp-hflt.tar.gz
Linux ARM 64 Hard Float ABI	74.83 MB	jdk-8u121-linux-arm64-vfp-hflt.tar.gz
Linux x86	162.41 MB	jdk-8u121-linux-i586.rpm
Linux x86	177.13 MB	jdk-8u121-linux-i586.tar.gz
Linux x64	159.96 MB	jdk-8u121-linux-x64.rpm
Linux x64	174.76 MB	jdk-8u121-linux-x64.tar.gz
Mac OS X	223.21 MB	jdk-8u121-macosx-x64.dmg
Solaris SPARC 64-bit	139.64 MB	jdk-8u121-solaris-sparcv9.tar.Z
Solaris SPARC 64-bit	99.07 MB	jdk-8u121-solaris-sparcv9.tar.gz
Solaris x64	140.42 MB	jdk-8u121-solaris-x64.tar.Z
Solaris x64	96.9 MB	jdk-8u121-solaris-x64.tar.gz
Windows x86	189.36 MB	jdk-8u121-windows-i586.exe
Windows x64	195.51 MB	jdk-8u121-windows-x64.exe

iii. 다운로드한 파일 실행하여 설치

B. INTELLIJ IDEA IDE 다운로드 및 설치

i. 아래 링크를 통해서 접속

<https://www.jetbrains.com/idea/#chooseYourEdition>

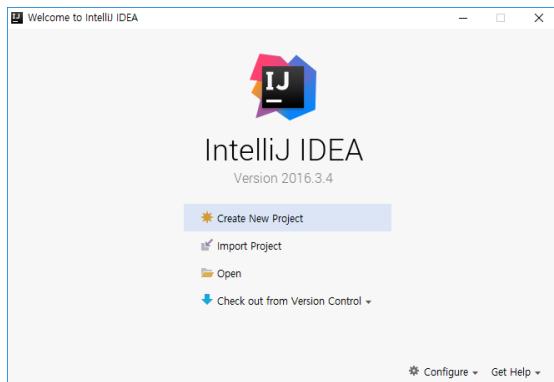
ii. Community 버전 다운로드

iii. 다운로드한 파일 실행하여 설치

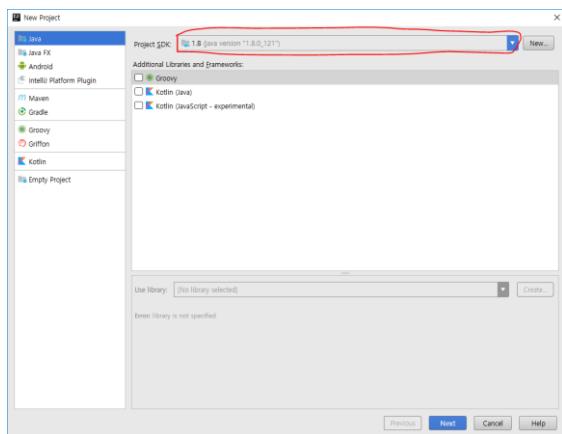
2. INTELLIJ IDEA IDE 를 이용한 자바 프로그램 개발

A. 프로젝트 생성

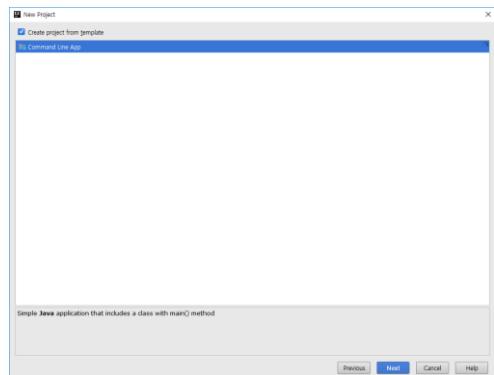
i. Create New Project 메뉴 선택



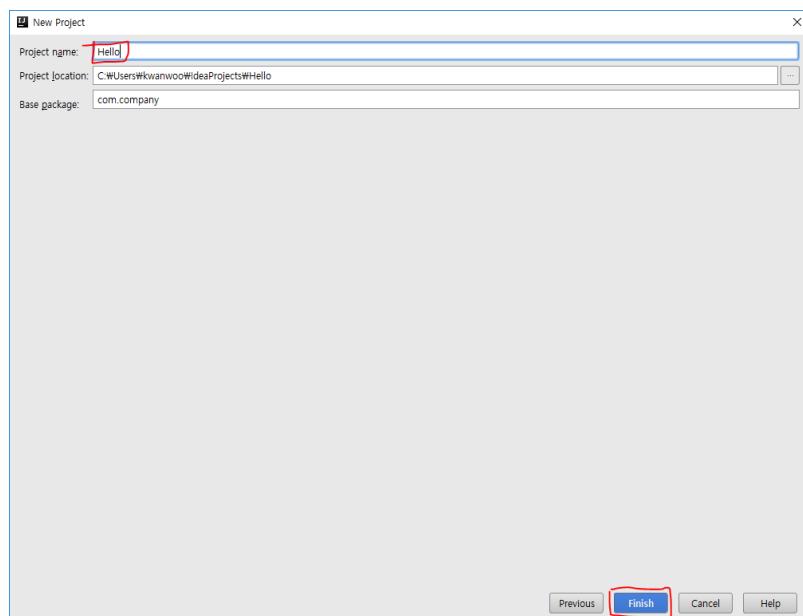
ii. Project SDK 의 드롭다운 메뉴에서 설치된 Java SDK 를 선택 (Java SDK 가
안보이는 경우에는 New 버튼을 누른 후, 설치된 Java SDK 폴더를 선택 후, OK
버튼 클릭)



iii. "Create project from template" 선택후, Next 버튼 클릭



iv. Project Name 칸에 새로운 프로젝트 이름을 입력 후, Finish 버튼 클릭



B. 자바 소스 작성

i. 자바 소스 코드 작성

```
package com.company;

public class Main {

    public static void main(String[] args) {
        int year = 2017;
        System.out.println("Welcome to " + year + " Java Class" );
    }
}
```

The screenshot shows the IntelliJ IDEA interface with the project 'Hello' open. The code editor displays the following Java code:

```
1 package com.company;
2
3 public class Main {
4
5     public static void main(String[] args) {
6         int year = 2017;
7         System.out.println("Welcom to " + year + " Java Class");
8     }
9 }
10
```

C. 컴파일 및 실행

The screenshot shows the IntelliJ IDEA interface with the project 'Hello' open. The code editor displays the same Java code as before. The 'Run' tool window at the bottom has a green play button icon highlighted with a red box. The terminal window shows the command 'java -version' and the output 'java version "1.8.0_121"'. A red box highlights the terminal output.