기초사항

514760
2020년 봄학기
3/24/2020
박경신

자바란 무엇인가?

- 1991년 그린 프로젝트(Green Project)
  - 선마이크로시스템즈의 제임스 고슬링(James Gosling)에 의해 시작
    - 가전 제품에 들어갈 소프트웨어를 위해 개발
- 목적
  - 플랫폼 호환성 문제 해결
    - 기존 언어로 작성된 프로그램은 PC, 유닉스, 메인 프레임 등 플랫폼 간에 호환성 없음
    - 소스를 다시 컴파일하거나 프로그램을 재 작성해야 하는 단점
  - 플랫폼 독립적인 언어 개발
    - 모든 플랫폼에서 호환성을 갖는 프로그래밍 언어 필요
    - 네트워크, 특히 웹에 최적화된 프로그래밍 언어의 필요성 대두
  - 메모리 사용량이 적고 다양한 플랫폼을 가지는 가전 제품에 적용
    - 가전 제품 : 작은 양의 메모리를 가지는 제어 장치
    - 내장형 시스템 요구 충족
- 2009년에 선마이크로시스템즈를 오라클에서 인수

Java Version

https://en.wikipedia.org/wiki/Java_version_history

- JDK 1.0
  - Jan 1996
  - JDK 1.0 발표, Applet 도입
- JDK 1.1
  - Feb 1997
  - Inner class, JavaBeans, JDBC, RMI(Remote Method Invocation), reflection, JIT(Just-In-Time) compiler, Internationalization and Unicode
- J2SE 1.2
  - Dec 1998
  - ME, SE, EE version 발표
  - Swing이 SE에 포함, Corba IDL, Collection framework
- J2SE 1.3
  - May 2000
  - HotSpot(Sun에서 만든 JIT구현) JVM(Java Virtual Machine), JNDI(Java Naming and Directory Interface), JPDA, JavaSound 포함
- J2SE 1.4
  - Feb 2002
  - Regular expression, assert, Security 2 version, IPv6, Non Blocking IO(NIO), Logging API, Image IO API, XML Parser, XSLT

Java Version

- **J2SE 5.0**
  - Sep 2004
  - 기능적으로 가장 많은 변화가 생긴 버전
  - Generics, annotation, auto boxing, enum, varargs, foreach, static import, Concurrent API, Scanner class
- Java SE 6
  - Dec 2006
  - 보안, 성능강화 주력
- Java SE 7
  - Jul 2011
  - String in switch, try-resource, generics에서 타입추론, 숫자에서 underscore 사용
  - JavaFX가 기본으로 포함
- **Java SE 8 (LTS)**
  - Mar 2014
  - 오라클로 인수된 후 첫번째 버전 Java 5 이후 가장 큰 언어적 변화
  - Lambda expression, Default method interface, functional programming
  - Date and Time API, stream API

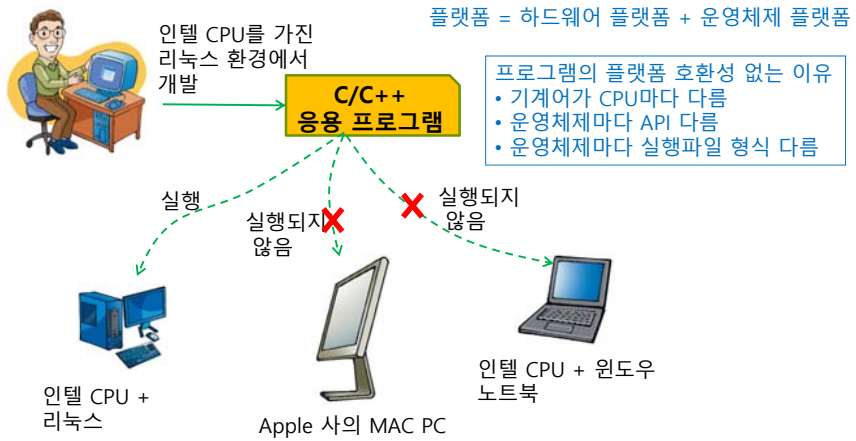
Java Version

- Java SE 9
  - Sep 2017
  - Modular system(Jigsaw) jshell (Java Shell), G1 garbage collector
- Java SE 10
  - Mar 2018
  - Local-variable type interference(var), Parallel Full GC for G1
- **Java SE 11 (LTS)**
  - Sep 2018
  - 자바 11은 Long-term support 버전
  - Nest-based access controls, ZGC, Flight Recorder, HTTP API, TLS(Transport Layer Security) 1.3, 람다에서 var 변수
  - JavaFX, Java EE & CORBA 모듈, Web Start, Applets 삭제
- Java SE 12
  - Mar 2019
- Java SE 13
  - Sep 2019

자바의 특징

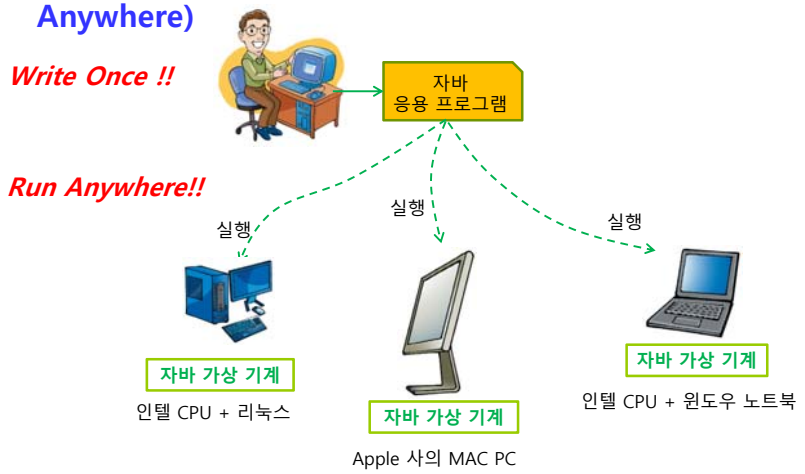
- **WORA(Write Once Run Anywhere)**
  - 한번 작성된 코드는 모든 플랫폼에서 바로 실행되는 자바의 특징
  - C/C++ 등 기존 언어가 가진 플랫폼 종속성 극복
    - OS, H/W에 상관없이 자바 프로그램이 동일하게 실행
  - 네트워크에 연결된 어느 클라이언트에서나 실행
    - 웹 브라우저, 분산 환경 지원
- WORA를 가능하게 하는 자바의 특징
  - **바이트 코드(byte code)**
    - 자바 소스를 컴파일한 목적 코드
    - CPU에 종속적이지 않은 중립적인 코드
    - JVM에 의해 해석되고 실행됨
  - **JVM(Java Virtual Machine)**
    - 자바 바이트 코드를 실행하는 자바 가상 기계(소프트웨어)

플랫폼 종속성 (Platform Dependency)



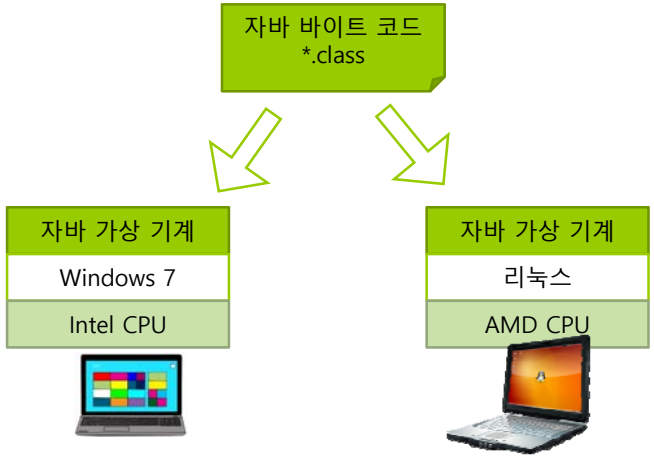
플랫폼 독립성

- **자바는 플랫폼 독립성 - WORA(Write Once Run Anywhere)**



자바 가상 기계

- 자바 컴파일러는 특정한 컴퓨터가 아닌 가상적인 기계(virtual machine)를 위한 바이트 코드를 생성함

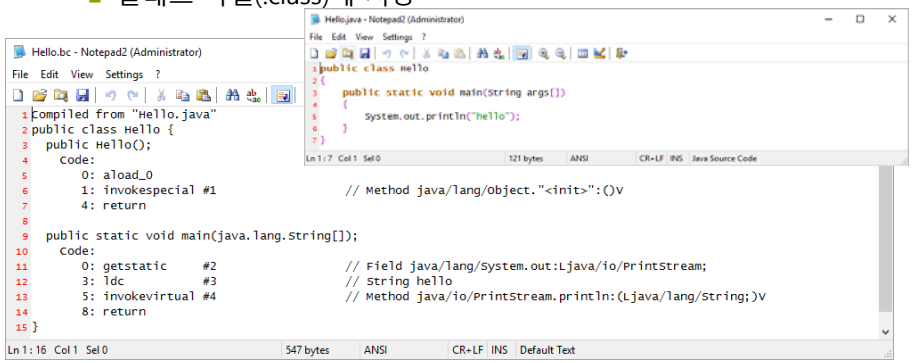


자바 가상 기계

- **자바 가상 기계 (JVM : Java Virtual Machine)**
  - 각기 다른 플랫폼에 설치
  - 동일한 자바 실행 환경 제공
  - 자바 가상 기계 자체는 플랫폼에 종속적
    - 자바 가상 기계는 플랫폼마다 각각 작성됨
    - 예) 리눅스에서 작동하는 자바 가상 기계는 윈도우에서 작동하지 않음
  - 자바 가상 기계 개발 및 공급
    - 자바 개발사인 오라클 외 IBM, MS 등 다양한 회사에서 제작 공급
- 자바의 실행
  - 자바 가상 기계가 클래스 파일(.class) 바이트 코드 실행

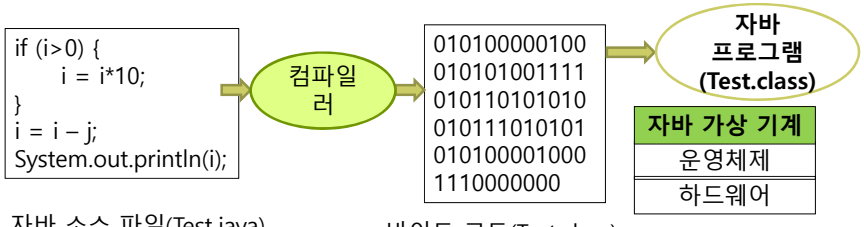
바이트 코드

- 바이트 코드
  - 자바 가상 기계에서 실행 가능한 바이너리 코드
    - 바이트 코드는 컴퓨터 CPU에 의해 직접 실행되지 않음
    - 자바 가상 기계가 작동 중인 플랫폼에서 실행
    - 자바 가상 기계가 인터프리터 방식으로 바이트 코드 해석
  - 클래스 파일(.class)에 저장



자바와 C/C++의 실행 환경 차이

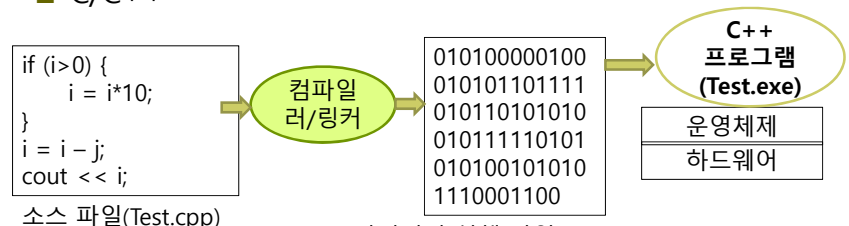
- 자바



자바 소스 파일(Test.java)

바이트 코드(Test.class)

- C/C++



소스 파일(Test.cpp)

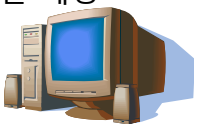
바이너리 실행 파일(Test.exe)

자바와 C/C++의 실행 환경 차이

- 자바
  - 자바는 링크 과정 없이 컴파일러가 바로 바이트 코드 생성
  - 바이트 코드는 JVM에서만 실행 가능
  - 자바는 필요한 클래스들을 프로그램 실행 중에 동적으로 로딩
    - 동적 로딩은 JVM에 포함된 클래스 로더에 의해 이루어짐
    - ClassLoader 클래스를 이용하여 개발자가 직접 클래스 로딩가능
- C/C++
  - 컴파일
    - C/C++에서는 컴파일러가 중간 단계인 목적 코드를 생성
  - 링크
    - 링커가 목적 코드와 라이브러리 연결, 실행 가능한 최종 실행 파일 생성
  - 정적 라이브러리는 실행 파일에 포함
    - 실행 파일 크기가 커짐
  - 목적 코드 및 실행 파일은 플랫폼에 따라 다름
    - 플랫폼이 바뀌면 다시 컴파일 및 링크

자바의 종류

- 오라클은 개발 환경에 따라 다양한 자바 배포판 제공
- **Java SE**
  - **자바 표준 배포판(Standard Edition)**
  - 데스크탑과 서버 응용 개발 플랫폼

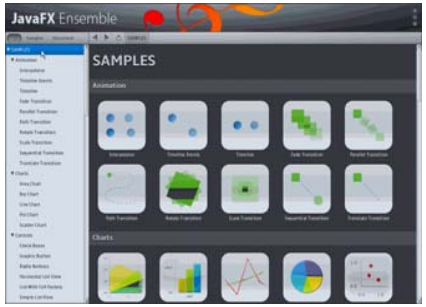


- Java EE
  - 자바 기업용 배포판
    - 자바를 이용한 다중 사용자, 기업용 응용 개발을 위한 플랫폼
  - Java SE + 인터넷 기반의 서버사이드 컴퓨팅 관련 API 추가
  - 응용 서버, 웹서버, J2EE API, 엔터프라이즈 자바 빈즈(JavaBeans) 지원, 자바 서블릿 API 와 JSP 등을 포함

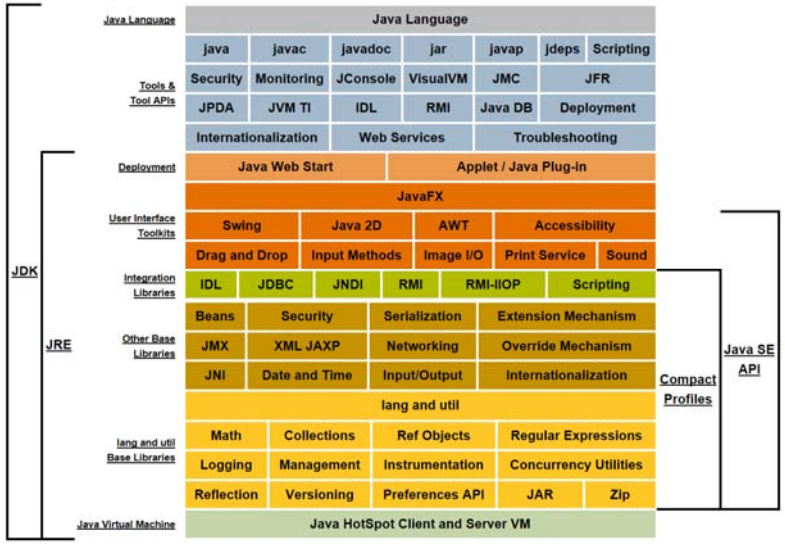
- Jave ME
  - 자바 마이크로 배포판
    - 휴대 전화나 PDA, 셋톱박스 등 제한된 리소스를 갖는 하드웨어에서 응용 개발을 위한 플랫폼
    - 가장 작은 메모리 풋프린트
  - Java SE의 서브셋 + 임베디드 및 가전 제품을 위한 API 정의

자바의 종류

- Jave FX
  - 자바 리치-인터넷-애플리케이션(RIA: Rich Internet Application)을 생성하고 배포하기 위한 자바 클라이언트 플랫폼
  - RIA은 서로 인터넷 상의 다양한 플랫폼에서도 동일한 외관으로 실행
  - JavaFX 플랫폼은 자바 기술에 기반을 둔 고성능의 하드웨어 가속 그래픽과 미디어 엔진 API를 제공



Java SE

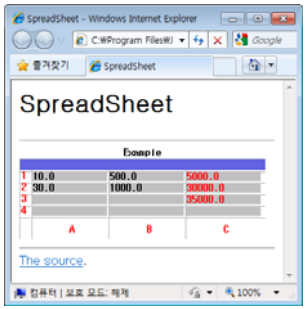


JDK와 JRE

- JDK(Java Development Kit)
  - 자바 응용 개발 환경. 개발에 필요한 도구 포함
    - 컴파일러, JRE (Java Runtime Environment), 클래스 라이브러리, 샘플 등 포함
- JRE(Java Runtime Environment)
  - 자바 실행 환경. JVM 포함
  - 자바 실행 환경만 필요한 경우 JRE만 따로 다운 가능
- JDK와 JRE의 개발 및 배포
  - 오라클의 Technology Network의 자바 사이트에서 다운로드
    - http://www.oracle.com/technetwork/java/index.html
- JDK의 bin 디렉터리에 포함된 주요 개발 도구
  - javac - 자바 소스를 바이트 코드로 변환하는 컴파일러
  - java - JRE의 bin 디렉터리에도 있는 자바 응용프로그램 실행기
  - jar - 자바 아카이브 파일 (JAR)의 생성 및 관리하는 유틸리티
  - jdb - 자바 디버거
  - appletviewer - 웹 브라우저 없이 애플릿을 실행하는 유틸리티

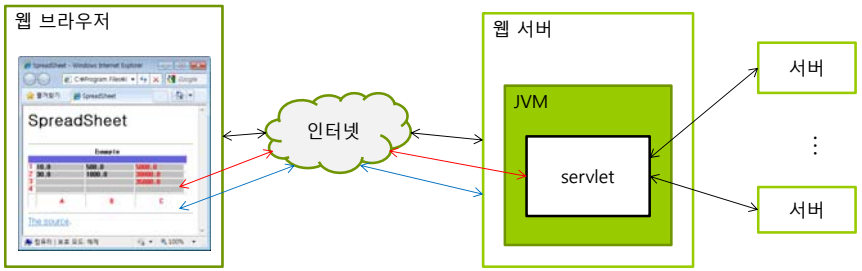
자바로 만들 수 있는 것

- 자바 애플리케이션(Java application)
  - 독립적으로 실행될 수 있는 일반 응용 프로그램
- 자바 애플릿(Java applet)
  - 웹 브라우저 안에서 실행되는 작은 프로그램
  - 애플릿은 사용할 수 있는 자원 접근에 제약 있음



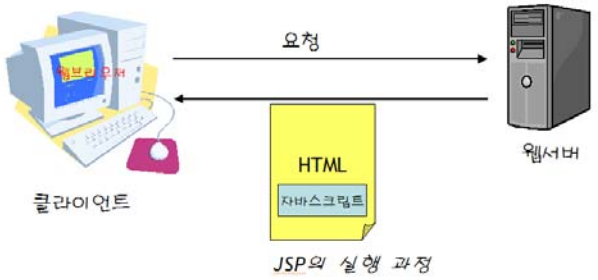
자바로 만들 수 있는 것

- 자바 서블릿(Java servlet)
  - 웹서버에서 동작하는 서버 모듈로서 클라이언트의 요구를 받아서 그에 대한 처리를 한 후에, 실행 결과를 HTML 문서 형태로 클라이언트 컴퓨터로 전송
  - 데이터베이스 서버 및 기타 서버와 연동하는 복잡한 기능 구현시 사용. 웹 서버에 의해 실행 통제 받음.



자바로 만들 수 있는 것

- JSP (Java Server Pages)
  - HTML안에 자바 스크립트 코드를 넣으면 웹페이지를 사용자와 상호작용하도록 만들 수 있음
  - JSP는 서버에서 실행되고 클라이언트 요청에 따라서 동적으로 HTML, XML 웹페이지를 생성

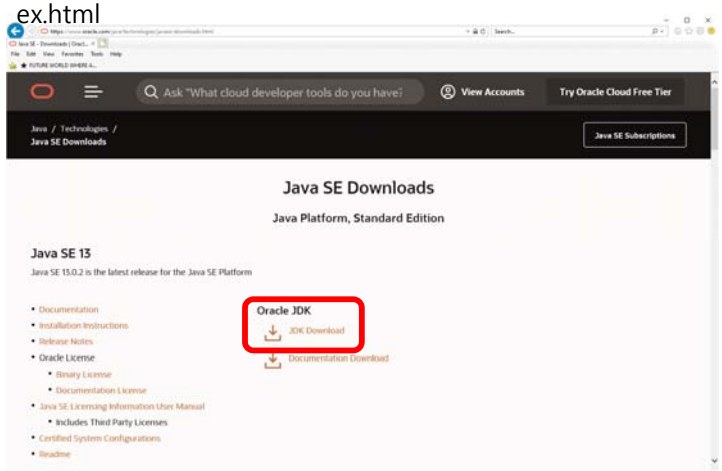


자바로 만들 수 있는 것

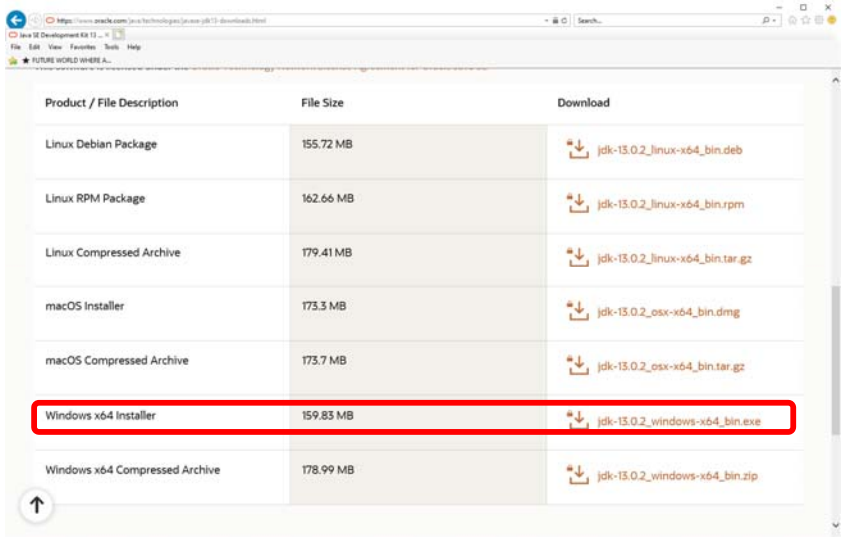
- 안드로이드 애플리케이션
  - 안드로이드 개발자들은 자바의 SE 버전 중에서 AWT와 스윙(swing)을 제외한 거의 모든 패키지를 사용함
  - 개발 언어는 자바를 사용하나 효율성때문에 자체적인 달빅(Dalvik) 가상 머신을 구현함. 달빅은 기존 바이트 코드와 호환성이 없어 변환이 필요함.

JDK 설치

- Java SE Development Kit (JDK13.0.2) Download
  - http://www.oracle.com/technetwork/java/javase/downloads/index.html

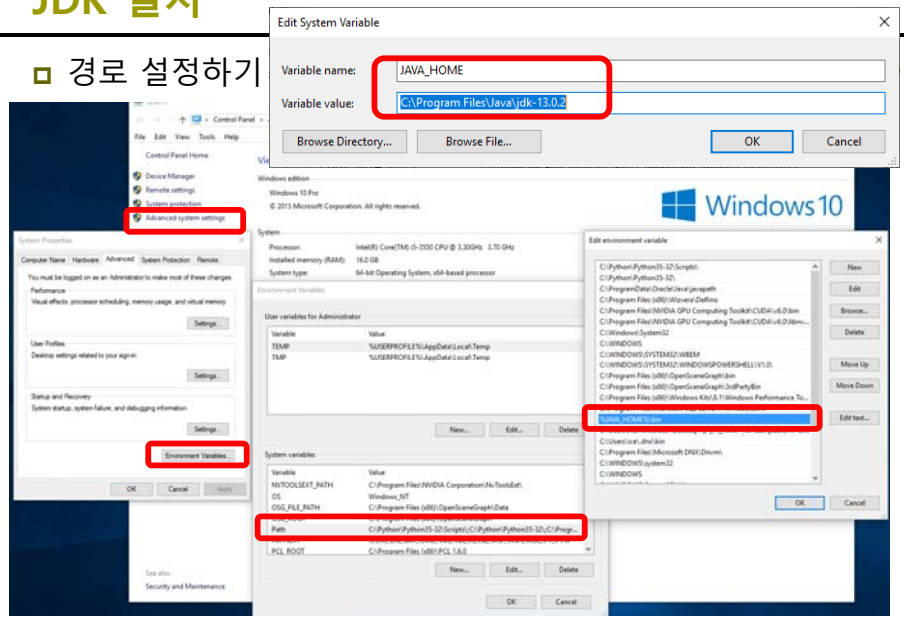


JDK 설치



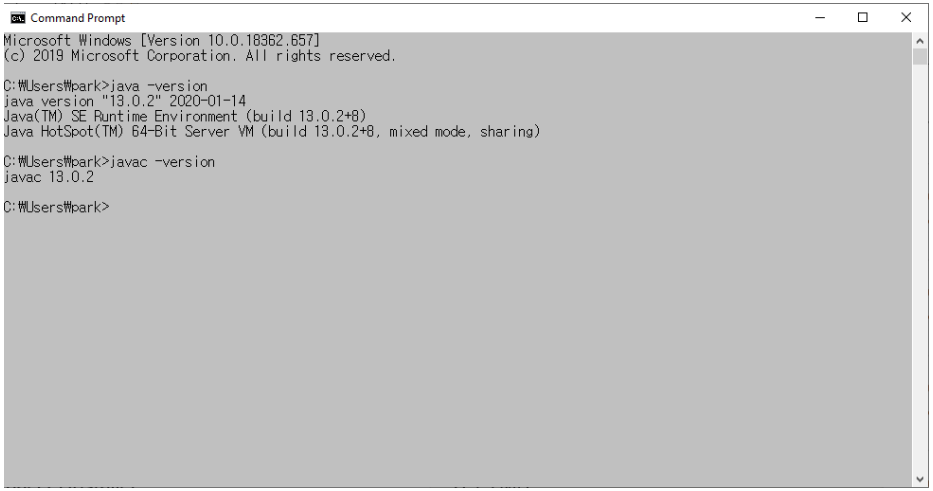
JDK 설치

- 경로 설정하기



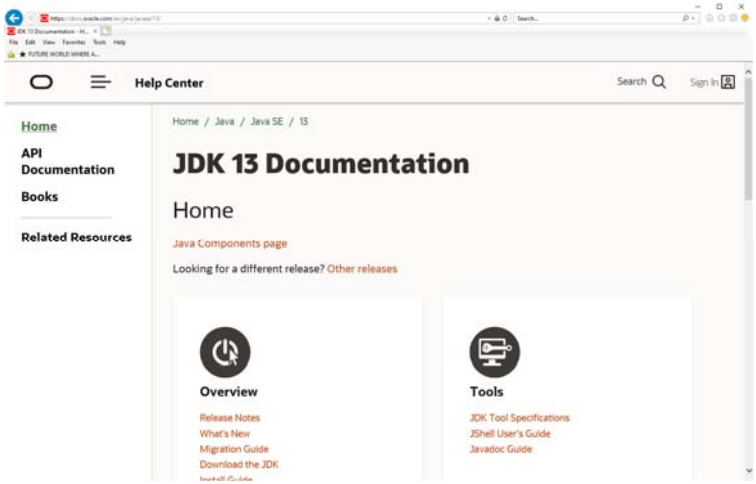
JDK 설치

- 설치된 자바 버전 확인



자바 API 문서

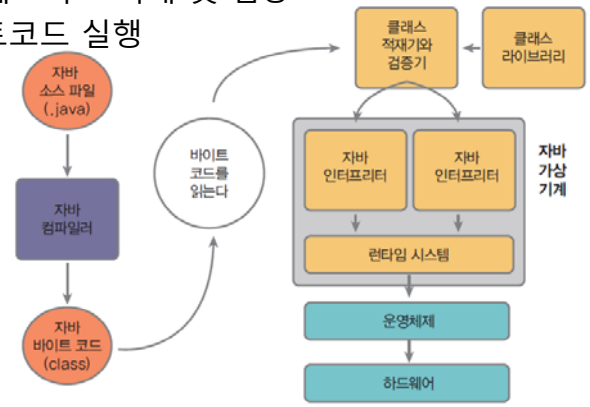
- https://docs.oracle.com/en/java/javase/13/



JDK 설치 후 디렉터리 구조

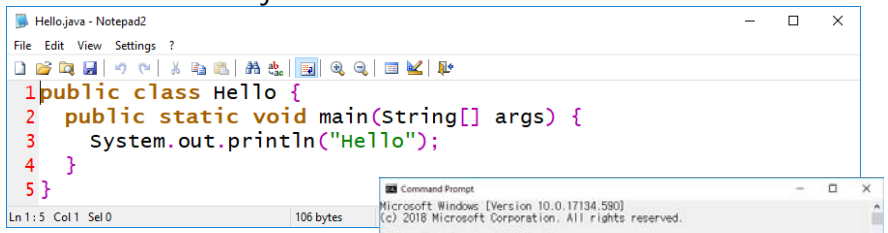
자바 프로그램 개발 단계

- 소스 파일 .java
- 컴파일 후 바이트코드로 변환 .class
- 바이트코드를 메모리로 적재 및 검증
- JVM에서 바이트코드 실행

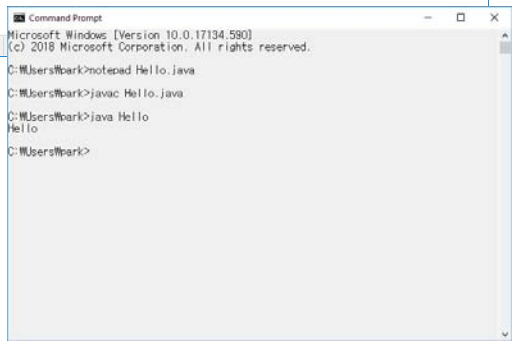


자바 프로그램 개발 단계

- 소스코드 Hello.java

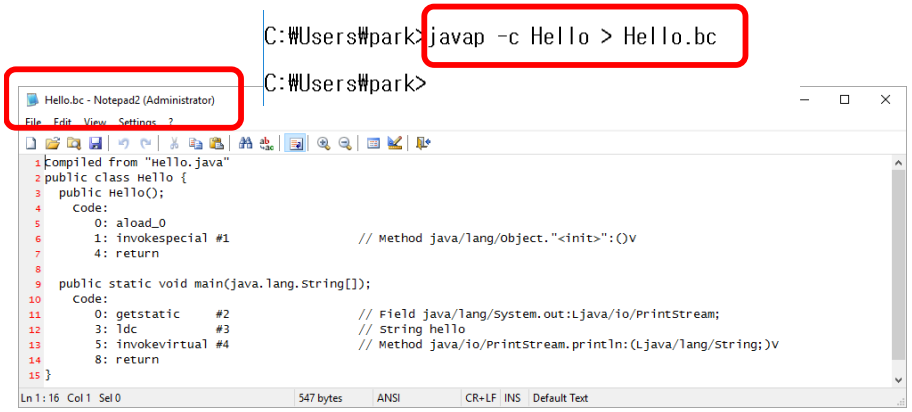


- 컴파일 javac Hello.java
- 실행 java Hello



자바 프로그램 개발 단계

- Disassemble
  - 클래스 파일에 들어있는 바이트 코드를 텍스트로 볼 수 있게 변환하는 작업 (JDK의 javap.exe 이용)
  - Hello.class 파일을 디스어셈블한 결과 Hello.bc 파일 생성



자바 API

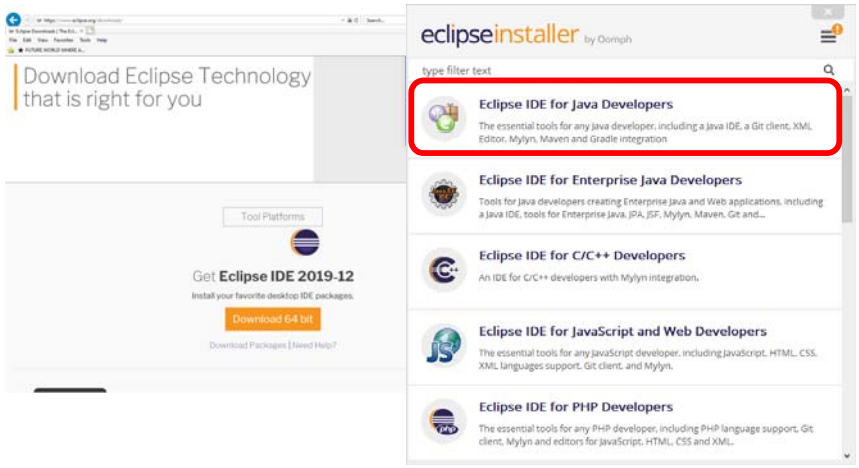
- 자바 API(Application Programming Interface)
  - JDK에 포함된 클래스 라이브러리
    - 주요한 기능들을 미리 구현한 코드(클래스 라이브러리)의 집합
  - 개발자는 API를 이용하여 쉽고 빠르게 자바 프로그램 개발
    - API에서 정의한 규격에 따라 클래스 사용
- 자바 패키지(Package)
  - 서로 관련된 클래스들을 분류하여 묶어 놓은 것
  - 계층구조로 되어 있음
    - 클래스의 이름에 패키지 이름도 포함
    - 다른 패키지에 동일한 이름의 클래스 존재 가능
  - 자바 API(클래스 라이브러리)는 JDK에 패키지 형태로 제공됨
    - 필요한 클래스가 속한 패키지만 import하여 사용
  - 개발자 자신의 패키지 생성 가능

자바 IDE 소개와 설치

- 자바 IDE (Integrated Development Environment)
  - 통합 개발 환경
  - 편집, 컴파일, 디버깅을 한번에 할 수 있는 통합된 개발 환경
- 이클립스(Eclipse)
  - 자바 응용 프로그램 개발을 위한 통합 개발 환경
  - IBM에 의해 개발된 오픈 소스 프로젝트
  - http://www.eclipse.org/downloads/ 에서 다운로드
- 넷빈즈(Netbeans)
  - https://netbeans.org/
- IntelliJ Idea
  - http://www.jetbrains.com/idea/download/

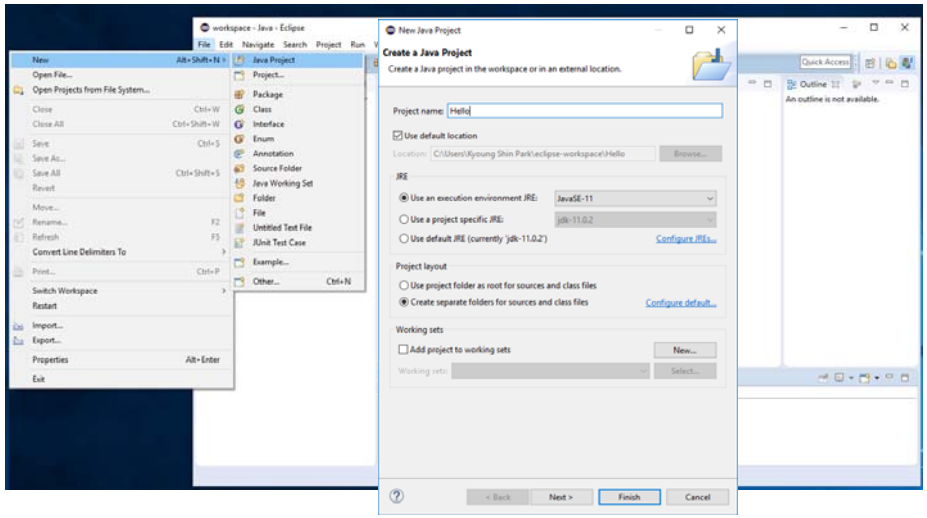
Eclipse 설치하기

- 이클립스 IDE 2019-12 (eclipse-inst-win64.exe)
  https://www.eclipse.org/downloads/ 다운로드



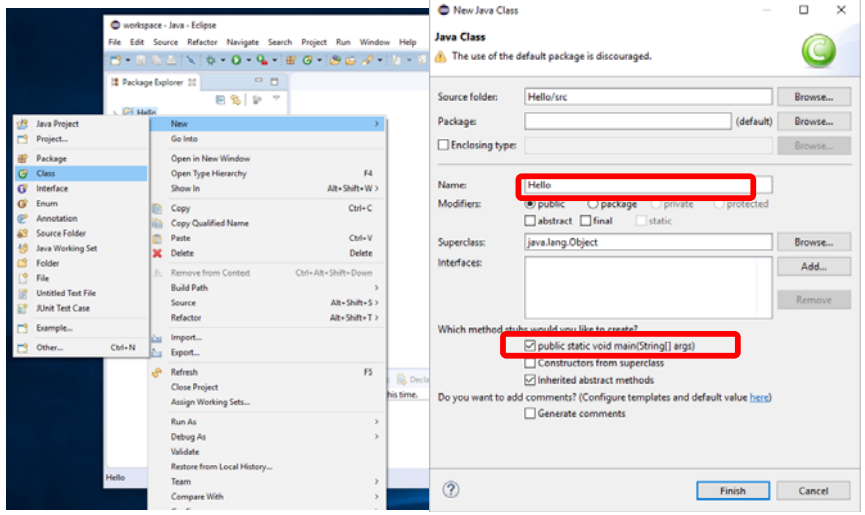
자바 프로그램 작성

- 이클립스 프로젝트 생성 File->New->Java Project



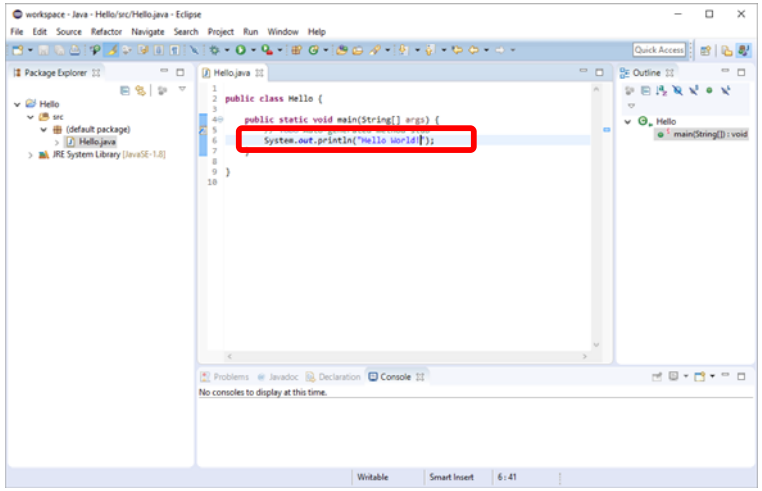
자바 프로그램 작성

- 클래스 생성 [프로젝트]->New->Class



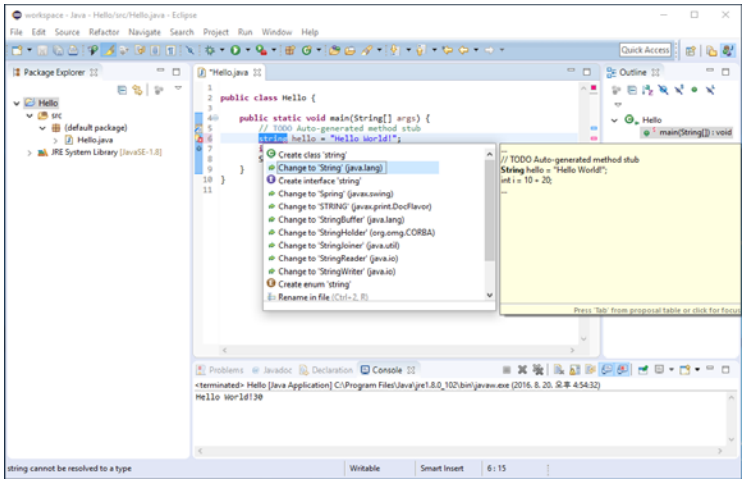
자바 프로그램 작성

- 클래스에 소스 코드 추가



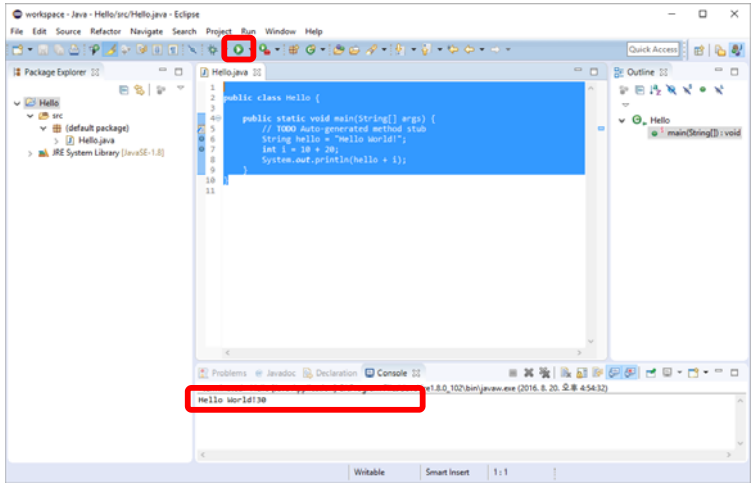
자바 프로그램 컴파일 오류

- 컴파일 오류시 quick fix를 눌러서 수정



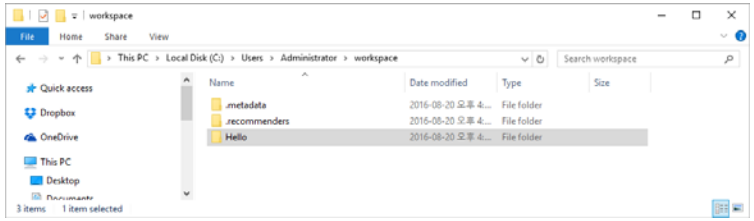
자바 프로그램 실행

- 프로그램 실행 Run (Ctrl+F11)



자바 프로그램 실행

- 이클립스 프로젝트 디렉토리



- 또는 콘솔 창에서 실행

LAB0

- HelloWorld 프로그램을 작성한다.
  - JDK 설치
  - IDE 설치
  - 환경설정
  - 이클립스를 사용한 자바 프로그램 작성
  - 프로젝트 디렉토리 안에 보고서 (1~2장)를 넣고 Lab0_학번_이름.zip 압축한 후 e-learning으로 제출