[개요]

생코 강의를 들으면서 모든 내용을 워드에 정리하려고 하다보니 너무도 많은 시간이 걸리고, 수업에도 집중하기 힘들었다. 그리고 그렇게 고생해서 적은 내용들을 나중에 한 달, 두 달 뒤에 찾아보면 다시 이해하는데 또 시간이 걸릴뿐더러 텍스트 투성이의 공부 내용을 줄줄이 읽기가 참 힘들었다. 분명 읽기 쉽도록 구어체로, 나름 농담도 섞어가면서 적었음에도 불구하고 말이다.

이러한 문제를 인지하고 앞으로 수업을 들을 때에는 메모보다는 강의내용에 집중하고, 정말 강의에서 중요하다, 잊고 싶지 않다 싶은 내용들만 필기하기로 했다.

[jdk]

Java Development Kit 의 줄임말이다.

즉 자바를 사용해서 소프트웨어를 개발하는데에 필요한 도구들의 모음이다.

자바를 이용해서 프로그램을 개발할 때 사용하는 여러가지 도구들을 모아서 제공하는 개발자용 버전 자바.

[jdk vs jre]

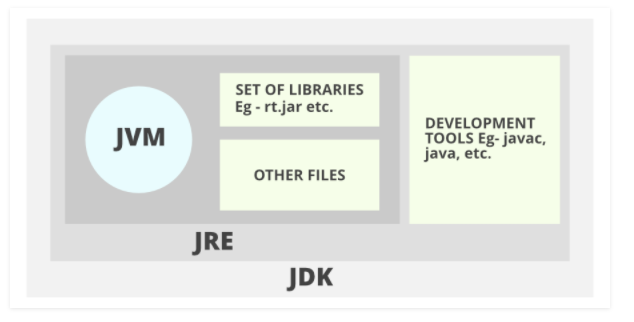
이건 강의에 없었고 궁금해서 찾아봤다.

자 옛날에 들어봤던 용언데. jre도 있고 jdk도 있다. 이것들은 대체 무엇인가?

jdk는 상술했듯 도구들의 집합이고, jre는 java runtime environment, 즉 자바 실행 환경이라는 뜻이다.

자바로 만들어진 프로그램을 실행하기 위해서 필요한 것이 jre라는 것이다.

그럼 jdk랑 무슨 차이가 있는가? 단적으로 말하자면 jre는 jdk에 포함된다.



jdk가 자바 프로그램 개발을 위해 필요한 툴의 모음이라고 했는데, 그 뿐 아니라 자바 실행 환경을 포함하고 있다.

즉 jdk는 “자바 실행이 가능한 환경(JRE) + 자바 프로그래밍 도구들” 이다.

jre는 자바 프로그램을 실행만 하기 위한 “자바 가상 머신 + 각종 클래스 라이브러리들” 이다.

출처 : https://www.geeksforgeeks.org/difference-between-jdk-and-jre-in-java/

[자바의 .class 파일과 자바 코드 실행]

자바에서 코드를 짜면 .java 파일에 짤 것이다. 즉 .java 파일에 source가 담겨있다.

그리고 .java 파일을 저장하면 컴파일을 해서 .class 파일을 만든다.

텍스트이(가) 표시된 사진

자동 생성된 설명

기본적인 Hello World 코드를 짠 source의 클래스 파일을 열어보면 위와 같다. 사람이 읽도록 만든 것은 아니라고 한다.

eclipse에서 run을 하면, 소스파일인 hello world.java가 아니라, hello world.java 파일을 컴파일해서 만들어진 hello world.class 파일이 실행되는 것이다.

[자바 동작원리]

java source(확장자가 java인 파일)를 컴파일하면 java applicastion, 즉 확장자가 class인 파일이 나온다. 그 class 파일을 java binary 파일이라고도 한다. 그걸 JRE안에 있는 JVM이 실행해준다.

JVM은 환경에 의존적이긴 하지만, java binary 파일은 어떤 환경의 JVM이라도 똑같이 동작시킬 수 있다.

JAVA 라는 Language는 이렇게 확장성을 확보하는 것이다.

JRE는 JVM의 구현모델이다. JVM을 각기 다른 환경에서 구현하기 위해서 각종 라이브러리 파일들, 기타 파일들을 가지고 있다.

[자바 출력]

**public** **class** Datatype{

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

System.***out***.println(6); // Number

System.***out***.println();

}

}

위와 같이 System.out.println 해주면 된다.

그런데 파이썬에선 그냥 print() 해주면 될걸 System.out.prinln() 이라고 적는 것이 너무 길다. 세미콜론도 포함되고.

따라서 sysout 까지만 치고 ctrl+spacebar 하면 자동으로 eclipse에서 System.out.println()을 적어준다.

[자바의 작은 따옴표]

파이썬에선 작은 따옴표나 큰 따옴표나 문자를 나타내지만,

자바에서 작은 따옴표는 character를 뜻한다. 단 한 문자를 의미한다는 것이다.

[이스케이프(escape) 문자란?]

자바의 문자열에서, 문자열 안에 큰 따옴표 따위를 출력하기 위해서 역슬래시를 붙이는데(“\”말\”” 이런 식으로), 그 때 사용되는 \(역슬래시) 문자를 이스케이프 문자라고 한다.

[매개변수, parameter란?]

대충 알고있다고 생각하고 지금까지 무시했겠지. 매개변수? 그거 대충 인자아니냐.

근데 막상 프로그래머들 대화에서 인자 인수 매개변수가 혼용된다면 전부 다 빠르게 구분해서 이해할 수 있을까?

**public** **class** OkJavaGoInHomeInput {

// parameter

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

String id = JOptionPane.*showInputDialog*("Enter a ID");

String bright = JOptionPane.*showInputDialog*("Enter a Bright level");

위 코드에서 main() 메소드에 인자로 들어간게 String[] args이고, 그게 매개변수, parameter다.

인자는 argument, 라고 한다. 그래서 매개변수의 이름이 arguments가 된 것이고.

매개변수에서 매개라는 말은 둘 사이를 이어준다고 하는 것인데, 여기서는 사용자와 프로그램을 잇는다는 것을 말한다.

[IDE 없이 java 실행하기]

.java 확장자 파일을 class 파일로 컴파일하고 JRE에서 그 class 파일을 실행시켜준다. 고했다.

이클립스로 이런 동작을 자동화했지만, 이클립스가 없을 땐 어떻게 하면 좋을까?

윈도우라고 가정한다면, 우선 개념은 당연히 javac 명령어 로 .java 확장자 파일을 컴파일하고, 생성된 class 파일들을 java 라는 명령어로 실행하면 된다.

텍스트이(가) 표시된 사진

자동 생성된 설명

그런데 자 생각해보자. 자바 코드를 작성할 때 import를 엄청나게 한다.

import를 할 때 패키지 경로들을 적어주어야 한다.

그러한 패키지들의 경로가 바뀐다면? 혹은 다른 곳에 있다면 어떻게 실행하면 좋을까 할 때 사용하는 것이 위 코드다.

javac로 컴파일할 때 -cp(class path의 약자) 옵션을 주고, 클래스 경로들을 입력한다.

클래스 경로를 입력하는 방법은 큰따옴표 사이에 경로를 입력하면 된다.

여러 경로를 입력할 경우 경로는 세미콜론으로 구분한다.

위 사진에서 경로는 .(마침표, 현재폴더를 뜻함)와 lib 폴더다.

즉 OkJavaGoInHome.java 파일에서 필요로하는 패키지들을 현재 폴더와 lib 폴더에서 찾아서 컴파일해라 라는 명령어다.

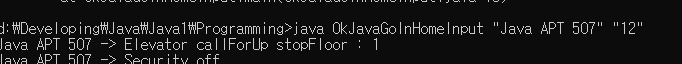
텍스트이(가) 표시된 사진

자동 생성된 설명

물론 실행할 때에도 이렇게 class path를 설정해주어야 한다.

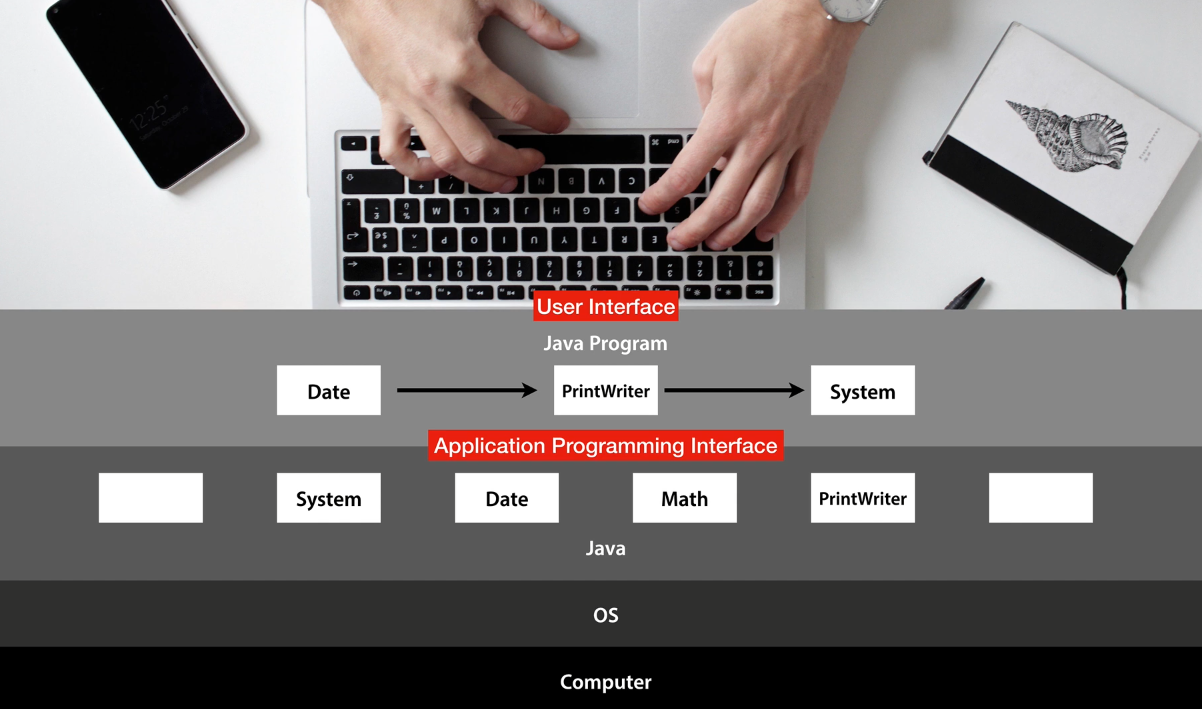
이러면 class 파일을 읽을 때 .(현재폴더)와 lib 폴더에서 class 파일들을 찾아서 실행한다.

[IDE 없이 java 실행하기 – 인자 넣기]

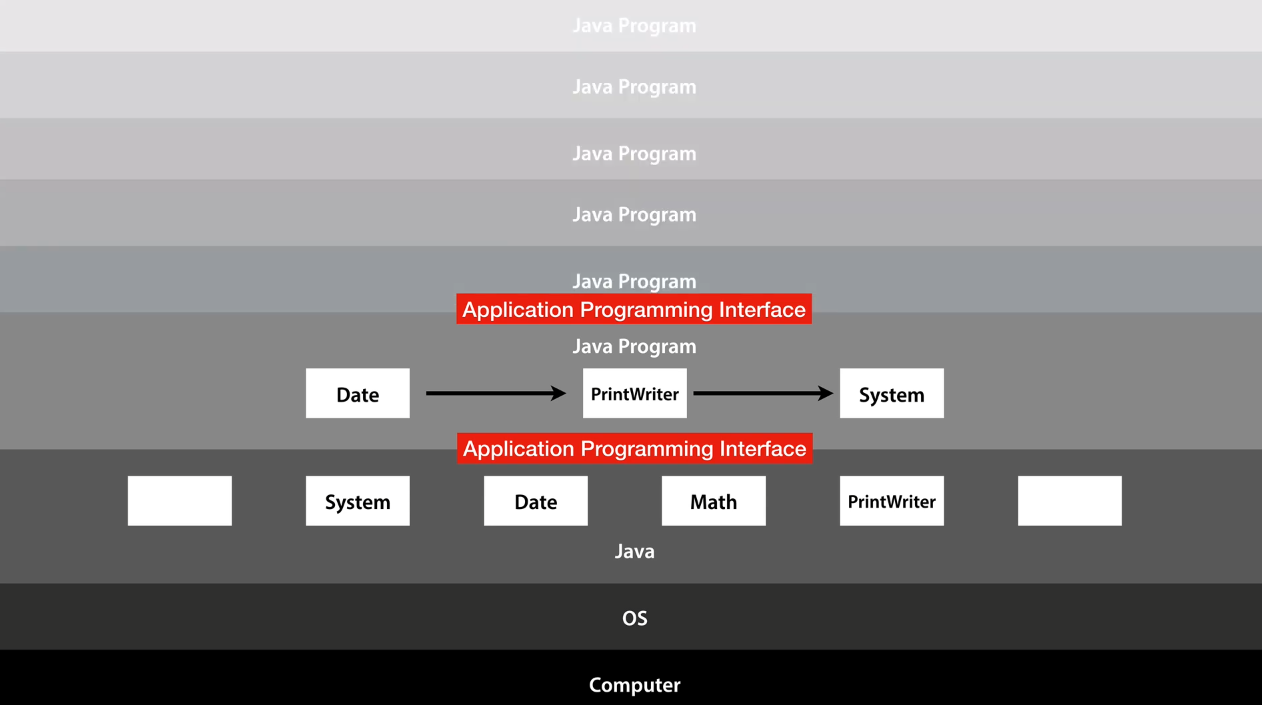


위의 명령어를 잘 보면 java로 실행할 때 큰따옴표로 묶어서 매개변수를 전달할 수 있다.

[API/ UI란?]



이 계층도가 ui와 api가 무엇인지 말해준다.



이런 식으로 중첩될수도 있다.

[override]

부모 클래스의 메소드를 덮어쓰기 하는걸 메소드 오버라이딩이라고 한다.

[자바에서 필드(Field)란]

전역변수를 의미한다.