2019.03.04

Java

프로그래밍 언어란?
고급언어
: 컴퓨터는 이해할 수 없도록 만든 언어 중에서
 사람이 쉽게 이해할 수 있는 언어
- 컴파일 과정을 통해 기계어로 변환한 후 컴퓨터에서 사용

저급언어
: 기계어에 가까운 언어 (ex 어셈블리어)

자바의 특징
① 이식성이 높다.
- 이식성
- 서로 다른 환경을 가진 시스템 간에 프로그램을
 옮겨 실행할 수 있는 것.
* 자바언어로 개발된 프로그램은 소스파일을 수정하지
 않아도 JRE(Java Runtime Environment)가
 설치되어 있는 모든 OS에서 실행가능하다.

② 객체지향언어이다.
 부품에 해당하는 객체들을 먼저 만들고,
 이것들을 하나씩 조립하고 연결하여 전체 프로그램을
 완성하는 개념 (OOP: Object oriented Programming)
- 자바는 OOP의 캡슐화, 상속, 다형성을 완벽히 지원

③ 함수적 스타일 코딩 지원
함수적 프로그래밍은 대용량 데이터의 병렬처리,
 이벤트 지향 프로그래밍을 위해 적합하다.
→ 자바는 Lambda Expression을 Java 8부터
 지원하기 때문에 함수적 프로그래밍이 가능하다.
람다식 사용시>
컬렉션의 요소를 필터링, 매핑, 집계하는데 쉬우며
코드가 간결해진다

Programming is art and cook
What is programming?
"Programming" is the art and Science of translating a set of ideas into a program - a list of instructions a computer can follow
Higher level languages abstract an increasing amount of the platform from the programmer
Lower level languages are more closely tied to the platform they are targeted ~~for~~

Java> Characteristics ~~and feature~~ of Java

① Java is a Simple language
② An object - oriented model
③ platform independent
④ Robust
⑤ It supports multi- threading
⑥ High performance
⑦ It offers Strong Security
⑧ A portable language
⑨ Architectural neutral
⑩ It is distributed

① Portable
- Java programs can execute in any environment for which there is a JVM
- Java programs can be run on any platform
- Java programs can be transfered over World wide web

② Object oriented

Object-oriented means we organize our software as a combination of different types of objects that incorporates both data and behavior

[The basic concepts of OOP]

- Object
- class
- Inheritance
- Polymorphism
- Abstraction
- Encapsulation

- Java programming is object-oriented programming language
- Like C++ java provides most of the object oriented features
- Java is pure oop language (while C++ is semi object oriented

③ Java8 - Functional programming

Java 8 introduced a new cool feature to support functional programming, called lambda expression. Before Java8, anonymous class was the only way to implement functional style code

Java7>

```
public static void main(String[] args){
  ...
  button.addActionListener(new ActionListener(){
     public void actionPerformed(ActionEvent e){
       System.out.println("You clicked me!");
     }
  });
}
```

Java8>

```
public static void main(String[] args){
 ...
  button.addActionListener(
     event -> {System.out.println("You clicked me!")}
  );
}
```

<Using Lambda>

It makes it easier to filter, map and aggregate the elements of a collection and simplify the code

④메모리를 자동관리한다
자바는 개발자가 직접 메모리에 접근할 수 없도록 설계되었으며, 메모리는 자바가 직접관리한다.
객체 생성시 자동적으로 메모리영역을 찾아서 할당하고, 사용이 완료되면 쓰레기 수집기를 실행시켜 자동적으로 사용하지 않는 객체를 제거시켜준다.
→개발자는 메모리관리의 수고를 덜고, 핵심기능 코드 작성에 집중할 수 있다

※ C++은 메모리에 생성된 객체를 제거하기 위해 개발자가 직접 코드를 작성해야한다

④Memory management(Robust)
Robust means Strong
Java uses Strong memory management.
There are lack of pointers that avoids Security problems.
There is automatic garbage collection in Java
There is exception handling and type checking mechanism in Java
All these points makes Java robust.
But the main areas which Java improved were mishandled Exceptions by introducing automatic Garbage Collector and Exception handling

Java SE (Standard Edition)
=JVM + Library API

need $\boxed{\text{JDK}}$

⑤멀티 스레드를 쉽게 구현
- 멀티 스레드 프로그래밍
: 하나의 프로그램이 동시에 여러 가지 작업을 처리해야 할 경우나 대용량 작업을 빨리 처리하기 위해 서브 작업으로 분리해서 병렬로 처리하기 위해 필요

⑤ Multi threaded
·Java provides Integrated Support for multithreaded programming.

ⓒ 동적로딩(Dynamic loading)지원
객체가 필요한 시점에 클래스를 동적으로
로딩해서 객체를 생성한다.
개발 후 유지보수가 발생하더라도
해당 클래스만 수정하면 된다.
(전체 App 컴파일 필요 X)
⇒ 유지보수를 쉽고 빠르게 진행할 수 있다.

Create an object by loading the class
dynamically at the point where the object
is needed
~~After Development~~
Need maintenance after development.
Only need to modify the classes
⇒ Maintanance can be done quickly and easily.

<OOP : object - oriented Programming>

① Encapsulation.
: 객체의 필드, 메소드를 하나로 묶고
실제 구현 내용을 감추는 것을 말한다.
- 캡슐화 하여 보호하는 이유는
외부의 잘못된 사용으로 인해 객체가 손상되지
않도록 하는 데 있다.
- 접근 제한자 (Access Modifier)의 사용
: 객체의 필드와 메소드의 사용 범위를
제한함으로써 외부로부터 보호한다.

② Inheritance
상위 객체는 자기가 가지고 있는 필드와
메소드를 하위 객체에게 물려주어
하위 객체가 사용할 수 있도록 해준다.
상속)은 상위 객체를 재사용해서
하위 객체를 쉽고 빠르게 설계할 수 있도록
해준다.
반복된 코드의 중복을 줄여준다.
유지 보수 시간을 최소화시켜준다.

③ Polymorphism
같은 타입이지만 실행 결과가 다양한 객체를
이용할 수 있는 성질.

① Encapsulation
Defined as the wrapping up of data under a
single unit.
It's the mechanism that binds together code
and the data it manipulates.
It's a protective shield that prevents the
data from being accessed by the code
outside this shield
『Advantage of Encapsulation』
- Data Hiding
- Increased Flexibility
- Reusability
- Testing code is easy

② Inheritance
A mechanism in which one object acquires
all the properties and behaviors of a
parent object.
Why use?
→ For Method overriding, Code Reusability.

③ Polymorphism
Polymorphism is the ability of an
object to take on many forms.
The most common use of Polymorphism
in OOP occurs when a parent class
reference is used to refer to a child
class object.