**<객체 지향 프로그래밍>**

**2015.04.29 수요일**

1. **객체란?**

-객체란 물리적으로 존재하거나 추상적으로 생각할 수 있는 것 중에서 자신의 속성을 가지고 있고 다른 것과 식별 가능한 것을 말한다.

-예) 자동차, 자전거, 책, 사람 등등..

-객체는 속성과 동작으로 구성되어 있고, 자바에서는 (속성: 필드, 동작: 메소드)라고 부른다.

1. **클래스란?**

-객체를 묶는 하나의 설계도라고 볼 수 있다. 이는 인스턴스의 객체를 위한 상태, 행위를 정의하는 코드가 된다.

-자바는 최소한 하나 이상의 클래스를 갖는다.

-모든 실행 가능한 문장은 반드시 클래스 안에 있어야 한다.

-추상클래스는 다른 클래스들을 위한 청사진을 제공한다.

-서브 클래스는 클래스의 능력을 확장한 또 하나의 클래스가 된다.

-클래스는 final로 선언 될 수 있으나, 그 클래스는 확장 될 수 없다.

1. **객체 지향 프로그래밍의 특징**

**2-1. 캡슐화 (Encapsulation)**

-캡슐화란 객체의 필드, 메소드를 하나로 묶고, 실제 구현 내용을 감추는 것을 말한다.

-외부 객체는 객체 내부의 구조를 알지 못하며, 객체가 노출해서 제공하는 필드와 메소드만 이용할 수 있다.

-필드와 메소드를 캡슐화하여 보호하는 이유: 외부의 잘못된 사용으로 인해 객체가 손상되지 않도록 하기 위해서이다.

-캡슐화를 위하여 접근 제한자 (Access Modifier)를 사용하여 객체의 필드와 메소드의 사용 범위를 제한함으로써 외부로부터 보호한다.

**2-2. 상속 (Inheritance)**

-객체 지향 프로그래밍에도 부모 역할의 상위 객체, 자식 역할의 하위 객체가 있다.

-상위 객체는 자기가 가지고 있는 필드와 메소드를 하위 객체에게 물려주어 하위 객체가 사용할 수 있도록 해준다.

-상속은 상위 객체를 재사용함으로, 하위 객체를 쉽고 빨리 설계할 수 있게 해주며, 재사용성을 증가시켜 반복된 코드의 중복을 줄인다.

**2-3. 다형성 (Polymorphism)**

-다형성이란 같은 타입이지만 실행 결과가 다양한 객체를 이용할 수 있는 성질을 뜻한다. (즉, ‘여러 가지 형태를 가질 수 있는 능력’)

-자바에서는 한 타입의 참조변수로 여러 타입의 객체를 참조할 수 있도록 함으로써 다형성을 프로그램적으로 구현하였다.

-구체적으로 말하자면, **조상클래스 타입의 참조변수로 자손클래스의 인스턴스를 참조할 수 있도록 하였다**는 것이다.

-예) 대표적인 다형성의 예: ‘오버로딩’

|  |
| --- |
| package org.opentutorials.javatutorials.polymorphism;  class O{  public void **a**(int param){  System.out.println("숫자출력");  System.out.println(param);  }  public void **a**(String param){  System.out.println("문자출력");  System.out.println(param);  }  }  public class PolymorphismOverloadingDemo {  public static void main(String[] args) {  O o = new O();  o.a(1);;  o.a("one");  }  } |

클래스 O의 메소드 a는 두개의 본체를 가지고 있다. 동시에 두개의 본체는 하나의 이름인 a를 공유하고 있다. 같은 이름이지만 서로 다른 동작 방법을 가지고 있기 때문에 오버로딩은 다형성의 한 예라고 할 수 있다.