

클래스 개념 정리.

OOP(Object Oriented Project)

Object : 객체

Oriented : 지향

Programming : 프로그래밍

절차지향 프로그래밍은 데이터가 함수 주변을 맴돌지만

객체지향은 함수가 프로그래밍 주의를 맴돈다.

실제 사물들의 개념을 컴퓨터에 이식시켜 객체간 이벤트가 발생하게 하는 것.

객체(object): 하나로서 개념을 가지는 모든 것들

- 속성(Attribute): 값 또는 data라고 한다 이 곳을 정보은닉을 해야한다.(Private) (getter/setter)
- 기능(): 동작, 행동

클래스(Class): 객체의 속성과 기능을 모아둔 객체를 만들기 위한 설계도

객체지향프로그래밍의 4가지 핵심 : 캡상추다


캡슐화 (Encapsulation)

- 직접 접근 제한의 원칙에 따라 정보은닉 하는 것.
 - 이 기능 덕분에 외부에서 어떤 방식이든 손상 시키는 것을 방지 할 수 있다.
- 속성 그리고 기능을 하나의 클래스안에 묶어 보호하는 거
- 객체 각각은 독립적으로 작용할 수 있도록 응집도 강해야 하고 다른 모듈을 참조하는 결합도는 낮아야 한다.

상속

- 부모 자식으로 나뉘져 있어 자식이 부모클래스+자신으로 구성 되는것.
- 상속은 자식 클래스를 외부로 부터 은닉하는 캡슐화의 일종이다.
 - ※ 코드 재사용을 목적으로 하는 상속 행위는 '엄격하게 금지한다'
 - 1.부모 클래스의 변경이 불편해진다
 - 부모 클래스에 의존하는 자식 클래스가 많을때 부모 클래스의 변경이 필요하다면 이를 의지하는 자식클래스들이 영향을 받는다
 - 2.불필요한 클래스의 증가
 - 유사 기능 확장시 필요 이상의 불필요한 클래스를 만들어야할 수도 있다.
 - 3.잘못된 상속 사용
 - 같은 종류가 아닌 클래스의 구현을 재사용하기 위해 상속을 받게 되면 문제가 발생할 수 있다
 - 상승받는 클래스가 부모클래스와 IS-A관계가 아닐 때 발생

추상화 (Abstraction)

- 객체들이 공통적으로 필요로 하는 속성 혹은 동작을 하나로 추출해내는것
- 객체의 속성들중 세부적인 것들을 제외하고 연관성이 있는 것들만 모아 간략화하는것(데이터의 공통된 성질)
- 추상적인 개념에 의존하면서 설계해야 유연함을 갖출 수 있다 즉 세부적인 사물들의 공통적인 특징을 파악한후 하나의 묶음으로 만들어 내는 것이 추상화다
 - 자동차의 추상화(엔진,타이어,핸들,브레이크 같이 모든 자동차가 필요한 기본적인 기능을 모으는 것)
 - 
 -

다형성 (Polymorphism)

- 서로 다른 클래스의 객체가 같은 동작 수행 명령을 받았을때 각자의 특성에 맞는 방식으로 동작하는 것.
- 다형성 구현을 통해 코드를 간결하게 해주고 유연함을 갖추게 해준다.
- 상속관계에 있다면 새로운 자식 클래스가 추가되어도 부모 클래스의 함수를 참조해오면 되기 때문에 다른 클래스는 영향을 받지 않는다.

클래스(Class)

- 클래스의 정의 : 객체를 정의해 놓는것
- 클래스의 용도 : 객체를 생성하는데 사용

- 객체의 정의 : 실제로 존하는 것 사물 혹은 개념
- 객체의 용도 : 객체가 가지고 있는 기능과 속성에 따라 다름

TV설계도 (클래스) -> TV(제품)

붕어빵 기계 (클래스) -> 붕어빵(제품)

- 객체의 속성 : 크기 길이 높이 색상 채널
- 객체의 기능 : 켜기 끄기 볼륨높이기 채널 바꾸기등

객체 : 모든 인스턴스를 대표하는 일반적인 용어

인스턴스 : 특정 클래스부터 생성된 객체(tv인스턴스)

클래스(설계도) -----> 객체(인스턴스) 하는 것을 인스턴스화 한다라고 표현한다.

클래스(설계도)가 왜 필요한가?

객체를 생성하기 위해(제품)

객체가 왜 필요한가?

객체의 기능을 사용하기 위해.

객체를 사용한다는 것은?

객체가 가진 속성과 기능(method)을 사용하려고

public 접근 제한자가 있는 경우 소스파일의 이름은 반드시 public class의 이름과 일치해야 한다

하지만 public 접근 제한자가 하나도 없는 경우 소스파일의 이름은 모두 가능하다

- public 클래스는 반드시 하나만 있어야 한다.
- public 클래스의 이름은 소스 파일 이름과 일치해야 한다
- 대 소문자를 구분하기 때문에 소스파일과 public 클래스의 이름을 조심해야 한다.

객체의 생성

클래스명 변수명; // 클래스의 객체를 참조하기 위한 참조 변수 선언

변수명 = new 클래스명(); // 클래스의 객체를 생성 후 객체의 주소를 참조 변수에 저장

Tv t; // Tv 클래스 타입의 참조 변수 t를 선언

t = new Tv(); // Tv 인스턴스를 생성한 후, 생성된 Tv 인스턴스의 주소를 t에 저장

Tv t = new Tv();

t.channel=7; // Tv 인스턴스의 멤버 변수 channel의 값을 7로 한다

t.channelDown(); // Tv 인스턴스의 메소드 channelDow() 을 호출한다

System.out.println("현재 채널은" + t.channel + "입니다");

멤버는 속성과 메소드를 합친 개수

Tv t1 = new Tv();

Tv t2 = new Tv();

변수명이 다르니깐 서로 다른 값을 가진다.

t2 = t1; // 참조 변수 t1의 값을 t2에 저장.

클래스의 명명 규칙

[접근제한자][예약어] class 클래스명 { ... }

접근제한자

- public
- protected
- (default)
- private

필드(Field): 객체의 속성을 작성하는 클래스 내부 영역.

멤버 변수 : 메소드 외부에서 작성된 변수

- 멤버 변수는 클래스 외부에서 접근할 수 있게 public으로 선언한다.

인스턴스 변수 : 필드 영역에서 작성된 변수

클래스 변수 : 필드에 static 예약어가 작성된 변수(같은 클래스로 만들어진 객체가 값을 공유할 수 있기 때문에)

[접근제한자] [예약어] 자료형 변수명 [=초기값] 으로 구성된다.

필드의 접근 제한자의 뜻 : 직접 접근이 가능한 범위를 나타낸다