Ch7
상속

객체지향프로그래밍(기본) 2019

경상대학교 항공우주및소프트웨어공학전공

7. 상속

상속(Inheritance) 이란 부모로부터 자식이 무엇인가를 물려받는 것을 말한다.

객체 지향에서 상속이란 부모 클래스로부터 자식클래스가 특성을 계승 받는 것을 말한다.

자바에서 상속은 extends 키워드로 부모클래스로부터 접근이 허가된 인스턴스 변수나 메소드를 자식클래스에서 따로 선언 하지 않고 사용 할 수 있는 것을 말한다.

인스턴스 변수와 메소드는 클래스의 멤버라 하며 상속은 자식클래스가 부모클래스의 멤버를 물려 받는 것을 의미한다.

상속

상속을 받았다고 해서 부모클래스의 모든 자원에 접근 할 수 있는 것은 아니다. 자식 클래스에서 private 으로 선언된 부보 클래스의 인스턴스 변수는 직접 접근 할 수 없다.

public으로 선언된 메소드의 경우 자식 클래스 내에 선언이 되어 있지 않아도 사용 할 수 있다.

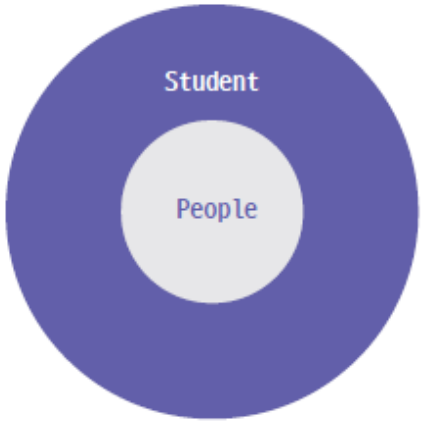

그림 7-1 People을 상속받은 Student 객체

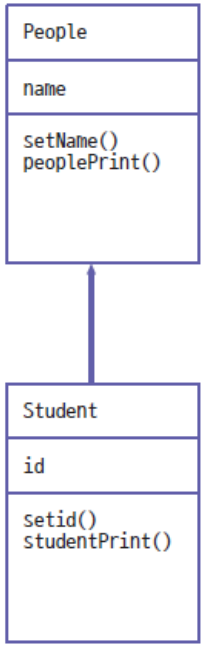

그림 7-2 상속 다이어 그램

7.1 상속의 개념

Student 클래스는 People 클래스를 상속 받았다. Student는 People에 속한다고 할 수 있다.

상속의 관계를 맺을 때 단방향의 IS-A 관계가 성립될 경우 상속 관계를 맺는 것이 좋다.

"학생은 사람이다."(Student is a People.)로 표현 할 수 있는 경우 상속관계를 맺는다.

이때 반대로 "사람은 학생이다."의 관계는 성립되지 않는다.

HAS-A 관계

상속을 검증 할 때 많이 범하게 되는 실수는 HAS-A 관계일 경우이다.

"학교에 학생이 있다."(School has a Student)와 같이 포함되는 관계일 경우 상속을 이용하기 보단 클래스의 인스턴스 변수로 객체를 포함 하도록 설계한다.

```
class School{
 Student std;
}
```

Object 클래스

자바에서 모든 클래스는 Object 클래스를 자동으로 상속 받는다.

모든 객체는 Object 클래스의 메소드가 사용 가능하다.

Object 클래스의 주요 메소드는 다음과 같다.

메소드	기능
boolean equals(Object obj)	두 객체가 같은지 비교
String toString()	객체를 문자열로 반환
Class getClass()	객체의 클래스 형을 반환
int hashCode()	객체의 코드 값을 반환

7.2 클래스의 상속

상속은 동일한 행동을 하는 클래스가 여러 개 존재 할 때 부모 클래스를 생성하며 다른 클래스들을 품을 수 있도록 한다.

```
public class People {
  private String name;
```

```
public class Student extends People{
  private int id;
```

```
public class Professor extends People{
  private int office;
```

상속 관계 에서 생성자 호출

상속 관계에서 객체를 생성 할 때 먼저 부모클래스의 생성자가 먼저 실행된 후 자식클래스의 생성자가 실행된다.

부모클래스의 인스턴스 변수와 메소드를 자식클래스에서 사용하기 때문에 부모클래스의 생성자가 먼저 호출된다.

```
Student std = new Student();

Professor prof = new Professor();
```

```
People 생성자 호출
Student 생성자 호출
People 생성자 호출
Professor 생성자 호출
```

7.3 메소드 오버라이드

부모 클래스에 정의된 메소드를 동일한 이름과 형태로 자식 클래스에서 재정의하여 사용하는 것을 메소드 오버라이드라 한다.

부모클래스의 오버라이드된 메소드에 접근하기 위해 super 키워드를 사용 한다.

```
public class People {
  private String name;
  public void peoplePrint()
  {
    System.out.println("이름: "+ this.name);
  }
}
```

```
public class Student extends People{
  private int id;

  public void peoplePrint()  <-- 메소드 오버라이드
  {
    System.out.println("---학생---");
    super.peoplePrint();  <-- 부모클래스의 peoplePrint()
    System.out.println("학번: "+this.id);
  }
}
```

객체 배열과 메소드 오버라이드

```
Student std = new Student();
Professor prof = new Professor();


std.setName("Kim");
std.setId(12345);


prof.setName("Park");
prof.setOffice(365);


People p[] = new People[2];
p[0] = std;
p[1] = prof;


for(int i = 0; i<2; i++)
  p[i].peoplePrint();
```

People 클래스 객체 배열 생성

메소드 오버라이드를 이용하여 동일한 이름의 메소드를 생성 할 경우 부모 클래스의 객체 배열을 이용하여 반복문으로 여러 객체의 메소드 호출이 가능하다.

```
---학생---
이름: Kim
학번: 12345
---교수---
이름: Park
사무실: 365
```