10장 상속과 인터페이스

상속과 인터페이스

◈ 상속

이미 존재하는 클래스를 확장해서 새로운 클래스를 만드는 기술

기존 클래스가 가지고 있는 데이터 구조와 기능을 그대로 물려받아 사용하는 기술

클래스 간의 계층적 관계를 구성함으로써 다형성의 문법적 토대를 마련

extends 키워드 사용

상속 받는 클래스를 자식 클래스, 상속을 해주는 클래스를 부모 클래스라 한다.

자바에서는 기본적으로 하나의 부모클래스만을 가질 수 있는 단일 상속만 허용한다.

상속과 인터페이스

◈ 상속과 생성자

생성자를 통한 부모 클래스 필드 초기화 방법

생성자가 있는 부모 클래스를 상속하는 방법

◈ super

현재 객체의 상위인 부모 클래스(super 클래스)의 객체 변수로, 부모 클래스로부터 상속 받은 필드나 메서드를 자식 클래스에서 참조하는 데 사용

◈ super()

부모클래스의 생성자를 의미하며, 부모클래스의 멤버를 초기화하기 위해서는 자식 클래스의 생성자에서 부모 클래스의 생성자까지 호출해야 한다.

상속과 인터페이스

◈ 메서드 오버라이딩(Overriding)

자식클래스에서 부모클래스의 메서드를 재정의하는 것

부모클래스의 메서드와 똑같은 메서드를 선언해서 상위클래스의 메서드를 무시하거나 기능을 추가한다.

◈ 다형성(Polymorphism)

클래스 변수의 다형성은 여러 종류의 객체들을 똑같은 로직으로 처리하는 프로그램을 작성할 수 있다.

◈ 객체간 형 변환(Casting)

서로 상속 관계에 있는 객체들 사이에는 형 변환을 할 수 있다.

다형성에 이용한다.

상속과 인터페이스

◈ 추상 클래스(abstract class)

미완성 클래스로, 하나 이상의 추상 메서드를 포함하면 추상 클래스가 됨

자체적으로 객체를 생성하지 못하며, 반드시 상속을 통해 하위 클래스에서 추상 클래스의 추상 메서드를 오버라이딩해야 객체 생성이 가능하다.

◈ 추상 메서드(abstract method)

프로토타입(prototype)만을 가진 메서드로, 몸체[블록]가 없는 메서드

```
public abstract class Account {
    public abstract void deposit();
    public abstract void withdraw();
}
```

상속과 인터페이스

◈ 인터페이스(Interface)

인터페이스는 클래스 작성할 때 기본이 되는 틀을 제공하기 위해 사용한다.

오로지 추상 메서드와 상수 만을 포함

다중 상속을 가능하게 한다.

인터페이스 간 상속 및 다중 상속이 가능하다.

콜백함수(Callback)를 구현하는 데에도 쓰인다.

상속과 인터페이스

◈ Instanceof

객체의 타입을 확인하는데 사용

객체 + Instanceof + 클래스 의 형태로 사용하며, 좌측 객체가 우측 클래스로 형변환이 가능한지에 대해 true/false로 값을 리턴한다.

```
public static void main(String[] args)  {
    Child child = new Child();
    Parent parent = new Parent();
    Parent childToParent = new Child();

    System.out.println(child instanceof Parent);     // true, 자식객체가 부모 객체로 형변환이 가능
    System.out.println(child instanceof Child);      // true, 객체 본인의 클래스로는 항상 true 리턴
    System.out.println(parent instanceof Parent);
    System.out.println(parent instanceof Child);              // false, 부모객체가 자식 객체로는 형변환할 수 없다
    System.out.println(childToParent instanceof Child);       // true, 자식클래스를 부모클래스의 객체로 생성한 경우엔 자식클래스로 형변환이 가능하다.
    System.out.println(childToParent instanceof Parent);      // true, 객체 본인의 클래스
}
```

패키지와 접근제어

◈ 패키지(package)

서로 연관된 클래스나 인터페이스를 한 데 묶어서 관리하는 단위 (폴더)

단순한 폴더의 개념보다는 식별자의 역할도 한다.

◈ 타 패키지의 클래스 사용법

import 및 절대 경로명

```
import java.time.LocalDateTime;
import java.time.format.DateTimeFormatter;
```