스마트콘텐츠와 웹 융합
응용sw개발자 양성과정

강사 - Innova Lee(이상훈)

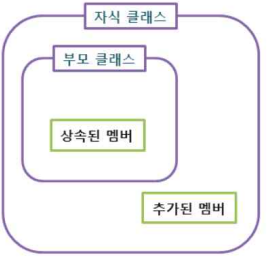
학생- jonghyeon Yoo(유종현)

상속의 개념

상속(inheritance)이란?

상속(inheritance)이란 기존의 클래스에 기능을 추가하거나 재정의하여 새로운 클래스를 정의하는 것을 의미합니다.
이러한 상속은 캡슐화, 추상화와 더불어 객체 지향 프로그래밍을 구성하는 중요한 특징 중 하나입니다.

상속을 이용하면 기존에 정의되어 있는 클래스의 모든 필드와 메소드를 물려받아, 새로운 클래스를 생성할 수 있습니다.
이때 기존에 정의되어 있던 클래스를 부모 클래스(parent class) 또는 상위 클래스(super class), 기초 클래스(base class)라고도합니다.
그리고 상속을 통해 새롭게 작성되는 클래스를 자식 클래스(child class) 또는 하위 클래스(sub class), 파생 클래스(derived class)라고도 합니다.



상속의 장점

자바에서 클래스의 상속은 다음과 같은 장점을 가집니다.

1. 기존에 작성된 클래스를 재활용할 수 있습니다.
2. 자식 클래스 설계 시 중복되는 멤버를 미리 부모 클래스에 작성해 놓으면, 자식 클래스에서는 해당 멤버를 작성하지 않아도 됩니다.
3. 클래스 간의 계층적 관계를 구성함으로써 다형성의 문법적 토대를 마련합니다.

자식 클래스(child class)

자식 클래스(child class)란 부모 클래스의 모든 특성을 물려받아 새롭게 작성된 클래스를 의미합니다.

자바에서 자식 클래스는 다음과 같은 문법을 통해 선언합니다.

문법

```
class 자식클래스이름 extend 부모클래스이름 { ... }
```

예제

```
class Parent {
    private int a = 10; // private 필드
    public int b = 20;  // public 필드
}

class Child extends Parent {
    public int c = 30;  // public 필드
    void display() {
①       // System.out.println(a); // 상속받은 private 필드 참조
②       System.out.println(b);    // 상속받은 public 필드 참조
③       System.out.println(c);    // 자식 클래스에서 선언한 public 필드 참조
    }
}

public class Inheritance01 {
    public static void main(String[] args) {
        Child ch = new Child();
        ch.display();
    }
}
```

실행 결과

```
20
30
```

위 예제의 ②번 라인에서는 자식 클래스의 메소드에서 부모 클래스에서 상속받은 public 필드를 참조하고 있습니다.
이처럼 자식 클래스에서 따로 선언하지 않은 필드라도 해당 이름의 필드를 부모 클래스에서 상속받았다면 문제가 없습니다.
하지만 주석 처리된 ①번 라인처럼 해당 필드가 부모 클래스의 private 필드라면 접근할 수 없으므로, 오류를 발생시킬 것입니다.
또한, 자식 클래스에서는 ③번 라인처럼 자신만의 필드나 메소드를 선언하여 사용할 수 있습니다.

super

super 키워드

super 키워드는 부모 클래스로부터 상속받은 필드나 메소드를 자식 클래스에서 참조하는 데 사용하는 참조 변수입니다.

인스턴스 변수의 이름과 지역 변수의 이름이 같을 경우 인스턴스 변수 앞에 this 키워드를 사용하여 구분할 수 있었습니다.
이와 마찬가지로 부모 클래스의 멤버와 자식 클래스의 멤버 이름이 같을 경우 super 키워드를 사용하여 구별할 수 있습니다.

이렇게 자바에서는 super 참조 변수를 사용하여 부모 클래스의 멤버에 접근할 수 있습니다.
this와 마찬가지로 super 참조 변수를 사용할 수 있는 대상도 인스턴스 메소드뿐이며, 클래스 메소드에서는 사용할 수 없습니다.

예제

```
class Parent { int a = 10; }

class Child extends Parent {
    void display() {
        System.out.println(a);
        System.out.println(this.a);
        System.out.println(super.a);
    }
}

public class Inheritance02 {
    public static void main(String[] args) {
        Child ch = new Child();
        ch.display();
    }
}
```

실행 결과

```
10
10
10
```

위의 예제에서 int형 변수 num는 부모 클래스인 Parent 클래스에서만 선언되어 있습니다.
따라서 지역 변수와 this 참조 변수 그리고 super 참조 변수 모두 같은 값을 출력합니다.

예제

```
class Parent {
    int a = 10;
}

class Child extends Parent {
    int a = 20;

    void display() {
        System.out.println(a);
        System.out.println(this.a);
        System.out.println(super.a);
    }
}

public class Inheritance03 {
    public static void main(String[] args) {
        Child ch = new Child();
        ch.display();
    }
}
```

실행 결과

```
20
20
10
```

하지만 위의 예제에서 int형 변수 num는 자식 클래스인 Child 클래스에서도 선언되어 있습니다.

따라서 지역 변수와 this 참조 변수는 자식 클래스에서 대입된 값을 출력하며, super 참조 변수만이 부모 클래스에서 대입된 값을 출력하게 됩니다.

추상화란?

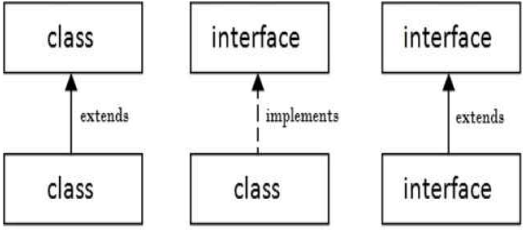
- 추상의 뜻 : 여러 가지 사물이나 개념에서 공통되는 특성이나 속성 따위를 추출하여 파악하는 작용.

- 일반화라고 생각하면 된다.

- 추상화를 사용하면 코드의 재사용성, 가독성을 높이고, 생산성의 증가, 에러의 감소, 유지 보수에 있어 많은 시간을 줄일수 있다.

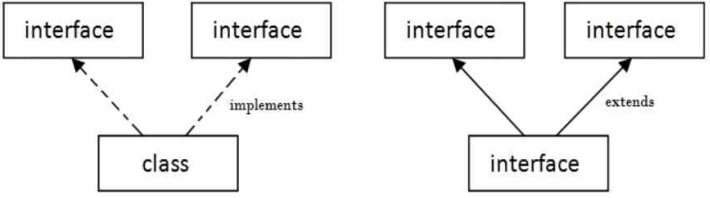
클래스와 인터페이스 사이의 관계 이해하기.

아래의 이미지에서 보는 것과 같이, 클래스는 다른 클래스를 extends 하고 인터페이스는 다른 인터페이스를 extends 하지만 클래스는 인터페이스를 implements 합니다.



인터페이스에 의한 자바에서의 다중 상속.

만약 한 클래스가 여러 인터페이스들을 implements 한다면, 또는 한 인터페이스가 여러 인터페이스들을 extends 한다면 알려진 것과 같이 다중상속 입니다.



인터페이스 상속

클래스는 인터페이스를 implements 하지만 인터페이스는 다른 인터페이스를 extends 합니다.

인터페이스의 장점

인터페이스를 사용하면 다중 상속이 가능할 뿐만 아니라 다음과 같은 장점을 가질 수 있다.

- 1. 대규모 프로젝트 개발 시 일관되고 정형화된 개발을 위한 표준화가 가능합니다.
- 2. 클래스의 작성과 인터페이스의 구현을 동시에 진행할 수 있으므로, 개발 시간을 단축할 수 있습니다.
- 3. 클래스와 클래스 간의 관계를 인터페이스로 연결하면, 클래스마다 독립적인 프로그래밍이 가능합니다.