KH 정보교육원

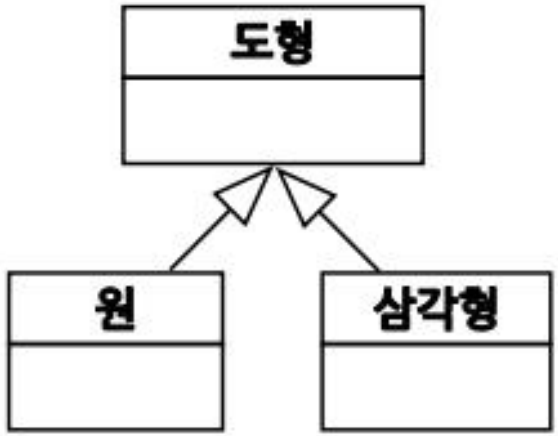
다형성
(polymorphism)

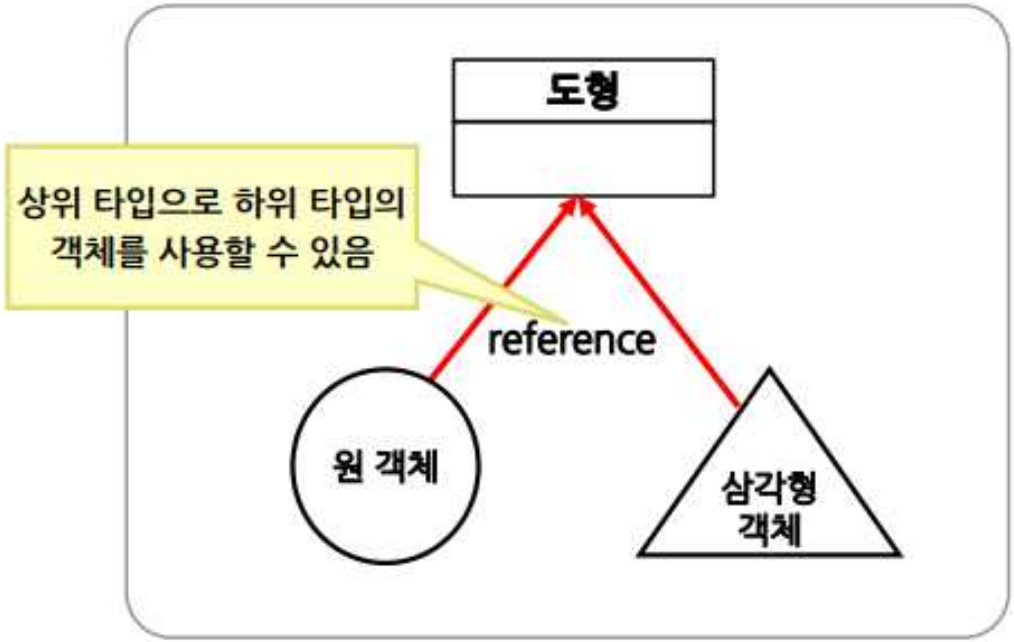
다형성

KH정보교육원

다형성이란?

- '여러 개의 형태를 갖는다'는 의미, 객체지향 프로그래밍의 3대 특징 중 하나
- 상속을 이용한 기술로, 자식 객체를 부모객체의 타입의 변수로 다룰 수 있는 기술





KH정보교육원

클래스 형변환 – up casting

상속관계에 있는 부모, 자식 클래스 간에 **부모타입의 참조형 변수**가 모든 자식 타입의 객체의 주소를 받을 수 있다.

예시) //Sonata클래스가 Car클래스의 후손임

```
Car c = new Sonata();
```

Sonata 클래스형 -> Car 클래스 형으로 바뀜

※ 자식객체의 주소를 전달받은 부모참조변수를 통해서 사용할 수 있는 자식의 정보는, 원래 부모의 것이었던 멤버만 참조 가능

KH정보교육원

클래스 형변환 – down casting

자식객체의 주소를 받은 부모 참조형 변수를 가지고 자식의 멤버를
참조해야 할 경우, 후손 클래스 타입으로 참조형 변수를 형 변환해야 한다.
이 변환을 down casting이라고 하며, 자동으로 처리되지 않기 때문에
반드시 후손 타입을 **명시**해서 형 변환 해야 한다.

예시) //Sonata클래스가 Car클래스의 후손임

```
Car c = new Sonata();
((Sonata)c).moveSonata();
```

※ 클래스간의 형 변환은 **반드시 상속관계**에 있는 클래스끼리만 가능함

KH정보교육원

instanceof 연산자

현재 참조형 변수가 어떤 클래스 형의 객체 주소를 참조하고 있는지 확인할 때 사용, 클래스타입이 맞으면 true, 아니면 false값을 반환

표현식

```
if( 레퍼런스 instanceof  클래스타입){
        참일 때 처리할 내용
         //해당 클래스 타입으로 down casting
}
```

객체배열과 다형성

다형성을 이용하여 상속관계에 있는 여러 개의 자식클래스를 부모 클래스의 배열에 저장 가능

** 객체배열 : 같은 자료의 클래스를 여러개 보관할 수 있는 저장공간

예시

```
Car[] carr = new Car[5]; //Car 부모클래스

carr[0] = new Sonata(); //자식클래스
carr[1] = new Avante(); //자식클래스
carr[2] = new Spark();
carr[3] = new Morning();
carr[4] = new Granduer();
```

KH정보교육원

추상클래스(abstract class)

몸체 없는 메소드(abstract가 있는 매소드)를 포함한 클래스

추상 클래스일 경우 클래스 선언부에 abstract 키워드를 사용

표현식

[접근제한자] abstract class 클래스명 {}

예시

```
public abstract class Car {  // 추상클래스
        private int x, y;
        abstract void move(int x);
}
```

다형성

KH정보교육원

추상메소드(abstract method)

몸체({}) 없는 메소드를 추상 메소드라고 한다.

추상 메소드의 선언부에 abstract 키워드를 사용한다.

표현식

[접근제한자] abstract void 메소드명();

예시

```
public abstract class Car {
        private int x, y;
        abstract void move(int x); //추상매소드
}
```

KH정보교육원

추상클래스의 특징

- 미완성 클래스(abstract 키워드 사용)
- abstract 메소드가 포함된 클래스 -> 반드시 abstract 클래스
- 자체적으로 객체 생성 불가 -> 반드시 상속하여 객체 생성
- 일반적인 메소드, 변수도 포함할 수 있다.
- abstract메소드가 없어도 abstract 클래스 선언 가능하다.
- 객체 생성은 안되나, 참조형 변수 type으로는 사용 가능하다.

다형성

KH정보교육원

인터페이스란?

상수형 필드와 추상 메소드만을 작성할 수 있는 추상 클래스의 변형체이다.

메소드의 통일성을 부여하기 위해서 추상 메소드만 따로 모아 놓은것으로,

상속시 인터페이스 내에 정의된 모든 추상 메소드를 구현해야 한다.

표현식

```
[접근제한자] interface 인터페이스명 {
        //상수도 멤버로 포함할 수 있음
        public static final 자료형 변수명 = 초기값;

        //추상메소드만 선언 가능
        [public abstract] 반환자료형 메소드명([자료형 매개변수]);
        //public abstract가 생략되기 때문에 오버라이딩시
        //반드시 public 표기 해야 함
}
```

다형성

KH정보교육원

인터페이스의 특징

- 모든 인터페이스의 메소드는 묵시적으로 public이고 abstract이다.
- 변수는 묵시적으로 public static final이다.
  -> 따라서 인터페이스 변수의 값 변경 시도는 컴파일시 에러를 발생
- 객체 생성은 안되나, 참조형 변수로서는 가능하다.

인터페이스의 장점

- 상위 타입의 역할로 다형성을 지원하여 연결해주는 역할 수행
- 해당 객체가 다양한 기능을 제공시에도 인터페이스에 해당하는 기능만을 사용하게 제한할 수 있다.
- 공통 기능상의 일관성 제공 / 공동 작업을 위한 인터페이스 제공

다형성

KH정보교육원

인터페이스 VS 추상메소드

구분	인터페이스	추상클래스
상속	• 다중상속	• 단일상속
구현	• implements 사용	• extends 사용
추상메소드	• 모든 메소드는 abstract	• abstract 메소드 0개 이상
abstract	• 묵시적으로 abstract	• 명시적 사용
객체	• 객체 생성불가	• 객체 생성불가
용도	• reference 타입	• reference 타입