

DAY12 - JAVA

≡ 태그	
📅 날짜	@2022년 10월 17일

상속 (Inheritance)

상속은 실제 세계에서의 상속처럼 부모의 자산을 자식이 물려 받아서 마치 자기의 것처럼 사용할 수 있듯이, 특정 클래스를 상속 받게 되면 그 클래스의 자산(메서드, 변수...)을 그대로 가져다가 사용할 수 있다.

다른 언어와는 달리 자바는 다중 상속을 지원하지 않는다.

일반적인 경우 자식 클래스는 부모 클래스의 모든 자산에 플러스 자신의 자산을 가지게 되므로 대부분 부모보다 자식 클래스가 더 많은 기능을 가지게 된다.

```
animal.java

package day12_1017;

public class animal {

    String name;

    void setName(String name) {
        this.name = name;
    }
}
```

Dog.java

```
package day12_1017;

public class Dog extends animal{ //extends = animal 클래스를 상속

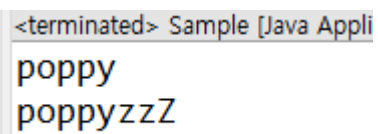
    void sleep() {
        System.out.println(this.name + "zzZ");
    }
}
```

Sample.java

```
package day12_1017;

public class Sample {

    public static void main(String[] args) {
        // TODO Auto-generated method stub
        Dog dog = new Dog();
        dog.setName("poppy");
        System.out.println(dog.name);
        dog.sleep();
    }
}
```



```
<terminated> Sample [Java Appli
poppy
poppyzzZ
```

Dog 클래스는 Animal 클래스 보다 좀 더 많은 기능

여기서는 Sleep()메서드를 가지게 된다.

보통 부모 클래스 보다 자식 클래스가 더 많은 기능을 가지게 되는게 일반적이다.

주의!!!

Dog 클래스는 animal 클래스를 상속했다, 따라서 Dog은 animal의 하위 개념이라고 볼 수 있다.

이 경우 Dog은 animal에 포함되기 때문에 ‘개는 동물이다.’ 라고 표현할 수 있다.

하지만 동물은 개만 될 수는 없다.

```
Dog dog = new Dog(); (O)
```

```
animal dog = new Dog(); // 개는 동물이다. (O)
```


```
Dog dog = new animal(); // 동물은 개다. (X) 안된다.
```

개념적으로 생각해보면 동물로 만든 객체는 개 자료형이다. 로 읽을 수 있을 것이다.

하지만 말이 안된다. 호랑이, 사자, 말 등은 개가 될 수 없다.

HouseDog.java

```
HouseDog housedog = new HouseDog();  
    housedog.setName("hongsi"); // animal(할아버지)이 가진 기능  
    housedog.sleep();           // Dog(아버지)이 가진 기능
```



```
<terminated> Sample [Java Application] C:\Program Files\Java\  
hongsiZZ
```

method overriding

부모클래스(Dog.java)가 가진 sleep() 메서드를 사용하지 않고 자식(HouseDog)이 동일한 이름의 메서드로 작성하면 자식이 가진 메서드가 호출된다.

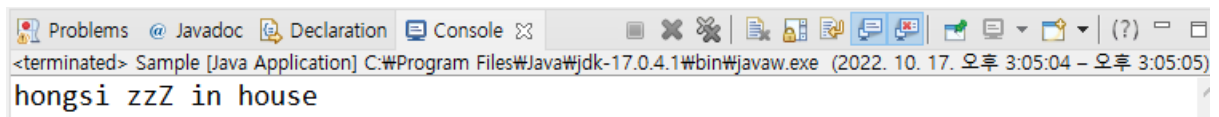
```

package day12_1017;

public class HouseDog extends Dog{
    // Dog과 Dog위에 animal을 상속받는다.

    void sleep() {
        System.out.println(this.name + " zzZ in house");
        // 부모의 기능이 마음에 안들면 새로이 만들어서 method overriding
        // 새로운걸 덮어씌우기
    }
}

```



The screenshot shows an IDE's console window. The title bar includes tabs for 'Problems', 'Javadoc', 'Declaration', and 'Console'. The console text reads: '<terminated> Sample [Java Application] C:\Program Files\Java\jdk-17.0.4\bin\javaw.exe (2022. 10. 17. 오후 3:05:04 - 오후 3:05:05)' followed by the output 'hongsi zzZ in house'.

method overloding

HouseDog.java

```

void sleep() {
    System.out.println(this.name + " zzZ in house");
    // 부모의 기능이 마음에 안들면 새로이 만들어서 method overriding
    // 새로운걸 덮어씌우기
}

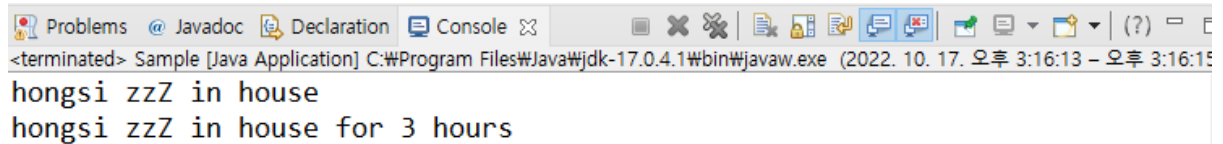
void sleep(int hour) {
    System.out.println(this.name + " zzZ in house for"
        + hour + "hours");
}

// 인수에 따라서 출력이 달라짐
// 아무인수 없이 출력하면 위에 sleep, 인수를 입력하면 아래의 sleep을 출력

```

Sample.java

```
housedog.sleep();  
housedog.sleep(3);
```



```
<terminated> Sample [Java Application] C:\Program Files\Java\jdk-17.0.4\bin\javaw.exe (2022. 10. 17. 오후 3:16:13 - 오후 3:16:15)  
hongsi zzZ in house  
hongsi zzZ in house for 3 hours
```

동일한 이름의 메서드(sleep)가 여러 개 있을 때 인수의 종류 또는 개수로 실행되는 내용을 바꾸고자 할 때는 메서드 오버로딩을 이용한다.

- 실제 사용

웹 개발 - 에러처리 클래스를 상속 받아서 에러해결

생성자 Constructor

- 생성자는 클래스명과 동일해야한다.
- 생성자는 리턴을 정의하지 않는다. void도 사용하지 않는다.

HouseDog.java

```
HouseDog(String name){  
    this.setName(name);  
}
```

Sample.java

```
HouseDog housedog = new HouseDog("hongsi"); // 인수를 던져주지 않으면 에러가 발생한다.  
// ex) HouseDog housedog = new HouseDog(); <-- 에러
```

- 생성자 자동 생성

source - generate constructor using field - generate

- 클래스에는 반드시 생성자가 존재해야만 한다.
- 개발자가 클래스를 만들때 생성자를 만들지 않으면 자바가 자동으로 기본 생성자를 생성해준다.
- 하나라도 생성자를 개발자가 직접 만들게 되면 자바는 기본생성자를 추가로 생성하지는 않는다.

기본생성자 (Default constructor)

```

animal.java
package day12_1017;

public class animal {

-----

    // 기본생성자 (Default constructor)
    // 아무것도 없는 생성자를 만들어준다.
    animal(){
    }
    -----

    String name;

    void setName(String name) {
        this.name = name;
    }
}

```

생성자 overloading

```

HouseDog.java

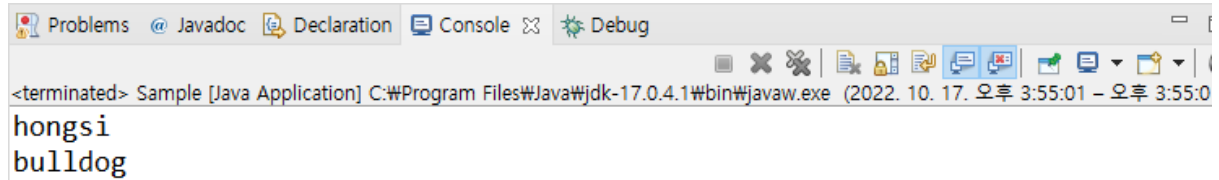
HouseDog(int type){
    if(type == 1) {
        this.setName("yorkshire");
    }else if (type == 2)
        this.setName("bulldog");

}

```

Sample.java

```
HouseDog housedog1 = new HouseDog("hongsi");  
HouseDog housedog2 = new HouseDog(2);  
  
System.out.println(housedog1.name);  
System.out.println(housedog2.name);
```



1. 소나타 자동차 객체를 생성하세요.
2. 소나타 자동차 마일리지를 10으로 설정하세요.
3. 현재 소나타 자동차의 마일리지를 출력하세요.
-현재 소나타의 마일리지는 10 Mile입니다.

