

제 25 강

다형성

교재:p129~136

목차

1. 다형성과 다형성실습

1. 다형성 개념
 - 1) 업 캐스팅
 - 2) 동적 바인딩
 - 3) 다운 캐스팅
2. 다형성 실습

클래스와 객체 복습

클래스란?

객체를 만들기 위한 설계도

객체란?

독립적으로 존재할 수 있는 것

인스턴스란?

클래스로 실제 메모리에 생성된 객체(객체가 더 포괄적인 의미)

상속 복습

상속이란?

부모 클래스의 필드를 자식 클래스가 받아오는 것

상속 방법?

[구조]

```
class 클래스명 extends 부모클래스명{  
    ...  
}
```

오버라이딩 복습

오버라이딩?

부모 클래스로부터 받아온 메서드를 자식에 맞게끔 재정의 하는 것



```
class Student{  
    void learn() {  
        System.out.println("배우기");  
    }  
    void eat() {  
        System.out.println("밥 먹기");  
    }  
    void say() {  
        System.out.println("선생님 안녕하세요~");  
    }  
}
```



```
class Leader extends Student{  
    void lead() {}  
    void say() {  
        System.out.println("선생님께 인사"); //오버라이딩  
    }  
}
```

1. 다형성 개념

다형성?

한 가지의 타입이 여러 가지 형태의 인스턴스를 가질 수 있는 것

다형성의 여러 방법: 부모 자식간의 casting(형 변환)

1. 업 캐스팅(upcasting)
2. 다운 캐스팅(downcasting)

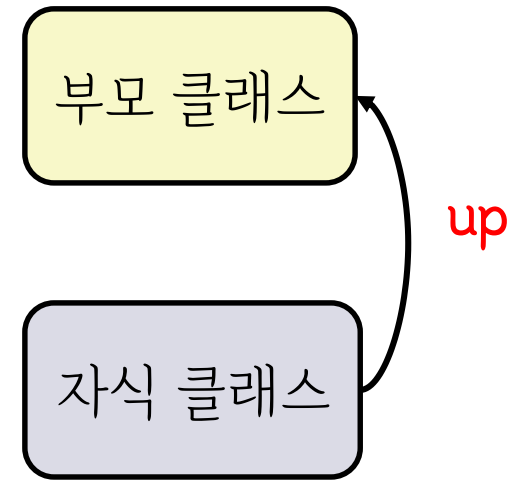
1-1) 업 캐스팅(upcasting)

- 업 캐스팅 정의

: 자식 클래스의 객체가 부모클래스의 참조 변수로 형 변환 되는 것

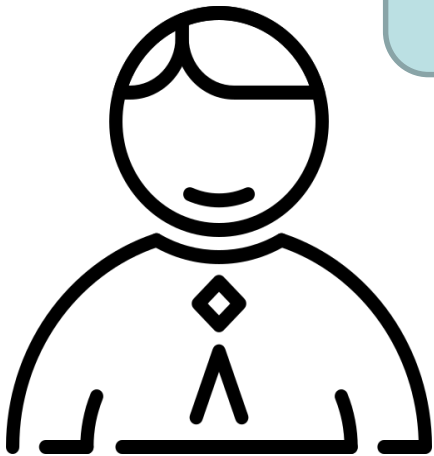
<구조>

부모클래스 변수 = 자식 객체값;



1-1) 업 캐스팅(upcasting)

<실습> Upcasting1.java



사람 클래스

이름
나이
취미



학생 클래스

이름
나이
취미
학년

1-1) 업 캐스팅(upcasting)

<실습> Upcasting1.java

```
class Human{
    String name;
    int age;
    String hobby;

    public Human(String name, int age, String hobby) {
        this.name = name;
        this.age = age;
        this.hobby = hobby;
    }

    void info() {
        System.out.println("name:"+name);
        System.out.println("age:"+age);
        System.out.println("hobby:"+hobby);
    }
}
```

```
class Student extends Human{
    int grade;

    public Student(String name, int age, String hobby, int grade) {
        super(name, age, hobby);
        this.grade = grade;
    }

    void info() {
        super.info();
        System.out.println("grade:"+grade+"학년");
    }

    void study() {
        System.out.println("공부 중~~");
    }
}
```

1-1) 업 캐스팅(upcasting)

<실습> Upcasting1.java

<구조>

부모 클래스 변수 = 자식 객체 값;

```
public class Upcasting1 {  
    public static void main(String[] args) {  
        Student s1= new Student("나길동",30,"잠자기",3);  
        s1.info();  
        s1.study();  
  
        Human h1=new Student("가길동",20,"게임하기",1);  
        h1.info();  
  
    }  
}
```

<출력 결과>

```
name:가길동  
age:20  
hobby:게임하기  
grade:1학년  
name:나길동  
age:30  
hobby:잠자기  
grade:3학년  
공부 중~~
```

1-1) 업 캐스팅(upcasting)

<실습> Upcasting1.java

```
Human h1=new Student("가길동",20,"게임하기",2);  
h1.info();  
System.out.println(h1.grade);
```

업 캐스팅이 되었기 때문에 Human 클래스의 객체 변수는 grade를 저장할 수 없다.

1-1) 업 캐스팅(upcasting)

<실습> Upcasting1.java

```
Human h1=new Student("가길동",20,"게임하기",2);  
h1.info();  
System.out.println(h1.grade);  
h1.study();
```

업 캐스팅이 되었기 때문에

Human 클래스의 객체 변수는 grade 변수 사용 불가,
Human 클래스의 객체 변수는 study() 메서드 호출 불가!

1-1) 업 캐스팅(upcasting)

<실습> Upcasting1.java

```
Human h1=new Student("가길동",20,"게임하기",1);  
h1.info();
```

<Human 클래스의 info() >

```
void info() {  
    System.out.println("name:"+name);  
    System.out.println("age:"+age);  
    System.out.println("hobby:"+hobby);  
}
```

<출력 결과>

```
name:가길동  
age:20  
hobby:게임하기  
grade:1학년
```

<Student 클래스의 info() >

```
void info() {  
    super.info();  
    System.out.println("grade:"+grade+"학년");  
}
```

업 캐스팅 시, Student클래스의
info()가 호출되는 이유?

→ 동적 바인딩!

1-2) 동적 바인딩 (Dynamic Binding)

- 동적 바인딩 정의

: 컴파일 시점에선 부모클래스의 필드로 인지하나, 런 타임 시점에선 자식클래스의 필드로 호출 할 메서드를 묶는 것

```
Human h1=new Student("가길동",20,"게임하기",1);  
h1.info();
```

컴파일 시점 바인딩

<Human 클래스의 info() >

```
void info() {  
    System.out.println("name:"+name);  
    System.out.println("age:"+age);  
    System.out.println("hobby:"+hobby);  
}
```

런타임 시점 바인딩

<Student 클래스의 info() >

```
void info() {  
    super.info();  
    System.out.println("grade:"+grade+"학년");  
}
```

1-1) 업 캐스팅(upcasting)

<실습> Exam-50.java

```
class A{
    void methodA() {
        System.out.println("methodA");
    }
}

class B extends A{
    void methodB() {
        System.out.println("methodB");
    }
}
```

```
public class Polymorphism1 {
    public static void main(String[] args) {

        A obj = new B(); //업캐스팅
        obj.methodA();
        obj.methodB();
    }
}
```

<출력 결과>

methodA

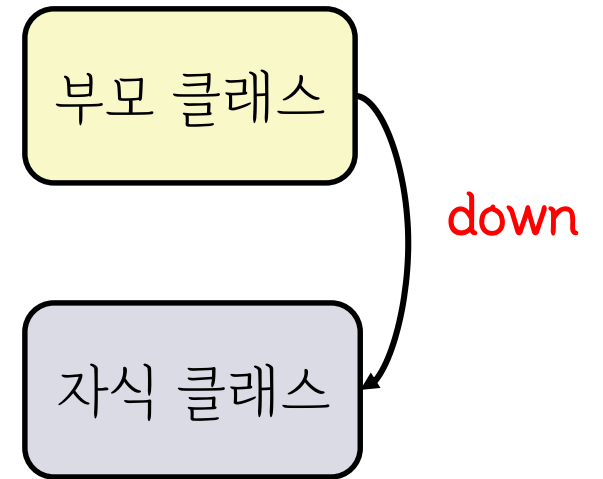
1-3) 다운 캐스팅(Down Casting)

- 다운 캐스팅 정의

: 업 캐스팅 된 부모 클래스의 객체가
자식클래스의 참조 변수로 형 변환 되는 것

<구조>

자식 클래스 변수 = (자식 클래스) 업 캐스팅 된 부모 참조 변수;



1-3) 다운 캐스팅(Down Casting)

- 구조

자식 클래스 변수 = (자식 클래스) 업 캐스팅 된 부모 참조 변수;

- 다운 캐스팅 시 주의할 점

1. 업 캐스팅 된 변수만 가능
2. 명시적 형 변환

1-3) 다운 캐스팅(Down Casting)

<실습> Downcasting1.java

```
public class Downcasting1 {  
    public static void main(String[] args) {  
        Human h1=new Student("가길동",20,"게임하기",2);  
        Student s1=(Student)h1;  
        s1.info();  
        s1.study();  
    }  
}
```

업 캐스팅으로 인해 사용하지 못했던 필드를
다운 캐스팅을 통해 다시 사용가능하도록 함

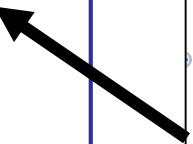
1-3) 다운 캐스팅(Down Casting)

<실습> Exam-53.java

```
class Animal{  
    void breadth() {  
        System.out.println("숨 쉬기");  
    }  
}
```

```
class ZooKeeper{  
    void feed(Animal animal) {  
        System.out.println(animal+"에게 먹이주기!");  
    }  
}
```

```
class Lion extends Animal{  
    public String toString(){  
        return "사자";  
    }  
}  
  
class rabbit extends Animal{  
    public String toString(){  
        return "토끼";  
    }  
}  
  
class Monkey extends Animal{  
    public String toString(){  
        return "원숭이";  
    }  
}
```



1-3) 다운 캐스팅(Down Casting)

<실습> Exam-53.java

```
public class Polymorphism4 {  
    public static void main(String[] args) {  
        Animal lion= new Lion();  
        Animal rabbit= new Rabbit();  
        Animal monkey= new Monkey();  
  
        ZooKeeper james=new ZooKeeper();  
  
        james.feed(lion);  
        james.feed(rabbit);  
        james.feed(monkey);  
    }  
}
```

<출력 결과>

사자에게 먹이주기!
토끼에게 먹이주기!
원숭이에게 먹이주기!