

Week 11-1 : Class Hierarchy

Part1. Inheritance

Inheritance between classes. When a child class is inherited from parent class, a child class has parent class' member.

Parent class is a superclass, and derived class is a subclass.

Inheritance keyword : extends

- Needs of Inheritance

Don't need to declare member repeatedly.

simplifying a class by reusing fields and methods

managing and classifying classes hierarchically

- Example of Inheritance

```
class Point {
    int x, y;
    void set(int x, int y) {
        this.x = x; this.y = y;
    }
    void showPoint() {
        System.out.println("x :" + x + " / y : " + y );
    }
}

public class ColorPoint extends Point {
    String color;
    void setColor(String color) {
        this.color = color;
    }
    void showColorPoint() {
        System.out.print(color);
        showPoint();
    }
    public static void main(String[] args) {
```

```
        ColorPoint cp = new ColorPoint();
        cp.set(3,4);
        cp.setColor("red");
        cp.showColorPoint();
    }
}
```

☞ ColorPoint class inherit Point class. ColorPoint type cp has ColorPoint's member and Point's member. Don't need to implement another for coordinates. Only reuse the method implemented in the Point. ColorPoint has set method about color information and show method about color information and coordinates.

- Access modifier

Class accessing member	Member's access modifier			
	default	private	protected	public
Same package class	O	X	O	O
Different package class	X	X	X	O
Same package's subclass	O	X	O	O
Different package's subclass	X	X	O	O

```
class Person {
    public String name;
    protected int height;
    private int weight;
    int age;

    public void setWeight(int weight) {
        this.weight = weight;
    }

    public void getWeigth() {
        return weight;
    }
}
```

```
public class Student extends Person {
    void set() {
        name = "Gildong Hong";
        height = 180;
        seetWeight(70);
        age = 20;
    }

    public static void main(String[] args) {
        Student s = new Student();
        s.set();
    }
}
```

public String name : Possibly accessing a Student class

protected int height : Possibly accessing a Student class
(inheritance)

private int weight : Not possibly accessing a Student class (using
methods)

int age : Possibly accessing a Student class

* this Constructor : using this constructor when parameter name
and member name are same.

Part2. Constructor

- Inheritance and Constructor

Superclass' constructor executes first, after that subclass'
constructor executes.

```
class Person {
    public String name;

    public Person() {
        System.out.println("Person Constructor");
    }
}
```

```
}  
  
public class Student extends Person {  
    public Student() {  
        System.out.println("Student Constructor");  
    }  
}  
  
public class Constructor {  
    public static void main(String[] args) {  
        Student s = new Student();  
    }  
}
```

☞ Person constructor executes first, after that student constructor executes.

[Exercise]

Make Person, Student, Constructor Classes. A Student class inherit a Person class.

1. Person Class

String name variable

no parameter constructor.

one parameter constructor : print out "person : a parameter value"

2. Student Class

no parameter constructor.

one parameter constructor :

 first line super(parameter name);

 print out "student : a parameter value"

3. Constructor Class

main method

generate Student type object ; parameter ("Minsu");

Question

1. Run the program. What is super() method?
2. In the Student class, change the two line in the Student(String x) Constructor. What happen?