Java Programming

— Inheritance and Overriding —

Waseda University

Inheritance

Inheritance

Subclass can reuse the fields and methods of the existing class(superclass) using Inheritance.

- A subclass can inherit fields and methods from a superclass.
- Constructors in a superclass are not inherited.

Template of Inheritance

```
Modifier class subclassName extends superclassName {
    Class Body
}
```

Example of superclass

Person class

Define the class `Person` which has a name, height, weight and age as fields and the method to display its fields.

Person.java

```
public class Person {
    String name;
    double height, weight;
    int age;
    public Person(){
            :
    }
    public Person(String s, double h, double w, int a){
            :
    }
    public void print(){
            :
    }
}
```

Example of a superclass

Person.java

```
public class Person {
    String name;
    double height, weight;
    int age;
    public Person(){
        name = ""; height = 0.0; weight = 0.0; age = 0;
    }
    public Person(String s, double h, double w, int a){
        name   = s; height = h; weight = w; age = a;
    }
    public void print(){
        System.out.println("name : " + name);
        System.out.println("height : " + height + " cm");
        System.out.println("weight : " + weight + " kg");
        System.out.println("age : " + age + " years old");
    }

}
```

Example of inheritance

Student.java

```
public class Student extends Person{
    String id;
    public static void main(String[] args){
        Student st1 = new Student();
        st1.name = "Frank";
        st1.height = 175.0;
        st1.weight = 63.0;
        st1.age     = 18;
        st1.id = "1w00B000";
        st1.print();
    }
}
```

Result

```
name : Frank
height : 175.0 cm
weight : 63.0 kg
age : 18 years old
```

Inheritance of the fields and methods

Person.java

```
public class Person {
    String name;
    double height, weight;
    int age;
    public Person(){
        name = ""; height = 0.0; weight = 0.0; age = 0;
    }
    public Person(String s, double h, double w, int a){
        name   = s; height = h; weight = w; age = a;
    }
    public void print(){
        System.out.println("name : " + name);
        System.out.println("height : " + height + " cm");
        System.out.println("weight : " + weight + " kg");
        System.out.println("age : " + age + " years old");
    }

}
```

Overriding and super

Extend `print` method

The print method in `Person` class displays its name, height, weight and age. The print method in `Student` class want to display its name, height, weight, age and id.

- To override the print method ⇒ overriding
- To refer the print method in Person ⇒ super

Overriding of methods

You can define the same method which has same name, parameters and return type as the method in superclass.

The keyword super

if you want to access the fields or methods in the superclass, then you use The keyword super.(e.g. super.print();)

Overriding of method

Student.java

```
public class Student extends Person{
    String id;
    @Override
    public void print(){
        super.print();
        System.out.println("id : "+ id );
    }
    public static void main(String[] args){
        Student st1 = new Student();
        st1.name = "Frank";
        st1.height = 175.0;
        st1.weight = 63.0;
        st1.age     = 18;
        st1.id = "1w00B000";
        st1.print();
    }
}
```

Constructors in a subclass (1)

Student.java

```
public class Student extends Person{
    String id;

    public Student(){
        super();
    }
    public Student(String n, double h, double w, int a){
        super(n, h, w,a);
    }
    public Student(String n, double h, double w, int a, String sID){
        super(n, h, w,a);
        id =sID;
    }
    @Override
    public void print(){
        super.print();
        System.out.println("id : "+ id );
    }
}
```

Constructors in a subclass (2)

SampleStudent.java

```
public class SampleStudent {
    public static void main(String[] args){
        Student st1 = new Student("Frank", 175.0, 63.5,18);
        st1.id      = "1w00B000";
        st1.print();

        Student st2 = new Student("Thomas", 177.0, 72.0,18,"1w00C000");
        st2.print();
    }
}
```

Constructors in a subclass (3)

Important Point

If you call the constructor in superclass, you must write the keyword super at first line in constructor.

```
public Student(String n, double h, double w, String sID){
    super(n, h, w);
    id =sID;
}
```