Object Oriented Programming using Java

Prepared By:
Suyel, PhD
Assistant Professor
Dept. of CSE, NIT Patna

Outline

1. Method Overriding
2. Examples of Method Overriding
3. Difference Between Method Overloading and Method Overriding

Method Overriding

- If subclass (child class) has the same method as declared in the parent class, it is known as **method overriding** in Java.
- Method of a subclass overrides the method of the superclass
- If an object of a parent class is used to invoke the method, the version in the parent class is executed. If an object of the subclass is used to invoke the method, the version in the child class is executed.
- Uses of Method Overriding
  - It is used to provide the specific implementation of the method that has been declared by one of its parent class.
  - Method overriding is used for runtime polymorphism.

Method Overriding (Cont…)

- **Rules for method overriding**
  - Both the superclass and the subclass must have the same method name, the same return type and the same parameter list.
  - The overriding method access specifier cannot be more restrictive than the parent class access specifier. For example, if the parent method is public, the child class cannot be default or private or protected because all of these access specifiers are more restrictive.
  - static, final, and private access restrict the method from overriding. This means that any parent class method which has a private access specifier or is static or final in nature cannot be overridden by any of its child classes.
  - We also cannot override the main method in Java.

Method Overriding (Cont…)

- **Rules for method overriding (Cont…)**
  - If the class is implementing another class, which is abstract or implementing an interface, then the class has to override all of the functions, unless the class is an abstract class in itself.
  - We should always override abstract methods of the superclass.
  - The overriding method may have unchecked exceptions because the implementation is different from the method in the parent class. However, there should be no additional checked exceptions other than the exceptions in the parent class method.
  - The class should follow an "IS-A" relationship, i.e., the classes should be implementing inheritance.
  - Constructors cannot be overridden.

Examples of Method Overriding

```
class Animal
{
   public void disp()
   {
      System.out.println("It is an animal");
   }
}

class Dog extends Animal
{
   public void disp()
   {
      System.out.println("It is actually a dog");
   }
}

class Over1
{
   public static void main(String args[])
   {
      Dog obj = new Dog();
      obj.disp();
   }
}
```

Examples of Method Overriding (Cont…)

- Output

  It is actually a dog

Examples of Method Overriding (Cont…)

```
class GrandFather
{
   void color()
   {
      System.out.println("Medium fair");
   }
}

class Father extends GrandFather
{
   void color()
   {
      System.out.println("Medium fair");
   }
}

class Child extends Father
{
   void color()
   {
      System.out.println("fair");
   }

   public static void main(String args[])
   {
      Child obj = new Child();
      obj.color();
   }
}
```

Examples of Method Overriding (Cont…)

- Output

  fair

Examples of Method Overriding (Cont…)

```
class RBI
{
   public double roi;
   double interest(double SI, double pr, double t)
   {
      roi = SI/(pr*t);
      return roi;
   }
}

class SBI extends RBI
{
   double interest(double SI, double pr, double t)
   {
      roi = SI/((pr+1)*t);
      return roi;
   }
}

class ICICI extends RBI
{
   double interest(double SI, double pr, double t)
   {
      roi = SI/((pr+2)*t);
      return roi;
   }
}

class Over2
{
   public static void main(String args[])
   {
      double a, b, c, d;
      SBI s = new SBI();
      ICICI i = new ICICI();
      System.out.println("Rate of Interest of SBI: " + s.interest(500, 40000, 2));
      System.out.println("Rate of Interest of ICICI: " + i.interest(600, 40000, 2));
   }
}
```

Examples of Method Overriding (Cont…)

- Output

```
Rate of Interest of SBI: 0.0062498437539061525
Rate of Interest of ICICI: 0.007499625018749062
```

Examples of Method Overriding (Cont…)

```
class Animal
{
   protected void disp()
   {
      System.out.println("It is an animal");
   }
}

class Dog extends Animal
{
   public void disp()
   {
      System.out.println("It is actually a dog");
   }

   public static void main(String args[])
   {
      Dog obj = new Dog();
      obj.disp();
   }
}
```

Examples of Method Overriding (Cont…)

- Output

  It is actually a dog

Examples of Method Overriding (Cont…)

```
class Father
{
   private void color()
   {
      System.out.println("Medium fair from Father Class");
   }

   protected void work()
   {
      System.out.println("Employee from Father class");
   }
}

class Child extends Father
{
   public void color()
   {
      System.out.println("Fair from Child class");
   }

   public void work()
   {
      System.out.println("Asst. Prof. from Child class");
   }
}

class Over3
{
   public static void main(String args[])
   {
      Father obj1 = new Father();
      obj1.work();
      Father obj2 = new Child();
      obj2.work();
   }
}
```

Examples of Method Overriding (Cont…)

- Output

  Employee from Father class

  Asst. Prof. from Child class

Examples of Method Overriding (Cont…)

```
class Father
{
   private void color()
   {
      System.out.println("Medium fair from Father Class");
   }

   protected void work()
   {
      System.out.println("Employee from Father class");
   }
}

class Child extends Father
{
   public void color()
   {
      System.out.println("Fair from Child class");
   }

   public void work()
   {
      System.out.println("Asst. Prof. from Child class");
   }
}

class Over4
{
   public static void main(String args[])
   {
      Father obj1 = new Father();
      obj1.color();
      Father obj2 = new Child();
      obj2.color();
   }
}
```

Examples of Method Overriding (Cont…)

- Output

```
Over4.java:33: error: color() has private access in Father
      obj1.color();
           ^
Over4.java:35: error: color() has private access in Father
      obj2.color();
           ^
```

Examples of Method Overriding (Cont…)

```
class Father
{
   static void color()
   {
      System.out.println("Medium fair from Father Class");
   }

   void work()
   {
      System.out.println("Employee from Father class");
   }
}

class Child extends Father
{
   static void color()
   {
      System.out.println("Fair from Child class");
   }

   public void work()
   {
      System.out.println("Asst. Prof. from Child class");
   }
}

class Over5
{
   public static void main(String args[])
   {
      Father obj1 = new Father();
      obj1.color();
      obj1.work();
      Father obj2 = new Child();
      obj2.color();
      obj2.work();
   }
}
```

Examples of Method Overriding (Cont…)

- Output

```
Medium fair from Father Class
Employee from Father class
Medium fair from Father Class
Asst. Prof. from Child class
```

Examples of Method Overriding (Cont…)

```
class Father
{
   public void color()
   {
      System.out.println("Medium fair from Father Class");
   }
}

class Child extends Father
{
   public void color()
   {
      System.out.println("Fair from Child class");
   }

   public void work()
   {
      System.out.println("Asst. Prof. from Child class");
   }
}

class Over6
{
   public static void main(String args[])
   {
      Father obj1 = new Father();
      obj1.color();
      Father obj2 = new Child();
      obj2.color();
      obj2.work();
   }
}
```

Examples of Method Overriding (Cont…)

- Output

```
Over6.java:31: error: cannot find symbol
      obj2.work();
           ^
  symbol:   method work()
  location: variable obj2 of type Father
1 error
```

Difference Between Method Overloading and Method Overriding

Method Overloading	Method Overriding
It is used to increase the readability of the program.	It is used to provide the specific implementation of the method that is already provided by its super class.
Method overloading is performed within class.	Method overriding occurs in two classes that have IS-A relationship.
Here, parameter must be different.	Here, parameter must be same.
Method overloading is the example of compile time polymorphism.	Method overriding is the example of run time polymorphism.
Method overloading cannot be performed by changing return type of the method only.	Return type must be same or covariant in method overriding.

thank you . . .

Slides are prepared from various sources, such as Book, Internet Links and many more.