Method Overriding

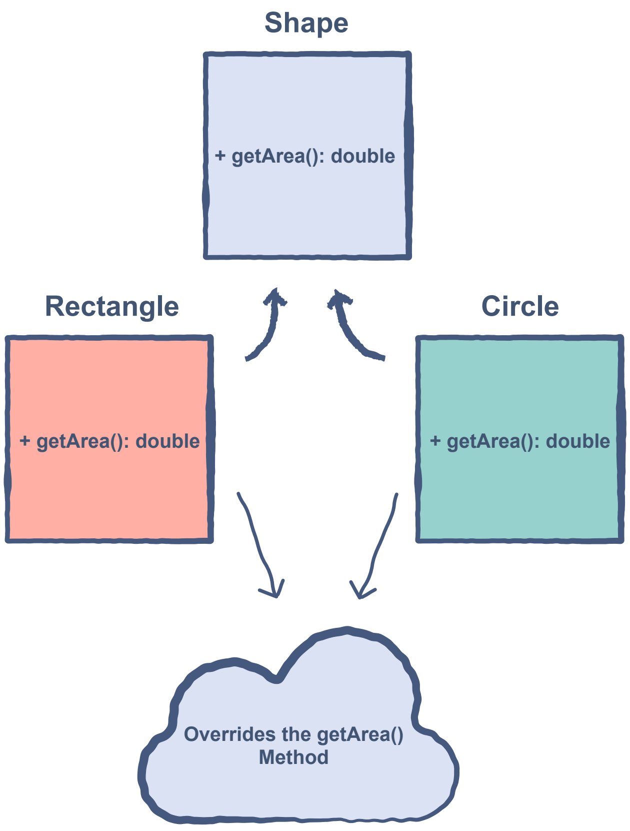
In this lesson, you'll be learning about what method overriding is and how to achieve it in Java.

WE'LL COVER THE FOLLOWING

A Brief Introduction

> *Method overriding* is the process of redefining a parent class's method in a subclass.

In other words, if a subclass provides the specific implementation of a method that has been declared by one of its parent classes, it is known as **method overriding.**



In the previous example, the Rectangle and Circle classes were overriding the `getArea()` method from the Shape class.

Overriding is done so that a child class can give its own implementation to a method which is already provided by the parent class.

In this case:

- The method in the parent class is called **overridden method.**
- The methods in the child classes are called **overriding methods.**

We have already seen the implementation of the `getArea()` method in the previous lesson, which depicts the concept of overriding. The *highlighted* portions show where method overriding is happening.

Let's have a look!

```
// A sample class Shape which provides a method to get the Shape's area
class Shape {

  public double getArea() {
    return 0;
  }

}

// A Rectangle is a Shape with a specific width and height
class Rectangle extends Shape {   // extended form the Shape class

  private double width;
  private double height;

  public Rectangle(double width, double height) {
    this.width = width;
    this.height = height;
  }

  public double getArea() {
    return width * height;
  }
}

// A Circle is a Shape with a specific radius
class Circle extends Shape {

  private double radius;

  public Circle(double radius) {
    this.radius = radius;
  }
  public double getArea() {
    return 3.14 * radius * radius;
  }

}



class driver {

  public static void main(String args[]) {
    Shape[] shape = new Shape[2]; // Creating shape array of size 2

    shape[0] = new Circle(2); // creating circle object at index 0
    shape[1] = new Rectangle(2, 2); // creating rectangle object at index 1

    // Shape object is calling children classes method
    System.out.println("Area of the Circle: " + shape[0].getArea());
    System.out.println("Area of the Rectangle: " + shape[1].getArea());
  }

}
```

Advantages of the Method Overriding

Method overriding is very useful in OOP. Some of its advantages are stated below:

- The derived classes can give their own specific implementations to inherited methods without modifying the parent class methods.
- For any method, a child class can use the implementation in the parent class or make its own implementation.

Key Features of the Method Overriding

Here are some key features of the *Method Overriding:*

- Method Overriding needs inheritance and there should be at least one derived class.
- Derived class/es must have the same declaration, i.e., name, same parameters and same return type of the method as of the base class.
- The method in the derived class/es must have different implementation from each other.
- The method in the base class must need to be overridden in the derived

class.

- Base class/method must not be declared as the `Final` class.

Now that we are familiar with the concept of method overriding let's understand the difference between method overloading and method overriding in the next lesson.