Java重写(Override)&重载(Overload)

重写(Override)概述

Java面向对象编程中的重写（override）指的是子类可以重写其父类中的非private方法，使得子类在调用该方法时会使用自己的实现而不是父类的实现。

重写(Override)讲解

定义一个名为 Animal 的父类和一个名为 Cat 的子类，其中 Cat 类继承了 Animal 类。Animal 类中有一个名为 move() 的方法，Cat 类可以对这个方法进行重写。

Animal 类中的 move() 方法不是private 类型，因此它可以被其子类重写。在 Cat 类中，使用相同的名称和参数列表来重新定义了 move() 方法，并且使用 @Override 注解向编译器说明这是一个重写方法。

Java | 复制代码

```java
class Animal {
    public void move() {
        System.out.println("动物可以移动");
    }
}

class Cat extends Animal {
    public void move() {
        System.out.println("猫可以走和跑");
    }
}
```

```java
public class Test {
    public static void main(String[] args) {
        Animal a = new Animal(); // Animal 对象
        Animal b = new Cat(); // Cat 对象

        a.move(); // 执行 Animal 类的方法
        b.move(); // 执行 Cat 类的方法
    }
}
```

重载(Overload)概述

Java 面向对象中的重载（Overload）指的是在同一个类中声明多个方法，它们拥有相同的名称，但是参数类型或数量不同。这样做的目的是为了提高代码的复用性和可读性。

重载(Override)讲解

这段代码展示了如何在同一个类中使用方法重载的方式来实现不同类型的处理，在 OverloadDemo 类中声明了三个方法，它们的名称相同，但是参数类型不同。

Java | 复制代码

```java
public class OverloadDemo {
    public void print(int n) {
        System.out.println("Print integer: " + n);
    }

    public void print(double d) {
        System.out.println("Print double: " + d);
    }

    public void print(String s) {
        System.out.println("Print string: " + s);
    }

    public static void main(String[] args) {
        OverloadDemo demo = new OverloadDemo();
        demo.print(123);
        demo.print(3.14);
        demo.print("Hello, world!");
    }
}
```