# Java 9 钻石操作符(Diamond Operator)

钻石操作符是在 java 7 中引入的，可以让代码更易读，但它不能用于匿名的内部类。

在 java 9 中， 它可以与匿名的内部类一起使用，从而提高代码的可读性。

考虑以下 Java 9 之前的代码：

## 实例

public class Tester { public static void main(String[] args) { Handler<Integer> intHandler = new Handler<Integer>(1) { @Override public void handle() { System.out.println(content); } }; intHandler.handle(); Handler<? extends Number> intHandler1 = new Handler<Number>(2) { @Override public void handle() { System.out.println(content); } }; intHandler1.handle(); Handler<?> handler = new Handler<Object>("test") { @Override public void handle() { System.out.println(content); } }; handler.handle(); } } abstract class Handler<T> { public T content; public Handler(T content) { this.content = content; } abstract void handle(); }

执行输出结果为：

12Test

在 Java 9 中，我们可以在匿名类中使用 <> 操作符，如下所示：

## 实例

public class Tester { public static void main(String[] args) { Handler<Integer> intHandler = new Handler<>(1) { @Override public void handle() { System.out.println(content); } }; intHandler.handle(); Handler<? extends Number> intHandler1 = new Handler<>(2) { @Override public void handle() { System.out.println(content); } }; intHandler1.handle(); Handler<?> handler = new Handler<>("test") { @Override public void handle() { System.out.println(content); } }; handler.handle(); } } abstract class Handler<T> { public T content; public Handler(T content) { this.content = content; } abstract void handle(); }

执行输出结果为：

12Test