# 1 内部类

## 1.1 什么是内部类

在一个类的内部，再定义一个类，那么这个类就是内部类。

特点：内部类中可以直接访问其外部类的成员，包括私有的成员。

内部类分为成员内部类和局部内部类：在类的成员位置定义的类，被称为成员内部类；在类的方法中(局部位置)定义的类，被称为局部内部类。例如：

|  |
| --- |
| **public class** Outer {  **private int outerNum** = 100;  **class** Inner1 {  *// 这是成员内部类*  **public void** show() {  System.***out***.println(**outerNum**); *// 可访问外部类私有变量*  }  }  **public void** method() {  *// 这是局部内部类*  **class** Inner2 {  **public void** show() {  System.***out***.println(**outerNum**); *// 也可访问外部类私有变量*  }  }  }  } |

内部类生产的class文件为 “外部类$内部类.class”，为了标明该内部类是属于具体哪个外部类的。

成员内部类和局部内部类都能访问外部类的私有变量，下面主要介绍一些细节和内部类是如何被外界访问的。

## 1.2 成员内部类

成员内部类可用访问修饰符修饰，比如public private和默认。实际开发中，为了保护内部类，将内部类用private修饰，不让外界直接访问，这样只能在该其外部类使用这个内部类。

成员内部类还能用static修饰，

（1）当不加static修饰时，成员内部类中不能有静态成员，并且外部类和外界访问该内部类的方法如下：

外部类名.内部类名 对象名 = 创建外部类对象.创建内部类对象。例如：

|  |
| --- |
| **public class** Outer {  **private int num** = 100;  **public class** Inner {  *// 成员内部类*  **public void** showNum() {  System.***out***.println(**num**);  }  }  **public void** show() {  *// 在外部类中可以直接写Inner*  Inner i = **new** Outer().**new** Inner();  i.showNum();  }  } |

Demo使用类中：

|  |
| --- |
| **public class** Demo {  **public static void** main(String[] args) {  Outer outer = **new** Outer();  outer.show();  *// 外界必须写成Outer.Inner*  Outer.Inner i = **new** Outer().**new** Inner();  i.showNum();  }  } |

（2）当用static修饰时，成员内部类中只能访问外部类的静态成员；内部类中静态和实例成员都能有（实例方法也只能访问外部类的静态成员），并且这时外界访问更加方便，格式如下：

使用内部类静态方法，直接用：外部类名.内部类名.方法名()；

使用内部类实例方法：

1.创建内部类对象格式：

外部类名.内部类名 对象名 = new 外部类名.内部类名()；

2.使用实例方法：对象名.实例方法名()。

例子：

外部类Outer：

|  |
| --- |
| **public class** Outer {  **private int num1** = 100;  **private static int** *num2* = 200;  **public static class** Inner {  *// 内部类静态方法*  **public static void** showStaticNum() {  System.***out***.println(*num2*); *// 只能访问外部静态变量*  }  *// 内部类实例方法*  **public void** showNum() {  System.***out***.println(*num2*); *// 实例方法也只能访问外部变量*  }  }  **public void** showInner() {  *// 外部类中Outer.都能省略*  Inner.*showStaticNum*(); *// 直接调用内部类静态方法*  *// 创建对象调用内部类实例方法*  Inner inner = **new** Inner();  inner.showNum();  }  } |

Demo使用类：

|  |
| --- |
| **public class** Demo {  **public static void** main(String[] args) {  *// 测试外部类访问*  Outer outer = **new** Outer();  outer.showInner();  *// 外界访问不能省略Outer.*  Outer.Inner.*showStaticNum*();  Outer.Inner inner = **new** Outer.Inner();  inner.showNum();  }  } |

案例补充：

|  |
| --- |
| **public class** Outer {  **public int num** = 10;  **public class** Inner {  **public int** num = 20;  **public void** show() {  **int** num = 30;  System.***out***.println(num); *// 直接访问的是本局部变量的num，值是30*  System.***out***.println(**this**.num); *// 用this.num访问的是内部类的num，值是20*  System.***out***.println(Outer.**this**.**num**); *// 用Outer.this.num表示的是外部类的num，而不能用Outer.num*  }  }  } |

## 1.3 局部内部类

局部内部类不能写任何修饰符，包括public和private。同样，局部内部类也不能用static关键字修饰，并且局部内部类中也不能有静态成员。

在类的静态方法和实例方法中都可以写局部内部类，局部内部类只能在包含该局部内部类的方法中使用，在该方法中创建局部内部类对象并使用，外部的别的地方无法直接使用。

例子：

（1）Outer类

|  |
| --- |
| **public class** Outer {  **private int num1** = 100;  **private static int** *num2* = 200;  **public static void** staticMethod() {  *// 静态方法中的内部类*  **class** Inner1 {  *// 内部类的方法*  **public void** show() {  System.***out***.println(*num2*); *// 静态方法中的内部类只能外部类的访问静态变量*  }  }  *// 在该方法中使用内部类*  Inner1 inner1 = **new** Inner1();  inner1.show();  }  **public void** instanceMethod() {  *// 实例方法中的内部类*  **class** Inner2 {  **public void** show() {  System.***out***.println(**num1** + *num2*); *// 都可访问，这是正常的*  }  }  *// 同样在该方法中使用内部类*  Inner2 inner2 = **new** Inner2();  inner2.show();  }  } |

（2）Demo测试类

|  |
| --- |
| **public class** Demo {  **public static void** main(String[] args) {  *// 测试*  Outer outer = **new** Outer();  outer.*staticMethod*(); *// 200*  outer.instanceMethod(); *// 300*  }  } |

特别需要注意的是，如果局部内部类中想要访问该方法的局部变量，那么必须将这个变量用final关键字修饰。原因：局部变量在外部类的这个方法中，该方法一旦执行完毕，变量就会被销毁释放，那么内部类就无法访问了，因此会出错。所以需要把内部类访问的局部变量用final关键字修饰，将这个变量提升为“类”的字段。如果不加，JDK8会自动加上隐含的final。建议还是自己加上。（这里所说的局部变量是外部类的包含这个局部内部类方法中的变量，而不是内部类自己的方法中的局部变量）

例子：

（1）Outer类

|  |
| --- |
| **public class** Outer {  **public void** method() {  **final int** num = 100; *// 用final修饰将要被局部内部类访问的变量*  *// 定义局部内部类*  **class** Inner {  **public void** show() {  System.***out***.println(num);  }  }  *// 使用局部内部类*  Inner inner = **new** Inner();  inner.show();  }  } |

（2）Demo测试类

|  |
| --- |
| **public class** Demo {  **public static void** main(String[] args) {  *// 测试*  Outer outer = **new** Outer();  outer.method();  }  } |