关于内部类，需要掌握的三点：

* 如何创建成员内部类的对象；

// 创建静态内部类的对象，直接通过外部类调用静态内部类的构造器。

Person.Dog dog = new Person.Dog();  
  
// 创建非静态的内部类的对象，必须先创建外部类的对象，通过外部类的对象调用内部类的构造器。  
Person person = new Person();  
Person.Bird bird = person.new Bird();

* 如何区分调用外部类、内部类的变量；

public void setName(String name) {

// 局部变量  
 System.***out***.println(name);  
 // 内部类对象的属性  
 System.***out***.println(this.name);  
 // 外层类对象的属性  
 System.***out***.println(Person.this.name);  
}

* 局部内部类的使用。（使用较少）
  + 常常使用一个方法，使其返回值为某个类或接口对象。而这个类或接口在方法内部创建。

public Comparable getComparable() {

/\*\*  
 \* 1、创建一个实现Comparable接口类：局部内部类  
 \*/  
 class MyComparable implements Comparable {  
  
 @Override  
 public int compareTo(Object o) {  
 return 0;  
 }  
 }  
 // 2、返回一个实现类的对象  
 return new MyComparable();  
}

* **内部类**
* 在Java中，允许一个类的定义位于另一个类的内部，称为内部类
* Inner class一般用在定义它的类或语句块之内，在外部引用它时必须给出完整的名称。
* **Inner class的名字不能与包含它的类名相同**。
* **内部类可以使用外部类的私有数据**，因为它是外部类的成员，同一个类的成员之间可以相互访问。而外部类要访问内部类中的成员需要：内部类.成员或者内部类对象.成员。
* **分类：**
* 成员内部类：声明在类内部且方法外（static成员内部类和非static成员内部类）
* 局部内部类：声明在类的方法里（不谈修饰符）、匿名内部类。
* **特性**
* **Inner class作为类的成员：**
  + 可以声明为**final**的；
  + 和外部类不同，Inner class可声明为**private**或**protected**；
  + Inner class可以声明为**static**的，但此时就不能再使用外层类的非static的成员变量。
* **Inner class作为类：**
  + 可以声明为**abstract**类，因此可以被其它的内部类继承。
* 注意：非static的内部类中的成员不能声明为static的，只有在外部类或static的内部类中才可声明static成员。
* 匿名内部类
* 匿名内部类不能定义任何静态成员、方法和类，只能创建匿名内部类的一个实例。一个匿名内部类一定是在new的后面，用其隐含实现一个接口或实现一个类。
* new 父类构造器（实参列表）|实现接口(){

//匿名内部类的类体部分

}

public Comparable getComparable1() {

/\*\*  
 \* 返回一个实现Comparable接口的匿名内部类的对象  
 \*/  
 return new Comparable(){  
 @Override  
 public int compareTo(Object o) {  
 return 0;  
 }  
 };  
}