#### 对接口中的函数默认实现

interface Study {

fun readBooks()

fun doHomework() {

println("do homework default implementation.")

}

}复制代码

如果像之前那么写，在实现接口时里面的两个方法都必须实现，如果改成这样的话，只需要强制实现 readBooks() 函数了，doHomework()可以选择写或者不写，不写的话则会打印do homework default implementation.。

#### 4.3.3 访问修饰符

在 Java 中一共由 public、private、protected、default（什么都不写）这四种修饰符，在 Kotlin 中有 public、private、protected、internal 这四种修饰符，想要使用那种修饰符时直接将修饰符写在 fun 前面即可。

首先 private 修饰符在两种语言中的作用一模一样，都表示只对当前类内部可见，public 修饰符的作用也是一致的，标识对所有类可见，但是**在 Kotlin 中 public 修饰符是默认项**，而在 Java 中是 default，前面书写的函数都没有加访问修饰符，那么这些函数的访问权限全部是 public。protected 在 Java 中表示对当前类，子类和同一个包路径下的类可见，**在 Kotlin 中则表示只对当前类和子类可见**。Kotlin 抛弃了 Java 中的 default 可见性（同一包路径下的类可见）。引入了一种新的可见性概念，**只对同一模块中的类可见**，使用的是 internal 修饰符。

比如我们开发了一个模块给别人使用，但是有一些函数只允许在模块内部调用，不想暴露给外部，就可以将函数声明为 internal修饰的。

Java 和 Kotlin 可见性修饰符对照表



#### 4.4.2 Kotlin 中的数据类

data class Cellphone(val brand: String, val price: Double)复制代码

在 Kotlin 中只需要这一行代码即可，其中神奇的地方在于 class 前面的 data 关键字，有了这个关键字就表明我们想要声明一个数据类，Kotlin 会根据主构造函数中的参数帮你将 equals()、hashCode()、toString()方法自动生成，从而减少了开发的工作量。

编写 main 函数进行测试

fun main() {

val cellphone1 = Cellphone("Samsung", 1299.99)

val cellphone2 = Cellphone("Samsung", 1299.99)

println(cellphone1)

println("cellphone1 equals cellphone2 " + (cellphone1 == cellphone2))

}

object Singleton {

fun singletonTest() {

println("singletonTest is called.")

}

}

#### Kotlin 中的单例类

在 Kotlin 中实现单例要比 Java 中简单的多，只需要使用 object 关键字即可，在这其中 Kotlin 帮我们创建了一个 Singleton 类的实例，并且保证全局只存在一个 Singleton 实例。

object Singleton { fun **singletonTest**() { println("singletonTest is called.") } }