深入理解Java中的final、finally和finalize



六尺帐篷 (/u/f8e9b1c246f1)

2017.09.12 14:22 字数 1356 阅读 94 评论 0 喜欢 2

(/u/f8e9b1c246f1)

编辑文章 (/writer#/notebooks/10012174/notes/16986444)

final

final可以用于以下四个地方

- 定义变量,包括静态的和非静态的。
- 定义方法的参数。
- 定义方法。
- 定义类。

定义变量

如果final修饰的是一个基本类型,就表示这个变量被赋予的值是不可变的,即它是个常量;如果final修饰的是一个对象,就表示这个变量被赋予的引用是不可变的,这里需要提醒大家注意的是,不可改变的只是这个变量所保存的引用,并不是这个引用所指向的对象。在第二种情况下,final的含义与第一种情况相同。实际上对于前两种情况,有一种更贴切的表述final的含义的描述,那就是,如果一个变量或方法参数被final修饰,就表示它只能被赋值一次,但是JAVA虚拟机为变量设定的默认值不记作一次赋值。

被final修饰的变量必须被初始化。初始化的方式有以下几种:

- 在定义的时候初始化。
- final变量可以在初始化块中初始化,不可以在静态初始化块中初始化。
- 静态final变量可以在静态初始化块中初始化,不可以在初始化块中初始化。
- final变量还可以在类的构造器中初始化,但是静态final变量不可以。

上面初始化的方式有的非常绕口,很难记忆,但实际上我们只要搞清楚,静态变量是依赖类的,而非静态变量变量是依赖对象的,初始化块和构造方法是初始化对象的,而静态初始化块是初始化类的,先有类的加载,才有对象的生成。 所以上述的初始化方式变得很好理解。

用final修饰的变量(常量)比非final的变量(普通变量)拥有更高的效率,因此我们在实际编程中应该尽可能多的用常量来代替普通变量,这也是一个很好的编程习惯。

定义方法

当final用来定义一个方法时,会有什么效果呢?正如大家所知,它表示这个方法不可以被子类重写,但是它这不影响它被子类继承。

定义类

final用于类的情况。这个大家应该也很熟悉了,因为我们最常用的String类就是final的。由于final类不允许被继承,编译器在处理时把它的所有方法都当作final的,因此final类比普通类拥有更高的效率。而由关键字abstract定义的抽象类含有必须由继承自它的子类重





&

载实现的抽象方法,因此无法同时用final和abstract来修饰同一个类。同样的道理,final 也不能用来修饰接口。final的类的所有方法都不能被重写,但这并不表示final的类的属性 (变量) 值也是不可改变的, 要想做到final类的属性值不可改变, 必须给它增加final修饰

finally

finally用在异常处理中定义总是执行代码,无论try块中的代码是否引发异常,catch是否 匹配成功,finally块中的代码总是被执行,除非JVM被关(System.exit(1)),通常用作 释放外部资源(不会被垃圾回收器回收的资源)。

那么,有没有一种情况使finally语句块得不到执行呢? 大家可能想到了return、 continue、break这三个可以打乱代码顺序执行语句的规律。

很明显, return、continue和break都没能阻止finally语句块的执行。从输出的结果来看, return语句似乎在finally语句块之前执行了,事实真的如此吗?我们来想想看, return语 句的作用是什么呢?是退出当前的方法,并将值或对象返回。如果finally语句块是在 return语句之后执行的,那么return语句被执行后就已经退出当前方法了,finally语句块 又如何能被执行呢?因此,正确的执行顺序应该是这样的:编译器在编译return new ReturnClass();时,将它分成了两个步骤, new ReturnClass()和return, 前一个创建对象 的语句是在finally语句块之前被执行的,而后一个return语句是在finally语句块之后执行 的,也就是说finally语句块是在程序退出方法之前被执行的。同样, finally语句块是在循 环被跳过 (continue) 和中断 (break) 之前被执行的。

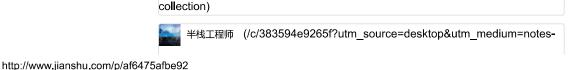
finalize

included-collection)

included-collection)

finalize()方法是Object类中定义的方法,当垃圾回收器将无用对象从内存中清除时,该对





程序员 (/c/NEt52a?utm_source=desktop&utm_medium=notes-included-

Android... (/c/5139d555c94d?utm_source=desktop&utm_medium=notes-

🤵 Android开发 (/c/d1591c322c89?utm_source=desktop&utm_medium=notes-

企

ಹ

included-collection)

Android... (/c/58b4c20abf2f?utm_source=desktop&utm_medium=notes-included-collection)

Moderal Android知识 (/c/3fde3b545a35?utm_source=desktop&utm_medium=notesincluded-collection)

java (/c/4f285596f858?utm_source=desktop&utm_medium=notes-included-collection)

企

&