Static和final的区别

Static强调只有一份，不管类被初始化多少次,static初始化的成员值初始化一次。Final表示常量，不可改变。

一、Static：

Static是静态修饰关键字，可以修饰变量和成员方法(包括程序块和类的方法)，也可以形成静态static代码块。

被static修饰的成员变量和成员方法独立于该类的任何对象，也就是说，它不依赖于类特定的实例，被类的所有实例共享，只要这个类被加载,java虚拟机就可以根据类名在运行数据区的方法区内找到他们。所以,static对象可以在它的任何对象创建之前访问，无需任何引用对象。

1.1 static变量

当定义一个static的变量的时候，jvm会将其分配在内存堆上，所有程序对它的引用都会指向这一个地址而不会重新分配内存。

按照是否静态，类的成员变量可以分为两种：一种是被static修饰的变量，叫做静态变量或类变量，另一种是没有被static修饰的变量，叫做实例变量。两者的区别是：

--对于静态变量，它在内存中只有一个拷贝（节省内存）。Jvm只为静态变量分配一次内存，在加载类的过程中完成对静态变量的内存分配，可以直接通过类名访问，也可以通过对象来访问(但不推荐这样)。

--对于实例变量，每创建一个实例，就会为实例变量分配一个内存，实例变量可以再内存中有多个拷贝，互不影响。

1.2 静态代码块

当修饰一个程序块的时候(也就是直接将代码写在static中时)，虚拟机会优先加载静态块中的代码，用于系统初始化。

Static代码块（也叫做静态代码块），实在类中独立于类成员的static语句块，可以有多个，jvm会按照他们在类中出现的先后顺序依次执行他们，每个代码块都只会被执行一次。如以下代码：

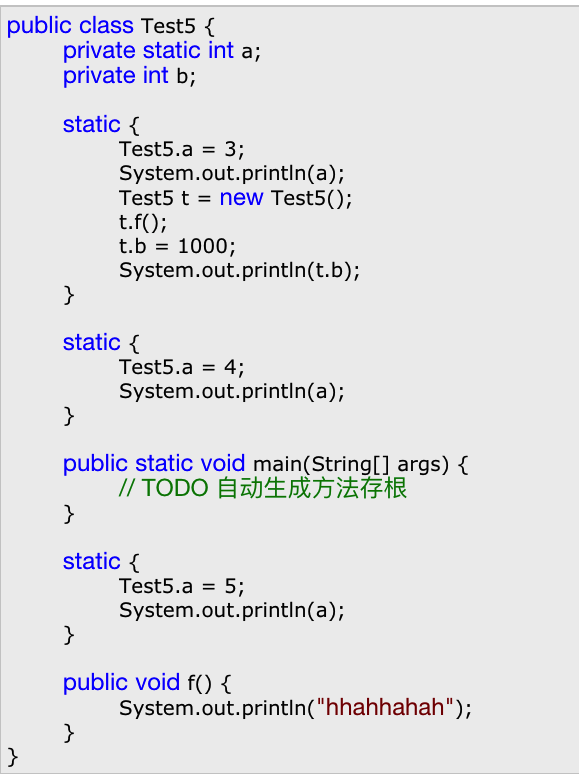


图2：static代码块

它的执行结果是：



图3：static代码块执行结果

我们可以发现所有的类中，都可以有一个static的main方法，这样的话jvm在运行main方法的时候可以直接调用，而不用创建实例。

1.3 静态方法

当修饰一个类方法时，叫做静态方法，静态方法可以直接通过类名调用，任何的实例也都可以调用（同静态变量相似）。Static和“this ,super”势不两立，static与具体对象无关,而this和super正好跟具体对象有关。因此静态方法中不能用this和super关键字，不能直接访问所属类的实例变量和实例方法（也就是那些不带static的成员变量和成员方法，注意我们平时在定义static main方法时注意的一些细节），他只能访问所属类的静态成员变量和成员方法。

因为static方法独立于任何实例，因此static方法必须被实现，而不能是抽象的abstract。

二、final

final可以被用来修饰变量、方法和类。

final修饰变量时，表示常量，只能被赋值一次，赋值后值不再改变，jvm会将它分配到常量池中，程序不可以改变它的值。

final修饰一个方法时,该方法在子类中将不能被重写；但是可以被继承。

final修饰一个类时，该类不能被继承，没有子类，final类中的方法默认是final的。

注意：父类的private成员方法是不能被子类方法覆盖的，因此private类型的方法默认是final的。

2.1 final变量（常量）

用final修饰的成员变量表示常量，值一旦给定后就无法改变了。

用final修饰的变量有三种：静态变量、实例变量和局部变量，分别表示三种类型的常量。一旦给final变量初值后，值就不能再改变了。

此外，final变量定义的时候，可以先声明，而不给初值，这种变量也成为final空白，这种final空白可以实现，一个类中的final数据成员可以依据对象的不同而有所不同，但是依然保持其恒定不变的特征。（也就是说对类进行实例化时，可以在初始化时对final空白赋值，这样就实现了一个类的不同实例对象有不同的final值。）

代码如下：

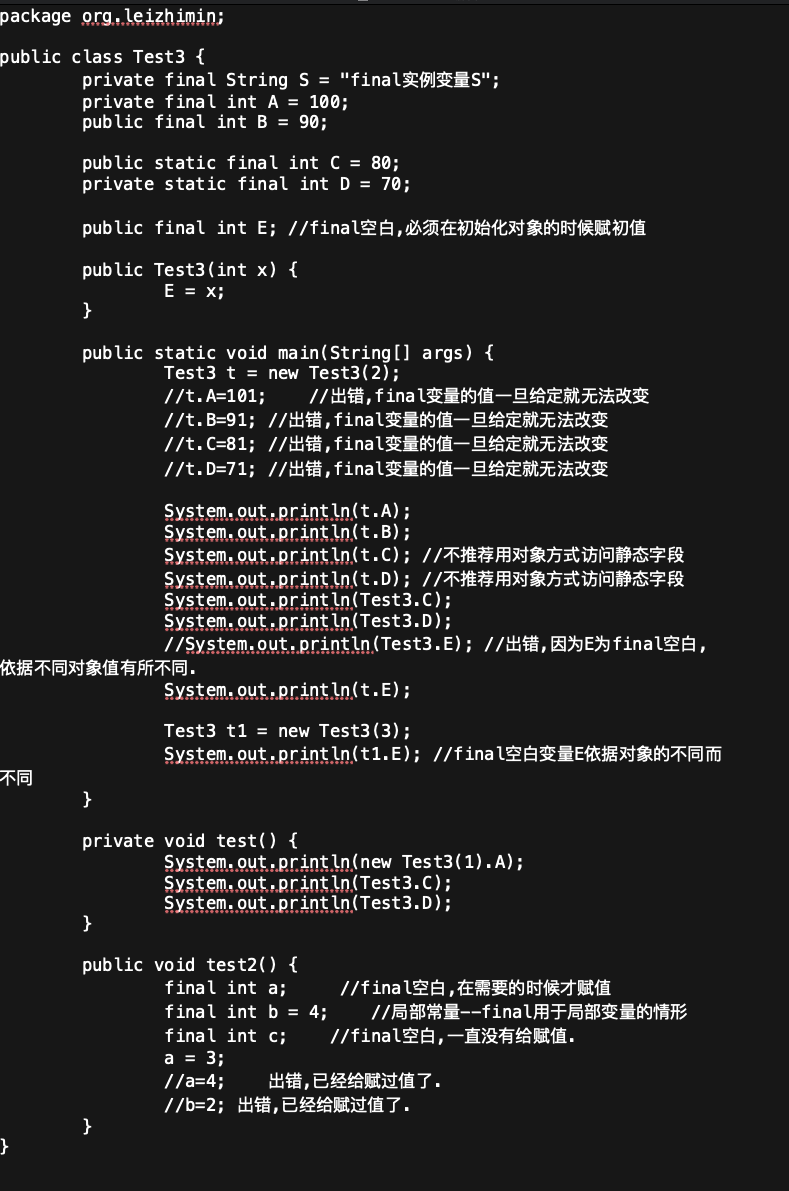


图4：final变量（常量）

2.2 final参数

当函数的参数为final类型时，我们可以读取使用这个参数，但是无法改变这个参数的值。

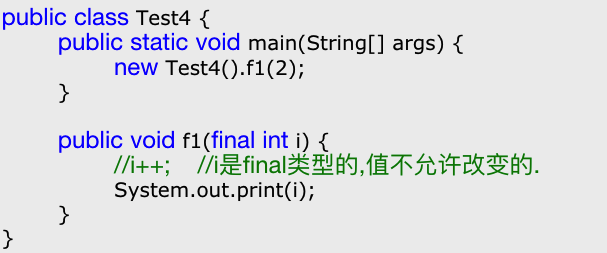


图5：final参数

2.3 final类

final类不能被继承，因此final类的成员方法没有机会被覆盖，默认都是final的。

2.4 final方法

如果一个类不允许其他的子类覆盖某个方法，则可以将这个方法声明为final方法。

使用final方法的原因有两个：

--1）把方法锁定，防止任何继承类修改它的意义和实现

--2）高效，编译器在遇到调用final方法时会转入内嵌机制，大大提高执行效率。

一个例子如下：

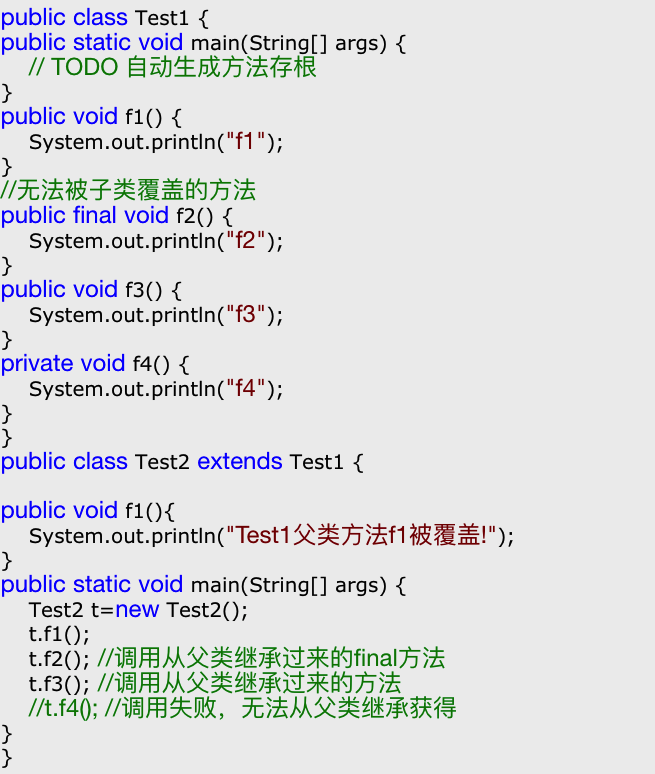


图6：final方法

三、 static和final一起用

static final用来修饰成员变量和成员方法，可以简单理解为全局变量。对于变量，表示一旦给定值就不可修改，并且可以通过类名访问；对于方法，表示不可覆盖，并且可以通过类名直接访问。

但是对于被static和final修饰过的实例常量，实例本身不能再改变了，但对于一些容器类型（比如ArrayList，HashMap）的实例变量，不可以改变容器变量本身，但是可以改变容器中存放的对象。

下面是一个例子，对于statin final修饰的实例常量不能修改值，但是对于容器，它们内部的值却可以修改。



图7：实例变量和容器的区别

实验结果如下：



图8：实验结果