final

一、基本介绍

- final可以修饰 **类、属性、方法 和 局部变量**.

final使用时机:

1.当不希望被继承时，可以用final修饰.

```
class fianl A{//希望A类不被任何的类继承，可以用fianl

}
class B extends A{

}
```

.

2.当不希望父类的某个方法被子类 覆盖 或 重写时，可以用final.

```
class C{
    //若要求hi()不能被重写,可以用final
    public final void hi(){}
}
class D extends C{
    @Override
    public void hi() {
'hi()' cannot override 'hi()' in 'com.final_.C'; overridden method is fina
Make 'C.hi' not final Alt+Shift+Enter  More actions... Alt+Enter
}
```

.

3.当不希望 类的某个属性的值 被修改，可以用final.

```java
public class final01 {
    public static void main(String[] args) {
        E e = new E();
        e.TAX_RATE = 1;
    }
}
/*.
/*.
//当不希望某个属性的值被修改，可以final
class E{
    public final double TAX_RATE = 0.08;
}
```

Cannot assign a value to final variable 'TAX_RATE'

Make 'TAX_RATE' not final Alt+Shift+Enter More actions... Alt+

```
com.final_.E
public final double TAX_RATE = 0.08
```

.

4.当不希望 某个局部变量 被修改，可以用final.

```java
//当不希望某个局部变量被修改
class F{
    public void G(){
        //这时NUM称为局部常量
        final double NUM =0.03;
        NUM = 0;
    }
}
```

二、final细节：

Detail 01：

1. **final属性又叫常量,一般用 XX_XX_XX来命名(比如更刚刚写的TAX_RATE变量名).**

2. **final的属性定义时，必须赋值，以后该值的地址都不能再修改。该地址内的值可被修改（赋值可在如下区域）**

- ```
  public final double TAX_RATE = 0.08;//声明之后直接初始化
  ```
- **在构造器中.**
- **在代码块中.**

```
class AA {
    /*
    1. 定义时：如 public final double TAX_RATE=0.08;
    2. 在构造器中
    3. 在代码块中
     */
    public final double TAX_RATE = 0.08;//1.定义时赋值
    public final double TAX_RATE2 ;
    public final double TAX_RATE3 ;

    public AA() {//构造器中赋值
        TAX_RATE2 = 1.1;
    }
    {//在代码块赋值
        TAX_RATE3 = 8.8;
    }
}
```

.

3. **若final修饰的属性是 静态的，则初始化的位置<只能是.**

- **定义时.**
- **在静态代码块.**
- **不能在构造器中赋值.**
```

```java
class BB {
    /*
    如果final修饰的属性是静态的，则初始化的位置只能是
    1 定义时   2 在静态代码块  不能在构造器中赋值。
     */
    public static final double TAX_RATE = 99.9;
    public static final double TAX_RATE2 ;
    public static final double TAX_RATE3 ;

    public BB() {
        TAX_RATE3 = 8.8;
    }

    static {
        TAX_RATE2 = 3.3;
    }
}
```

.

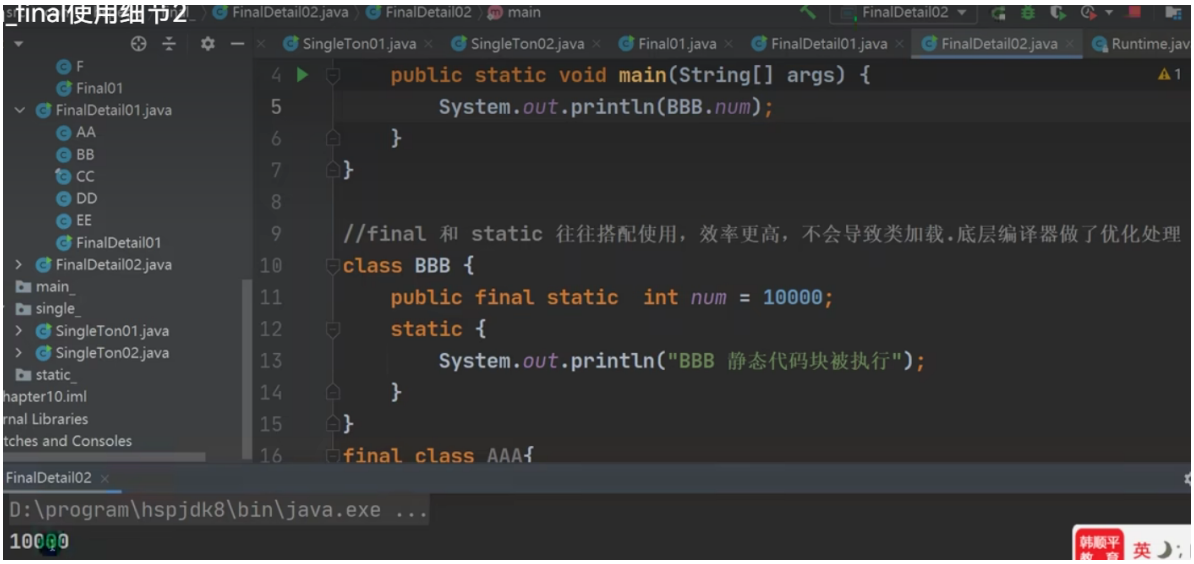
- **因为属性为static，在加载类的时候 会先 调用static属性,调用完static之后 再 调用构造器的，所以final的这个属性 没有被赋值。**

4. **final 类 不可以继承，但可以 实例化对象.**
5. **若 类 不是 final类，但含有final方法，则该方法不能重写，但可以被继承.**

Detail 02:

1. **一般来说,一个类已经是final的话，本类中的方法就不用再写final了,因为 类 没有机会被继承， 方法也没有机会被重写.**
2.
```java
final class AAA{
    //一般来说，如果一个类已经是final类了，就没有必要再将方法修饰成final方法
    public final void cry() {}
}
```
。
3. **final不能修饰构造方法.**

4. **final 和static往往搭配使用,效率更高，（不会导致类加载），底层编译器做了优化处理.**



。

5. **包装类(Integer,Double,Float,Boolean,都是final),String 也是final类,所以不能够被继承的.**

Excercise

Excerpts01：

```
package com.final_.Final_Excer;
/*编写能够计算圆形的面积，圆周率为3.14
* 赋值的位置3个三个方式都写一下
* 1.属性声明时直接赋值
* 2.构造器赋值
* 3.在代码块中赋值*/
public class Excer01 {
    public static void main(String[] args) {
        Circle circle = new Circle(5.0);
        System.out.println("面积：" + circle.calArea());
    }
}
class Circle{
    private double radius;
    private final double PI ;//= 3.14;//声明时直接赋值

    public Circle(double radius) {
        this.radius = radius;
       //构造器赋值 PI = 3.14;
    }
    {
        PI = 3.14;//代码块赋值
    }

    public double calArea(){
        return PI * radius * radius;
    }
}
```

Excer02：

```
public class Something {  //FinalExercise02.java
public int addOne(final int x) { //下面的代码是否有误，为什么?
 ++x;  //错误,原因是不能修改 final x的值
return x + 1; //这里是可以.
}
}
```

。