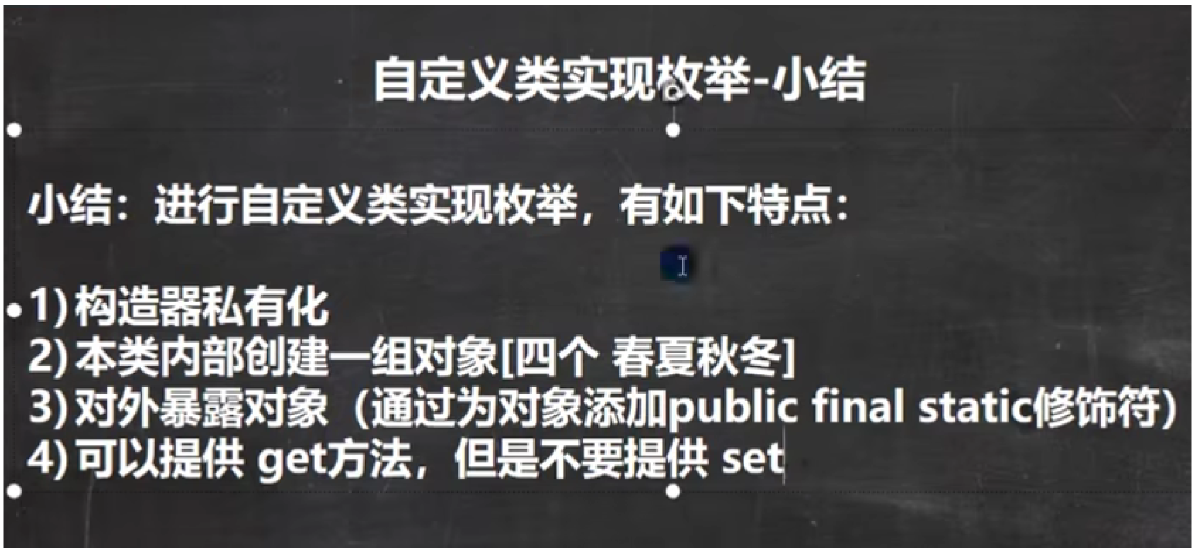
九、枚举法

一、引入枚举法 & 例子

```java
package com;
/*
 *@author 杨宗霖
 *@version 1.0
 */
/*引出枚举法:
* 例如:四个季节,只有春夏秋冬,不可能有其他情况
* 那么这种固定情况就可以用枚举法来限定.
* 特点:
* 1.季节的值是有限的几个值(spring,summer,autumn,winter)
* 2.只读,不需要修改
* 枚举属于一种特殊的类,里面只包含一组(有限的)(特定的)对象*/

public class Enumeration01 {
    public static void main(String[] args) {
        System.out.println(Season.AUTUMN);
        System.out.println(Season.SUMMER);
        System.out.println(Season.WINTER);
        System.out.println(Season.SPRING);
    }
}
class Season{
    private String name;
    private String desc;
    /*枚举法的写法:
    * 1.将构造器私有化,目的防止,直接 new
    * 2.去掉set相关方法,防止属性被修改
    * 3.在Season类内部,直接创建固定的对象
    * 4.优化,可以再添加final,目的是可以防止static多次加载*/

    public static final Season SPRING = new Season("春天","温暖");
    public static final Season WINTER = new Season("冬天","寒冷");
    public static final Season AUTUMN = new Season("秋天","凉爽");
    public static final Season SUMMER = new Season("夏天","炎热");

    private Season(String name, String desc) {
        this.name = name;
        this.desc = desc;
    }

    public String getName() {
        return name;
    }


    public String getDesc() {
        return desc;
    }
    @Override
    public String toString() {
        return "Season{" +
                "name='" + name + '\'' +
                ", desc='" + desc + '\'' +
                '}';
    }
}
```



自定义类实现枚举-小结

小结：进行自定义类实现枚举，有如下特点：

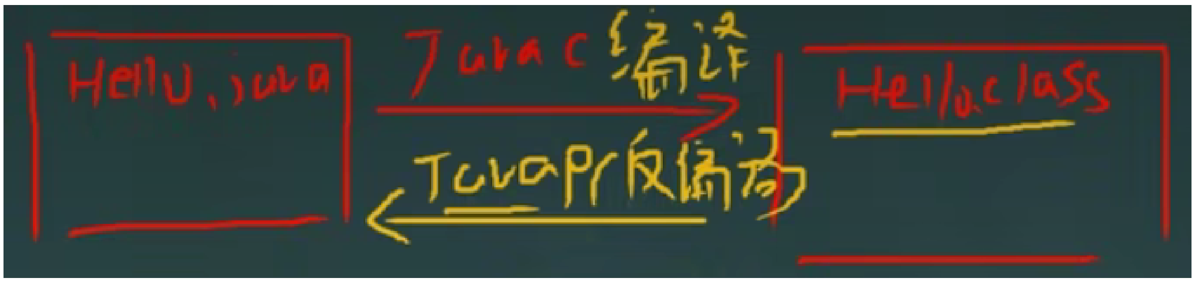
1) 构造器私有化
2) 本类内部创建一组对象[四个 春夏秋冬]
3) 对外暴露对象（通过为对象添加public final static修饰符）
4) 可以提供 get方法，但是不要提供 set

.

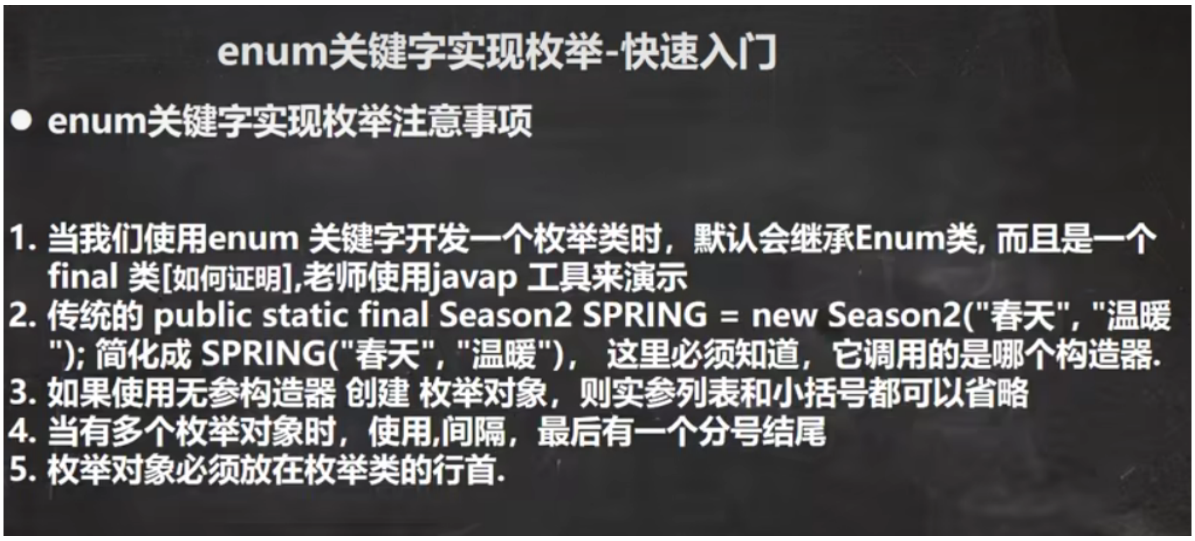
二、枚举关键字 & 使用

```
package com;

/*
 *@author 杨宗霖
 *@version 1.0
 */public class Enumeration02 {
    public static void main(String[] args) {
        System.out.println(Season2.SPRING);
        System.out.println(Season2.AUTUMN);
        System.out.println(Season2.SUMMER);
        System.out.println(Season2.WINTER);
    }
}
//实现enum关键字
/*步骤:
1.使用enum关键字 代替 class
2.public static final (修饰符部分)
    Season (类型部分)
    SPRING (对象名部分) = new Season("春天","温暖");
直接代替成(两者等价)
SPRING("春天","温暖")
3.若有多个对象,使用 逗号 隔开即可
4.枚举对象必须写在枚举类的行首
5.若用无参构造器创建常量对象,则可以省略()
*/
```

```
enum Season2{//1.使用enum关键字 代替 class
    //要求将常量对象写在最前面
    SPRING("春天","温暖"),
    WINTER("冬天","寒冷"),
    AUTUMN("秋天","凉爽"),
    SUMMER("夏天","炎热");
    private String name;
    private String desc;

 /*   public static final Season SPRING = new Season("春天","温暖");
    public static final Season WINTER = new Season("冬天","寒冷");
    public static final Season AUTUMN = new Season("秋天","凉爽");
    public static final Season SUMMER = new Season("夏天","炎热");*/



    private Season2(String name, String desc) {
        this.name = name;
        this.desc = desc;
    }

    public String getName() {
        return name;
    }



    public String getDesc() {
        return desc;
    }

    @Override
    public String toString() {
        return "Season{" +
                "name='" + name + '\'' +
                ", desc='" + desc + '\'' +
                '}';
    }
}
```



三、枚举写法底层实现

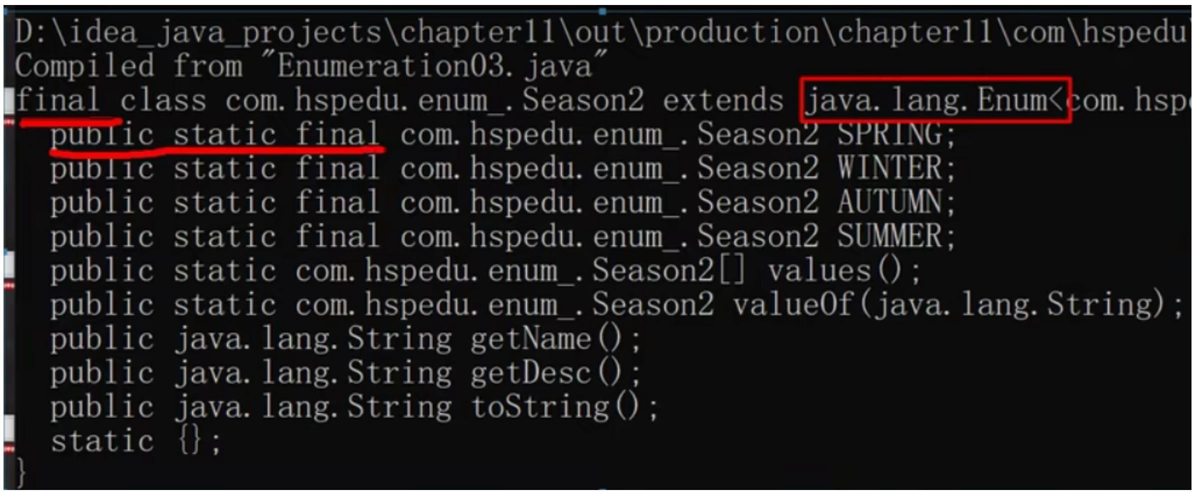


enum关键字实现枚举-快速入门

● enum关键字实现枚举注意事项

1. 当我们使用enum 关键字开发一个枚举类时，默认会继承Enum类, 而且是一个 final 类[如何证明],老师使用javap 工具来演示
2. 传统的 public static final Season2 SPRING = new Season2("春天", "温暖"); 简化成 SPRING("春天", "温暖")， 这里必须知道，它调用的是哪个构造器.
3. 如果使用无参构造器 创建 枚举对象，则实参列表和小括号都可以省略
4. 当有多个枚举对象时，使用,间隔，最后有一个分号结尾
5. 枚举对象必须放在枚举类的行首.

.

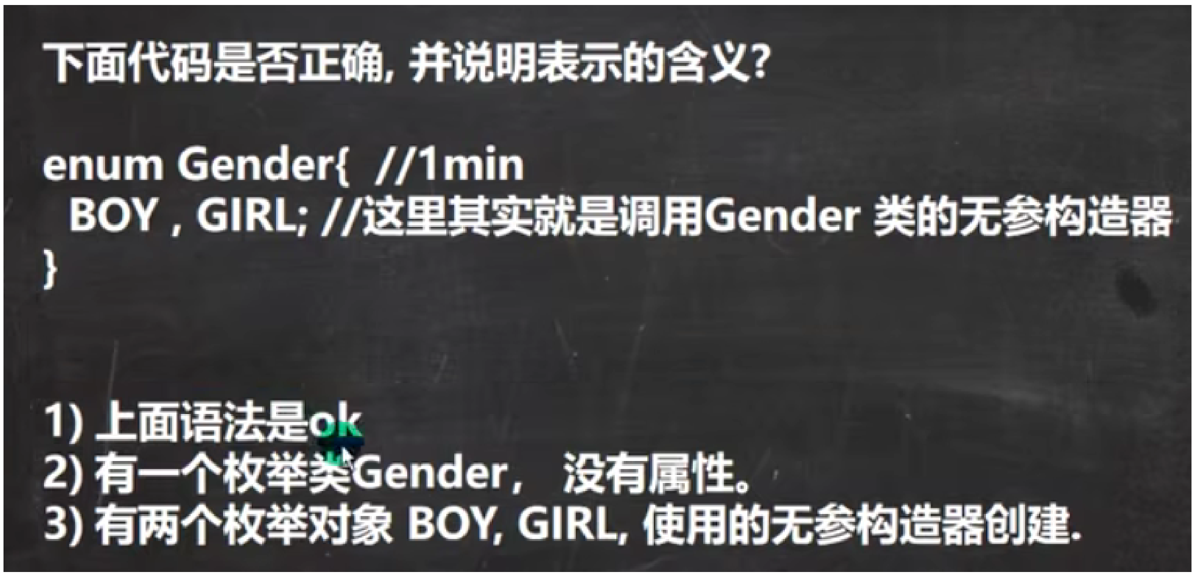


```
D:\idea_java_projects\chapter11\out\production\chapter11\com\hspedu
Compiled from "Enumeration03.java"
final class com.hspedu.enum_.Season2 extends java.lang.Enum<com.hsp
  public static final com.hspedu.enum_.Season2 SPRING;
  public static final com.hspedu.enum_.Season2 WINTER;
  public static final com.hspedu.enum_.Season2 AUTUMN;
  public static final com.hspedu.enum_.Season2 SUMMER;
  public static com.hspedu.enum_.Season2[] values();
  public static com.hspedu.enum_.Season2 valueOf(java.lang.String);
  public java.lang.String getName();
  public java.lang.String getDesc();
  public java.lang.String toString();
  static {};
}
```

.

四、枚举练习

第一题:



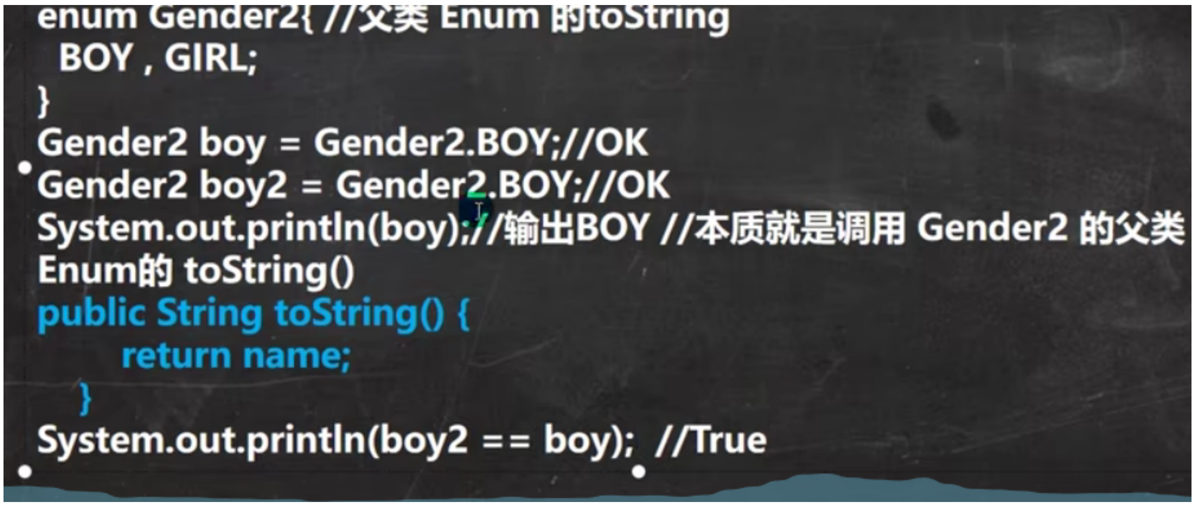
下面代码是否正确, 并说明表示的含义?

```
enum Gender{  //1min
  BOY , GIRL; //这里其实就是调用Gender 类的无参构造器
}
```

1) 上面语法是ok
2) 有一个枚举类Gender， 没有属性。
3) 有两个枚举对象 BOY, GIRL, 使用的无参构造器创建.

.

第二题:


```
enum Gender2{ //父类 Enum 的toString
  BOY , GIRL;
}
Gender2 boy = Gender2.BOY;//OK
Gender2 boy2 = Gender2.BOY;//OK
System.out.println(boy);//输出BOY //本质就是调用 Gender2 的父类
Enum的 toString()
public String toString() {
      return name;
   }
System.out.println(boy2 == boy);  //True
```

.

五、枚举的成员方法

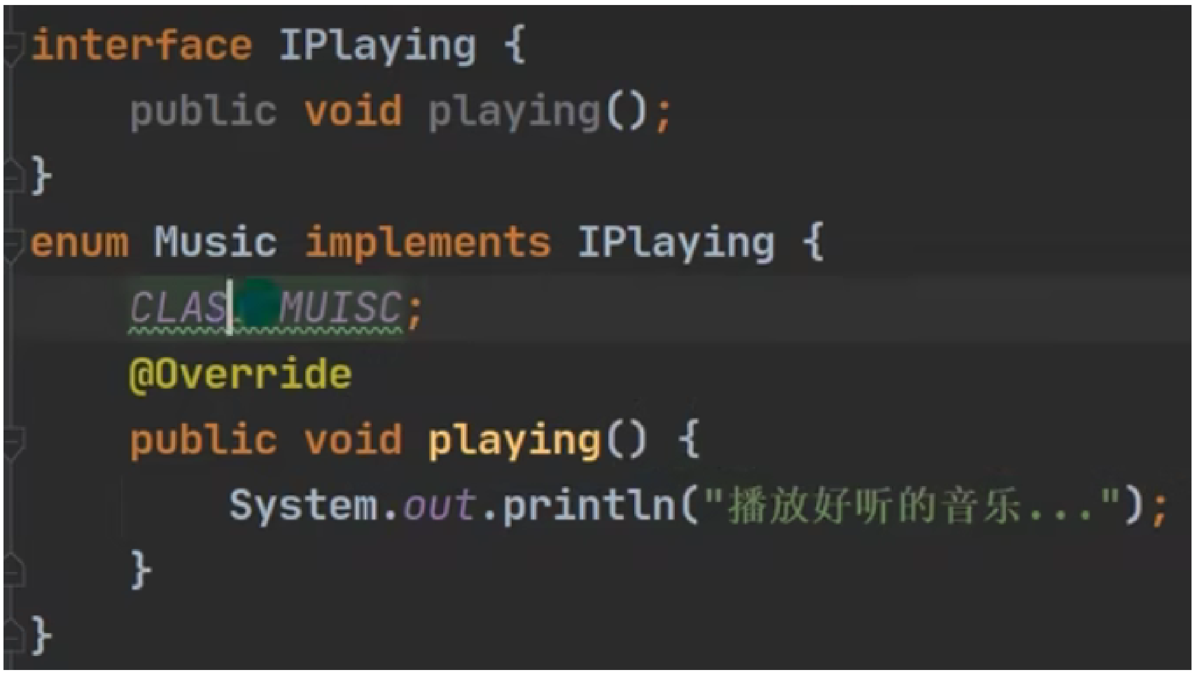
```
package com;
/*
 *@author 杨宗霖
 *@version 1.0
 */
public class EnumMethod {
    public static void main(String[] args) {
        //使用Season2 枚举类来演示各种Enum方法
        Season2 autumn = Season2.AUTUMN;
        System.out.println(autumn);
        //输出枚举对象的名字
        System.out.println(autumn.name());
        //ordinal(),输出的是枚举对象的编号
        System.out.println(autumn.ordinal());
        //从反编译可以看出 values方法(隐藏的),返回Season2[]
        //含有定义的所有枚举对象
        Season2[] values = Season2.values();
        //强力for循环
        /*eg:
        int[] a = {1,2,3}
        for(int i : a){
              System.out.println(a);
        }*/
        for(Season2 season2 : values){
            System.out.println(season2);
        }

        //valueof将字符串转换成枚举对象,要求字符串必须为已有对象,否则报异常
        /*流程:
        * 1.根据你输入的"AUTUNM" 到 Season2的枚举对象去查找
        * 2.若找到了,则返回,若找不到则报错
        * */
        Season2 autumn1 = Season2.valueOf("AUTUMN");//输入的字符串应该在枚举数组内存在
对应的对象,否则报错
        System.out.println("autumn1 = " + autumn1);
        System.out.println(autumn == autumn1);
        //compareTo:比较两个枚举常量,比较的就是编号
        /*解析:
        * 1.Season2.AUTUMN 的编号与 Season2.SUMMER的编号进行比较
        * 2.输出的结果是 前一个枚举对象的编号 - 后面的枚举对象编号*/
        System.out.println(Season2.AUTUMN.compareTo(Season2.SUMMER));
    }
}
enum Season2{//1.使用enum关键字 代替 class
    //要求将常量对象写在最前面
    SPRING("春天","温暖"),
    WINTER("冬天","寒冷"),
    AUTUMN("秋天","凉爽"),
    SUMMER("夏天","炎热");
    private String name;
    private String desc;

 /*   public static final Season SPRING = new Season("春天","温暖");
    public static final Season WINTER = new Season("冬天","寒冷");
    public static final Season AUTUMN = new Season("秋天","凉爽");
    public static final Season SUMMER = new Season("夏天","炎热");*/



    private Season2(String name, String desc) {
        this.name = name;
        this.desc = desc;
    }

    public String getName() {
        return name;
    }



    public String getDesc() {
        return desc;
    }

    @Override
    public String toString() {
        return "Season{" +
                "name='" + name + '\'' +
                ", desc='" + desc + '\'' +
                '}';
    }
}
```

六、练习 & Detail

```
package com.EnumExcer;
/*1.声明Week枚举类,其中包含周一到周日的定义
* MONDAY,TUESDAY,WEDENSAY,THURSDAY,FRIDAY,SATURDAY,SUNDAY
* 2.使用values返回枚举对象的所有数组,并遍历*/
public class Excer01 {
    public static void main(String[] args) {
        Week[] values = Week.values();//valus(): 将枚举的所有对象以数组的形式赋给
values
```

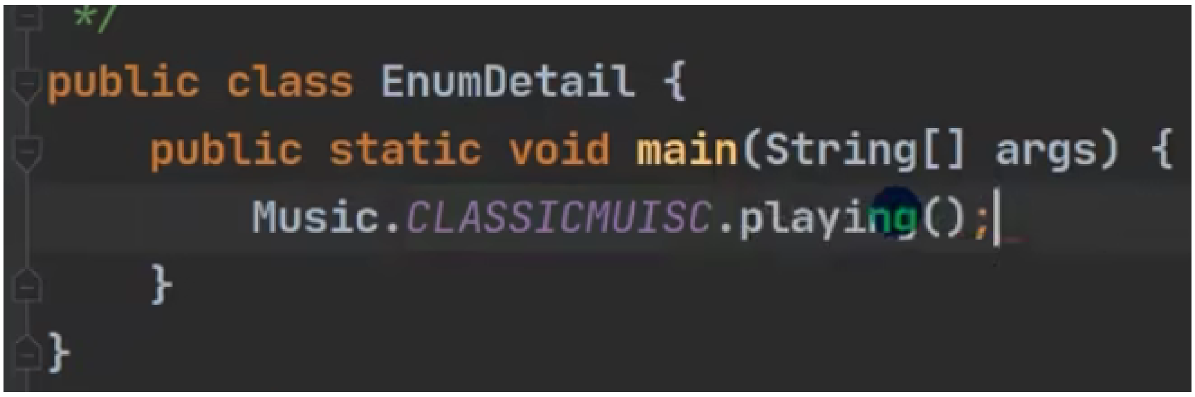
```
        for (Week day : values){//加强for循环的遍历
            System.out.println(day);
        }
    }
}

enum Week{//Detail01:Week没有办法再继承其它类了,因为已经隐式地继承了enum()了
        //Detail02:枚举跟普通类一样可以实现接口的
        /*eg:enum 类名 implements 接口1 ,接口2{}*/
    MONDAY("星期一"),
    TUESDAY("星期二"),
    WEDNESDAY("星期三"),
    THURSDAY("星期四"),
    FRIDAY("星期五"),
    SATURDAY("星期六"),
    SUNDAY("星期日");

    private String day;

    private Week(String day) {//注意:私有
        this.day = day;
    }

    public String getDay() {
        return day;
    }

    @Override
    public String toString() {
        return day ;//调整对象的输出方式
    }
}
```

Detail02:



```
interface IPlaying {
    public void playing();
}
enum Music implements IPlaying {
    CLASSMUISC;
    @Override
    public void playing() {
        System.out.println("播放好听的音乐...");
    }
}
```

.



```
 */
public class EnumDetail {
    public static void main(String[] args) {
        Music.CLASSICMUISC.playing();
    }
}
```

.