你知道枚举的这些神仙用法吗？

Original JAVA旭阳 JAVA旭阳 2023-03-23 07:25 Posted on 浙江

概述

Java中的枚举，大家在项目中经常使用吧，主要用来定义一些固定值，在一个有限的集合内，比如在表示一周的某一天，一年中的四季等。那你了解枚举的本质吗，或者说底层是什么样的？了解枚举的一些神仙用法吗？

枚举介绍和使用

枚举主要用来定义一个有限集合内的固定值。

枚举定义方式如下：

```
// 简单的定义
enum WeekEnum {
    MONDAY, TUESDAY
}

// 有属性的定义
enum StatusEnum {
    ENABLE("1", "启用"), DISABLE("0", "禁用");

    private String code;

    private String name;

    StatusEnum(String code, String name) {
        this.code = code;
        this.name = name;
    }
}
```

枚举编译后实际上继承了java.lang.Enum这个类，后面详细讲解，我们看下这个类的关键方法：

- static Enum valueOf(Class enumClass, String name)

返回指定名字、给定类的枚举常量

- String toString()

返回枚举常量名

- int ordinal()

返回枚举常量在enum中的位置，从0开始

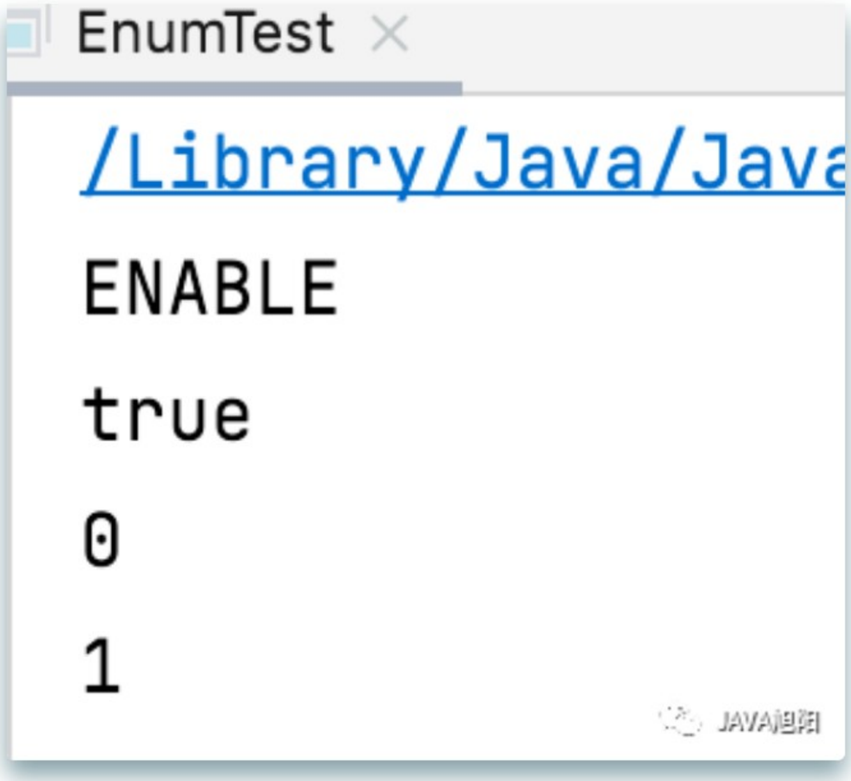
- int compareTo(E other)

如果枚举场景出现在other之前，则返回一个负值，如果this == other,则返回0，否则返回正值。

```
public static void main(String[] args) {
        // 根据字符串获取枚举
        StatusEnum enable = Enum.valueOf(StatusEnum.class, "ENABLE");
        System.out.println(enable);
        //枚举比较直接用==
        System.out.println(enable == StatusEnum.ENABLE);

        // values方法获取所有的枚举
        StatusEnum[] values = StatusEnum.values();
        for (StatusEnum statusEnum : values) {
            // 打印枚举的位置
            System.out.println(statusEnum.ordinal());
        }
    }
```

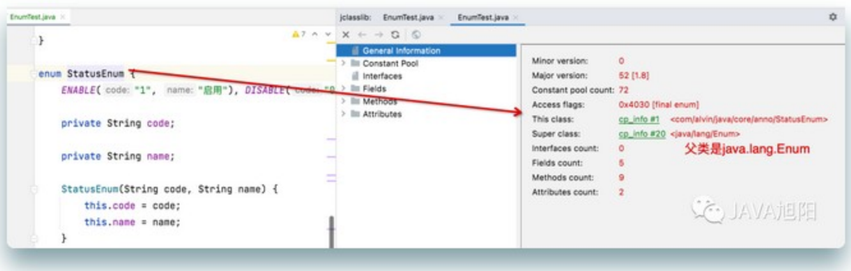
运行结果：



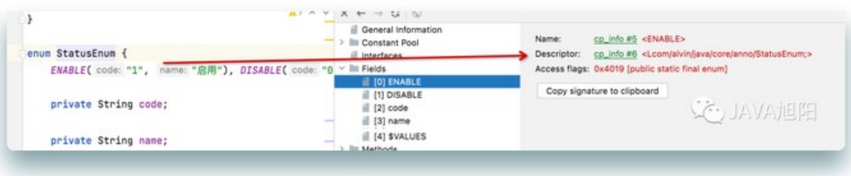
枚举的本质

枚举的本质其实是一个类，继承了java.lang.Enum这个类。我们可以用idea的插件看下生成的字节码如下：

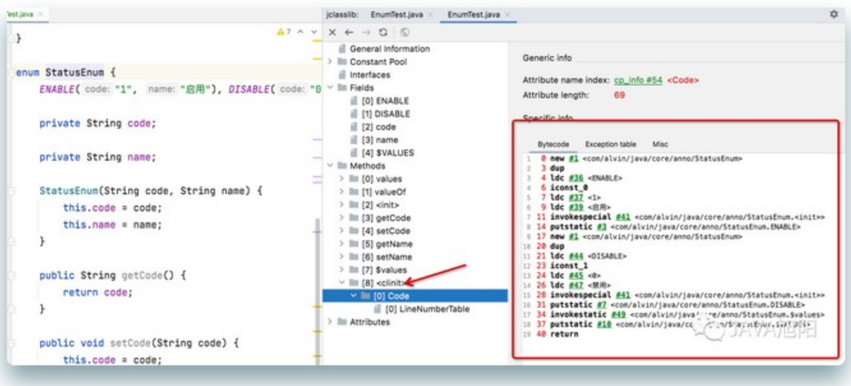
1. 继承了java.lang.Enum



2. ENABLE、DISABLE最终转换为静态字段，类型其实StatusEnum这个类



3. 类加载的时候初始化ENABLE、DISABLE这两个属性。



方法是类加载的初始化阶段就是执行的，它这里的主要逻辑就是创建了两个对象，设置到ENABLE，DISABLE上。

这下你明白枚举是怎么一会事情了吧。

枚举常见用途

枚举创建单例

枚举可以作为单例模式的最佳方式，能够保证单例对象的唯一性。

```
public class User {
    //私有化构造函数
    private User(){ }

    //定义一个静态枚举类
    static enum SingletonEnum{
        //创建一个枚举对象，该对象天生为单例
        INSTANCE;
        private User user;
        //私有化枚举的构造函数
        private SingletonEnum(){
            user = new User();
        }
        public User getInstnce(){
            return user;
        }
    }

    //对外暴露一个获取User对象的静态方法
    public static User getInstance(){
        return SingletonEnum.INSTANCE.getInstnce();
    }
}
```

- 枚举INSTANCE会在类加载初始化的时候创建,而Java类的加载和初始化过程都是线程安全的。
- 枚举可避免反序列化破坏单例。

枚举抽象方法

枚举类可以定义抽象方法，然后让各个具体的枚举实现，可以通过枚举实现不同的策略实现。

```
enum OperEnum {
    ADD(1, 2) {
        @Override
        public Integer operate() {
            return this.getA() + this.getB();
        }
    }, MULTIPY(1, 2) {
        @Override
        public Integer operate() {
            return this.getA() * this.getB();
        }
    };

    private Integer a;

    private Integer b;

    OperEnum(Integer a, Integer b) {
        this.a = a;
        this.b = b;
    }

    public abstract Integer operate();

    public Integer getA() {
        return a;
    }

    public void setA(Integer a) {
        this.a = a;
    }

    public Integer getB() {
        return b;
    }

    public void setB(Integer b) {
        this.b = b;
    }
}
```

总结

本文主要讲解了枚举的本质，以及枚举的常见用法，希望对你们有帮助。

既然看到这了，还请帮忙点赞、在看、转发一下，码字不易，非常感谢！

欢迎点击关注公众号，利用碎片化时间学习，每天进步一点点。