链滴

Java 开发过程中枚举类的封装

作者：zhangzeshan

原文链接：https://ld246.com/article/1602666502441

来源网站：链滴

前景

以前做php的时候,因为自己的代码修养不够,导致一个项目开发完成之后,出现一个很坑爹的问题,就是很多魔术数字,比如用户的状态:正常1 冻结2 删除3 整套项目下来 如果很多地方都会调用到的话,一旦现这些枚举值要进行改动的场景,那简直是一场灾难,甚至枚举类的重要性!

创建枚举接口

这个借口创建之后,其他各种类型的枚举会去实现接口

```
package com.center.common.enumtype;


/**
 * 枚举基类.
 */
public interface BaseEnum {


    /**
     * 获取描述
     * @return
     */
    String getDesc();

    /**
     * 获取编码值(数据库存的)
     * @return
     */
    Byte getCode();
```

```
}
```

创建数字类型的枚举类

整型的数字,字符串的数字,长整型的数字

这个主要用在一些判断结果值之类的场景

```
package com.center.common.enumtype;

/**
 * 枚举
 * @author javashishijieshangzuihaodeyuyan
 */
public class Constants {


    public interface Ints {
        Integer MINUS_ONE = -1;
        Integer ZERO = 0;
        Integer ONE = 1;
        Integer TWO = 2;
        Integer THREE = 3;
        Integer FOUR = 4;
        Integer FIVE = 5;
        Integer SIX = 6;
        Integer SEVEN = 7;
        Integer TEN = 10;
    }

    public interface Strings {
        String ZERO = "0";
        String ONE = "1";
        String TWO = "2";
        String THREE = "3";
        String FOUR = "4";
        String HUNDRED="100";
        String TEN = "10";

    }

    public interface Longs {
        Long ZERO = 0L;
        Long ONE = 1L;
        Long TWO = 2L;
        Long THREE = 3L;
        Long FOUR = 4L;
        Long FIVE = 5L;
        Long SIX = 6L;
        Long SEVEN = 7L;
        Long TEN = 10L;
    }
}
```

使用示范

```
# 读取的值 是整型的数字5
Constants.Ints.FIVE
```

映射数据库的一些tinyint类型字段的枚举

比如订单类型:0代表提现订单,1充值订单,2通道订单

```
package com.center.common.enumtype;

/**
 * Description:订单类型 0用户订单1测试订单2正式订单
 *
 */
public enum OrderTypeEnum implements BaseEnum {

    CASHOUT((byte)0, "用户订单"),
    RECHARGE((byte)1, "测试订单"),
    CHANNEL((byte)2, "正式订单");

    private byte code;
    private String desc;


    OrderTypeEnum(byte code, String desc) {
        this.code = code;
        this.desc = desc;
    }

    @Override
    public String getDesc() {
        return this.desc;
    }

    @Override
    public Byte getCode() {
        return this.code;
    }
}
```

使用示范

```
# 获取 数字  如:1
OrderTypeEnum.CASHOUT.getCode()
# 获取描述  如:测试订单
OrderTypeEnum.CASHOUT.getDesc()
```

缓存常量枚举

这个是给redis读key的时候使用,不然每个地方都去写死字符串,

要改动的话,直接当场去世!

```
package com.pay.common;

/**
 * 缓存常量
 */
public interface CacheKeyContants {
    String IMG_CODE = "IMG_CODE";
    String SESSION_USER = "SESSION_USER";



    // 库
    int DATABASE = 0;
    int ROB_DATABASE = 1;

    // KEY
    String KEY_ROB_USER= "KEY_ROB_USER";



    String KEY_FUUID= "fuuid";

    //时间

    // 分钟
    int ONE_MINUTE = 60;
    //一周
    int ONE_WEEK = 86400 * 7;
    // 一天
    int ONE_DAY = 86400;
    // 一小时
    int ONE_HOUR = 3600;
    // 五分钟
    int FIVE_MINUTES = 300;
    //key

}
```

使用示范

```
#读取value
CacheKeyContants.ROB_DATABASE
```

大致的枚举就可以按照第二个如法炮制

通过这样处理,整个项目的一些值清晰易懂,想改动也只需要改动一个地方!