[在此处输入文章标题]

# 1、String

## 1 isNotEmpty ：

判断某字符串是否非空  
StringUtils.isNotEmpty(null) = false  
StringUtils.isNotEmpty("") = false  
StringUtils.isNotEmpty(" ") = true  
StringUtils.isNotEmpty("bob") = true

## 2 isNotBlank：判断经历过trim后是否为空

判断某字符串是否不为空且长度不为0且不由空白符(whitespace)构成，  
下面是示例：  
StringUtils.isNotBlank(null) = false  
StringUtils.isNotBlank("") = false  
StringUtils.isNotBlank(" ") = false  
StringUtils.isNotBlank("\t \n \f \r") = false

# 2、 Java中compareTo用法

## 解释：java中的compareto方法，返回参与比较的前后两个字符串的asc码的差值

String a = "a";

String b = "b";

System.out.println(a.compareTo(b));

输出值-1

String a = "b";

String b = "a";

System.out.println(a.compareTo(b));

输出值1

String a = "a";

String b = "a";

System.out.println(a.compareTo(b));

输出0

## 1、两个字符串首字母不同，则该方法返回首字母的asc码的差值

String a = "abc";

String b = "bcdfg";

System.out.println(a.compareTo(b));

输出-1

## 2、参与比较的两个字符串如果首字符相同，则比较下一个字符，直到有不同的为止，返回该不同的字符的asc码差值

String a = "abc";

String b = "abedfg";

System.out.println(a.compareTo(b));

输出-2

## 3、两个字符串不一样长，可以参与比较的字符又完全一样，则返回两个字符串的长度差值

String a = "abc";

String b = "abcdefg";

System.out.println(a.compareTo(b));

输出-4

String a = "abcde";

String b = "abcd";

System.out.println(a.compareTo(b));

输出1

# 3、Integer

### 1、int比较

public static final int ***TYPE\_SHIKE\_OTHER*** = 4;  
public static final int ***TYPE\_SHIKE\_ENTERPRISE\_LOGIN*** = 5;  
public static final int ***TYPE\_SHIKE\_ENTERPRISE\_LOGIN\_NO*** = 6;

public String getTypeStr() {  
 if(this.type == null) return "未知";  
 if(this.type.intValue() == ***TYPE\_SHIKE***){  
 return "试客";  
 }else if(this.type.intValue() == ***TYPE\_OUTSIDE***){  
 return "OPEN平台";  
 }else if(this.type.intValue() == ***TYPE\_MOREAPPS***){  
 return "MOREAPPS";  
 }else if(this.type.intValue() == ***TYPE\_SHIKE\_OTHER***){  
 return "试客其它";  
 }else if(this.type.intValue() == ***TYPE\_SHIKE\_ENTERPRISE\_LOGIN***){  
 return "试客企业版登录";  
 }else if(this.type.intValue() == ***TYPE\_SHIKE\_ENTERPRISE\_LOGIN\_NO***){  
 return "试客企业版无登录";  
 }else {  
 return "未知平台";  
 }  
}

### 2、integer 的比较

public String getHaveBackstageDesc(){  
 if (this.haveBackstage == null){  
 return "未知";  
 }  
 if (this.haveBackstage.compareTo(0) == 0){  
 return "无后台";  
 } else if (this.haveBackstage.compareTo(1) == 0){  
 return "有后台";  
 }else {  
 return "未知";  
 }  
  
}

# 4、list取得有逗号分隔的集合

List<String> ccStrList = Arrays.asList(ccIds.split(","));

public static List<EnumCrmUserStatus> getTypeList() {  
 return Arrays.asList(values());  
 }

## 2、list. subList根据包头不包尾部截取

//包头不包尾比如1 2 曲子list index 0 1  
List<Long> idParams = ids.subList(key.intValue()-1, map.get(key).intValue());

for(Long key:map.keySet()){  
 List<Long> idParams = ids.subList(key.intValue()-1, map.get(key).intValue());  
 couponItemGoods = couponItemGoodMapper.dataCpuponItemGoodfindByParamsIds(idParams);  
 if (listKeyMonitor(couponItemGoods, flag)){  
 return;  
 }  
}

## 3、String集合加入逗号，变成字符串

this.tagidList = StringUtils.join(tagid\_list,",");

## 5、比较字符串是否相等

if (!StringUtils.equals(remoteAccount,sessionAccount)) {

# 6、集合类的包装类遍历

List<SuperTaskRecordVertify> receiveList = clearSuperTaskRecordMapper.getSuperMissionVertifyInfo(userId);  
CollectionUtils.emptyIfNull(receiveList).stream().forEach(item -> {  
 if ("1".equals(item.getValidateType())){  
 item.setValidateType("手机");  
 }else if ("2".equals(item.getValidateType())){  
 item.setValidateType("idfa验证");  
 }  
});

# 7、正则表达式判断字符串

/\*\*  
 \* 判断是不是由数字组成  
 \*  
 \* @param str  
 \* @return  
 \*/  
public static boolean isNumeric(String str){  
 Pattern pattern = Pattern.compile("[0-9]\*");  
 return pattern.matcher(str).matches();  
}

# 8、字符中和集合之间的相互转化

String array[] = {"1","2","3"};  
 List<String> list ;  
 String str ;  
//1、数组转化为集合  
 list = Arrays.stream(array).collect(Collectors.toList());  
 //2、  
list = Arrays.asList(array);  
  
 //2、集合转换为数组  
 array = list.stream().toArray(String[]::new);  
 //2、  
 array = list.toArray(new String[0]);  
  
 //集合转化为逗号的字符串  
 //1、  
 str = list.stream().collect(Collectors.joining(","));  
 System.***out***.println(str);  
 //2、  
 str = StringUtils.join(list,"," );  
 System.***out***.println(str);  
  
 //1、数组转化为带逗号的字符串  
 str = Arrays.stream(array).collect(Collectors.joining(","));  
 System.***out***.println(str);  
  
 //逗号字符串转化为数组  
 array = str.split(",");  
 //逗号字符串转化为集合  
 list = Arrays.asList(str.split(",")) ;

# 9、linkedlist用法

## <https://blog.csdn.net/gongchuangsu/article/details/51527042>

# 10、替换变量

/\*\*  
 新字符串使用本地语言环境，制定字符串格式和参数生成格式化的新字符串。  
 format(String format, Object... args)  
  
 %s 字符串类型 "mingrisoft" ，也可以放入其他类型，比如，整数，小数等等  
 %c 字符类型 'm'  
 %b 布尔类型 true  
 %d 整数类型（十进制） 99 %6d  
 %.4f%% 小数 55.1510%

\*/  
@Test  
public void strisngFormat(){  
  
 String s ="\_\_\_\_%s\_\_\_\_";  
 //\_\_\_\_mingrisoft\_\_\_\_  
 System.***out***.println(String.format(s, "mingrisoft"));  
  
 String c ="\_\_\_\_%c\_\_\_\_";  
 //\_\_\_\_c\_\_\_\_  
 System.***out***.println(String.format(c, 'c'));  
  
 String b = "\_\_\_\_%b\_\_\_\_";  
 //\_\_\_\_true\_\_\_\_  
 System.***out***.println(String.format(b, true));  
  
 String d = "\_\_\_\_%d\_\_\_\_";  
 //\_\_\_\_125\_\_\_\_  
 System.***out***.println(String.format(d, 125));  
  
 String str = "\_\_\_\_%s\_\_\_\_"+ "\_\_\_\_%c\_\_\_\_"+"\_\_\_\_%b\_\_\_\_"+"\_\_\_\_%d\_\_\_\_" ;  
 System.***out***.println(String.format(str, "mingrisoft",'c',true,123));  
}

@Test  
public void format(){  
  
 //1、展位  
 String msg = "{0},{1},{2},{3},{4},{5},{6},{7},{8}";  
 Object [] array = new Object[]{"A","B","C","D","E","F",1525.12,10,BigDecimal.***ZERO***};  
 String value = MessageFormat.format(msg, array);  
 // 输出：A,B,C,D,E,F,1,525.12,10,0  
 System.***out***.println(value);  
  
  
 // 2、格式化字符串时，两个单引号才表示一个单引号，单个单引号会被省略，除非中文单引号不会被省略，如：  
 value = MessageFormat.format("oh, {0} is 'a' ''pig''", "ZhangSan");  
 // oh, ZhangSan is a 'pig'  
 System.***out***.println(value);  
  
}

/\*\*  
 \* %05d   
 \* 0 代表前面补充0   
 \* 5 代表长度为5   
 \* d 代表参数为正数型   
 \* 结果:00024   
 \*\*/

int count = scfCreditBillReceivableManager.countPackOnBussNo(query);  
String countStr = String.format("%03d", count);