回顾:复习文档

# 语言包 lang包

java中内置的基础包,使用的时候会默认导入

# 枚举定义

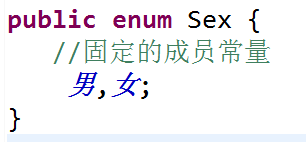
枚举类型可以取代以往的常量的定义方式,即将常量封装在类或接口中,此外,它还提供了安全检查功能. 枚举本质上还是以类的形式存在,枚举类型增强了程序代码的健壮性

枚举定义:有很多现实的对象属性的属性值其实是固定:性别,季节,节气

关键字:enum

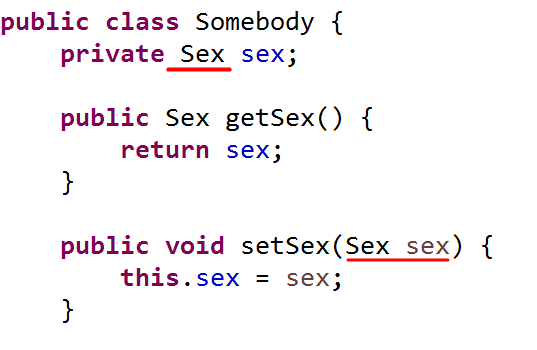
## 语法:

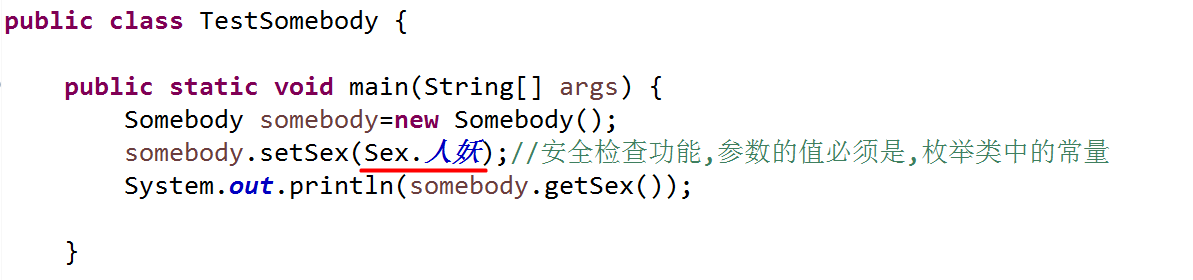
定义一个枚举类的语法: public enum 类名{成员常量1,成员常量2....}



## 实现:

将枚举类作为参数的数据类型,那么参数值就必须是枚举类中的常量值





## 枚举的好处

类型安全,易于输入,代码清晰

# 包装类：

## 概念：

Java中包含2大数据类型：基本数据类型（原始数据类型）、引用数据类型  
  
之前学习的基本数据类型在某一些场合使用不了，比如集合（集合存放只能是引用数据类型）。  
Java提供一系列包装类，将原始数据类型变成引用数据类型。  
  
在java.lang包中，对于每个原始数据类型都有一个相应的包装类。

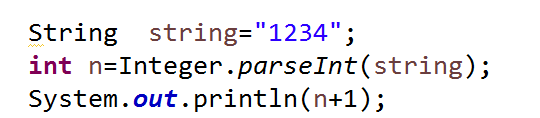
包装类->引用类型

|  |  |
| --- | --- |
| ***原始数据类型*** | ***包  装  类*** |
| ***boolean（布尔型）*** | ***Boolean*** |
| ***byte（字节型）*** | ***Byte*** |
| ***char（字符型）*** | ***Character*** |
| ***short（短整型）*** | ***Short*** |
| ***int（整型）*** | ***Integer*** |
| ***long（长整型）*** | ***Long*** |
| ***float（浮点型）*** | ***Float*** |
| ***double（双精度浮点型）*** | ***Double*** |

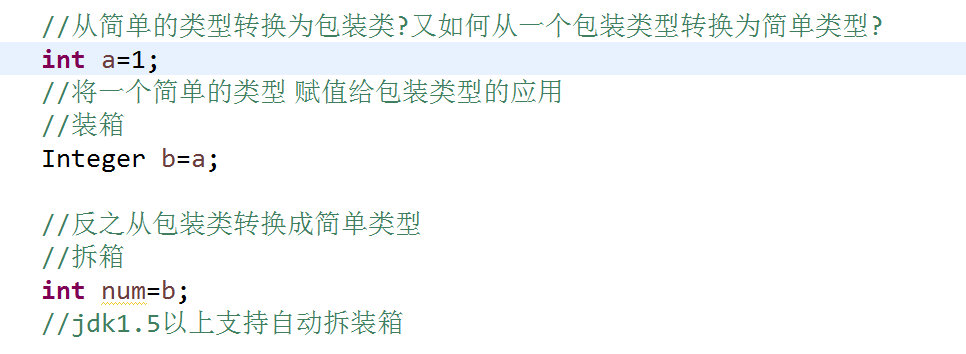
已经有了基本类型，为什么还要提供包装类呢?

1. 和面向对象思想一致(普通类型没有方法属性 )；

2. 和其他类型进行转换，没有包装类，比较麻烦(例如将字符串转换为int)



从简单的类型转换为包装类?又如何从一个包装类型转换为简单类型?



## 包装类的自动拆装箱

JDK5.0以后提供了自动拆装箱的操作。

自动装箱： 自动将 基本数据类型 变成 包装类

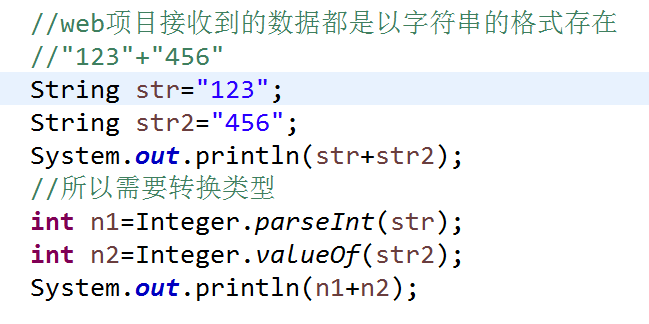
自动拆箱： 自动将 包装类 变成 基本数据类型

## 包装类的常用方法

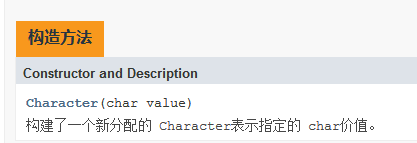
1.字符串类型和简单类型(包装类)之间如何转换

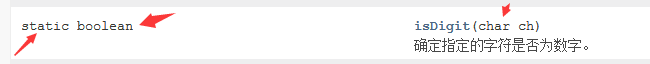


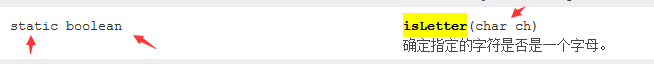




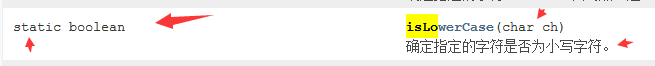
## Character类常用方法

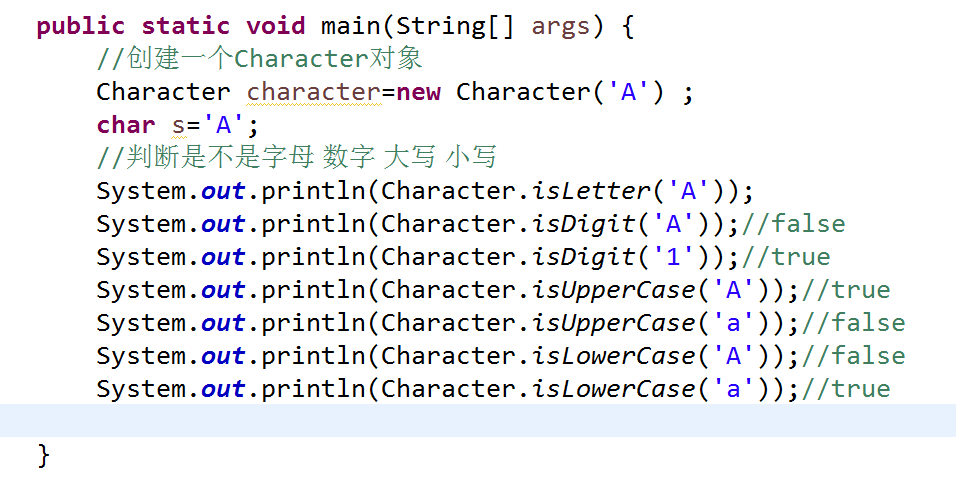












注意：包装类并不是用来取代基本类型的。

之前学习的基本数据类型8种在某一些场合用不了，有一些类（传递）的参数只能是引用数据类型(例如:集合)

## 常量池：

为了一定程度上减少频繁创建对象，将一些对象保存到一个"容器"中。

Byte,Short,Integer,Long,Character。这5种整型的包装类的对象池范围在-128~127之间，也就是说，超出这个范围的对象都会开辟自己的堆内存。 另外两种浮点数类型的包装类则没有实现；

当这些有常量池的包装类对应的值在-128到127的范围内，用==判断的时候是相等，因为指向同一个地址。



## 【面试题】

① Integer和int的区别

1、Integer是int的包装类，int则是java的一种基本数据类型

2、Integer变量必须实例化后才能使用，而int变量不需要

3、Integer实际是对象的引用，当new一个Integer时，实际上是Integer的引用指向此对象；而int则是直接存储数据值 。

4、Integer的默认值是null，int的默认值是0

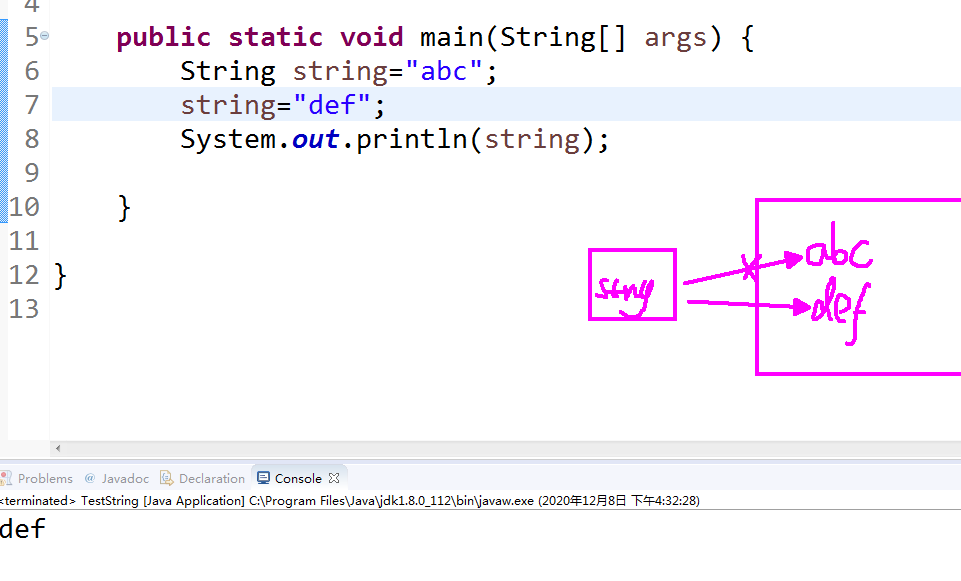
② 解释自动拆装箱

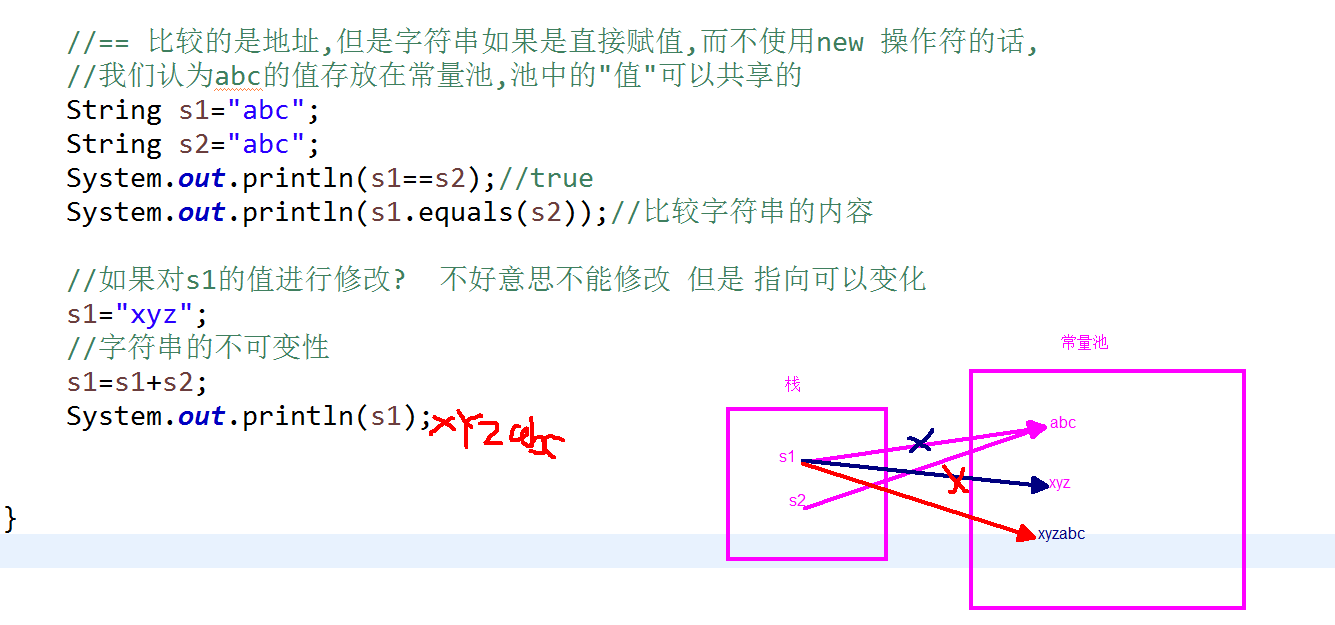
自动拆装箱在Java5(就是Java1.5,后改了命名)时被引入，自动装箱就是Java自动将基础类型值转换成对应的包装类对象，比如将int的变量转换成Integer对象，这个过程叫做装箱，反之将Integer对象转换成int类型值，这个过程叫做拆箱。

装箱过程是通过调用包装器的valueOf方法（譬如Integer的valueOf(int)）实现的，而拆箱过程是通过调用包装器的 xxxValue方法（譬如Integer的intValue()）实现的

# String







以上的代码请大家好好理解

下面给出了一些如何使用字符串的更多示例：

System.out.println("abc");

String cde = "cde";

System.out.println("abc" + cde);

String c = "abc".substring(2,3);

String d = cde.substring(1, 2);

String 类包括的方法可用于检查序列的单个字符、比较字符串、搜索字符串、提取子字符串、创建字符串副本并将所有字符全部转换为大写或小写。大小写映射基于 [Character](mk:@MSITStore:E:\\API帮助文档\\jdk1.6\\JDK_API_1_6.CHM::/java/lang/../../java/lang/Character.html" \o "java.lang 中的类) 类指定的 Unicode 标准版。

|  |  |
| --- | --- |
| **构造方法摘要** | |
| **[String](mk:@MSITStore:E:\\API帮助文档\\jdk1.6\\JDK_API_1_6.CHM::/java/lang/../../java/lang/String.html" \l "String())**()            初始化一个新创建的 String 对象，使其表示一个空字符序列。 |  |
| **[String](mk:@MSITStore:E:\\API帮助文档\\jdk1.6\\JDK_API_1_6.CHM::/java/lang/../../java/lang/String.html" \l "String(char[]))**(char[] value)            分配一个新的 String，使其表示字符数组参数中当前包含的字符序列。   |  | | --- | | **[String](mk:@MSITStore:E:\\API帮助文档\\jdk1.6\\JDK_API_1_6.CHM::/java/lang/../../java/lang/String.html" \l "String(java.lang.String))**([String](mk:@MSITStore:E:\\API帮助文档\\jdk1.6\\JDK_API_1_6.CHM::/java/lang/../../java/lang/String.html" \o "java.lang 中的类) original)            初始化一个新创建的 String 对象，使其表示一个与参数相同的字符序列；换句话说，新创建的字符串是该参数字符串的副本。 | | **[String](mk:@MSITStore:E:\\API帮助文档\\jdk1.6\\JDK_API_1_6.CHM::/java/lang/../../java/lang/String.html" \l "String(java.lang.StringBuffer))**([StringBuffer](mk:@MSITStore:E:\\API帮助文档\\jdk1.6\\JDK_API_1_6.CHM::/java/lang/../../java/lang/StringBuffer.html" \o "java.lang 中的类) buffer)            分配一个新的字符串，它包含字符串缓冲区参数中当前包含的字符序列。 | | **[String](mk:@MSITStore:E:\\API帮助文档\\jdk1.6\\JDK_API_1_6.CHM::/java/lang/../../java/lang/String.html" \l "String(java.lang.StringBuilder))**([StringBuilder](mk:@MSITStore:E:\\API帮助文档\\jdk1.6\\JDK_API_1_6.CHM::/java/lang/../../java/lang/StringBuilder.html" \o "java.lang 中的类) builder)            分配一个新的字符串，它包含字符串生成器参数中当前包含的字符序列。 | |  |
|  |  |

**常见方法:**

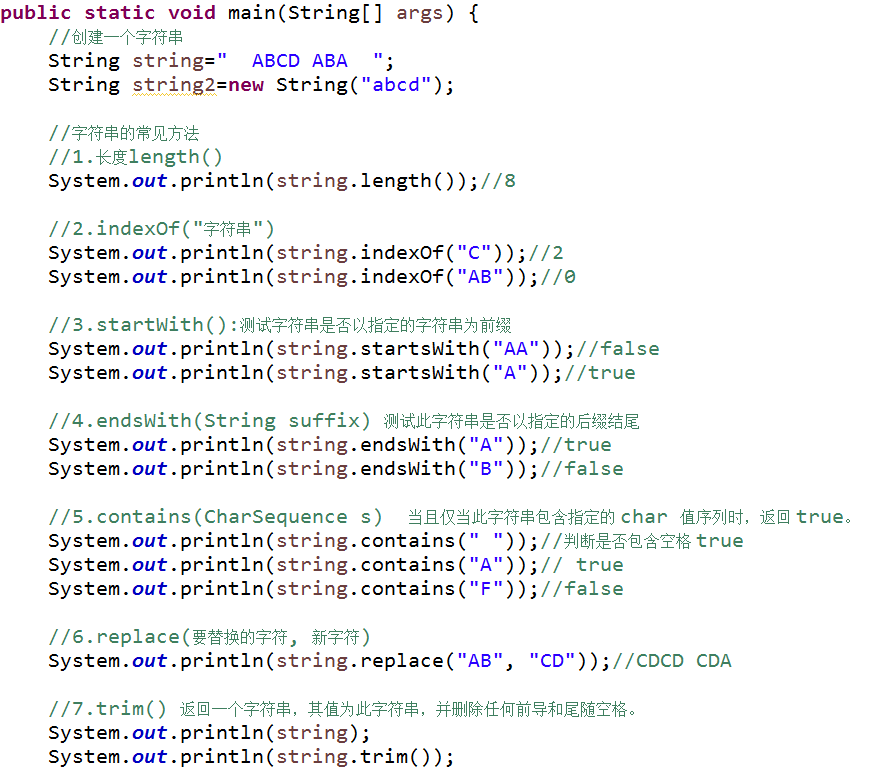
|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| char | | **[charAt](mk:@MSITStore:E:\\标准化教案\\API帮助文档\\JDK_API_1_6.CHM::/java/lang/String.html" \l "charAt(int))**(int index)  返回指定索引处的 char 值。 | | | |
| boolean | | **[contains](mk:@MSITStore:E:\\标准化教案\\API帮助文档\\JDK_API_1_6.CHM::/java/lang/String.html" \l "contains(java.lang.CharSequence))([CharSequence](mk:@MSITStore:E:\\标准化教案\\API帮助文档\\JDK_API_1_6.CHM::/java/lang/CharSequence.html" \o "java.lang 中的接口) s)  当且仅当此字符串包含指定的 char 值序列时，返回 true。** | | | |
| boolean | | **[equals](mk:@MSITStore:E:\\标准化教案\\API帮助文档\\JDK_API_1_6.CHM::/java/lang/String.html" \l "equals(java.lang.Object))([Object](mk:@MSITStore:E:\\标准化教案\\API帮助文档\\JDK_API_1_6.CHM::/java/lang/Object.html" \o "java.lang 中的类) anObject)  将此字符串与指定的对象比较。** | | | |
| int | | **[indexOf](mk:@MSITStore:E:\\标准化教案\\API帮助文档\\JDK_API_1_6.CHM::/java/lang/String.html" \l "indexOf(java.lang.String))([String](mk:@MSITStore:E:\\标准化教案\\API帮助文档\\JDK_API_1_6.CHM::/java/lang/String.html" \o "java.lang 中的类) str)  返回指定子字符串在此字符串中第一次出现处的索引。** | | | |
| int | | **[length](mk:@MSITStore:E:\\标准化教案\\API帮助文档\\JDK_API_1_6.CHM::/java/lang/String.html" \l "length())()  返回此字符串的长度。** | | | |
| boolean | | **[isEmpty](mk:@MSITStore:E:\\标准化教案\\API帮助文档\\JDK_API_1_6.CHM::/java/lang/String.html" \l "isEmpty())()  当且仅当 [length()](mk:@MSITStore:E:\\标准化教案\\API帮助文档\\JDK_API_1_6.CHM::/java/lang/String.html" \l "length()) 为 0 时返回 true。** | | | |
| boolean | | **[matches](mk:@MSITStore:E:\\标准化教案\\API帮助文档\\JDK_API_1_6.CHM::/java/lang/String.html" \l "matches(java.lang.String))([String](mk:@MSITStore:E:\\标准化教案\\API帮助文档\\JDK_API_1_6.CHM::/java/lang/String.html" \o "java.lang 中的类) regex)  告知此字符串是否匹配给定的[正则表达式](mk:@MSITStore:E:\\标准化教案\\API帮助文档\\JDK_API_1_6.CHM::/java/util/regex/Pattern.html" \l "sum)。** | | | |
| static [String](mk:@MSITStore:E:\\标准化教案\\API帮助文档\\JDK_API_1_6.CHM::/java/lang/String.html" \o "java.lang 中的类) | | **[valueOf](mk:@MSITStore:E:\\标准化教案\\API帮助文档\\JDK_API_1_6.CHM::/java/lang/String.html" \l "valueOf(int))(int i)  返回 int 参数的字符串表示形式。** | | | |
| boolean | **[equalsIgnoreCase](mk:@MSITStore:E:\\API帮助文档\\jdk1.6\\JDK_API_1_6.CHM::/java/lang/../../java/lang/String.html" \l "equalsIgnoreCase(java.lang.String))**([String](mk:@MSITStore:E:\\API帮助文档\\jdk1.6\\JDK_API_1_6.CHM::/java/lang/../../java/lang/String.html" \o "java.lang 中的类) anotherString)            将此 String 与另一个 String 比较，不考虑大小写。 | | | |
| [String](mk:@MSITStore:E:\\API帮助文档\\jdk1.6\\JDK_API_1_6.CHM::/java/lang/../../java/lang/String.html" \o "java.lang 中的类) | [toUpperCase](mk:@MSITStore:E:\\API帮助文档\\jdk1.6\\JDK_API_1_6.CHM::/java/lang/../../java/lang/String.html" \l "toUpperCase())()            使用默认语言环境的规则将此 String 中的所有字符都转换为大写。 | | | |
| [String](mk:@MSITStore:E:\\API帮助文档\\jdk1.6\\JDK_API_1_6.CHM::/java/lang/../../java/lang/String.html" \o "java.lang 中的类) | [toLowerCase](mk:@MSITStore:E:\\API帮助文档\\jdk1.6\\JDK_API_1_6.CHM::/java/lang/../../java/lang/String.html" \l "toLowerCase())()            使用默认语言环境的规则将此 String 中的所有字符都转换为小写。 | | | |
| boolean | | | | | [startsWith](mk:@MSITStore:D:\\赣南师大16级软件工程\\课程内容\\jdk%20api%201.8_google.CHM::/java/lang/../../java/lang/String.html" \l "startsWith-java.lang.String-)([String](mk:@MSITStore:D:\\赣南师大16级软件工程\\课程内容\\jdk%20api%201.8_google.CHM::/java/lang/../../java/lang/String.html" \o "class in java.lang) prefix)  测试此字符串是否以指定的前缀开头。 |
| boolean | | | [endsWith](mk:@MSITStore:D:\\赣南师大16级软件工程\\课程内容\\jdk%20api%201.8_google.CHM::/java/lang/../../java/lang/String.html" \l "endsWith-java.lang.String-)([String](mk:@MSITStore:D:\\赣南师大16级软件工程\\课程内容\\jdk%20api%201.8_google.CHM::/java/lang/../../java/lang/String.html" \o "class in java.lang) suffix)  测试此字符串是否以指定的后缀结尾。 | | |
| [String](mk:@MSITStore:D:\\赣南师大16级软件工程\\课程内容\\jdk%20api%201.8_google.CHM::/java/lang/../../java/lang/String.html" \o "class in java.lang) | | | | [replace](mk:@MSITStore:D:\\赣南师大16级软件工程\\课程内容\\jdk%20api%201.8_google.CHM::/java/lang/../../java/lang/String.html" \l "replace-char-char-)(char oldChar, char newChar)  返回从替换所有出现的导致一个字符串 oldChar在此字符串 newChar 。 | |
| [String](mk:@MSITStore:D:\\赣南师大16级软件工程\\课程内容\\jdk%20api%201.8_google.CHM::/java/lang/../../java/lang/String.html" \o "class in java.lang) | | | | [trim](mk:@MSITStore:D:\\赣南师大16级软件工程\\课程内容\\jdk%20api%201.8_google.CHM::/java/lang/../../java/lang/String.html" \l "trim--)()  返回一个字符串，其值为此字符串，并删除任何前导和尾随空格。 | |

## 如何创建一个字符串对象:

1. String str=”我是字符串”;
2. String str=new String(“我也是字符串”);
3. String str=new String(char字符数组);
4. String str=new String(stringbuffer);

以上几种比较常见,还包含了其他的方法,请见api文档中的构造方法

## 【面试题】String常见方法：



## 单元测试:

### 导包:



### 语法要求

