# 尚马教育 JAVA 基础课程

# 字符串类

文档编号：A12

创建日期： 2017-04-12

最后修改日期：2019-09-23

版 本 号：V3.0

电子版文件名：尚马教育-第一阶段-12.字符串类专题课程.docx

**文档修改记录：**

|  |  |  |  |
| --- | --- | --- | --- |
| 更新日期 | 更新作者 | 更新说明 | 版本号 |
| 2017-07-30 | 张元林 | 初始版本 | V1.0 |
| 2018-08-01 | 王绍成 | Java基础版本更新 | V2.0 |
| 2019-08-09 | 徐丽莎 | Java基础版本更新 | V3.0 |

**主讲人：**

**徐丽莎**

目录

[尚马教育 JAVA 基础课程 1](#_Toc14490)

[字符串类 1](#_Toc21184)

[1. String 2](#_Toc15425)

[1.1. 常用构造 2](#_Toc5523)

[1.2. 常用方法 3](#_Toc20239)

[1.2.1. 比较判断方法 3](#_Toc32120)

[1.2.2. 字符串搜索 4](#_Toc8214)

[1.2.3. 提取字符串 5](#_Toc13150)

[1.2.4. 大小写转换 5](#_Toc14859)

[1.2.5. 数据格式转化 6](#_Toc3749)

[1.2.6. 正则表达式 6](#_Toc6242)

[2. StringBuffer 8](#_Toc24635)

[2.1. 常用构造 8](#_Toc9845)

[2.2. 常用方法 9](#_Toc21416)

[2.3. 案例 9](#_Toc216)

[3. StringBuilder 10](#_Toc32157)

[4. 区别 10](#_Toc21056)

[5. 作业 10](#_Toc11191)

## String

* Java中，字符串是String类的对象；
* 可以通过使用String类提供的方法来完成对字符串的操作；
* 创建一个字符串对象之后，将不能更改构成字符串的字符；
* 每当更改了字符串版本时，就创建了一个新的字符串对象，并在其内包含所做的修改，原始字符串保持不变。

### 常用构造

* String类的构造方法共有13种重载方式，以下是常用的几个

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  |  | | --- | --- | | **构造方法** | **说    明** | | **String()** | **将创建一个空字符串** | | **String(String original)** | **将新建一个字符串作为指定字符串的副本** | | **String(char[] value)** | **将根据字符数组构造一个新字符串** | | **String(byte[] tytes)** | **将通过转换指定的字节数组新建一个字符串** | | **String(byte[] bytes, Charset charset)** | **使用指定编码格式对字节数组进行解码** | | **[String](mk:@MSITStore:C:\\Users\\Administrator\\Desktop\\jdk%20api%201.8_中文.CHM::/java/lang/../../java/lang/String.html" \l "String-byte:A-int-int-)(byte[] bytes, int offset, int length)** | **将通过转换指定的字节数组新建一个字符串** | |
| **public** **class** StringDemo {  **public** **static** **void** main(String[] args) {  **char**[] aryChar = { 'J', 'A', 'V', 'A' };  String str1 = "JAVASM";  // 利用一个字符串常量值创建新的字符串  String str2 = **new** String("JAVASM");  // 利用一个字符型数组创建新的字符串  String str3 = **new** String(aryChar);  System.***out***.println(str1);  System.***out***.println(str2);  System.***out***.println(str3);  }  } |

### 常用方法

#### 比较判断方法

|  |
| --- |
|  |
|  |

#### 字符串搜索

* 如果需要搜索某个字符（或某个子串）在字符串中是否出现过，这就要使用到indexOf方法和lastIndexOf方法。

|  |
| --- |
|  |
| **public** **class** StringDemo {  **public** **static** **void** main(String[] args) {  String strEmail = "java@sun.com";  **int** index;  System.***out***.println("E-mail地址：" + strEmail);  index = strEmail.indexOf('@');  System.***out***.println("@字符出现的索引：" + index);  index = strEmail.indexOf("sun");  System.***out***.println("字符串\"sun\"出现的索引：" + index);  index = strEmail.lastIndexOf('a');  System.***out***.println("a字符最后一次出现的索引：" + index);  }  } |

#### 提取字符串

|  |
| --- |
|  |
| **public** **class** StringDemo {  **public** **static** **void** main(String[] args) {  String str1 = " Java is OOP";  String str2 = **new** String("Java");  System.***out***.println(str1.charAt(2));  System.***out***.println(str1.substring(5));  System.***out***.println(str1.substring(2, 9));  System.***out***.println(str1.concat(str2));  System.***out***.println(str1 + str2);  System.***out***.println(str1.replace('a', 'e'));  System.***out***.println(str1.trim());  }  } |

#### 大小写转换

* 有时候，我们需要将字符串中字符的大小写形式进行转换

|  |
| --- |
|  |
| **public** **class** StringDemo {  **public** **static** **void** main(String[] args) {  String str1 = "Java is OOP";  String str2;  str2 = str1.toLowerCase();  System.***out***.println(str2);  str2 = str1.toUpperCase();  System.***out***.println(str2);  }  } |

#### 数据格式转化

|  |
| --- |
|  |

#### 正则表达式

* 正则表达式就是结合特殊的语法以及特殊的字符组成的字符串内容。
* 用来校验，替换与正则模式相符的文本内容。
* 正则在线手册：https://tool.oschina.net/uploads/apidocs/jquery/regexp.html
* 语法:"^([]{})([]{})([]{})$"
* 场景:一般都是校验用户录入的数据是否符合正则要求。
* String类里面支持正则表达式的方法
  + String[] split(String regex)
  + String replaceFirst(String regex, String replacement)
  + String replaceAll(String regex, String replacement)
  + boolean matches(String regex)
* 案例：
  + 校验用户名是否合法。

|  |
| --- |
| **public** **class** StringDemo {  **private** **static** **void** testRegex() {  //正则表达式校验用户提交的用户名以及密码的数据  String name = "aaaa#11";// 字母\_数字 \_特殊符号  //用户名首字母必须大写 有且只有一个 A-Z  //3-5个小写的字母  //特殊符号 \_ 1  //数字 1,3个  // String nameRegex = "^[A-Za-z0-9]{4,7}[\_|#|\*][0-9]{1,3}$";  String nameRegex = "^([\\w]{4,7})[\_|#|\*][0-9]{1,3}$";  System.***out***.println(name.matches(nameRegex));//true  //先去校验正则表达式语法是否是正确的  Pattern pattern = Pattern.compile(nameRegex);  //创建比较器对象  Matcher matcher = pattern.matcher(name);  //校验 匹配  System.***out***.println(matcher.matches());  }  } |

* + 使用当前字符替换字符串里面出现>=1次的字符内容

|  |
| --- |
| **public** **class** StringDemo {  **private** **static** **void** testRegex1() {  String str = "aaabbaaaaaaaaccddaa我我我";  System.***out***.println(str.replace(" ", ""));  System.***out***.println(str.replaceAll(" ", ""));  System.***out***.println(str.replaceAll("\\s{1,}", ""));  //需求：将字符串里面出现>=1字符 使用自己进行替换  // System.out.println(str.replaceAll("\\s\*", ""));  // System.out.println(str.replaceAll("[a]{1,}", "a"));  //通配任意的字符  String regex = "(.)\\1+";// \1 第一个域段里面的内容 $1: 获得第一个域段里面的数据  regex = "(.)\\1\*";  System.***out***.println(str.replaceAll(regex, "$1"));  }  } |

## StringBuffer

* StringBuffer类用于表示可以修改的字符串；
* 使用+运算符的字符串将自动创建字符串缓冲对象；
* 在Java中一旦创建了字符串就不能直接更改，这就是字符串String的不变性；而StringBuffer类正是针对此问题而提供的字符可变序列；StringBuffer与String是同等的类，唯一区别是可以进行更改。
* 以下是StringBuffer类的构造方法有4种重载方式，以下是常用的几个：

### 常用构造

|  |
| --- |
|  |

### 常用方法

|  |
| --- |
|  |

### 案例

|  |
| --- |
| **public** **class** StringBufferDemo {  **public** **static** **void** main(String[] args) {  StringBuffer strBuf = **new** StringBuffer("Java");  strBuf.append(" Guide Ver1/"); //连接  System.***out***.println(strBuf);  strBuf.append(3);  System.***out***.println(strBuf);  strBuf.insert(5, "Student"); //插入  System.***out***.println(strBuf);  strBuf.setCharAt(20, '.'); //替换字符  System.***out***.println(strBuf);  strBuf.reverse(); //倒序  System.***out***.println(strBuf);  String str = strBuf.toString();  System.***out***.println(str);  }  } |

## StringBuilder

* JDK1.5以后，提供了一个新的字符缓冲区：StringBuilder。
* 提供和StringBuffer相似的API ，非线程安全

## 区别

|  |
| --- |
|  |

## 作业

1. 用户从控制台输入一个字符串，要求判断是否为合法的email地址，如果合法，请判断是否为sina的邮箱地址。(PS:记住异常处理) 合法邮箱规则:
   * 有且仅有一个@和.
   * @在.之前 且@不能是第一位 .不能是最后一位
   * @与.不能相邻
   * 新浪邮箱应当以@sina.com结尾
2. 用户从控制台输入一个字符串，请判断是否为回文字符串。
3. 编写自定义的字符串一致性匹配方法，只要两个字符串包含同样的字符，不管字符的顺序如何，都认为两个字符串一致，如：”aabbcc”和”abcabc”被认为是一致的
4. 从控制台实现输入字符串（路径），分别把盘符，文件路径（不包含盘符），文件名，文件类型打出来如：E:\jboss-4.0.4.GA\server\minimal\conf\readme.txt 解析路径，输入 盘符，文件路径，文件名，文件类型。
   * 盘符：E:
   * 文件路径：jboss-4.0.4.GA\server\minimal\conf\
   * 文件名：readme
   * 文件类型：txt
5. 把数组中的数据按照指定格式拼接成一个字符串。
   * 举例：int[] arr = {1,2,3}; 输出结果：[1, 2, 3]
6. 字符串反转。 举例：键盘录入”abc” 输出结果：”cba”
7. 去掉一个字符串中的所有空格(包括中间的空格) 使用正则
8. 获取一个字符串中指定子串出现的次数。比如说“hanbasdnbafllgnbahjnbakqqqqlnbaxi” 在这个字符串中，有几个nba？
9. 在控制台输入一个文件路径（例如：C:\hello\image\hello.jpg），然后获得文件名（hello）和扩展名(jpg) 注意双斜杠表示一个斜杠

1. 练习：让用户在控制台不断录入英文字母（每次录入两个字母即可），直到总长度超过6个停止，然后统计一下一共有几个a
2. 让用户在控制台输入几个文件名，中间用“,”隔开，然后只把jpg文件换行输出来。例如：hello.java,a.txt,x1.jpg,t1.ppt,m.jpg
3. 现有一个字符串“我我....我...我.要...要要...要学....学学..学.编..编编.编.程.程.程..程”。将该字符串变成:“我要学编程”。使用正则