String类：本身就是一个类（重点）

String 实例化的方法有两种：

使用直接赋值的方式完成

通过new关键字进行实例化

如果使用直接赋值的方式，那么系统会先去堆内存查找是否有相同的内容空间，如果有，则直接指向此空间，如果没有，则开辟一个新的空间并且赋值

我们可以看到，使用“==”符号对字符串进行比较，比较的其实是地址

比较字符串的内容，使用的equals方法

public Boolean equals(Object anObject):比较字符串的内容

|  |
| --- |
| **package** com.wanczy.StringDemo;  **public** **class** StringDemo01 {  **public** **static** **void** main(String[] args) {  String str1 ="123";  String str2 ="123";  String str3 =str2;  System.*out*.println(str1==str2);  System.*out*.println(str1==str3);  System.*out*.println(str2==str3);    }      } |

String  提供一个length()方法来获得字符串的长度

事实上，在java里面用””引起来的就是一个字符串的匿名对象

|  |
| --- |
| **package** com.wanczy.StringDemo;  **public** **class** StringDemo02 {  **public** **static** **void** main(String[] args) {  String str1="Hello";  System.*out*.println("Hello".equals(str1));  }  } |

String 是一种字符串常量：

一旦声明了，他就无法改变

StringBuffer这就是一种动态的可以修改的字符串类：

|  |
| --- |
| **package** com.wanczy.StringDemo;  **public** **class** StringDemo02 {  **public** **static** **void** main(String[] args) {  StringBuffer str1=**new** StringBuffer("123");  StringBuffer str2=str1;  System.*out*.println(str2==str1);  str1=str1.append("!");  System.*out*.println(str2==str1);  //    }  } |

一般只有一种情况我们使用StringBuffer而不使用String

如果程序中需要有字符串的多次叠加，我们使用StringBuffer

在我们堆内存里，如果存在没有变量名指向的值，系统是不会马上进行回收，只有当你的代码块执行完了之后，才会去回收。所以说，多次叠加的时候如果使用String，将会消耗大量系统资源。

字符串大写变小写的方法: public [String](mk:@MSITStore:E:\DH02\课堂笔记\JAVASE\JDK_API_1_6_zh_CN.CHM::/java/lang/String.html) **toLowerCase**()

字符串小写变大写的方法：public [String](mk:@MSITStore:E:\DH02\课堂笔记\JAVASE\JDK_API_1_6_zh_CN.CHM::/java/lang/String.html) **toUpperCase**()

字符串转换成char[]:public char[] **toCharArray**()

把字符数组转变成字符串的方法

public **String**(char[] value,int offset, int count)

public String(char[])

|  |
| --- |
| **package** com.wanczy.StringDemo;  **public** **class** StringDemo02 {  **public** **static** **void** main(String[] args) {  **char** c[]={'H','E','L','L','O'};  // String str1= new String(c);  String str1=**new** String(c,1,4);    }  } |

字符串可以使用+或者使用concat()方法

根据字符串的索引找到指定字符的方法public char **charAt**(int index)

字符串是否以指定的字符开头或者结尾

开始：public boolean **startsWith**([String](mk:@MSITStore:E:\DH02\课堂笔记\JAVASE\JDK_API_1_6_zh_CN.CHM::/java/lang/String.html) prefix)

结束：public boolean **endsWith**([String](mk:@MSITStore:E:\DH02\课堂笔记\JAVASE\JDK_API_1_6_zh_CN.CHM::/java/lang/String.html) prefix)

字符的替换：replace

public [String](mk:@MSITStore:E:\DH02\课堂笔记\JAVASE\JDK_API_1_6_zh_CN.CHM::/java/lang/String.html) **replace**(char oldChar, char newChar)

字符串的替换：replaceAll

public [String](mk:@MSITStore:E:\DH02\课堂笔记\JAVASE\JDK_API_1_6_zh_CN.CHM::/java/lang/String.html) **replaceAll**([String](mk:@MSITStore:E:\DH02\课堂笔记\JAVASE\JDK_API_1_6_zh_CN.CHM::/java/lang/String.html) regex,[String](mk:@MSITStore:E:\DH02\课堂笔记\JAVASE\JDK_API_1_6_zh_CN.CHM::/java/lang/String.html) replacement)

字符串的截取:

全部截取: public [String](mk:@MSITStore:E:\DH02\课堂笔记\JAVASE\JDK_API_1_6_zh_CN.CHM::/java/lang/String.html) **substring**(int beginIndex)

部分截取：public [String](mk:@MSITStore:E:\DH02\课堂笔记\JAVASE\JDK_API_1_6_zh_CN.CHM::/java/lang/String.html) **substring**(int beginIndex,int endIndex)

字符串的拆分：public [String](mk:@MSITStore:E:\DH02\课堂笔记\JAVASE\JDK_API_1_6_zh_CN.CHM::/java/lang/String.html)[] **split**([String](mk:@MSITStore:E:\DH02\课堂笔记\JAVASE\JDK_API_1_6_zh_CN.CHM::/java/lang/String.html) regex)

字符串的查找：public int **indexOf**([String](mk:@MSITStore:E:\DH02\课堂笔记\JAVASE\JDK_API_1_6_zh_CN.CHM::/java/lang/String.html) str)

如果查找到了，返回str在目标字符串中首字母的下标

用法最多的是:if(str.indexOf(str1)!=-1){

}

字符串还提供了一个去空格的方法：

trim();

|  |
| --- |
| **package** com.wanczy.StringDemo;  **public** **class** StringDemo02 {  **public** **static** **void** main(String[] args) {  String str=" 123 321 ";  str=str.trim();  System.*out*.println(str);  }  }  输出的是“123 321”  所以，只是去掉左右两边的空格 |

总结：

·String类对象的实例化

·要去理解String类的一些构造方法（byte数组转换成指定编码的字符串，char数组转换成字符串）

·将StringBuffer对象转换成String类对象（toString()方法，还有就是new String（StringBuffer sb））

·学会使用String类中的方法。

练习：第一个去预习正则表达式，Java中其他常用的API（包装类、Math、Date、Calendar、DateFormat），集合。