# Java中的String类

String类用于描述字符串事物，它提供了多个方法对字符串进行操作。

1. 字符串的声明

String A = "koter"; //A在内存中有一个对象

String B = **new** String ("koter"); //B在内存中有两个对象，字符串的不变型。一旦在内存中声明，改变即创建新的对象。

String C = "koter";

**boolean** a = A==B; //比较内存地址 false

**boolean** c = A==C; //true ，比较的是一个对象。

**boolean** b = A.equals(B); //String类重写了equals方法，比较内容

//判断一个字符串是否等于某字符串。""也是一个对象。A如果为null则会报空指针异常

**boolean** d = "".equals(A);

1. String类中的常用方法

1． 获取。

1.1 int length():获取长度。

1.2 根据位置获取位置上某个字符。

char charAt(int index):

1.3 根据字符获取该字符在字符串中位置。

int indexOf(int ch):返回的是ch在字符串中第一次出现的位置。

int indexOf(int ch, int fromIndex) :从fromIndex指定位置开始，获取ch在字符串中出现的位置。

int indexOf(String str):返回的是str在字符串中第一次出现的位置。

int indexOf(String str, int fromIndex) :从fromIndex指定位置开始，获取str在字符串中出现的位置。

int lastIndexOf(int ch) :返回的是ch在字符串中倒序第一次出现的位置

2. 判断。

2.1 字符串中是否包含某一个子串。

boolean contains(str):

特殊之处：indexOf(str):可以索引str第一次出现位置，如果返回-1.表示该str不在字符串中存在。

所以，也可以用于对指定判断是否包含。

if(str.indexOf("aa")!=-1)

而且该方法即可以判断，有可以获取出现的位置。

2.2 字符中是否有内容。

boolean isEmpty(): 原理就是判断长度是否为0.

2.3 字符串是否是以指定内容开头。

boolean startsWith(str);

2.4 字符串是否是以指定内容结尾。

boolean endsWith(str);

2.5 判断字符串内容是否相同。复写了Object类中的equals方法。

boolean equals(str);

2.6 判断内容是否相同，并忽略大小写。

boolean equalsIgnoreCase();

3. 转换。

3.1 将字符数组转成字符串。

构造函数:

String(char[])

String(char[],offset,count):将字符数组中的一部分转成字符串。

静态方法:

static String copyValueOf(char[]);

static String copyValueOf(char[] data, int offset, int count)

static String valueOf(char[]):

3.2 将字符串转成字符数组。\*\*

char[] toCharArray():

3.3 将字节数组转成字符串。

String(byte[])

String(byte[],offset,count):将字节数组中的一部分转成字符串。

3.4 将字符串转成字节数组。

byte[] getBytes():

3.5 将基本数据类型转成字符串。

static String valueOf(int)

static String valueOf(double)

//3+"";直接转换//String.valueOf(3);

特殊：字符串和字节数组在转换过程中，是可以指定编码表的。

4. 替换

String replace(oldchar,newchar);

5. 切割

String[] split(regex);

6. 子串。获取字符串中的一部分。

* String substring(begin);
* String substring(begin,end);

7. 转换，去除空格，比较。

7.1 将字符串转成大写或则小写。

String toUpperCase();

String toLowerCase();

7.2 将字符串两端的多个空格去除。

String trim();

7.3 对两个字符串进行自然顺序的比较。

int compareTo(string);

# StringBuffer是字符串缓冲区。

1，存储。

StringBuffer append():将指定数据作为参数添加到已有数据结尾处。

StringBuffer insert(index,数据):可以将数据插入到指定index位置。

2，删除。

StringBuffer delete(start,end):删除缓冲区中的数据，包含start，不包含end。

StringBuffer deleteCharAt(index):删除指定位置的字符。

3，获取。

char charAt(int index)

int indexOf(String str)

int lastIndexOf(String str)

int length()

String substring(int start, int end)

4，修改。

StringBuffer replace(start,end,string);

void setCharAt(int index, char ch) ;

5，反转。

StringBuffer reverse();

6，

将缓冲区中指定数据存储到指定字符数组中。

void getChars(int srcBegin, int srcEnd, char[] dst, int dstBegin)

# JDK1.5 版本之后出现了StringBuilder.

StringBuffer是线程同步。

StringBuilder是线程不同步。

以后开发，建议使用StringBuilder

升级三个因素：

1，提高效率。

2，简化书写。

3，提高安全性。

# 基本数据类型对象包装类。

byte Byte

short short

int Integer

long Long

boolean Boolean

float Float

double Double

char Character

Character.isLetter(char c) //判断c是否是一个字母

基本数据类型和字符串类型之间做转换

基本数据类型转成字符串。

1. 基本数据类型+""
2. 基本数据类型.toString(34);

字符串转成基本数据类型。

int a = Integer.parseInt("123");

double b = Double.parseDouble("12.23");

boolean b = Boolean.parseBoolean("true");

Integer i = new Integer("123");

int num = i.intValue();

十进制转成其他进制。

toBinaryString();

toHexString();

toOctalString();

其他进制转成十进制。

parseInt(string,radix);

自动拆箱和自动装箱

Integer x = 4;//自动装箱。//new Integer(4)

x = x+ 2;//x+2:x 进行自动拆箱。变成成了int类型。和2进行加法运算。

//再将和进行装箱赋给x。

# 正则表达式:符合一定规则的表达式。

作用：用于专门操作字符串。

特点：用于一些特定的符号来表示一些代码操作。这样就简化书写。

所以学习正则表达式，就是在学习一些特殊符号的使用。

好处：可以简化对字符串的复杂操作。

弊端：符号定义越多，正则越长，阅读性越差。

具体操作功能：

1，匹配：String matches方法。用规则匹配整个字符串，只要有一处不符合规则，就匹配结束，返回false。

2，切割：String split();

3，替换：String replaceAll(regex,str);如果regex中有定义组，可以在第二参数中通过$符号获取正则表达式中的已有的组。