**JAVA基础**

# Math

|  |
| --- |
| **public** **class** MathDemo {  **public** **static** **void** main(String[] args) {  System.***out***.println(Math.***PI***);    // 绝对值  System.***out***.println(Math.*abs*(-5));  System.***out***.println(Math.*abs*(5));  // 取下整  System.***out***.println(Math.*floor*(5.5));// 5.0  System.***out***.println(Math.*floor*(5.6));// 5.0  // 取上整  System.***out***.println(Math.*ceil*(5.5));// 6.0  System.***out***.println(Math.*ceil*(5.4));// 6.0  // 先除再取整  System.***out***.println(Math.*floorDiv*(6, 4));// 1  // 先取余在取整  System.***out***.println(Math.*floorMod*(6, 4));// 2    System.***out***.println(Math.*max*(5, 10));// 10  System.***out***.println(Math.*min*(5, 10));// 5    // 返回乘积，如果超过int类型的数据返回，会抛出异常 ArithmeticException（算术异常）  // System.out.println(Math.multiplyExact(611111111, 111115));    // 浮点数精度问题，有可能会在计算的结果后面多出多位小数,遇到这样的问题，  // 要给数据保留2位小数....(计算金额时，这种问题很常见)  System.***out***.println(3.1\*3.0);// 9.300000033    // 比他小的最接近他的浮点数  System.***out***.println(Math.*nextDown*(-199));  System.***out***.println(Math.*nextDown*(199));    // 平方根  System.***out***.println(Math.*pow*(3, 2));// 9.0  System.***out***.println(Math.*pow*(3, 3));// 27.0    // 随机数[0.0,1.0), 0.8978961066466424  System.***out***.println(Math.*random*());    // 开方根  System.***out***.println(Math.*sqrt*(9));// 3.0  // 四舍五入  System.***out***.println(Math.*round*(5.5));// 6 long  System.***out***.println(Math.*round*(5.4));// 5 long  System.***out***.println(Math.*round*(5.8F));// 6 int    // 弧度角度计算  // Math.toDegrees(angrad)  // Math.toRadians(angdeg)  }  } |

# String

|  |
| --- |
| **public** **class** StringDemo {  **static** String *str* ="";  **public** **static** **void** main(String[] args) {  System.***out***.println(*str*.isEmpty());// 空串""    String strDemo1 = **new** String("海内存知己,存,天涯若比邻");  **char**[] chArr = {'a','b','c'};  String strDemo2 = **new** String(chArr);  System.***out***.println(chArr);  // 0代表其实位置，2代表取字符数组中字符的个数  String strDemo3 = **new** String(chArr, 0, 2);  System.***out***.println(strDemo3);  // 将字符串转换成字符数组  **char**[] chs = strDemo2.toCharArray();  System.***out***.println(chs[0]);  // 判断两个字符串是否相等，相等返回true  System.***out***.println(strDemo1.equals(strDemo2));// false    // 字符串从下标为2的位置开始，截取当前字符串中的数据，得到子串  System.***out***.println(strDemo1.substring(2));//存知己天涯若比邻  // 截取当前字符串中下标[2,4)  System.***out***.println(strDemo1.substring(2, 4));// 存知  // 字符在字符串中第一次出现的下标  System.***out***.println(strDemo1.indexOf(78));// -1  System.***out***.println(strDemo1.indexOf('a'));// -1    // 字符串第一次出现的位置  System.***out***.println(strDemo1.indexOf("存"));// 2  // 从指定位置开始查找字符串，若发现时，则从头开始计算下标  System.***out***.println(strDemo1.indexOf("存", 1));// 2    System.***out***.println(strDemo1.lastIndexOf('b'));// -1  // 指定字符串在源字符串中最后一次出现的位置  System.***out***.println(strDemo1.lastIndexOf("存,"));// 6  // 查找字符串在源字符串中最后一次出现的位置,源字符串的长度为参数中指定的长度  // 如本题中8表示源字符串为"海内存知己,存,天"  System.***out***.println(strDemo1.lastIndexOf("存,", 8));// 6    // 取下标为指定参数的内容  System.***out***.println(strDemo1.charAt(1));// 内  // 字符串长度  System.***out***.println(strDemo1.length());    // 先以最小长度的那个字符串进行比较，如果他们从前向后，根据最小串长度进行比较  // 相等，则在直接进行字符串的减法比较  System.***out***.println(strDemo1.compareTo("海内存知己,存,天涯若比邻"));// 0  System.***out***.println("abgd".compareTo("abd"));// 3  System.***out***.println("abcd".compareTo("abc"));// 1  // 忽略大小写进行比较  System.***out***.println("abcd".compareToIgnoreCase("ABC"));// 1  // 以什么作为结尾  System.***out***.println(strDemo1.endsWith("比邻"));// true  System.***out***.println(strDemo1.endsWith("天涯"));// false  // 以什么开头  System.***out***.println(strDemo1.startsWith("比邻"));// false    // 从哪里开始查找是否以给定的字符串开头  System.***out***.println(strDemo1.startsWith("存,", 3));// false  System.***out***.println(strDemo1.startsWith("存,", 6));// true    // 海内存知己,存,天涯若比邻字符串  System.***out***.println(strDemo1.concat("字符串"));  // 是否包含指定的字符串  System.***out***.println(strDemo1.contains("存,"));// true  // 字符替换  System.***out***.println("abcdcf".replace('c', 'd'));// abdddf  // 字符串替换  System.***out***.println("abcdcdf".replace("cd", "c"));// abccf  // 根据正则表达式匹配，替换为后面的内容  System.***out***.println("abcdc123df".replaceAll("[a-z]", "1"));// 1111112311    // 大小写转换  System.***out***.println("ABC".toLowerCase());  System.***out***.println("abc".toUpperCase());  // 转换成字节数组  System.***out***.println("abc".getBytes()[1]);      **try** {  // 按照一定的编码方式，返回一个字节数组  System.***out***.println("abc".getBytes("UTF-8")[0]);  } **catch** (UnsupportedEncodingException e) {  // **TODO** Auto-generated catch block  e.printStackTrace();  }    // 按照指定字符进行分割  System.***out***.println("abc,def,ghi".split(",")[2]);  // 按照指定字符进行分割，但是分割字符串的个数，即数组的长度最大为指定的数2  System.***out***.println("abc,def,ghi".split(",",2));// abc def,ghi    String[] str = "abc,def,ghi".split(",",3);// abc def ghi          // 去掉收尾空格  System.***out***.println(" 海内存知己 ".trim());  System.***out***.println(" 海内存知己 ");      // 按照指定的分隔符，将字符串连接起来  System.***out***.println(String.*join*(",", "abc","def","hij"));    // 会保留5个字符  System.***out***.println(String.*format*("%.5s", "123.3432423"));    // 保留2位小数，返回一个string类型的数据  System.***out***.println(String.*format*("%.2f", 123.34323));  // 计算时需要再将字符串转换回来  System.***out***.println(Double.*parseDouble*(String.*format*("%.2f", 123.34323)));    }  } |

# StringBuffer

1. 在执行速度方面的比较：StringBuilder > StringBuffer   
2. StringBuffer与StringBuilder，他们是字符串变量，是可改变的对象，每当我们用它们对字符串做操作时，实际上是在一个对象上操作的，不像String一样创建一些对象进行操作，所以速度就快了。  
3. StringBuilder：线程非安全的  
　 StringBuffer：[线程安全](https://www.baidu.com/s?wd=%E7%BA%BF%E7%A8%8B%E5%AE%89%E5%85%A8&tn=44039180_cpr&fenlei=mv6quAkxTZn0IZRqIHckPjm4nH00T1YLrHuBrHbkuHIBmWR4nHDL0ZwV5Hcvrjm3rH6sPfKWUMw85HfYnjn4nH6sgvPsT6KdThsqpZwYTjCEQLGCpyw9Uz4Bmy-bIi4WUvYETgN-TLwGUv3EPjRvnHm3PH04)的

|  |
| --- |
| **public** **class** StringBufferDemo {  **public** **static** **void** main(String[] args) {  // final修饰的类，不可变长度的字符串  String x = "abc";  System.***out***.println(x.length());  x = "defgh";// 长度变化的假象  System.***out***.println(x.length());  // 可变长度的字符串  StringBuffer strBuff = **new** StringBuffer();  strBuff.append("a");// 追加内容  strBuff.append(17.898);  // 从StringBuffer对象到String对象,方法返回的字符串为String类型，不可变  // 并没有改变StringBuffer  System.***out***.println(strBuff.toString());  strBuff.append('c');  // 字符串长度  System.***out***.println(strBuff.length());    // 返回当前的容量，默认初始容量为16  System.***out***.println(strBuff.capacity());    strBuff.append("dsfadfdsafsdaf");  // 当容量不足时，strBuff会自动扩展容量，18的倍数  System.***out***.println(strBuff.capacity());    // 下标为3位置的字符  System.***out***.println(strBuff.charAt(3));  // StringBuffer equals方法比较的是地址，与==一样，若要比较值  // 需要使用toString方法转换成字符串之后再比较  System.***out***.println(strBuff.equals("123"));  // 指定字符串第一次出现的位置  System.***out***.println(strBuff.indexOf("89"));// 4  // 指定字符串最后一次出现的位置  System.***out***.println(strBuff.lastIndexOf("8"));// 4    // 从1开始到结束，其他截取字符串放同String  System.***out***.println(strBuff.substring(1));// 17.898cdsfadfdsafsdaf    // 从其实位置开始删除字符串到指定的结束位置，会修改当前的对象  System.***out***.println(strBuff.delete(9, strBuff.length()));    // 删除指定位置的字符，会修改当前的对象  System.***out***.println(strBuff.deleteCharAt(0));  System.***out***.println(strBuff);  // 从下表为1的位置开始插入字符串hello  System.***out***.println(strBuff.insert(1, "hello"));  // 从下标为3的位置开始插入字符串hello  System.***out***.println(strBuff.insert(3, **true**));    // replace setCharAt reverse  // 替换  System.***out***.println(strBuff.replace(3, 7, "false"));  // 设置某个位置的字符  strBuff.setCharAt(1, 'a');  System.***out***.println(strBuff);  // 字符串反转，也就是倒序输出  System.***out***.println(strBuff.reverse());    strBuff.setLength(10);    System.***out***.println(strBuff);    String str = "上海 苏州 南京 无锡";    // 了解  StringTokenizer st = **new** StringTokenizer(str);  **while**(st.hasMoreTokens()) {  System.***out***.println(st.nextToken());  }    }  } |

# Date Calendar

注意util下的Date包

|  |
| --- |
| **import** java.text.SimpleDateFormat;  **import** java.util.Calendar;  **import** java.util.Date;  **public** **class** DateDemo {  **public** **static** **void** main(String[] args) {  Date date1 = **new** Date();    System.***out***.println(date1.toString());  // 从1970 1.1 00:00:00到系统当前日期的毫秒数  System.***out***.println(date1.getTime());    Date date2 = **new** Date();  // 日期比较 date1 < date2 -1  // date2.setTime(1490169971198L);  // 相同为 0  // date2.setTime(date1.getTime());  // date1 > date2 1  date2.setTime(1490169980239L);  System.***out***.println(date1.compareTo(date2));    // date1在date2之后为true，时间越大月靠后  System.***out***.println(date1.after(date2));// true  // 带有横杠的房均为弃用的方法，不在使用，可以通过查看方api找到  // 替代方案  date1.~~getDate~~();    Calendar calendar = Calendar.*getInstance*();  calendar.setTime(date1);  // 当前日期所对应的年  System.***out***.println(calendar.get(Calendar.***YEAR***));  // 当前日期所对应的月, 值是从0开始，需要+1，才能得到当前的月份  System.***out***.println(calendar.get(Calendar.***MONTH***));  // 当前日期对应的日  System.***out***.println(calendar.get(Calendar.***DATE***));  // 代表星期几，从sunday开始计算，如星期三是4  System.***out***.println(calendar.get(Calendar.***DAY\_OF\_WEEK***));    // 方法很多，自己多练习    // 12小时制显示  System.***out***.println(calendar.get(Calendar.***HOUR***));  // 24小时制显示  System.***out***.println(calendar.get(Calendar.***HOUR\_OF\_DAY***));  // 在已有时间的基础上再加一小时  calendar.add(Calendar.***HOUR***, 1);  System.***out***.println(calendar.get(Calendar.***HOUR***));  System.***out***.println(calendar.get(Calendar.***HOUR\_OF\_DAY***));    // 在已有时间的基础上再加一年，负数就是减  calendar.add(Calendar.***YEAR***, 1);  System.***out***.println(calendar.get(Calendar.***YEAR***));    // 将当前年设置为2016年  calendar.set(Calendar.***YEAR***, 2016);  System.***out***.println(calendar.get(Calendar.***YEAR***));      // 日期格式转化  System.***out***.println(date1);  // 2017/3/22 16:22  // y 年 M 月 d 这一月的第几天  // h 12小时制 H 24小时制  // m 分钟 s表示秒  SimpleDateFormat sf = **new** SimpleDateFormat("yyyy-MM-dd HH:mm:ss");  System.***out***.println(sf.format(date1));    }  } |