stringBuffer

2018年7月1日 14:58

线程安全的可变字符串，类似于string的字符串缓冲区，但不能修改

stringBuffer和string的区别?
前者长度和内容可变，后者不可变
如果使用前者作字符串的拼接，不会浪费太多资源

```
package cn.itcast_01;

/*
 *	线程安全（多线程）
 *	安全--同步--数据是安全的
 *	不安全--不同步--效率高一些
 *	安全和效率问题
 *
 *	StringBuffer的构造方法:
 *		public StringBuffer():无参构造
 *		public StringBuffer(int capacity):指定容量的字符串缓冲对象
 *		public StringBuffer(String str):指定内容的字符串缓冲对象
 *
 *	StringBuffer的方法:
 *		public int capacity()：返回当前容量/理论值
 *		public int length()：返回长度/实际值
 */
public class StringBufferDemo {
	public static void main(String[] args) {
		// public StringBuffer() 较为常用
		StringBuffer sb = new StringBuffer();
		System.out.println("sb=" + sb);
		System.out.println(sb.capacity());// 默认为16个字符
		System.out.println(sb.length());
		System.out.println("----------------");

		// public StringBuffer(int capacity)
		StringBuffer sb2 = new StringBuffer(50);
		System.out.println("sb=" + sb2);
		System.out.println(sb2.capacity());
```

```
        System.out.println(sb2.length());
        System.out.println("----------------");

        // public StringBuffer(String str)
        StringBuffer sb3 = new StringBuffer("hello");
        System.out.println("sb=" + sb3);
        System.out.println(sb3.capacity());
        System.out.println(sb3.length());
        System.out.println("----------------");
    }
}
```

添加功能

2018年7月10日 21:48

```
/*
 * StringBuffer的添加功能
 *		public StringBuffer append(String str)
 *		可以把任意类型添加到字符串缓冲区中,并返回字符串缓冲区本身
 *		public StringBuffer insert(int offset, String str)
 *		在指定位置插入任意类型的数据
 */
public class StringBufferDemo {
	public static void main(String[] args) {
		StringBuffer sb = new StringBuffer();
		/*
		 * StringBuffer sb2=sb.append("hello"); System.out.println("sb="+sb);
		 * System.out.println("sb2="+sb2); System.out.println(sb==sb2);//true
		 */

		sb.append("hello");
		sb.append("sss");
		sb.append(3.55);
		// 链式编程
		sb.append("dd").append(3).append(444);
		System.out.println("sb=" + sb);

		sb.insert(5, "world");
		System.out.println("sb="+sb);
	}
}
```

删除功能

2018年7月10日 22:10

```
package cn.itcast_03;

/*
 * StringBuffer的删除功能
 *                  public StringBuffer deleteCharAt(int index)
 *                  public StringBuffer delete(int start,int end)
 */
public class StringBufferDemo {
        public static void main(String[] args) {
                StringBuffer sb = new StringBuffer();
                sb.append("hello").append("world").append("java");
                System.out.println("sb=" + sb);

                sb.deleteCharAt(1);
                System.out.println("sb=" + sb);

                // 删除world
                sb.delete(5, 10);
                System.out.println("sb=" + sb);// 包左不包右
                // 清空字符串缓冲区
                sb.delete(0, sb.length());
                System.out.println("sb=" + sb);
        }
}
```

替换功能

2018年7月10日 22:30

```
package cn.itcast_04;
/*
 * StringBuffer的替换功能
 * public StringBuffer replace(int start,int end,String str)
 */
public class StringBufferDemo {
	public static void main(String[] args) {
		StringBuffer sb =new StringBuffer();
		sb.append("hello").append("world").append("java");

		sb.replace(5, 10, "happy");
		System.out.println("sb="+sb);
	}
}
```

反转功能

2018年7月10日 22:42

```
//反转功能
public class StringBufferDemo {
	public static void main(String[] args) {
		StringBuffer sb = new StringBuffer();
		sb.append("王若潇");

		sb.reverse();
		System.out.println(sb);
	}
}
```

截取功能

2018年7月11日 0:26

```
package cn.itcast_05;
/*
 *      截取功能;                                                    注意返回值类型为string
 *              public String substring(int start)
 *              public String substring(int start,int end)
 */
public class StringBufferDemo2 {
        public static void main(String[] args) {
                StringBuffer sb = new StringBuffer();
                sb.append("hello").append("world").append("java");

                String s = sb.substring(5);
                System.out.println("s="+s);
                System.out.println("sb="+sb);
        }
}
```

String和StringBuffer转换

2018年7月11日 0:26

```
package cn.itcast_07;

//String和StringBuffer的相互转换
public class StringBufferTest {
	public static void main(String[] args) {
		String s = "hello";
		// 注意不能把字符串的值直接赋值给StringBuffer
		// StringBuffer sb="hello";
		// StringBuffer sb = s;

		// 转换方式1：构造方法
		StringBuffer sb = new StringBuffer(s);
		// 转换方式2：append
		StringBuffer sb2 = new StringBuffer();
		sb2.append(s);

		StringBuffer buffer = new StringBuffer("hello");
		// 方式1：构造方法
		String s1 = new String(buffer);
		// 方式2：toString()方法
		String s2 = buffer.toString();
	}
}
```

练习

2018年7月11日 0:38

把数组拼接成字符串

```
public class StringBufferTest2 {
	public static void main(String[] args) {
		int[] arr = { 11, 22, 3, 4, 45, 44 };
		String s = arrayToString(arr);
		System.out.println(s);
	}

	public static String arrayToString(int[] arr) {
		StringBuffer sb = new StringBuffer();
		sb.append("[");
		for (int x = 0; x < arr.length; x++) {
			if (x == arr.length - 1) {
				sb.append(arr[x]);
			} else {
				sb.append(arr[x]).append(", ");
			}
		}
		sb.append("]");
		String s = new String(sb);
		return s;
	}
}
```

字符串反转

```
public class StringBufferTest3 {
	public static void main(String[] args) {
		Scanner sc = new Scanner(System.in);
		System.out.println("请输入字符串");
		String s=sc.nextLine();
		String s1=myreverse(s);
		System.out.println(s1);
	}

	public static String myreverse(String s) {
		return new StringBuffer(s).reverse().toString();
	}
}
```

```
import java.util.Scanner;

//判断一个字符串是否是对称的
public class StringBufferTest4 {
	public static void main(String[] args) {
		Scanner sc = new Scanner(System.in);
		System.out.println("请输入一个字符
		串");
		String s = sc.nextLine();
		System.out.println(isSame(s));
	}

	public static boolean isSame(String s) {
		StringBuffer sb = new StringBuffer(s);
		return sb.reverse().toString().equals(s);
	}
}
```

面试题

2018年7月11日 11:45

string StringBuffer StringBuilder的区别
1.String是内容不可变的，而StringBuffer，StringBuilder都是内容可变的
2.StringBuffer是同步的，数据安全；StringBuilder是不同步的，数据不安全，效率高

2.StringBuffer和数组的区别?
二者都可以看作一个容器，装其它数据
但StringBuffer的数据最终是一个字符串数据
而数组可以放置多种数据，但必须是同一数据类型的

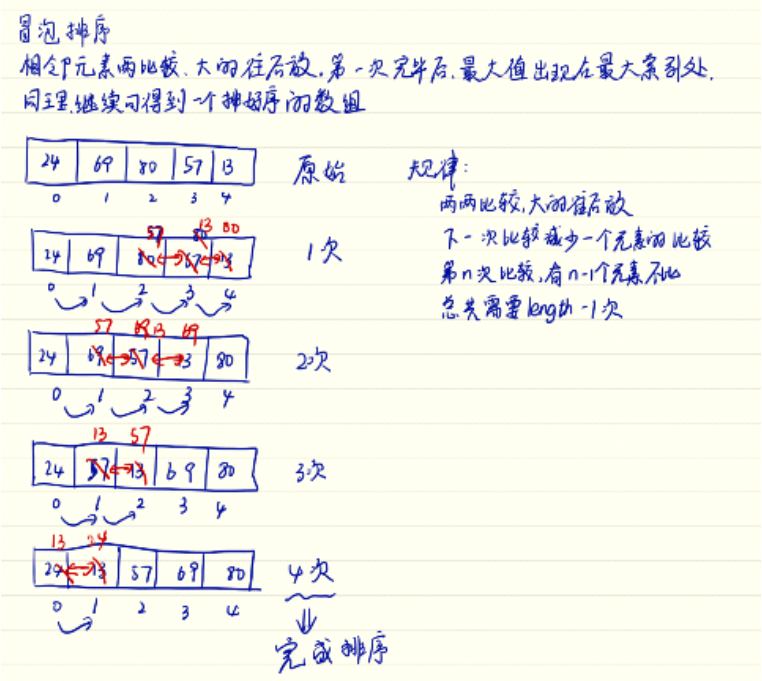
3.形式参数问题
String 作为参数传递，和基本类型作为参数传递效果相同
StringBuffer作为参数传递

```
package cn.itcast_08;

public class StringBufferDemo {
	public static void main(String[] args) {
		String s1 = "hello";
		String s2 = "world";
		System.out.println(s1 + "---" + s2);//hello---world
		change(s1, s2);
		//把String当作基本类型，因为它的参数传递特殊
		System.out.println(s1 + "---" + s2);//hello---world
		StringBuffer sb1 = new StringBuffer("hello");
		StringBuffer sb2 = new StringBuffer("world");
		System.out.println(sb1 + "---" + sb2);//hello---world
		change(sb1, sb2);
		System.out.println(sb1 + "---" + sb2);//hello---worldworld
	}

	public static void change(StringBuffer sb1, StringBuffer sb2) {
		sb1 = sb2;
		sb2.append(sb2);
	}

	public static void change(String s1, String s2) {
```

```
            s1 = s2;
            s2 = s1 + s2;
        }
    }
```

冒泡排序

2018年7月11日 12:11




```
public class ArrayDemo {
	public static void main(String[] args) {
		int[] arr = { 24, 69, 80, 57, 13 };
		printArray(arr);
		bubbleSort(arr);
		printArray(arr);

	}

	// 遍历数组
	public static void printArray(int[] arr) {
		System.out.print("[");
		for (int x = 0; x < arr.length; x++) {
			if (x == arr.length - 1) {
				System.out.println(arr[x] + "]");
			} else {
				System.out.print(arr[x] + ", ");
			}
		}
	}

	// 冒泡排序
	public static void bubbleSort(int[] arr) {
		for (int x = 0; x < arr.length - 1; x++) {
			for (int y = 0; y < arr.length - 1 - x; y++) {
				if (arr[y] > arr[y + 1]) {
					int temp = arr[y];
					arr[y] = arr[y + 1];
					arr[y + 1] = temp;
				}
			}
		}
	}
}
```

选择排序

2018年7月11日 15:45

从0索引开始，依次和后面元素比较，小的往前放，第一次完毕，最小值出现在最小索引处其它同理可得到排布后数组

规律：
1.第一次从0索引开始和其它比较
2.最后一次是数组长度-2和数组长度-1的元素比较

```
public class ArrayDemo2 {
	public static void main(String[] args) {
		int[] arr = { 24, 69, 80, 57, 13 };
		printArray(arr);
		selectSort(arr);
		printArray(arr);
	}

	// 遍历数组
	public static void printArray(int[] arr) {
		System.out.print("[");
		for (int x = 0; x < arr.length; x++) {
			if (x == arr.length - 1) {
				System.out.println(arr[x] + "]");
			} else {
				System.out.print(arr[x] + ", ");
			}
		}
	}

	public static void selectSort(int arr[]) {
		for (int x = 0; x < arr.length - 1; x++) {
			for (int y = x + 1; y < arr.length; y++) {
				if (arr[y] < arr[x]) {
					int temp = arr[y];
					arr[y] = arr[x];
					arr[x] = temp;
				}
			}
		}
	}
}
```

字符串排序

2018年7月11日 16:08

```
//把字符串中的字符进行排序
public class ArrayDemo3 {
	public static void main(String[] args) {
		String s = "dacgebf";
		char[] chs = s.toCharArray();
		bubbleSort(chs);
		String result =s.valueOf(chs);
		System.out.println(result);
	}

	public static void bubbleSort(char[] chs) {
		for (int x = 0; x < chs.length - 1; x++) {
			for (int y = 0; y < chs.length - 1 - x; y++) {
				if (chs[y] > chs[y + 1]) {
					char temp = chs[y];
					chs[y] = chs[y + 1];
					chs[y + 1] = temp;
				}
			}
		}
	}
}
```

二分法查找

2018年7月11日 16:57

思路：

1.定义最小索引，最大索引

2.计算出中间索引

3.那中间索引的值和要查找的元素进行比较

相等：直接返回当前中间索引

大了：在左边找

小了：在右边找

4.重写获取最小索引或者最大索引

max=mid-1

mid=max+1

```
package cn.itcast_02;

/*
 * 查找：
 *		基本查找:数组元素无续（从头找到尾）
 *		二分查找（折半查找）：数组元素有序
 */
public class ArrayDemo {
	public static void main(String[] args) {
		int arr[] = { 11, 22, 33, 44, 55, 66, 77 };
		int index = getIndex(arr, 35);
		System.out.println(index);

	}

	public static int getIndex(int[] arr, int value) {
		int max = arr.length - 1;
		int min = 0;
		int mid = (max + min) / 2;
		while (arr[mid] != value) {
			if (arr[mid] > value) {
				max = mid - 1;
			} else {
				min = mid + 1;
			}
			if(min>max) {
				return -1;
			}
			mid = (max + min) / 2;
		}
		return mid;
	}
}
```

Arrays类

2018年7月11日 20:45

Arrays：针对数组进行操作的工具类（排序、查找）

```
import java.util.Arrays;

/*
 * Arrays
 * 1.public static String toString(int[] a)
 * 2.public static void sort(int[] a)
 * 3.public static int binarySearch(int[] a,int key):二分查找
 */
public class ArraysDemo {
	public static void main(String[] args) {
		int[] arr = { 24, 69, 80, 57, 13 };

		// public static String toString(int[] a)
		System.out.println("排序前:" + Arrays.toString(arr));

		// public static void sort(int[] a)
		Arrays.sort(arr);//底层为快速排序
		System.out.println("排序后:" + Arrays.toString(arr));

		// public static int binarySearch(int[] a,int key)
		System.out.println("查找结果为:" + Arrays.binarySearch(arr, 57));
		System.out.println("查找结果为:" + Arrays.binarySearch(arr, 577));// -6
		//return -(low+1);
	}
}
```

Integer

2018年7月11日 21:29

```
package cn.itcast_01;
/*
 *      需求1：把100分别转换为二进制、八进制、十六进制
 *      需求2：判断一个数字是否是int范围内
 * 为了对基本数据类型进行更多操作，java就对每一种基本类型提供了对应的包装类类型
 * byte      Byte
 * short     Short
 * int       Integer
 * long      Long
 * float     Float
 * double    Double
 * char      Charactor
 * boolean   Boolean
 */
public class IntegerDemo {
	public static void main(String[] args) {
		//public static String toBinaryString(int i)
		System.out.println(Integer.toBinaryString(100));
		System.out.println(Integer.toOctalString(100));
		System.out.println(Integer.toHexString(100));

		System.out.println(Integer.MIN_VALUE);
		System.out.println(Integer.MAX_VALUE);
	}
}
```

Integer的构造

2018年7月11日 22:01

```
/*
 * Integer构造：
 *                  public Integer(int value)
 *                  public Integer(String s)
 *                  注意：字符串必须由数字字符组成
 */
public class IntegerDemo2 {
        public static void main(String[] args) {
                int i = 100;
                Integer ii = new Integer(i);
                System.out.println("ii" + ii);

                String s = "100";
                //s="abc"; NumberFormatException
                Integer iii = new Integer(s);
                System.out.println("iii=" + iii);
        }

}
```

int和String的转换

2018年7月11日 22:15

```
package cn.itcast_02;

//int类型和String类型的相互转换
public class IntegerDemo {
	public static void main(String[] args) {
		int number = 100;
		// int---String
		// 方式1
		String s1 = "" + number;
		System.out.println("s1=" + s1);

		// 方式2(推荐)
		String s2 = String.valueOf(number);
		System.out.println("s2=" + s2);

		// 方式3
		Integer ii = new Integer(number);
		String s3 = ii.toString();
		System.out.println("s3=" + s3);

		// 方式4
		String s4 = Integer.toString(number);
		System.out.println("s4=" + s4);

		// String---int
		String s = "100";
		// 方式1
		Integer i1 = new Integer(s);
		int num = i1.intValue();
		System.out.println("i=" + num);
		// 方式2
		int y = Integer.parseInt(s);
		System.out.println("i=" + y);

	}
}
```

进制转换

2018年7月11日 22:29

```
/*
 * 十进制转换为其它进制
 * public static String toString(int i,int radix)
 * 其它进制到十进制
 * public static int parseInt(String s,int radix)
 */
public class IntegerDemo {
	public static void main(String[] args) {
		System.out.println(Integer.toString(100, 5));// 五进制
		System.out.println(Integer.toString(100, 7));// 七进制
		System.out.println(Integer.toString(100, -7));// 进制不能为负
		// 进制的范围-->2~36 0-9+a-z共36个

		// 其它进制到十进制
		System.out.println(Integer.parseInt("100", 10));
		System.out.println(Integer.parseInt("100", 2));
		System.out.println(Integer.parseInt("100", 8));
	}
}
```

JDK5的新特性

2018年7月11日 22:47

```
/*
 * JDK5的新特性：
 *                自动装箱：把基本类型转换为包装类类型
 *                自动拆箱：把包装类类型转换为基本类型
 */
public class IntegerTest {
	public static void main(String[] args) {
		Integer i = new Integer(100);
		Integer ii = 100;// JDK5新特性
		ii += 200;
		// 在使用时，Integer=Null;空指针报错
	}
}
```

重要面试题

2018年7月11日 23:31

注意：Integer的数据直接赋值，如果在-128-127之间，直接从缓冲池中获取

```
public class IntegerTest2 {
	public static void main(String[] args) {
		Integer i1 = new Integer(127);
		Integer i2 = new Integer(127);
		System.out.println(i1 == i2);
		System.out.println(i1.equals(i2));
		System.out.println("------------");

		Integer i3 = new Integer(128);
		Integer i4 = new Integer(128);
		System.out.println(i3 == i4);
		System.out.println(i3.equals(i4));
		System.out.println("------------");

		Integer i5 = 127;
		Integer i6 = 127;
		System.out.println(i5 == i6);
		System.out.println(i5.equals(i6));
		System.out.println("------------");
		/*
		 * Integer ii=Integer.valueOf(127); 通过查看源码，针对-128~127之间的数据，建立了一个数据缓冲池
		 * 数据在该范围内，不创建新空间
		 */

		Integer i7 = 128;
		Integer i8 = 128;
		System.out.println(i7 == i8);
		System.out.println(i7.equals(i8));
		System.out.println("------------");

	}
}
```

输出结果：

```
False
true
------------
false
true
------------
true
true
------------
false
true
------------
```

Character类

2018年7月11日 23:52

```
import java.net.StandardSocketOptions;

/*
 * Character
 *        构造方法:
 *                Character(char value)
 */
public class CharacterDemo {
        public static void main(String[] args) {
                Character ch =new Character('a');
                System.out.println("ch="+ch);
                System.out.println(Character.isUpperCase('a'));
                System.out.println(Character.isUpperCase('A'));
                System.out.println(Character.isUpperCase('0'));
                System.out.println(Character.isLowerCase('b'));
                System.out.println(Character.isDigit('0'));
                System.out.println(Character.toLowerCase('C'));
                System.out.println(Character.toUpperCase('t'));
        }
}
```

练习

2018年7月12日 0:00

```
import java.util.Scanner;

public class CharactorTest {
	public static void main(String[] args) {
		int bigcount = 0;
		int smallcount = 0;
		int numcount = 0;
		Scanner sc = new Scanner(System.in);
		System.out.println("请输入一个字符串");
		String line = sc.nextLine();

		char[] chs = line.toCharArray();
		for (int x = 0; x < chs.length; x++) {
			char ch = chs[x];
			if (Character.isUpperCase(ch)) {
				bigcount++;
			} else if (Character.isLowerCase(ch)) {
				smallcount++;
			} else if (Character.isDigit(ch)) {
				numcount++;
			}
		}
		System.out.println(bigcount + "  " + smallcount + "   " + numcount);
	}
}
```