Java 核心技术

第八章 Java常用类

第三节 字符串相关类

华东师范大学 陈良育

字符串(1)

- String
  - Java中使用频率最高的类
  - 是一个不可变对象，加减操作性能较差
  - 以下方法需要牢记：charAt, concat, contains, endsWith,equals, equalsIgnoreCase, hashCode, indexOf, length, matches, replace, replaceAll, split, startsWith, subString, trim, valueOf
    - 查看StringTest.java

字符串(2)

- 可变字符串
  - StringBuffer（字符串加减，同步，性能好）
  - StringBuilder（字符串加减，不同步，性能更好）
- StringBuffer/StringBuilder： 方法一样，区别在同步
  - append/insert/delete/replace/substring
  - length 字符串实际大小，capacity字符串占用空间大小
  - trimToSize(): 去除空隙，将字符串存储压缩到实际大小
  - 如有大量append，事先预估大小，再调用相应构造函数
  - 查看相关代码

字符串(3)

- 总结
  - String(不可变对象，只读)
  - StringBuffer（字符串加减，同步，性能好）
  - StringBuilder（字符串加减，不同步，性能更好）

代码(1) StringTest.java


```java
public class StringTest {

    public static void main(String[] args) {
        String a = "123;456;789;123 ";
        System.out.println(a.charAt(0)); // 返回第0个元素
        System.out.println(a.indexOf(";")); // 返回第一个;的位置
        System.out.println(a.concat(";000")); // 连接一个新字符串并返回，a不变
        System.out.println(a.contains("000")); // 判断a是否包含000
        System.out.println(a.endsWith("000")); // 判断a是否以000结尾
        System.out.println(a.equals("000")); // 判断是否等于000
        System.out.println(a.equalsIgnoreCase("000"));// 判断在忽略大小写情况下是否等于000
        System.out.println(a.length()); // 返回a长度
        System.out.println(a.trim()); // 返回a去除前后空格后的字符串，a不变
        String[] b = a.split(";"); // 将a字符串按照;分割成数组
        for (int i = 0; i < b.length; i++) {
            System.out.println(b[i]);
        }

        System.out.println("====================");
```

代码(2) StringTest.java


```
        System.out.println(a.substring(2, 5)); // 截取a的第2个到第5个字符 a不变
        System.out.println(a.replace("1", "a"));
        System.out.println(a.replaceAll("1", "a")); // replaceAll第一个参数是正则表达式

        System.out.println("====================");

        String s1 = "12345?6789";
        String s2 = s1.replace("?", "a");
        String s3 = s1.replaceAll("[?]", "a");
        // 这里的[?] 才表示字符问号，这样才能正常替换。不然在正则中会有特殊的意义就会报异常
        System.out.println(s2);
        System.out.println(s3);
        System.out.println(s1.replaceAll("[\\d]", "a")); //将s1内所有数字替换为a并输出，s1的值未改变。

    }
}
```

代码(3)
StringBufferReferenceTest.java


```
public class StringBufferReferenceTest {

    public static void main(String[] args) {
        StringBuffer sb1 = new StringBuffer("123");
        StringBuffer sb2 = sb1;

        sb1.append("1234567890123456789012345678901234567890");
        System.out.println(sb2);  //sb1和sb2还是指向同一个内存的

    }

}
```

代码(4) StringAppendTest.java


```java
import java.util.Calendar;

public class StringAppendTest {

    public static void main(String[] args) {
        int n = 50000;
        Calendar t1 = Calendar.getInstance();
        String a = new String();
        for(int i=0;i<n;i++)
        {
            a = a + i + ",";
        }
        System.out.println(Calendar.getInstance().getTimeInMillis() - t1.getTimeInMillis());

        Calendar t2 = Calendar.getInstance();
        StringBuffer b = new StringBuffer("");
        for(int i=0;i<n;i++)
        {
            b.append(i);
            b.append(",");
        }
        System.out.println(Calendar.getInstance().getTimeInMillis() - t2.getTimeInMillis());

        Calendar t3 = Calendar.getInstance();
        StringBuilder c = new StringBuilder("");
        for(int i=0;i<n;i++)
        {
            b.append(i);
            b.append(",");
        }
        System.out.println(Calendar.getInstance().getTimeInMillis() - t3.getTimeInMillis());
    }
}
```

代码(5)
StringBufferCapacityTest.java


```java
public class StringBufferCapacityTest {

    public static void main(String[] args) {
        //StringBuffer的的初始大小为（16+初始字符串长度）即capacity=16+初始字符串长度
        //length 实际长度 capacity 存储空间大小
        StringBuffer sb1 = new StringBuffer();
        System.out.println("sb1 length: " + sb1.length());
        System.out.println("sb1 capacity: " + sb1.capacity());
        System.out.println("======================");

        StringBuffer sb2 = new StringBuffer("123");
        sb2.append("456");
        System.out.println("sb2 length: " + sb2.length());
        System.out.println("sb2 capacity: " + sb2.capacity());
        System.out.println("======================");

        sb2.append("7890123456789");
        System.out.println("sb2 length: " + sb2.length());
        System.out.println("sb2 capacity: " + sb2.capacity());
        System.out.println("======================");
```

代码(6)
StringBufferCapacityTest.java


```
        sb2.append("0");
        System.out.println("sb2 length: " + sb2.length());
        System.out.println("sb2 capacity: " + sb2.capacity());
        //一旦length大于capacity时，capacity便在前一次的基础上加1后翻倍；
        System.out.println("======================");

        //当前sb2length 20   capacity 40，再append 70个字符 超过(加1再2倍数额)
        sb2.append("12345678901234567890123456789012345678901234567890123456789012345678901234567890");
        System.out.println("sb2 length: " + sb2.length());
        System.out.println("sb2 capacity: " + sb2.capacity());
        //如果append的对象很长，超过(加1再2倍数额)，将以最新的长度更换

        System.out.println("======================");
        sb2.append("0");
        System.out.println("sb2 length: " + sb2.length());
        System.out.println("sb2 capacity: " + sb2.capacity());
        sb2.trimToSize();
        System.out.println("=====after trime================");
        System.out.println("sb2 length: " + sb2.length());
        System.out.println("sb2 capacity: " + sb2.capacity());
    }

}
```

谢 谢！