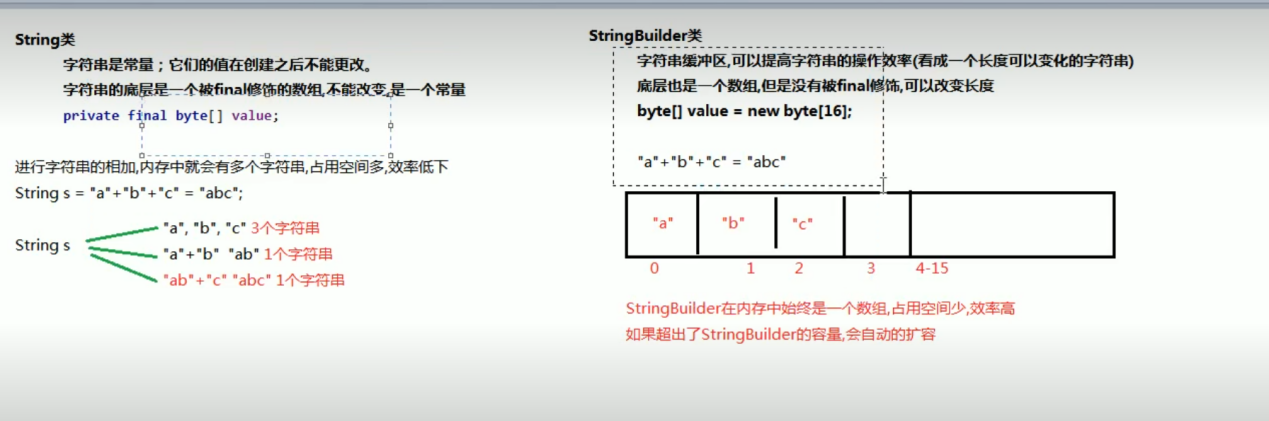
1. StringBuilder 类（也称：字符串缓冲区）

StringBuilder的原理：



字符串拼接问题

（1）StringBuilder类的构造方法

/\*  
java.long.StringBuilder()类：字符串缓冲区，可以提高字符串的效率  
构造方法：  
 - public StringBuilder():构造一个空的StringBuilder容器  
 - public StringBuilder(String str):构造一个StringBuilder容器，并将字符串添加进去。  
 \*/  
public class DemoStringBulider {  
 public static void main(String[] args) {  
  
 // 创建一个空的StringBuilder容器  
 StringBuilder bu1= new StringBuilder();  
 System.out.println("bu1:"+bu1); // bu1：  
  
 // 创建一个StringBuilder容器，并将字符串添加进去  
 StringBuilder bu2 = new StringBuilder("abc");  
 System.out.println("bu2:"+bu2); // bu2:abc  
 }  
}

Stringbulider 类的成员方法

package demo4;  
  
/\*  
StringBuilder()类的成员方法：  
 public StringBuilder append(...):添加任意类型数据的字符串形式，并返回当前对象自身。  
 参数：  
 可以时任意的数据类型  
 \*/  
public class DemoStringBulider {  
 public static void main(String[] args) {  
  
 // 创建StringBuilder对象  
 StringBuilder bu1 = new StringBuilder();  
  
 // 使用append方法往StringBuilder中添加数据。  
 StringBuilder bu2 = bu1.append("abc");  
 System.out.println(bu1); // abc  
 System.out.println(bu2); // abc  
  
 // 注意：其中append的返回值类型也是 StringBuilder，所以导致与所定义的空容器地址值一样。  
 // 使用append方法返回值是this，调用方法的对象bu1  
 System.out.println(bu1==bu2); // true  
 // true 这个为引用类型，比较的为两个对象的地址值，说明两个对象的地址值一样。  
  
 // 改进：使用append方法无需接收返回值  
 bu1.append("abc");  
 bu1.append('A');  
 System.out.println(bu1); // abcA  
 /\*  
 链式编程：方法的返回值是一个对象，可以根据对象继续调用方法。  
 \*/  
 // 因为返回值为 字符串 类型，所以可以继续调用字符串方法  
 System.out.println("abc".toString().toCharArray());  
 bu1.append("abc").append(1).append(true).append(8.8).append("中");  
 System.out.println(bu1); //abc1true8.8中  
 }  
}

总结：、

System 类

1.currentTimeMillis()方法

作用：获取当前系统的毫秒值

2.arraycopy()方法

作用：复制数组

import java.util.Arrays;  
public class Text {  
 public static void main(String[] args) {  
 // 获取一下当前系统毫秒值时间  
 long time = System.currentTimeMillis();  
 System.out.println(time); //1590825677442  
  
 // arraycopy(int[] src,int srcIndex,int[] dest,int destIndex,int count) 复制数组  
 int[] arr1 = {1,2,3,4,5};  
 int[] arr2 = {6,7,8,9,10};  
  
 // 将arr数组中的前四个数字复制到arr2数组中  
 System.arraycopy(arr1,0,arr2,0,4);  
  
 // 使用Array中的toString方法 打印一下复制后的数组arr2  
 System.out.println(Arrays.toString(arr2)); // [1, 2, 3, 4, 10]  
 }  
}

String类

构造方法

StringBuilder(): 创建一个空的字符串缓冲区对象

StringBilder(String str):根据传入的内容创建一个字符串缓冲区对象

成员方法

StringBuilder append(Object obj): 添加内容

StringBuilder reverse(): 反转内容

String toStrng(): 将缓冲区内容转为字符串。

public class Text01 {  
 public static void main(String[] args) {  
 StringBuilder sb = new StringBuilder();  
  
 // append方法  
 sb.append("abc");  
 sb.append(true);  
 sb.append("Hello").append("world").append("java"); // Helloworldjava  
  
 // toString 方法  
 System.out.println(sb);  
 }  
}

StringBuilder类中的toString方法  
/\*  
StringBuilder和String可以相互转化：  
 String -> StringBuilder:可以使用StringBuilder的构造方法  
 StringBuilder(String str):构造一字符串生成器，并初始化为指定的字符串内容。  
 StringBuilder -> String:可以使用StringBuilder中的toString方法  
 public String toString():将当前的StringBuilder对象转换为String对象  
 \*/  
public class toString {  
 public static void main(String[] args) {  
  
 // String -> StringBuilder  
 String str = "hello";  
 System.out.println("str:"+str); // str:hello  
 StringBuilder bu = new StringBuilder(str);  
  
 // 往StringBuilder中添加数据  
 bu.append("word");  
 System.out.println("bu:"+bu); //bu:helloword  
  
 // StringBuilder -> String  
 String s = bu.toString();  
 System.out.println("s:"+s); // s:helloword  
 }  
}