[**String类和StringBuffer类的区别**](http://www.cnblogs.com/rpp506815950/archive/2012/11/07/2758646.html)

**首先，String和StringBuffer主要有2个区别：**

**（1）String类对象为不可变对象，一旦你修改了String对象的值，隐性重新创建了一个新的对象，释放原String对象，StringBuffer类对象为可修改对象，可以通过append()方法来修改值**

**（2）String类对象的性能远不如StringBuffer类。**

关于以上具体解释如下：

在java中有3个类来负责字符的操作。

1.Character 是进行单个字符操作的，

2.String 对一串字符进行操作。不可变类。

3.StringBuffer 也是对一串字符进行操作，但是可变类。

String:  
是对象不是原始类型.  
为不可变对象,一旦被创建,就不能修改它的值.  
对于已经存在的String对象的修改都是重新创建一个新的对象,然后把新的值保存进去.  
String 是final类,即不能被继承.

StringBuffer:  
是一个可变对象,当对他进行修改的时候不会像String那样重新建立对象  
它只能通过构造函数来建立,  
StringBuffer sb = new StringBuffer();  
注意:不能通过赋值符号对他进行赋值.   
sb = "welcome to here!";//error  
对象被建立以后,在内存中就会分配内存空间,并初始保存一个null.向StringBuffer  
中赋值的时候可以通过它的append方法.  
sb.append("hello");

字符串连接操作中StringBuffer的效率要比String高:

String str = new String("welcome to ");  
str += "here";  
的处理步骤实际上是通过建立一个StringBuffer,让侯调用append(),最后  
再将StringBuffer toSting();  
这样的话String的连接操作就比StringBuffer多出了一些附加操作,当然效率上要打折扣.

并且由于String 对象是不可变对象,每次操作Sting 都会重新建立新的对象来保存新的值.  
这样原来的对象就没用了,就要被垃圾回收.这也是要影响性能的.

看看以下代码：  
将26个英文字母重复加了5000次，

1. String tempstr = "abcdefghijklmnopqrstuvwxyz";
2. int times = 5000;
3. long lstart1 = System.currentTimeMillis();
4. String str = "";
5. for (int i = 0; i < times; i++) {
6. str += tempstr;
7. }
8. long lend1 = System.currentTimeMillis();
9. long time = (lend1 - lstart1);
10. System.out.println(time);

得到的结果每次不一定，一般为 1563左右。  
我们再看看以下代码

1. String tempstr = "abcdefghijklmnopqrstuvwxyz";
2. int times = 5000;
3. long lstart2 = System.currentTimeMillis();
4. StringBuffer sb = new StringBuffer();
5. for (int i = 0; i < times; i++) {
6. sb.append(tempstr);
7. }
8. long lend2 = System.currentTimeMillis();
9. long time2 = (lend2 - lstart2);
10. System.out.println(time2);

得到的结果为 16 有时还是 0  
所以结论很明显，StringBuffer 的速度几乎是String 上万倍。当然这个数据不是很准确。因为循环的次数在100000次的时候，差异更大。不信你试试。

根据上面所说：

str += "here";  
的处理步骤实际上是通过建立一个StringBuffer,让侯调用append(),最后  
再将StringBuffer toSting();

所以str += "here";可以等同于

StringBuffer sb = new StringBuffer(str);

sb.append("here");

str = sb.toString();

所以上面直接利用"+"来连接String的代码可以基本等同于以下代码

1. String tempstr = "abcdefghijklmnopqrstuvwxyz";
2. int times = 5000;
3. long lstart2 = System.currentTimeMillis();
4. String str = "";
5. for (int i = 0; i < times; i++) {
6. StringBuffer sb = new StringBuffer(str);
7. sb.append(tempstr);
8. str = sb.toString();
9. }
10. long lend2 = System.currentTimeMillis();
11. long time2 = (lend2 - lstart2);
12. System.out.println(time2);

平均执行时间为1563左右。

总结: 如果在程序中需要对字符串进行频繁的修改连接操作的话.使用StringBuffer性能会更高