Difference between StringBuffer and StringBuilder

There are many differences between StringBuffer and StringBuilder. A list of differences between StringBuffer and StringBuilder are given below:

No.	StringBuffer	StringBuilder
1)		StringBuilder is <i>non-synchronized</i> i.e. not thread safe. It means two threads can call the methods of StringBuilder simultaneously.
2)	StringBuffer is <i>less efficient</i> than StringBuilder.	StringBuilder is more efficient than StringBuffer.

StringBuffer Example

```
public class BufferTest{
   public static void main(String[] args){
      StringBuffer buffer=new StringBuffer("hello");
      buffer.append("java");
      System.out.println(buffer);
   }
}
```

```
hellojava
```

StringBuilder Example

```
public class BuilderTest{
   public static void main(String[] args){
      StringBuilder builder=new StringBuilder("hello");
      builder.append("java");
      System.out.println(builder);
   }
}
```

hellojava

Performance Test of StringBuffer and StringBuilder

Let's see the code to check the performance of StringBuffer and StringBuilder classes.

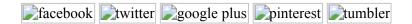
```
public class ConcatTest{
  public static void main(String[] args){
    long startTime = System.currentTimeMillis();
    StringBuffer sb = new StringBuffer("Java");
    for (int i=0; i<10000; i++){
        sb.append("Tpoint");
    }
    System.out.println("Time taken by StringBuffer: " + (System.currentTimeMillis() - startTime) + "ms");
    startTime = System.currentTimeMillis();
    StringBuilder sb2 = new StringBuilder("Java");
    for (int i=0; i<10000; i++){
        sb2.append("Tpoint");
    }
    System.out.println("Time taken by StringBuilder: " + (System.currentTimeMillis() - startTime) + "ms");
}
</pre>
```

```
Time taken by StringBuffer: 16ms
Time taken by StringBuilder: 0ms
```

 \leftarrow prev

 $next \rightarrow$

Share this page



Latest 4 Tutorials

