|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **[next →](https://www.javatpoint.com/difference-between-stringbuffer-and-stringbuilder)**[**← prev**](https://www.javatpoint.com/StringBuilder-class)  Difference between String and StringBuffer  There are many differences between String and StringBuffer. A list of differences between String and StringBuffer are given below:   |  |  |  | | --- | --- | --- | | **No.** | **String** | **StringBuffer** | | 1) | String class is immutable. | StringBuffer class is mutable. | | 2) | String is slow and consumes more memory when you concat too many strings because every time it creates new instance. | StringBuffer is fast and consumes less memory when you cancat strings. | | 3) | String class overrides the equals() method of Object class. So you can compare the contents of two strings by equals() method. | StringBuffer class doesn't override the equals() method of Object class. |   String vs StringBuffer  Performance Test of String and StringBuffer   1. **public** **class** ConcatTest{ 2. **public** **static** String concatWithString()    { 3. String t = "Java"; 4. **for** (**int** i=0; i<10000; i++){ 5. t = t + "Tpoint"; 6. } 7. **return** t; 8. } 9. **public** **static** String concatWithStringBuffer(){ 10. StringBuffer sb = **new** StringBuffer("Java"); 11. **for** (**int** i=0; i<10000; i++){ 12. sb.append("Tpoint"); 13. } 14. **return** sb.toString(); 15. } 16. **public** **static** **void** main(String[] args){ 17. **long** startTime = System.currentTimeMillis(); 18. concatWithString(); 19. System.out.println("Time taken by Concating with String: "+(System.currentTimeMillis()-startTime)+"ms"); 20. startTime = System.currentTimeMillis(); 21. concatWithStringBuffer(); 22. System.out.println("Time taken by Concating with  StringBuffer: "+(System.currentTimeMillis()-startTime)+"ms"); 23. } 24. }   Time taken by Concating with String: 578ms  Time taken by Concating with StringBuffer: 0ms  String and StringBuffer HashCode Test  As you can see in the program given below, String returns new hashcode value when you concat string but StringBuffer returns same.   1. **public** **class** InstanceTest{ 2. **public** **static** **void** main(String args[]){ 3. System.out.println("Hashcode test of String:"); 4. String str="java"; 5. System.out.println(str.hashCode()); 6. str=str+"tpoint"; 7. System.out.println(str.hashCode()); 9. System.out.println("Hashcode test of StringBuffer:"); 10. StringBuffer sb=**new** StringBuffer("java"); 11. System.out.println(sb.hashCode()); 12. sb.append("tpoint"); 13. System.out.println(sb.hashCode()); 14. } 15. }   Hashcode test of String:  3254818  229541438  Hashcode test of StringBuffer:  118352462  118352462 |