Java StringBuffer class

Java StringBuffer class is used to create mutable (modifiable) string. The StringBuffer class in java is same as String class except it is mutable i.e. it can be changed.

Note: Java StringBuffer class is thread-safe i.e. multiple threads cannot access it simultaneously. So it is safe and will result in an order.

Important Constructors of StringBuffer class

|  |  |
| --- | --- |
| **Constructor** | **Description** |
| StringBuffer() | creates an empty string buffer with the initial capacity of 16. |
| StringBuffer(String str) | creates a string buffer with the specified string. |
| StringBuffer(int capacity) | creates an empty string buffer with the specified capacity as length. |

Important methods of StringBuffer class

|  |  |  |
| --- | --- | --- |
| **Modifier and Type** | **Method** | **Description** |
| public synchronized StringBuffer | append(String s) | is used to append the specified string with this string.  The append() method is overloaded like append(char),  append(boolean), append(int), append(float),  append(double) etc. |
| public synchronized StringBuffer | insert(int offset, String s) | is used to insert the specified string with this string  at the  specified position.  The insert() method is overloaded like  insert(int, char), insert(int, boolean),  insert(int, int), insert(int, float),  insert(int, double) etc. |
| public synchronized StringBuffer | replace(int startIndex, int endIndex, String str) | is used to replace the string from specified  startIndex and endIndex. |
| public synchronized StringBuffer | delete(int startIndex, int endIndex) | is used to delete the string from  specified startIndex and endIndex. |
| public synchronized StringBuffer | reverse() | is used to reverse the string. |
| public int | capacity() | is used to return the current capacity. |
| public void | ensureCapacity(int minimumCapacity) | is used to ensure the capacity at least  equal to the given minimum. |
| public char | charAt(int index) | is used to return the character  at the specified position. |
| public int | length() | is used to return the length  of the string i.e. total number of characters. |
| public String | substring(int beginIndex) | is used to return the  substring from the specified beginIndex. |
| public String | substring(int beginIndex, int endIndex) | is used to return the substring  from the specified beginIndex and  endIndex. |

What is mutable string

A string that can be modified or changed is known as mutable string. StringBuffer and StringBuilder classes are used for creating mutable string.

1) StringBuffer append() method

The append() method concatenates the given argument with this string.

1. **class** StringBufferExample{
2. **public** **static** **void** main(String args[]){
3. StringBuffer sb=**new** StringBuffer("Hello ");
4. sb.append("Java");//now original string is changed
5. System.out.println(sb);//prints Hello Java
6. }
7. }

2) StringBuffer insert() method

The insert() method inserts the given string with this string at the given position.

1. **class** StringBufferExample2{
2. **public** **static** **void** main(String args[]){
3. StringBuffer sb=**new** StringBuffer("Hello ");
4. sb.insert(1,"Java");//now original string is changed
5. System.out.println(sb);//prints HJavaello
6. }
7. }

3) StringBuffer replace() method

The replace() method replaces the given string from the specified beginIndex and endIndex.

1. **class** StringBufferExample3{
2. **public** **static** **void** main(String args[]){
3. StringBuffer sb=**new** StringBuffer("Hello");
4. sb.replace(1,3,"Java");
5. System.out.println(sb);//prints HJavalo
6. }
7. }

4) StringBuffer delete() method

The delete() method of StringBuffer class deletes the string from the specified beginIndex to endIndex.

1. **class** StringBufferExample4{
2. **public** **static** **void** main(String args[]){
3. StringBuffer sb=**new** StringBuffer("Hello");
4. sb.delete(1,3);
5. System.out.println(sb);//prints Hlo
6. }
7. }

5) StringBuffer reverse() method

The reverse() method of StringBuilder class reverses the current string.

1. **class** StringBufferExample5{
2. **public** **static** **void** main(String args[]){
3. StringBuffer sb=**new** StringBuffer("Hello");
4. sb.reverse();
5. System.out.println(sb);//prints olleH
6. }
7. }

6) StringBuffer capacity() method

The capacity() method of StringBuffer class returns the current capacity of the buffer. The default capacity of the buffer is 16. If the number of character increases from its current capacity, it increases the capacity by (oldcapacity\*2)+2. For example if your current capacity is 16, it will be (16\*2)+2=34.

1. **class** StringBufferExample6{
2. **public** **static** **void** main(String args[]){
3. StringBuffer sb=**new** StringBuffer();
4. System.out.println(sb.capacity());//default 16
5. sb.append("Hello");
6. System.out.println(sb.capacity());//now 16
7. sb.append("java is my favourite language");
8. System.out.println(sb.capacity());//now (16\*2)+2=34 i.e (oldcapacity\*2)+2
9. }
10. }

7) StringBuffer ensureCapacity() method

The ensureCapacity() method of StringBuffer class ensures that the given capacity is the minimum to the current capacity. If it is greater than the current capacity, it increases the capacity by (oldcapacity\*2)+2. For example if your current capacity is 16, it will be (16\*2)+2=34.

1. **class** StringBufferExample7{
2. **public** **static** **void** main(String args[]){
3. StringBuffer sb=**new** StringBuffer();
4. System.out.println(sb.capacity());//default 16
5. sb.append("Hello");
6. System.out.println(sb.capacity());//now 16
7. sb.append("java is my favourite language");
8. System.out.println(sb.capacity());//now (16\*2)+2=34 i.e (oldcapacity\*2)+2
9. sb.ensureCapacity(10);//now no change
10. System.out.println(sb.capacity());//now 34
11. sb.ensureCapacity(50);//now (34\*2)+2
12. System.out.println(sb.capacity());//now 70
13. }
14. }