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

1) StringBuffer Class append() Method

**class** StringBufferExample{

**public** **static** **void** main(String args[]){

StringBuffer sb=**new** StringBuffer("Hello ");

sb.append("Java");//now original string is changed

System.out.println(sb);//prints Hello Java

}

}

**Output:**

Hello Java

### Graphical user interface, text, application Description automatically generated

### 2) StringBuffer insert() Method

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

**StringBufferExample2.java**

**class** StringBufferExample2{

**public** **static** **void** main(String args[]){

StringBuffer sb=**new** StringBuffer("Hello ");

sb.insert(1,"Java");//now original string is changed

System.out.println(sb);//prints HJavaello

}

}

**Output:**

HJavaello

### Graphical user interface, text, application Description automatically generated

### 3) StringBuffer replace() Method

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

**StringBufferExample3.java**

**class** StringBufferExample3{

**public** **static** **void** main(String args[]){

StringBuffer sb=**new** StringBuffer("Hello");

sb.replace(1,3,"Java");

System.out.println(sb);//prints HJavalo

}

}

**Output:**

HJavalo

### Graphical user interface, text, application Description automatically generated

### 4) StringBuffer delete() Method

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

**StringBufferExample4.java**

**class** StringBufferExample4{

**public** **static** **void** main(String args[]){

StringBuffer sb=**new** StringBuffer("Hello");

sb.delete(1,3);

System.out.println(sb);//prints Hlo

}

}

**Output:**

Hlo

### Graphical user interface, text, application Description automatically generated

### 5) StringBuffer reverse() Method

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

**StringBufferExample5.java**

**class** StringBufferExample5{

**public** **static** **void** main(String args[]){

StringBuffer sb=**new** StringBuffer("Hello");

sb.reverse();

System.out.println(sb);//prints olleH

}

}

**Output:**

olleH

### Graphical user interface, text, application Description automatically generated

### 6) StringBuffer capacity() Method

The capacity() method of the 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.

**StringBufferExample6.java**

**class** StringBufferExample6{

**public** **static** **void** main(String args[]){

StringBuffer sb=**new** StringBuffer();

System.out.println(sb.capacity());//default 16

sb.append("Hello");

System.out.println(sb.capacity());//now 16

sb.append("java is my favourite language");

System.out.println(sb.capacity());//now (16\*2)+2=34 i.e (oldcapacity\*2)+2

}

}

**Output:**

16

16

34

### Graphical user interface, text, application Description automatically generated

### 7) StringBuffer ensureCapacity() method

The ensureCapacity() method of the 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.

**StringBufferExample7.java**

**class** StringBufferExample7{

**public** **static** **void** main(String args[]){

StringBuffer sb=**new** StringBuffer();

System.out.println(sb.capacity());//default 16

sb.append("Hello");

System.out.println(sb.capacity());//now 16

sb.append("java is my favourite language");

System.out.println(sb.capacity());//now (16\*2)+2=34 i.e (oldcapacity\*2)+2

sb.ensureCapacity(10);//now no change

System.out.println(sb.capacity());//now 34

sb.ensureCapacity(50);//now (34\*2)+2

System.out.println(sb.capacity());//now 70

}

}

**Output:**

16

16

34

34

70

|  |
| --- |
| // Java program to demonstrate  // the charAt() Method.    **class** GFG {  **public** **static** **void** main(String[] args)      {            // create a StringBuilder object          StringBuilder str = **new** StringBuilder();            // add the String to StringBuilder Object          str.append("Geek");            // get char at position 1  **char** ch = str.charAt(1);            // print the result          System.out.println("StringBuilder Object"                             + " contains = " + str);          System.out.println("Character at Position 1"                             + " in StringBuilder = " + ch);      }  } |

**Output:**

StringBuilder Object contains = Geek

Character at Position 1 in StringBuilder = e

Graphical user interface, text, application

Description automatically generated

**Example 2:**

|  |
| --- |
| // Java program demonstrate  // the charAt() Method.    **class** GFG {  **public** **static** **void** main(String[] args)      {          // create a StringBuilder object          // with a String pass as parameter          StringBuilder              str              = **new** StringBuilder("WelcomeGeeks");            // print string          System.out.println("String is "                             + str.toString());            // loop through string and print every Character  **for** (**int** i = 0; i < str.length(); i++) {                // get char at position i  **char** ch = str.charAt(i);                // print char              System.out.println("Char at position "                                 + i + " is " + ch);          }      }  } |

**Output:**

String is WelcomeGeeks

Char at position 0 is W

Char at position 1 is e

Char at position 2 is l

Char at position 3 is c

Char at position 4 is o

Char at position 5 is m

Char at position 6 is e

Char at position 7 is G

Char at position 8 is e

Char at position 9 is e

Char at position 10 is k

Char at position 11 is s

Graphical user interface, text, application

Description automatically generated