Programming in Java

Topic: StringBuilder Class

Introduction

- Java StringBuilder class is used to create mutable (modifiable) string.
- We can add, insert or append new contents into a string builder, whereas the value of a String object is fixed, once the string is created.
- It is available since JDK 1.5.

StringBuilder Constructors

Following are some of the constructors defined for StringBuilder class:

- StringBuilder( )

  It creates an empty string Builder with the initial capacity of 16.

- StringBuilder(int length)

  It creates an empty string Builder with the specified capacity as length.

- StringBuilder(String str)

  It creates a string Builder with the specified string.

- The default constructor reserves room for 16 characters without reallocation.

Some Examples

```
StringBuilder sb = new StringBuilder();
System.out.println(sb.capacity());//Default capacity:16

StringBuilder sb = new StringBuilder(65);
System.out.println(sb.capacity());//Specified capacity:65

StringBuilder sb = new StringBuilder("A");
System.out.println(sb.capacity());
//Capacity: Default+No. of characters in the string, i.e. 16+1=17

StringBuilder sb = new StringBuilder('A');
System.out.println(sb.capacity());//Capacity:65[ASCII code of 'A']
```

StringBuilder Methods

length( ) and capacity( )

The current length of a StringBuilder can be found via the length( ) method, while the total allocated capacity can be found through the capacity( ) method.

int length( )

int capacity( )

The capacity() is the number of characters it is able to store without having to increase its size

The length() method returns the number of characters actually stored in the string builder

Example:

```
class StringBuilderDemo {
        public static void main(String args[]) {
                StringBuilder sb = new StringBuilder(“New Zealand");
                System.out.println("length = " + sb.length());//11
                System.out.println("capacity = " + sb.capacity());//27[16+11]
            }
        }
```

Important

- When the length of StringBuilder becomes larger than the capacity then memory reallocation is done:

- In case of StringBuilder, reallocation of memory is done using the following rule:

- If the new demand is exceeding the current capacity then new capacity will be:
- new_capacity = 2*(original_capacity + 1)
- If new_capacity can accommodate new demand, then it will remain as it is, otherwise new_capacity value will be set to the value of new demand.

```
public class StringBuilderCapacityExample3 {
    public static void main(String[] args) {
        StringBuilder sb=new StringBuilder();
        System.out.println(sb.capacity());//default 16
        sb.append("Hello");    5 characters took the space
        System.out.println(sb.capacity());//now 16   [Capacity will not change]
        sb.append("java is my favourite language");//After appending the current capacity will be exceeded, so reallocation will be performed
        System.out.println(sb.capacity());//now 2*(16+1)=34
        sb.append("string");   //It also exceeds the current capacity, so reallocation will be performed
        System.out.println(sb.capacity());//now 2*(34+1)=70
    }
}
```

ensureCapacity( )

- If we want to preallocate room for a certain number of characters after a StringBuilder has been constructed, we can use ensureCapacity( ) to set the size of the buffer.

- This is useful if we know in advance that we will be appending a large number of small strings to a StringBuilder.

void ensureCapacity(int capacity)

Example: ensureCapacity()

```
public class Main
{
        public static void main(String[] args) {
        StringBuilder sb=new StringBuilder(12);
   System.out.println(sb.capacity());//12
   sb.ensureCapacity(18);
   System.out.println(sb.capacity());//2*(12+1)=26
        }
}
```

setLength( )

- used to set the length of the buffer within a StringBuilder object.

void setLength(int length)

- Here, length specifies the length of the buffer.
- When we increase the size of the buffer, null characters are added to the end of the existing buffer.
- If the new length is less than the current length of the string builder, then string builder is truncated to contain exactly the number of characters given in the new length.

```
public class Main
{
        public static void main(String[] args) {
        StringBuilder sb = new StringBuilder("Hello");
        System.out.println(sb.length());
        sb.setLength(2);
    System.out.println("New length is:"+sb.length()+" with content:"+sb);
        }
}
```

Output:

5

New length is:2 with content He

charAt( ) and setCharAt( )

- The value of a single character can be obtained from a StringBuilder via the charAt( ) method.

- We can set the value of a character within a StringBuilder using setCharAt( ).

char charAt(int index)

void setCharAt(int index, char ch

Example:

```
public class Main
{
	public static void main(String[] args) {
	StringBuilder str = new StringBuilder("Welcome");
	System.out.println("String = " + str);
	// set char at index 2 to 'L'
	str.setCharAt(2, 'L');
	// print string
	System.out.println("After setCharAt() String = "+ str); //WeLcome
	System.out.println(str.charAt(0));//W
		}
}
```

getChars( )

- getChars( ) method is used to copy a substring of a StringBuilder into an array.
- void getChars(int srcBegin, int srcEnd, char[] dst, int dstBegin) method of StringBuilder class copies the characters starting at the given index:srcBegin to index:srcEnd-1 from String contained by StringBuilder into an array of char passed as parameter to function.

```
public class Main
{
        public static void main(String[] args) {
        StringBuilder str = new StringBuilder("WelcomeJava");
    char[] array = new char[7];
    str.getChars(0, 7, array, 0);
    System.out.print("Char array contains : ");
    for (int i = 0; i < array.length; i++) {
       System.out.print(array[i] + " "); //W e l c o m e
    }
        }
}
```

append() Method

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

```
class Example
{
public static void main(String args[])
{
StringBuilder sb=new StringBuilder("Hello ");
sb.append("Java");//now original string is changed
System.out.println(sb);//prints Hello Java
}
}
```

insert Method

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

```
class Example
{
public static void main(String args[])
{
StringBuilder sb=new StringBuilder("Hello ");
sb.insert(1,"Java");//now original string is changed
System.out.println(sb);//prints HJavaello
}
}
```

replace Method

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

StringBuilder replace(int startIndex, int endIndex, String str)

Thus, the substring at startIndex through endIndex–1 is replaced.

```
class Example
{
public static void main(String args[])
{
StringBuilder sb=new StringBuilder("Hello");
sb.replace(1,3,"Java");
System.out.println(sb);//prints HJavalo
}
}
```

delete( ) and deleteCharAt( )

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

StringBuilder delete(int startIndex, int endIndex)
StringBuilder deleteCharAt(int index)

- The delete( ) method deletes a sequence of characters from the invoking object (from startIndex to endIndex-1).

- The deleteCharAt( ) method deletes the character at the specified index.

- It returns the resulting StringBuilder

Example

```
public class Main
{
	public static void main(String[] args) {
StringBuilder sb = new StringBuilder("WelcomeJava");
	sb.delete(3, 7);
	System.out.println("After delete: " + sb);//WelJava
	sb.deleteCharAt(2);
System.out.println("After deleteCharAt: " + sb);//WeJava
	}
}
```

substring( )

- Used to obtain a portion of a StringBuilder by calling substring( ).

String substring(int startIndex)
String substring(int startIndex, int endIndex)

- The first form returns the substring that starts at startIndex and runs to the end of the invoking StringBuilder object.

- The second form returns the substring that starts at startIndex and runs through endIndex–1

Example:

```
public class Main
{
        public static void main(String[] args) {
         StringBuilder str = new StringBuilder("WelcomeJava");
    System.out.println("SubSequence = " + str.substring(7)); //Java
    System.out.println("SubSequence = " + str.substring(0,7)); //Welcome
        }
}
```

reverse() Method

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

```
class Example
{
public static void main(String args[])
{
StringBuilder sb=new StringBuilder("Hello");
sb.reverse();
System.out.println(sb);//prints olleH
}
}
```

Q1(Output)

```
public class Main
{
        public static void main(String[] args) {
         StringBuilder sb=new StringBuilder(2);
         sb.append("Exam");
         System.out.println(sb.capacity()+" "+sb.length());
        }
}
```

A.2 2

B.4 2

C.6 4

D.2 4

Q2(Output)

```
public class Main
{
        public static void main(String[] args) {
         StringBuilder sb = new StringBuilder("Programming");
         sb.setLength(7);
    System.out.println(sb.length()+” “+sb);

        }
}
```

A.7 Programming

B.11 Program

C.7 Program

D.11 Programming

Q3(Output)

```
public class Main
{
        public static void main(String[] args) {
         StringBuilder str = new StringBuilder("Evaluation");
    System.out.println(str.substring(1));
        }
}
```

A.v

B.valuation

C.va

D.valuatio

Q4(Output)

```
public class Main
{
        public static void main(String[] args) {
    StringBuilder str = new StringBuilder("Programming");
    char[] array = new char[5];
    str.getChars(0, 5, array, 0);
    System.out.print("Char array contains : ");
    for (int i = 0; i < array.length; i++) {
       System.out.print(array[i]);
    }
        }
}
```

A.Progr

B.Progra

C.Program

D.Programming

Q5(Output)

```
public class Main
{
        public static void main(String[] args) {
        StringBuilder sb = new StringBuilder("PollingQuestion");
        sb.delete(1, 4);
        System.out.println(sb);
        }
}
```

A.PngQuestion

B.PingQuestion

C.Polling

D.Question

Q6(Output)

```
public class Main
{
        public static void main(String[] args) {
        StringBuilder sb=new StringBuilder("Object");
    sb.insert(6,"ive");
    System.out.println(sb);
        }
}
```

A.Objecivet

B.Objective

C.ive

D.Object