StringBuilder – Dynamic String Object

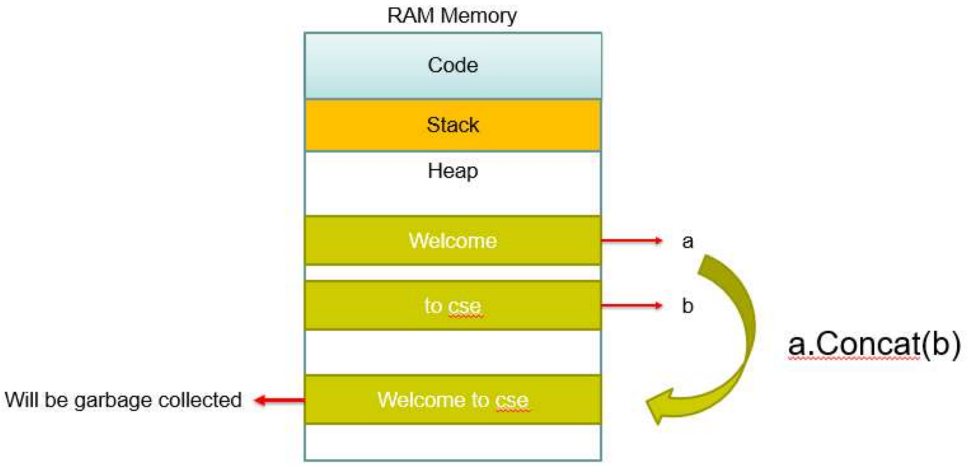
Strings are Immutable

This means, once we create a string, we cannot change that string.

For example, consider the following code.

```
string a = "Welcome ";
string b = "to cse";
a.Concat(b);
Console.WriteLine(a);
```

Output: Welcome



Here, we are using the Concat() method to add the string "Welcome " to the string “to cse”. But, the variable a still gives the old string “Welcome ”.

The reason behind this is, strings in C# are immutable objects. Hence, it creates a new object whenever the string are modified.

The new modified string object, "Welcome to cse", is released for garbage collection because no other variable holds a reference to it.

What is StringBuilder?

- StringBuilder is C# class System.Text.StringBuilder used to create Mutable String object.
- StringBuilder is a dynamic object. It doesn’t create a new object in the memory but dynamically expands the needed memory to accommodate the modified or new string.
- To avoid string replacing, appending, removing or inserting new strings in the initial string C# introduce StringBuilder concept.

How to create StringBuilder?

StringBuilder can be declared and initialized the same way as class object creation. We must include the namespace as given below.

```
using System.Text;
```

For example,

```
// Empty StringBuilder object
StringBuilder sb1 = new StringBuilder();
```

Or

```
// StringBuilder object with data
StringBuilder sb1 = new StringBuilder("Hello World");
```

Length and Capacity of StringBuilder

The StringBuilder class has two main properties:

- Length - which indicates the length of the string that it actually contains
- Capacity - which indicates the maximum length of the string in the memory allocation

```
StringBuilder sb1 = new StringBuilder();
//here the capacity – 16, length - 0

StringBuilder sb1 = new StringBuilder("Hello World");
//here the capacity – 16, length - 11
```

We can also set the minimum capacity and maximum capacity for StringBuilder object as given below.

```
StringBuilder sb1 = new StringBuilder(50,200);
//here the capacity – 50, length – 0, maxcapacity – 200
```

Difference between String and StringBuilder

String – Immutable Objects
StringBuilder – Mutable Objects

For example,

```
string a = "Welcome ";
string b = "to cse";
a.Concat(b);
```

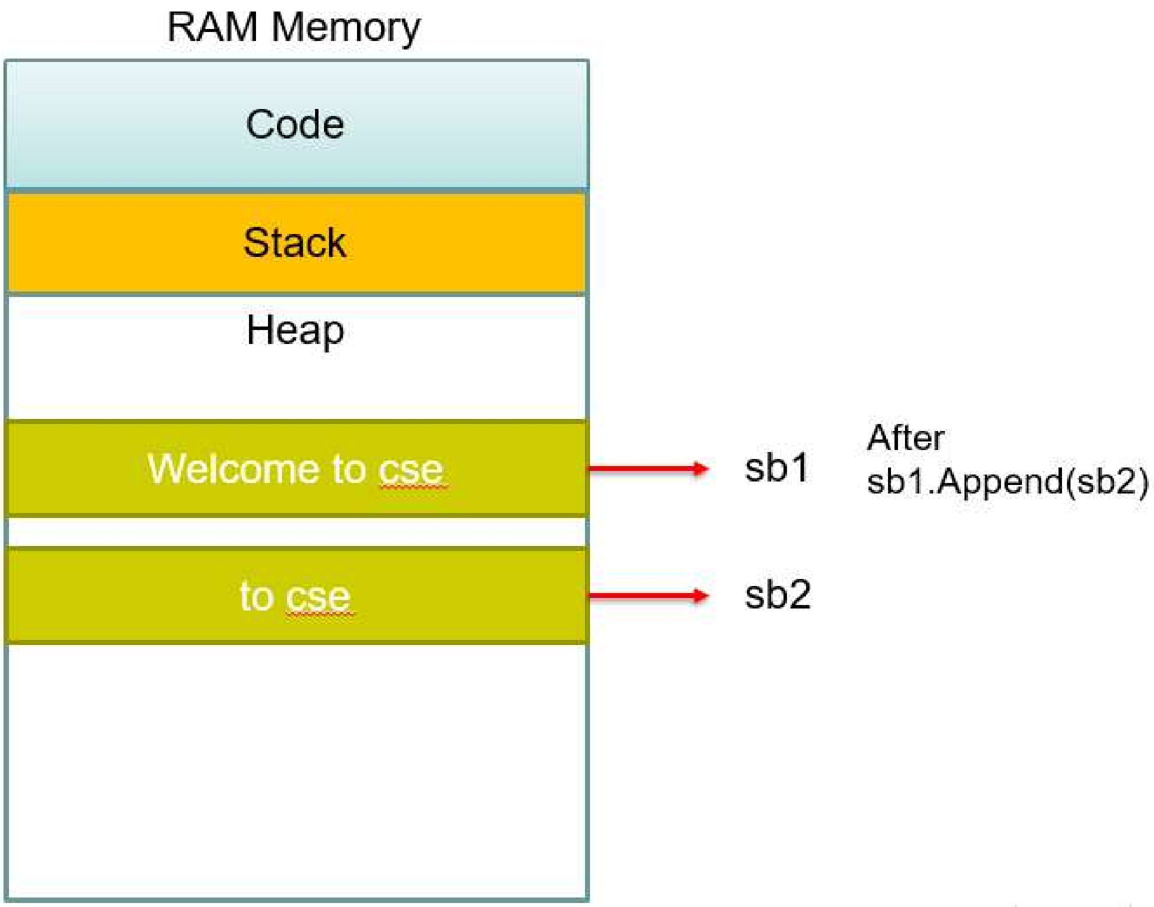
```
            Console.WriteLine(a);
```

Output: `Welcome`

But,

```
            StringBuilder sb1 = new StringBuilder("Welcome");
            StringBuilder sb2 = new StringBuilder(" to cse");
            sb1.Append(sb2);
            Console.WriteLine(sb1);
```

Output: `Welcome to cse`



Here, the StringBuilder allows in memory modification and does not create a new object.

Important Methods of StringBuilder Class:

- Append(string value)
- AppendFormat()
- Insert(int index, string value)

- Remove(int start, int length)
- Replace(old_val, new|_val)

Append(string value) method

The Append() method can be used to add or append a string value of an object to the end of a string represented by the current StringBuilder object.
AppendLine() method also come under this method. This method append the string with a newline at the end.
For example

```
using System;
using System.Text;
class Program
   {
        static void Main(string[] args)
        {
            StringBuilder sb1 = new StringBuilder("Welcome");
            sb1.Append(" to cse");
            sb1.AppendLine();
            sb1.Append("This is a Sample Text");
            sb1.AppendLine();
            sb1.Append("Hello World");
            Console.WriteLine(sb1);
            Console.ReadKey();
        }
   }
```

Output:

```
Welcome to cse
This is a Sample Text
Hello World
```

AppendFormat() method

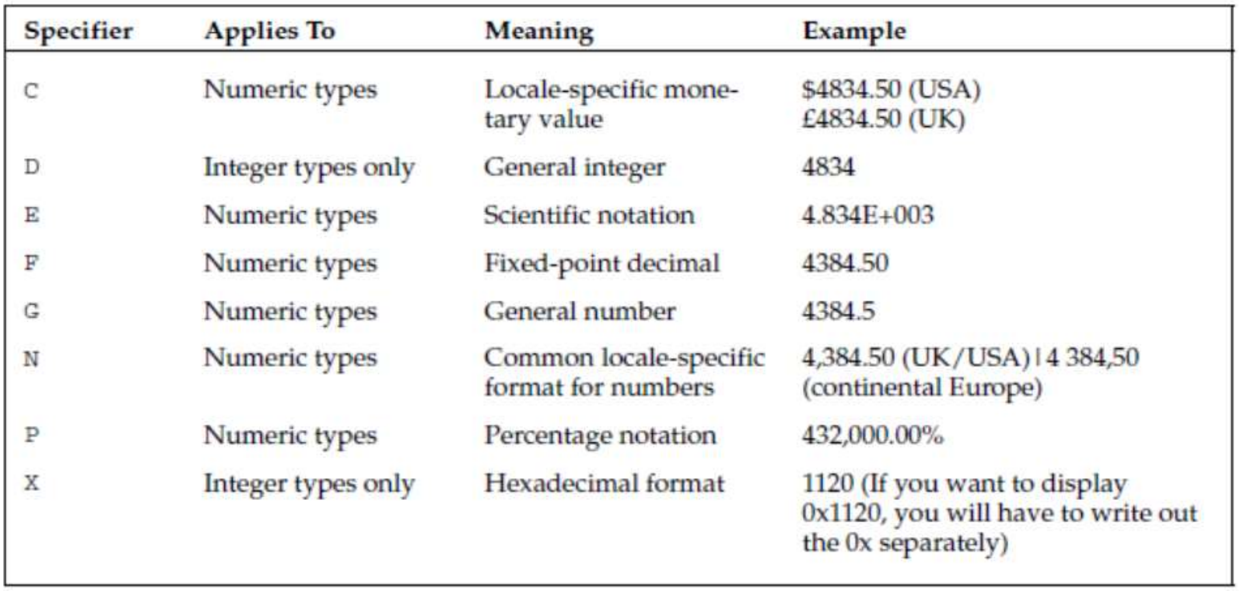
This method uses to format the input string into the specified format and then append it. This method also appends text to the end of the StringBuilder object.
For example,

```
using System;
using System.Text;
class Program
   {
        static void Main(string[] args)
        {
            string name = "John";
            int salary = 50000;
            StringBuilder sb1 = new StringBuilder();
            sb1.AppendFormat("Name is {0} and Salary is {1:C2}", name, salary);
            Console.WriteLine(sb1);
            Console.ReadKey();
        }
    }
```

Output:

Name is John and Salary is $50,000.00

Here, A format specifier, preceded by a colon (Ex: {1:C}). This indicates how you want the item to be formatted.



Specifier	Applies To	Meaning	Example
C	Numeric types	Locale-specific monetary value	$4834.50 (USA) £4834.50 (UK)
D	Integer types only	General integer	4834
E	Numeric types	Scientific notation	4.834E+003
F	Numeric types	Fixed-point decimal	4384.50
G	Numeric types	General number	4384.5
N	Numeric types	Common locale-specific format for numbers	4,384.50 (UK/USA) | 4 384,50 (continental Europe)
P	Numeric types	Percentage notation	432,000.00%
X	Integer types only	Hexadecimal format	1120 (If you want to display 0x1120, you will have to write out the 0x separately)

Insert(int index, string value) method

This method inserts the string at specified index in StringBuilder object.

For example,

```
using System;
using System.Text;
class Program
   {
          static void Main(string[] args)
          {
              StringBuilder sb1 = new StringBuilder("Apple");
              sb1.Insert(0, "I ");
              sb1.Insert(2, "like ");
              Console.WriteLine(sb1);
              Console.ReadKey();
```

```
        }
   }
```

Output:
I like Apple

Remove(int start, int length) Method

This method removes the specified number of characters from the current StringBuilder object.
The removing process beginning at a specified index and extends up to another index depends on the length
For example,

```
using System;
using System.Text;
class Program
   {
        static void Main(string[] args)
        {
            StringBuilder sb1 = new StringBuilder("abcdefghijkl");
            sb1.Remove(3, 4);
            Console.WriteLine(sb1);
            Console.ReadKey();
        }
   }
```

Output:
abchijkl

Replace(old_val, new_val) Method

This method is used to replace characters within the StringBuilder object with another specified character. For example,

```
using System;
using System.Text;
class Program
 {
   static void Main(string[] args)
   {
     StringBuilder sb1 = new StringBuilder("Welcome to cse");
     sb1.Replace('e','_');
```

```
        sb1.Replace("co", "TEST");
        Console.WriteLine(sb1);
        Console.ReadKey();
    }
}
```

Output:

```
W_ITESTm_ to cs_
```