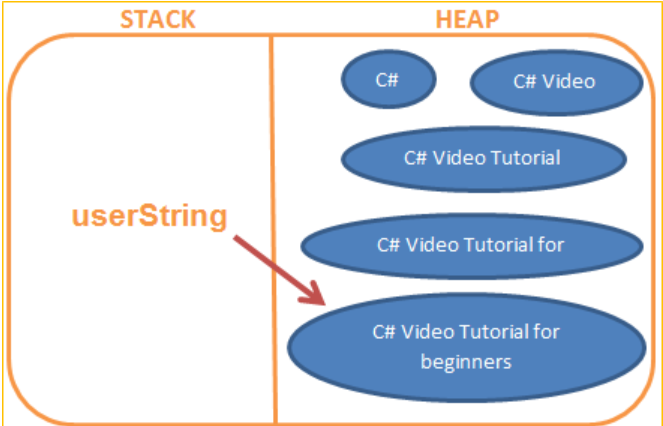
**Part 60 - C# Tutorial - difference between System.String and System.Text.StringBuilder**

Strings of type **StringBuilder**are mutable where as strings of type **System.String** are immutable.  As StringBuilder objects are mutable, they offer better performance than string objects of type System.String, when heavy string manipulation is involved.

**Let's understand the meaning of mutable and immutable strings with an example.**

using System;  
public class MainClass  
{  
    public static void Main()  
    {  
        string userString = "C#";  
        userString += " Video";  
        userString += " Tutorial";  
        userString += " for";  
        userString += " beginners";  
        Console.WriteLine(userString);  
    }  
}  
  
  
**In this example, userString variable is changed 5 times.**  
1. C#  
2. C# => C# Video  
3. C# Video => C# Video Tutorial  
4. C# Video Tutorial => C# Video Tutorial for  
5. C# Video Tutorial for => C# Video Tutorial for beginners  
  
  
Since, userString variable is of type System.String, and when we change this string 5 times, we end up with 5 string objects on the heap as shown in the diagram below. Immutable means, once a string object is created it cannot be changed, without creating another new string object. So in our example, When we initialize userString variable to **"C#"** we get one immutable string object on the heap. When we concatenate **" Video"** word to userString variable, the first created**"C#"**string object is orphaned(userString variable no longer points to this object). Now another new string object with words **"C# Video"** will be created to which the userString variable points to. So this process continues until, userString reference variable, points to the last string object (**C# Video Tutorial for beginners**), leaving the other 4 string onbjects on the heap(orphaned), until they are garbage collected, increasing the pressure on memory.



But on the other hand, StringBuilder string objects are mutable, meaning they can be changed inplace, without the need of creating another new StringBuilder object. The above example is rewritten using StringBuilder object.  
using System;  
using System.Text;  
namespace Pragim  
{  
    public class MainClass  
    {  
        public static void Main()  
        {  
            StringBuilder userStringBuilder =   
                new StringBuilder("C#");  
            userStringBuilder.Append(" Video");  
            userStringBuilder.Append(" Tutorial");  
            userStringBuilder.Append(" for");  
            userStringBuilder.Append(" beginners");  
            Console.WriteLine(userStringBuilder.ToString());  
        }  
    }  
}  
  
  
With StringBuilder, no matter how many times you manipulate a string, you will ever have only one instance.

Diagram

Description automatically generated

**So in brief, here are the differences between String and StringBuilderobjects.**  
1. Objects of type StringBuilder are mutable where as objects of type System.String are immutable.

2. As StringBuilder objects are mutable, they offer better performance than string objects of type System.String.  
3. StringBuilder class is present in System.Text namespace where String class is present in System namespace.  
  
  
Just imagine, the number of orphaned string objects that get created on the heap when you have a program as shown below.  
using System;  
namespace Pragim  
{  
    public class MainClass  
    {  
        public static void Main()  
        {  
            string strNumbers = string.Empty;  
            for (int i = 0; i < 1000; i++)  
            {  
                strNumbers += i.ToString() + " ";  
            }  
            Console.WriteLine(strNumbers);  
        }  
    }  
}