userString

STACK HEAP

string userString = "C#";

When you first initialize the string userString to C# userString reference variable get created in stack which will be pointing to the C# object initialy.

userString += " Video";

When we have this line what is gonna happen is the userString reference variable which is at the moment pointing to the C# object no longer points to the C# object it will insead create another new string object because a string object once created it’s immutable. That means you cannot change that object in memory there, instead when this line get executed what actually happens is C# object will remain in the memory and another new string object gets created in and C# object value will be copied over the newly build string object. Now you userString reference variable will pointing to the “C# Video”. And it goes on.

According to the above scenario in this example we will be creating 5 string objects and 4 are actually thrown away. They are no used at all. But they still remain in memory. Until the garbage collector in C# runs the memory will not be free.

Any time when you are manipulating string over and over again we will have so many object created.

When an object doesn’t have a reference variable pointing to it, it becomes garbage.

Immutable means : They cannot be changed once created.

userString

STACK HEAP

But string of type StringBuilder is mutable. Meaning no matter how many times you change your string builder object you wouldn’t get the new objects, the same object will be changed in memory.

using System;

using System.Text;

public class Program

{

public static void Main()

{

string userString = "C#";

userString += " Video";

userString += " Tutorial";

userString += " Pragim";

userString += " Tech";

Console.WriteLine(userString);

StringBuilder userString1 = new StringBuilder("C#");

userString1.Append(" Video");

userString1.Append(" Tutorial");

userString1.Append(" Pragim");

userString1.Append(" Tech");

Console.WriteLine(userString1.ToString());

Console.ReadLine();

}

}

