Selected files

3 printable files

02_STRING\02_STRING_EXAMPLE\02_STRING_EXAMPLE\IntegerNew.cs
02_STRING\02_STRING_EXAMPLE\02_STRING_EXAMPLE\Program.cs
02_STRING\02_STRING_EXAMPLE\02_STRING_EXAMPLE\StringNew.cs

02_STRING\02_STRING_EXAMPLE\02_STRING_EXAMPLE\IntegerNew.cs

```
1  using System;
2  using System.Collections.Generic;
3  using System.Diagnostics;
4  using System.Text;
5
6  namespace _02_STRING_EXAMPLE
7  {
8      class IntegerNew
9      {
10         public IntegerNew()
11         {
12             int ctr = 0;
13             Console.WriteLine("Loop Started");
14             var stopwatch = new Stopwatch();
15             stopwatch.Start();
16             for (int i = 0; i < 30000000; i++)
17             {
18                 ctr = ctr + 1;
19             }
20             stopwatch.Stop();
21             Console.WriteLine("Loop Ended");
22             Console.WriteLine("Loop Exceution Time in MS :" +
   stopwatch.ElapsedMilliseconds);
23         }
24     }
25 }
26
```

02_STRING\02_STRING_EXAMPLE\02_STRING_EXAMPLE\Program.cs

```
1  using System;
2  using System.Diagnostics;
3
4  namespace _02_STRING_EXAMPLE
5  {
6      class Program
7      {
8          /* What is String?
9           * In C#, the string is an object of the String class that represents a sequence of
   characters.
10          * We can perform many operations on strings such as concatenation, comparison,
   getting substring, search, trim, replacement, etc.
11          * */
12
13         /*
```

```
         * Before understanding strings are immutable, first, we need to understand two terms i.e. Mutable and Immutable.
         * Mutable means can be changed whereas Immutable means can not be changed. C# strings are immutable means C# strings cannot be changed.
         * */

        //string str = "DOTNET";
        //str = "TUTORIALS";
        /*
         * So, when the above two statements are executed, internally two memory locations are created.
         * When the first statement is executed, one object will be created which holds the value DotNet and that object will be referred
         * to by the str variable. When the second statement will be executed, another object will be created which holds the value
         * Tutorials and now the str variable will point to this newly created object. And the first object will be there and will be
         * available for garbage collection. So, the point that you need to remember is that each time, we assign a new value to the string variable,
         * a new object is created and that new object will be referred to by the string variable and older objects will be there for garbage collection
         * and this is the reason why said strings are immutable in C#.
         */

        //01. STRING EXAM
        static void Main(string[] args)
        {
            //Example For Everytime Create a New Object
            //It take Long time beacuse it will create everytime new object
            StringNew stringNew = new StringNew();

            //Example For Integer
            //It take small time because it will not create everytime new object
            IntegerNew integerNew = new IntegerNew();


            //Example For StringConcat
            //Here also take everytime new object to create
            StringConcat stringConcat = new StringConcat();

            //Example For StringBuilder
            StringBuilderExample stringBuilderExample = new StringBuilderExample();




            Console.ReadKey();
        }
    }
}

```

02_STRING\02_STRING_EXAMPLE\02_STRING_EXAMPLE\StringNew.cs

```
using System;
```

```
using System.Collections.Generic;
using System.Diagnostics;
using System.Text;

namespace _02_STRING_EXAMPLE
{
    //Example: String New Object
    class StringNew
    {
        public StringNew()
        {
            string str = "";
            Console.WriteLine("Loop Started");
            var stopwatch = new Stopwatch();
            stopwatch.Start();
            for (int i = 0; i < 30000000; i++)
            {
                str = Guid.NewGuid().ToString();    //create a new instance at everytime
            }
            stopwatch.Stop();
            Console.WriteLine("Loop Ended");
            Console.WriteLine("Loop Exceution Time in MS :" +
stopwatch.ElapsedMilliseconds);

        }
    }

    //Example: String with Same value in C#
    class StringSame
    {
        public StringSame()
        {
            string str = "";
            Console.WriteLine("Loop Started");
            var stopwatch = new Stopwatch();
            stopwatch.Start();
            for (int i = 0; i < 30000000; i++)
            {
                str = "DotNet Tutorials";
            }
            stopwatch.Stop();
            Console.WriteLine("Loop Ended");
            Console.WriteLine("Loop Exceution Time in MS :" +
stopwatch.ElapsedMilliseconds);

        }
    }

    //Example: String Concat
    //As We already Known everytimg concat string will create new object and old object
will be garbae collection.
    class StringConcat
    {
        public StringConcat()
```

```
        {
            string str = "";
            Console.WriteLine("Loop Started");
            var stopwatch = new Stopwatch();
            stopwatch.Start();
            for (int i = 0; i < 30000; i++)
            {
                str = "DotNet Tutorials" + str;
            }
            stopwatch.Stop();
            Console.WriteLine("Loop Ended");
            Console.WriteLine("Loop Exceution Time in MS :" +
stopwatch.ElapsedMilliseconds);
        }
    }

    //Example: String Builder
    class StringBuilderExample
    {
        public StringBuilderExample()
        {
            StringBuilder stringBuilder = new StringBuilder();
            Console.WriteLine("Loop Started");
            var stopwatch = new Stopwatch();
            stopwatch.Start();
            for (int i = 0; i < 30000; i++)
            {
                stringBuilder.Append("DotNet Tutorials");
            }
            stopwatch.Stop();
            Console.WriteLine("Loop Ended");
            Console.WriteLine("Loop Exceution Time in MS :" +
stopwatch.ElapsedMilliseconds);
        }
    }
}
```