New Features In .NET 7

* [](https://www.c-sharpcorner.com/members/munib-butt)

* [Munib Butt](https://www.c-sharpcorner.com/members/munib-butt)

* Nov 17, 2022

* 3.4k

* 0

* 2

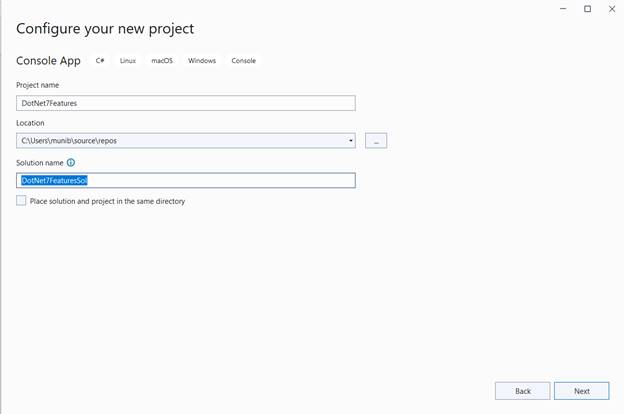
[Download Free .NET Java APIs for Office Documents](https://www.e-iceblue.com/download/free-products.html/?aff_id=100)

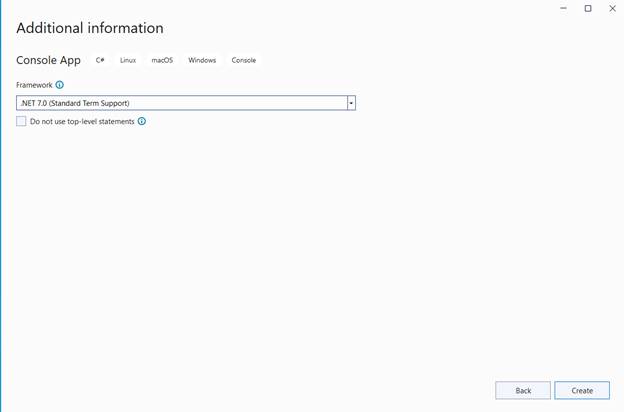
Introduction

Microsoft has just released .NET 7 on 14th November 2022. In this and the upcoming articles, we will look into some new features that have been introduced with C# in the latest version of .NET. Let us begin.

Improved way to handle string literals

Let us create a console application using Visual Studio 2022 Community edition.





Now, add the below code to the Program.cs file:

// Declare a string literal with quotes

//.NET 6

var firstValue = "<?xml version = \"1.0\"?>";

//.NET 6

var secondValue = @"<?xml version = ""1.0""?>";

//.NET 7

var thirdValue = """<?xml version = "1.0"?>""";

Console.WriteLine(firstValue);

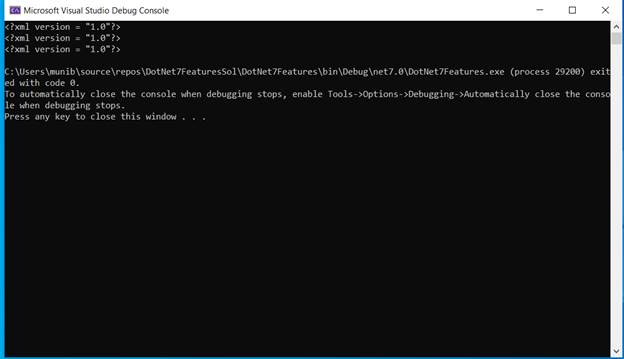
Console.WriteLine(secondValue);

Console.WriteLine(thirdValue);

C#

Copy

Run the application and you will see the below output,



Here, you see that the string literal with quotes has been displayed on the console three times. The first two methods were available in .NET 6 and earlier. These required some modifications to the text itself. However, in .NET 7 you do not need to make any updates within the string itself. Simply, adding three quotes makes the string useable. If your string contains three quotes, just change the prefix to four quotes. This simplifies the handling of XML and JSON type strings.

### **Microseconds and Nanoseconds to TimeStamp,**

### **DateTime, DateTimeOffset, and TimeOnly**

There are some improvements with DateTime where we have the ability get microsecond and nanosecond values in DateTime, which allows four more specific timing values coming out of the DateTime object.

namespace System

{

public struct DateTime

{

public DateTime(int year, int month, int day, int hour, int minute, int second, int millisecond, int microsecond);

public DateTime(int year, int month, int day, int hour, int minute, int second, int millisecond, int microsecond, System.DateTimeKind kind);

public DateTime(int year, int month, int day, int hour, int minute, int second, int millisecond, int microsecond, System.Globalization.Calendar calendar);

public int Microsecond { get; }

public int Nanosecond { get; }

public DateTime AddMicroseconds(double value);

}

public struct DateTimeOffset

{

public DateTimeOffset(int year, int month, int day, int hour, int minute, int second, int millisecond, int microsecond, System.TimeSpan offset);

public DateTimeOffset(int year, int month, int day, int hour, int minute, int second, int millisecond, int microsecond, System.TimeSpan offset, System.Globalization.Calendar calendar);

public int Microsecond { get; }

public int Nanosecond { get; }

public DateTimeOffset AddMicroseconds(double microseconds);

}

public struct TimeSpan

{

public const long TicksPerMicrosecond = 10L;

public const long NanosecondsPerTick = 100L;

public TimeSpan(int days, int hours, int minutes, int seconds, int milliseconds, int microseconds);

public int Microseconds { get; }

public int Nanoseconds { get; }

public double TotalMicroseconds { get; }

public double TotalNanoseconds { get; }

public static TimeSpan FromMicroseconds(double microseconds);

}

public struct TimeOnly

{

public TimeOnly(int hour, int minute, int second, int millisecond, int microsecond);

public int Microsecond { get; }

public int Nanosecond { get; }

}

}