

70-483 : Programming in C#

- Working with Dates -



Zouhair Rimale, Ph.D.

Expert technique .NET | SharePoint | ASP.NET MVC | WEB API | Angular

DateTime



Zouhair Rimale, Ph.D.

Expert technique .NET | SharePoint | ASP.NET MVC | WEB API | Angular

Program.cs

CSharpFundamentals.Program

Main(string[] args)

```
{  
    static void Main(string[] args)  
    {  
        var dateTime = new DateTime(2015, 1, 1);  
        var now = DateTime.Now;  
        var today = DateTime.Today;  
  
        //Console.WriteLine("Hour: " + now.Hour);  
        //Console.WriteLine("Minute: " + now.Minute);  
  
        var tomrrow = now.AddDays(1);  
        var yesterday = now.AddDays(-1);  
  
        Console.WriteLine(now.ToLongDateString());  
        Console.WriteLine(now.ToShortDateString());  
        Console.WriteLine(now.ToLongTimeString());  
        Console.WriteLine(now.ToShortTimeString());  
        Console.WriteLine(now.ToString("yyyy-MM-dd HH:mm"));  
    }  
}
```

Solution Explorer - CSharpFundamentals

Search Solution Explorer - CSharpFundamentals

Solution 'CSharpFundamentals' (1 project)

CSharpFundamentals

Properties

References

App.config

Program.cs

- MSDN Library
- .NET Development
- .NET Framework 4.6 RC and 4.5
- Development Guide
- Application Essentials
- Base Types
 - ▾ Formatting Types
 - Standard Numeric Format Strings
 - Custom Numeric Format Strings
 - Standard Date and Time Format Strings
 - Custom Date and Time Format Strings**
 - Standard TimeSpan Format Strings
 - Custom TimeSpan Format Strings
 - Enumeration Format Strings
 - Composite Formatting
 - Performing Formatting Operations

The following table describes the custom date and time format specifiers and displays a result string produced by each format specifier. By default, result strings reflect the formatting conventions of the en-culture. If a particular format specifier produces a localized result string, the example also notes the culture to which the result string applies. See the Notes section for additional information about using custom date and time format strings.

Format specifier	Description	Examples
"d"	The day of the month, from 1 through 31. More information: The "d" Custom Format Specifier.	2009-06-01T13:45:30 -> 1 2009-06-15T13:45:30 -> 15
"dd"	The day of the month, from 01 through 31. More information: The "dd" Custom Format Specifier.	2009-06-01T13:45:30 -> 01 2009-06-15T13:45:30 -> 15
"ddd"	The abbreviated name of the day of the week. More information: The "ddd" Custom Format Specifier.	2009-06-15T13:45:30 -> Mon (en-US) 2009-06-15T13:45:30 -> Пн (ru-RU) 2009-06-15T13:45:30 -> lun. (fr-FR)
"dddd"	The full name of the day of the week. More information: The "dddd" Custom Format Specifier.	2009-06-15T13:45:30 -> Monday (en-US) 2009-06-15T13:45:30 -> понедельник (ru-RU) 2009-06-15T13:45:30 -> lundi (fr-FR)
"f"	The tenths of a second in a date and time value. More information: The "f" Custom Format Specifier.	2009-06-15T13:45:30.6170000 -> 6 2009-06-15T13:45:30.05 -> 0
"ff"	The hundredths of a second in a date and time value. More information: The "ff" Custom Format Specifier.	2009-06-15T13:45:30.6170000 -> 61 2009-06-15T13:45:30.0500000 -> 00
"fff"	The milliseconds in a date and time value. More information: The "fff" Custom Format Specifier.	6/15/2009 13:45:30.617 -> 617 6/15/2009 13:45:30.0005 -> 000
"ffff"	The ten thousandths of a second in a date and time value.	2009-06-15T13:45:30.6175000 -> 6175

TimeSpan



Zouhair Rimale, Ph.D.

Expert technique .NET | SharePoint | ASP.NET MVC | WEB API | Angular

Program.cs

CSharpFundamentals.Program

Main(string[] args)

```
using System;

namespace CSharpFundamentals
{
    class Program
    {
        static void Main(string[] args)
        {
            // Creating
            var timeSpan = new TimeSpan(1, 2, 3);

            var timeSpan1 = new TimeSpan(1, 0, 0);
            var timeSpan2 = TimeSpan.FromHours(1);

            var start = DateTime.Now;
            var end = DateTime.Now.AddMinutes(2);
            var duration = end - start;
            Console.WriteLine("Duration: " + duration);

            // Properties
            Console.WriteLine("Minutes: " + timeSpan.Minutes);
            Console.WriteLine("Total Minutes: " + timeSpan.TotalMinutes);

            // Add
            Console.WriteLine("Add Example: " + timeSpan.Add(TimeSpan.FromMinutes(8)));
            Console.WriteLine("Subtract Example: " + timeSpan.Subtract(TimeSpan.FromMinutes(2)))
        }
    }
}
```

Solution Explorer - CSharpFundamentals

Search Solution Explorer - CSharpFundamentals

Solution 'CSharpFundamentals' (1 project)

CSharpFundamentals

- Properties
- References
- App.config
- Program.cs

