





**TIMER CLASS**

# TIMER CLASS



- Generates an event after a set interval, with an option to generate recurring events.

## Constructors

	Name	Description
	<code>Timer()</code>	Initializes a new instance of the Timer class, and sets all the properties to their initial values.
	<code>Timer(Double )</code>	Initializes a new instance of the Timer class, and sets the <code>Interval</code> property to the specified number of milliseconds.

# TIMER EVENTS

## Events

	Name	Description
	<a href="#">Disposed</a>	Occurs when the component is disposed by a call to the <a href="#">Dispose</a> method. (Inherited from <a href="#">Component</a> .)
	<a href="#">Elapsed</a>	Occurs when the interval elapses.

# TIMER CLASS

- The Timer component is a server-based timer that raises an [Elapsed](#) event in your application after the number of milliseconds in the [Interval](#) property has elapsed.
- You can configure the Timer object to raise the event just once or repeatedly using the [AutoReset](#) property.
- Typically, a Timer object is declared at the class level so that it stays in scope as long as it is needed.
- You can then handle its [Elapsed](#) event to provide regular processing. For example, suppose you have a critical server that must be kept running 24 hours a day, 7 days a week. You could create a service that uses a Timer object to periodically check the server and ensure that the system is up and running. If the system is not responding, the service could attempt to restart the server or notify an administrator.
- Source Code
- <https://referencesource.microsoft.com/#System/services/timers/system/timers/Timer.cs,897683f27faba082>

```

using System;
using System.Timers;

public class Example
{
    private static System.Timers.Timer aTimer;

    public static void Main()
    {
        SetTimer();

        Console.WriteLine("\nPress the Enter key to exit the application...\n");
        Console.WriteLine("The application started at {0:HH:mm:ss.fff}", DateTime.Now);
        Console.ReadLine();
        aTimer.Stop();
        aTimer.Dispose();

        Console.WriteLine("Terminating the application...");
    }

    private static void SetTimer()
    {
        // Create a timer with a two second interval.
        aTimer = new System.Timers.Timer(2000);
        // Hook up the Elapsed event for the timer.
        aTimer.Elapsed += OnTimedEvent;
        aTimer.AutoReset = true;
        aTimer.Enabled = true;
    }

    private static void OnTimedEvent(Object source, ElapsedEventArgs e)
    {
        Console.WriteLine("The Elapsed event was raised at {0:HH:mm:ss.fff}",
            e.SignalTime);
    }
}

// The example displays output like the following:
//     Press the Enter key to exit the application...
//
//     The application started at 09:40:29.068
//     The Elapsed event was raised at 09:40:31.084
//     The Elapsed event was raised at 09:40:33.100
//     The Elapsed event was raised at 09:40:35.100
//     The Elapsed event was raised at 09:40:37.116
//     The Elapsed event was raised at 09:40:39.116
//     The Elapsed event was raised at 09:40:41.117
//     The Elapsed event was raised at 09:40:43.132
//     The Elapsed event was raised at 09:40:45.133
//     The Elapsed event was raised at 09:40:47.148
//
//     Terminating the application...

```