

[-] Collapse All [+] Code: All

## Timer Control Overview

[See Also](#) [Send Feedback](#)

The ASP.NET AJAX [Timer](#) control performs postbacks at defined intervals. If you use the [Timer](#) control with an [UpdatePanel](#) control, you can enable partial-page updates at a defined interval. You can also use the [Timer](#) control to post the whole page.

This topic contains the following sections:

- [Scenarios](#)
- [Background](#)
- [Code Examples](#)
- [Class Reference](#)

### [-] Timer Control Scenarios

You use the [Timer](#) control when you want to do the following:

- Periodically update the contents of one or more [UpdatePanel](#) controls without refreshing the whole Web page.
- Run code on the server every time that a [Timer](#) control causes a postback.
- Synchronously post the whole Web page to the Web server at defined intervals.

### [-] Background

The [Timer](#) control is a server control that embeds a JavaScript component into the Web page. The JavaScript component initiates the postback from the browser when the interval that is defined in the [Interval](#) property has elapsed. You set the properties for the [Timer](#) control in code that runs on the server and those properties are passed to the JavaScript component.

An instance of the [ScriptManager](#) class must be included in the Web page when you use the [Timer](#) control.

When a postback was initiated by the [Timer](#) control, the [Timer](#) control raises the [Tick](#) event on the server. You can create an event handler for the [Tick](#) event to perform actions when the page is posted to the server.

Set the [Interval](#) property to specify how often postbacks will occur, and set the [Enabled](#) property to turn the [Timer](#) on or off. The [Interval](#) property is defined in milliseconds and has a default value of 60,000 milliseconds, or 60 seconds.

#### Note:

Setting the [Interval](#) property of a [Timer](#) control to a small value can generate significant traffic to the Web server. Use the [Timer](#) control to refresh the content only as often as necessary.

You can include more than one [Timer](#) control on a Web page if different [UpdatePanel](#) controls must be updated at different intervals. Alternatively, a single instance of the [Timer](#) control can be the trigger for more than one [UpdatePanel](#) control in a Web page.

### Using a Timer Control Inside an UpdatePanel Control

When the [Timer](#) control is included inside an [UpdatePanel](#) control, the [Timer](#) control automatically works as a trigger for the [UpdatePanel](#) control. You can override this behavior by setting the [ChildrenAsTriggers](#) property of the [UpdatePanel](#) control to **false**.

For [Timer](#) controls inside an [UpdatePanel](#) control, the JavaScript timing component is re-created only when each postback finishes. Therefore, the timed interval does not start until the page returns from the postback. For instance, if the [Interval](#) property is set to 60,000 milliseconds (60 seconds) but the postback takes 3 seconds to complete, the next postback will occur 63 seconds after the previous postback.

The following example shows how to include a [Timer](#) control inside an [UpdatePanel](#) control.

 [Copy Code](#)

```
<asp:ScriptManager runat="server" id="ScriptManager1" />
<asp:UpdatePanel runat="server" id="UpdatePanel1"
  UpdateMode="Conditional">
  <contenttemplate>
    <asp:Timer id="Timer1" runat="server"
      Interval="120000"
      OnTick="Timer1_Tick">
    </asp:Timer>
  </contenttemplate>
</asp:UpdatePanel>
```

### Using a Timer Control Outside an UpdatePanel Control

When the [Timer](#) control is outside an [UpdatePanel](#) control, you must explicitly define the [Timer](#) control as a trigger for the [UpdatePanel](#) control to be updated.

If the [Timer](#) controls is outside an [UpdatePanel](#) control, the JavaScript timing component continues to run as the postback is being processed. For example, if the [Interval](#) property is set to 60,000 milliseconds (60 seconds) and the postback takes 3 seconds to complete, the next postback will occur 60 seconds after the previous postback. The user will see the refreshed content in the [UpdatePanel](#) control for only 57 seconds.

You must set the [Interval](#) property to a value that enables one asynchronous postback to complete before the next postback is initiated. If a new postback is initiated while an earlier postback is being processed, the first postback is canceled.

The following example shows how to use the [Timer](#) control outside an [UpdatePanel](#) control.

 [Copy Code](#)

```
<asp:ScriptManager runat="server" id="ScriptManager1" />
<asp:Timer ID="Timer1" runat="server" Interval="120000"
  OnTick="Timer1_Tick">
</asp:Timer>
<asp:UpdatePanel ID="UpdatePanel1" runat="server">
  <Triggers>
    <asp:AsyncPostBackTrigger ControlID="Timer1"
      EventName="Tick" />
  </Triggers>
  <ContentTemplate>
    <asp:Label ID="Label1" runat="server" ></asp:Label>
  </ContentTemplate>
</asp:UpdatePanel>
```

### [-] Code Examples

The following example shows an [UpdatePanel](#) control that displays a randomly generated stock price and the time that the stock price was generated. By default, the [Timer](#) control updates the content in the [UpdatePanel](#) every 10 seconds. The user can decide to update the stock price every 10 seconds, every 60 seconds, or not at all. When the user chooses not to update the stock price, the [Enabled](#) property is set to **false**.

#### Visual Basic

```
<%@ Page Language="VB" AutoEventWireup="true" %>

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">

<html xmlns="http://www.w3.org/1999/xhtml" >
```

```

<head id="Head1" runat="server">
  <title>Timer Example Page</title>
  <script runat="server">
    Protected Sub Page_Load(ByVal sender As Object, ByVal e As EventArgs)
      OriginalTime.Text = DateTime.Now.ToLongTimeString()
    End Sub

    Protected Sub Timer1_Tick(ByVal sender As Object, ByVal e As EventArgs)
      StockPrice.Text = GetStockPrice()
      TimeOfPrice.Text = DateTime.Now.ToLongTimeString()
    End Sub

    Private Function GetStockPrice() As String
      Dim randomStockPrice As Double = 50 + New Random().NextDouble()
      Return randomStockPrice.ToString("C")
    End Function

    Protected Sub RadioButton1_CheckedChanged(ByVal sender As Object, ByVal e As System.EventArgs)
      Timer1.Interval = 10000
      Timer1.Enabled = True
    End Sub

    Protected Sub RadioButton2_CheckedChanged(ByVal sender As Object, ByVal e As System.EventArgs)
      Timer1.Interval = 60000
      Timer1.Enabled = True
    End Sub

    Protected Sub RadioButton3_CheckedChanged(ByVal sender As Object, ByVal e As System.EventArgs)
      Timer1.Enabled = False
    End Sub
  </script>
</head>
<body>
  <form id="form1" runat="server">
    <asp:ScriptManager ID="ScriptManager1" runat="server" />
    <asp:Timer ID="Timer1" OnTick="Timer1_Tick" runat="server" Interval="10000" />

    <asp:UpdatePanel ID="StockPricePanel" runat="server" UpdateMode="Conditional">
      <Triggers>
        <asp:AsyncPostBackTrigger ControlID="Timer1" />
      </Triggers>
      <ContentTemplate>
        Stock price is <asp:Label id="StockPrice" runat="server"></asp:Label><BR />
        as of <asp:Label id="TimeOfPrice" runat="server"></asp:Label>
      </ContentTemplate>
    </asp:UpdatePanel>
    <div>
      <asp:RadioButton ID="RadioButton1" AutoPostBack="true" GroupName="TimerFrequency" runat="server" Text="10 seconds" OnCheckedChanged="Radio
      <asp:RadioButton ID="RadioButton2" AutoPostBack="true" GroupName="TimerFrequency" runat="server" Text="60 seconds" OnCheckedChanged="Radio
      <asp:RadioButton ID="RadioButton3" AutoPostBack="true" GroupName="TimerFrequency" runat="server" Text="Never" OnCheckedChanged="RadioButt
      <br />
      Page originally created at <asp:Label ID="OriginalTime" runat="server"></asp:Label>
    </div>
  </form>
</body>
</html>

```

**C#**

```

<%@ Page Language="C#" AutoEventWireup="true" %>

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.1//EN" "http://www.w3.org/TR/xhtml11/DTD/xhtml11.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
<head id="Head1" runat="server">
  <title>Timer Example Page</title>
  <script runat="server">
    protected void Page_Load(object sender, EventArgs e)
    {
      OriginalTime.Text = DateTime.Now.ToLongTimeString();
    }

    protected void Timer1_Tick(object sender, EventArgs e)
    {
      StockPrice.Text = GetStockPrice();
      TimeOfPrice.Text = DateTime.Now.ToLongTimeString();
    }

    private string GetStockPrice()
    {
      double randomStockPrice = 50 + new Random().NextDouble();
      return randomStockPrice.ToString("C");
    }

    protected void RadioButton1_CheckedChanged(object sender, EventArgs e)
    {
      Timer1.Enabled = true;
      Timer1.Interval = 10000;
    }

    protected void RadioButton2_CheckedChanged(object sender, EventArgs e)
    {
      Timer1.Enabled = true;
      Timer1.Interval = 60000;
    }

    protected void RadioButton3_CheckedChanged(object sender, EventArgs e)
    {
      Timer1.Enabled = false;
    }
  </script>
</head>
<body>
  <form id="form1" runat="server">
    <asp:ScriptManager ID="ScriptManager1" runat="server" />
    <asp:Timer ID="Timer1" OnTick="Timer1_Tick" runat="server" Interval="10000" />

    <asp:UpdatePanel ID="StockPricePanel" runat="server" UpdateMode="Conditional">
      <Triggers>
        <asp:AsyncPostBackTrigger ControlID="Timer1" />
      </Triggers>
      <ContentTemplate>
        Stock price is <asp:Label id="StockPrice" runat="server"></asp:Label><BR />

```

```
        as of <asp:Label id="TimeOfPrice" runat="server"></asp:Label>
        <br />

    </ContentTemplate>
</asp:UpdatePanel>
<div>
<br />
Update stock price every:<br />
<asp:RadioButton ID="RadioButton1" AutoPostBack="true" GroupName="TimerFrequency" runat="server" Text="10 seconds" OnCheckedChanged="RadioButt
<asp:RadioButton ID="RadioButton2" AutoPostBack="true" GroupName="TimerFrequency" runat="server" Text="60 seconds" OnCheckedChanged="RadioButt
<asp:RadioButton ID="RadioButton3" AutoPostBack="true" GroupName="TimerFrequency" runat="server" Text="Never" OnCheckedChanged="RadioButton3_C
<br />
Page loaded at <asp:Label ID="OriginalTime" runat="server"></asp:Label>
</div>
</form>
</body>
</html>
```

---

## Tutorials

[Walkthrough: Introduction to the Timer Control](#)

[Walkthrough: Using the ASP.NET Timer Control with Multiple UpdatePanel Controls](#)

## Class Reference

The key server classes for the [Timer](#) control are shown in the following table.

[Timer](#)

Performs asynchronous or synchronous Web page postbacks at a defined interval.

## See Also

### Concepts

[Partial-Page Rendering Overview](#)

[UpdatePanel Control Overview](#)

---

Send [feedback](#) on this topic to Microsoft.