Lab 11: SLA Manager

Lab Objective: Build an add-on to you System Tracker page that displays Service Level Agreement information, as well as an escalation procedure for down servers. This lab will focus on implementing AJAX update panels, and coding. See mine

This a “Stretch” assignment meaning there will be portions of the programming requirements of the assignment that you’ll have to figure out. Give it your best try, and feel free to ask questions.

Due end of class 4/9. 25pts.

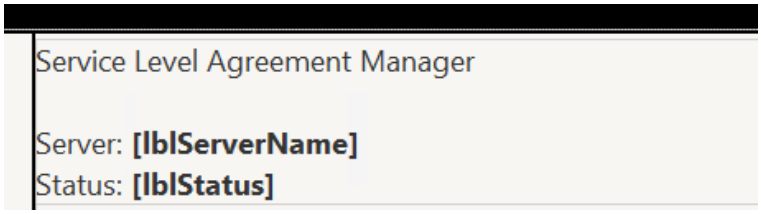
Point Breakdown = 19 Points to complete the assignment as is

+2 points for each of the three programming challenges indicated by the (**PROGRAMMING CHALLENGE**) tag

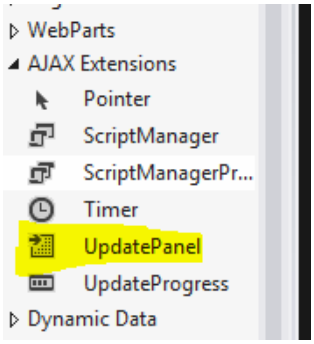
1: Layout

1. **Open** your System Tracker page
2. **Insert** a one row, two column, **table** below the existing datagrid
3. **Move** your existing **FormView** into the left cell
4. In the right cell, Type “**Service Level Agreement Manager**” at the top.
5. On the following lines, **add** two lines of **text**
    a. “Server:”
    b. “Status:”

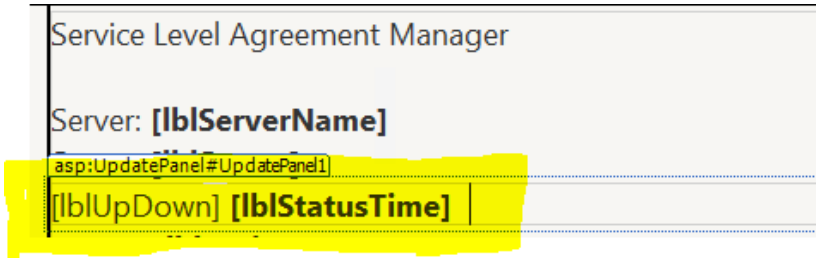
6. **Add** a **label** behind each new line. Call the labels lblServerName, and lblStatus respectively.
7. **Blank** the **text** out of each label.



8. On the next line add an **UpdatePanel** from the AJAX Extensions section of your server controls. **NOTE**: IF you elected not to use the default Visual Studio MasterPage, you must add a **ScriptManager** Control, from the AJAX extensions toolbox, to the top of the page.

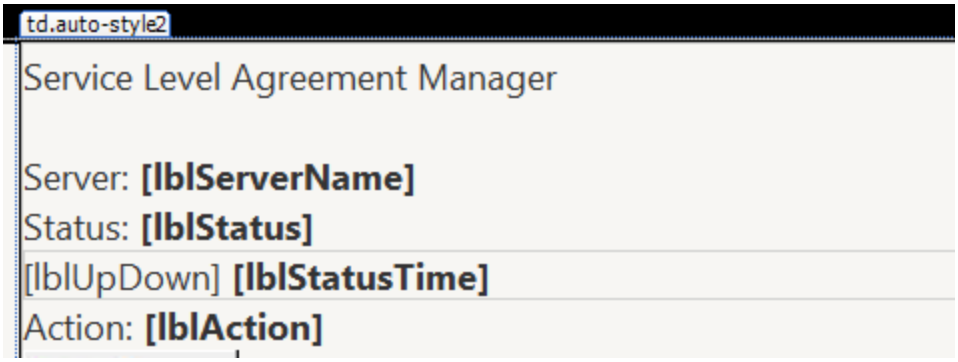


9. **Add** two **label** controls inside the UpdatePanel.
    a. Call the first one **lblUpDown**
    b. Call the second one **lblStatusTime**
10. **Blank** the **text** of each



11. Below, but not inside, the Update panel, **type** out the **text** “Action:”

12. **Add** a **label** control after the text, **change** the **ID** to lblAction. **Blank** the **text**

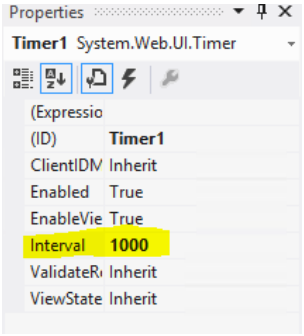


13. Below the text, **drag** a **Timer** control from the AJAX Extensions toolbox

2: Configure the Timer Control

14. **Select** the **Timer** Control

15. **Change** the **Interval** to 1000 (That’s 1000 milliseconds)



16. **Select** the **Events** button (The lightning bolt).
17. **Double-click** inside the field next to the **Tick** event. This will take you to the event handler code-behind for the Tick event.
18. Add the following lines of code

```
Dim DownSecs = DateDiff(DateInterval.Second, Session("Status_Change"), Date.Now)
Dim DownMins = DateDiff(DateInterval.Minute, Session("Status_Change"), Date.Now)
Dim DownHours = DateDiff(DateInterval.Hour, Session("Status_Change"), Date.Now)

If DownHours >= 1 Then
        lblStatusTime.Text = DownHours & ":" & (DownMins - (DownHours * 60)) & ":" &
((DownSecs) - (DownMins * 60))
Else
    lblStatusTime.Text = DownSecs

End If
```

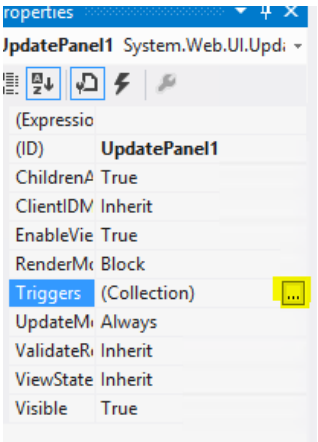
19. (**PROGRAMMING CHALLENGE**) **Modify** the IF Then **code** to account for a condition where the time since the last status change is between 60 seconds and 60 minutes and set the lblStatusTime.text accordingly.
20. While you're in the code-behind, Create a new Sub called **GetServerData.** Paste in the following code

```
Dim conn As SqlConnection
Dim cmd1, cmd2 As SqlCommand

Dim cmdString1 As String = "Select Status_Change FROM Servers WHERE Servers.ID = " &
GridView1.SelectedValue & ""
```

```
conn = New SqlConnection("Data Source=yourserverName\INSTANCENAME;Initial
Catalog=yourdatabaseName; User ID=sa; Password=yourSApassword")

"the previous command is should be on one line.  Replace the info in green with your database
information.

 cmd1 = New SqlCommand(cmdString1, conn)
 conn.Open()
 Dim dr As SqlDataReader
 dr = cmd1.ExecuteReader

  While dr.Read
     Session("Status_Change") = dr(0)
   End While
 conn.Close()
```

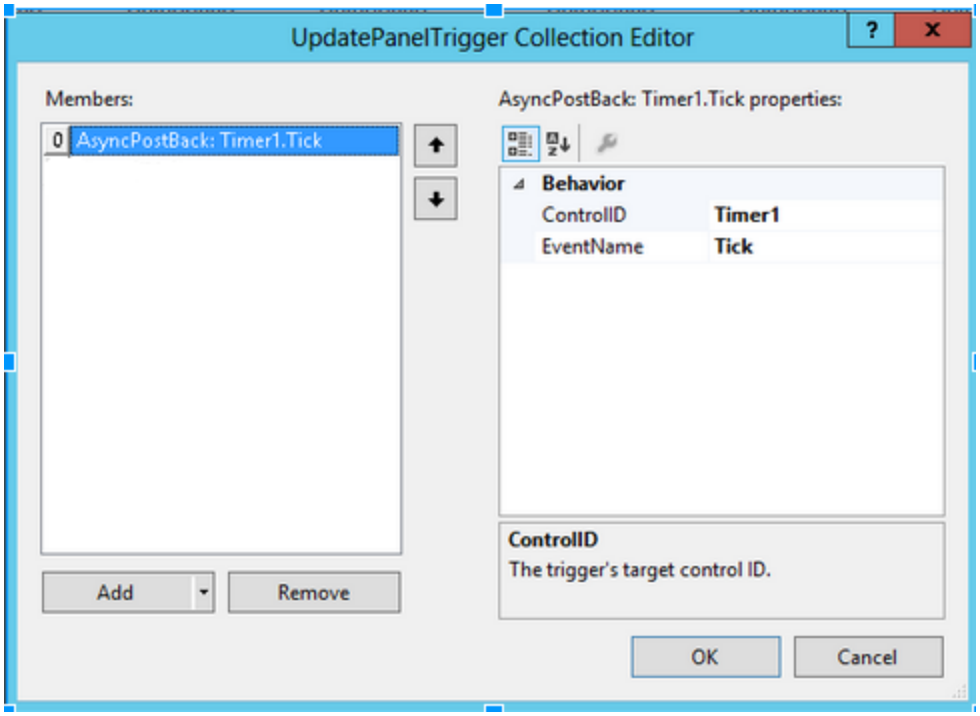
21. Back in the Design View (.aspx) page. **Select** the **updatepanel** you added in step 8.

22. **Click** the collection box in the **Triggers** attribute in the properties menu.



23. Select **Add**, to add an **AsyncPostBack**

24. **Select Timer1** for the ControlID, and **Tick** as the EventName. OK.



25. **Double-Click** on your Servers **Gridview**. This will take you the event handler code-behind.
26. Add this code to the event handler code-behind

```
GetServerData()
```

27. Save All.
28. **Visit** your System Tracker **page** on the web, select a Gridview Record. You should see the Timer start ticking, representing the time since the last status change. **NOTE**: For this to work this requires that you have changed the status in your System Tracker formview at least once

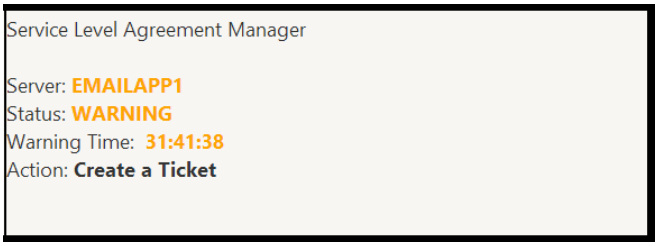
3: Change the text color based on Server Status

29. In the GridView1_SelectedIndexChanged event handler, add the following code

```
Dim R As GridViewRow
 R = GridView1.SelectedRow
```

```
lblServerName.Text = R.Cells(2).Text
lblStatus.Text = R.Cells(7).Text

If R.Cells(7).Text = "UP" Then
   lblServerName.ForeColor = Drawing.Color.Green
   lblStatus.ForeColor = Drawing.Color.Green
   lblStatusTime.ForeColor = Drawing.Color.Green
   lblUpDown.Text = "Up Time: "
   lblAction.Text = "Relax"
End If
```

30. (**PROGRAMMING CHALLENGE**) **Modify** the IF Then **code** to change the label colors for server DOWN and WARNING status. Down should be red, and Yellow should be Orange. lblUpDown should read “Down time” for a DOWN status, and “Warning time” for a WARNING status. Set the action for Warning to “Create a Ticket” See the following examples.



31. Create a new Sub called **DownActions**

32. Paste the following code:

```
Dim cmd2 As SqlCommand
    Dim conn As SqlConnection
    conn = New SqlConnection("Data Source=yourserverName\INSTANCENAME;Initial
Catalog=yourdatabaseName; User ID=sa; Password=yourSApassword")
```

```
"the previous command is should be on one line.  Replace the info in green with your database information.

        Dim getEMPLID1 As String = "SELECT Employee.Phone, Employee.Email, Employee.Last, Employee.First FROM Servers INNER JOIN Support " & _
    "ON Servers.SupportID = Support.SUPPORT_ID INNER JOIN Call_List ON Support.CALL_LIST_ID = Call_List.CALL_LIST_ID INNER JOIN Employee " &
    "ON Call_List.EmplID1 = Employee.EmplID Where Servers.ID=" & GridView1.SelectedValue & ""

        cmd2 = New SqlCommand(getEMPLID1, conn)

        conn.Open()
        Dim dr As SqlDataReader
        dr = cmd2.ExecuteReader

        dr.Read()
        Session("First") = Convert.ToString(dr("First"))
        Session("Last") = Convert.ToString(dr("Last"))
        Session("Email") = Convert.ToString(dr("Email"))
        Session("Phone") = Convert.ToString(dr("Phone"))

        conn.Close()

        lblAction.Text = "Contact: " & Session("First") & " " & Session("Last") & ", " & Session("Phone") & ", " & Session("Email")
```

33. **Call** the **DownActions** sub from inside the IfElse statement that you created in step 30, which handles the actions when a server status is DOWN.
34. Save and visit your site on the web. You should be able to:
    a. Select a server from the gridview and have the color of the text to reflect the color of the server status.
    b. See a timer indicating the time since the last status change.
    c. Get an initial course of action for each server status. Up = Relax, Warning = Create Ticket, Down = Call Primary contact.

4: Create an escalation policy

35. (**PROGRAMMING CHALLENGE**) When a server is down, depending on the acceptable downtime for that server, the following actions need to be expressed in your SLA manager

---

If Server has been down 1/4 of acceptable downtime, or less: **Call Primary support person.** (For 4 hour OK down time that's 1 hour, for 8 hour OK downtime that's 2 hours, etc)

If Server has been down ½ of acceptable down time: **Call Secondary Support Person**.

If Server has been down for ¾ of acceptable down time: **Call Third Level Support Person**.

If Server has been down for over acceptable down time: **Call Operations Manager**.

NOTES:

1. You won't have an Operations Manager defined in your database so you can hardcode that condition.
2. You'll apply this logic in the DownActions Sub
3. The DownActions Sub has most of what you need to work with.
4. Here is a starter. SNIPPET PENDING