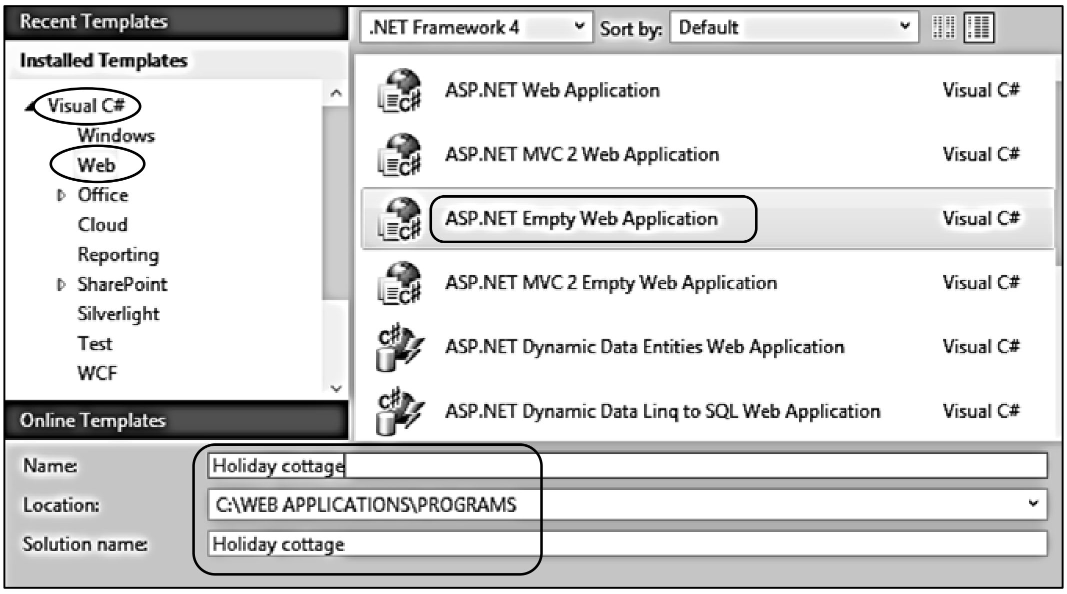
# 7 Holiday cottage bookings

This project examines the use of the ***Calendar*** component in C#.ASP. This is an extremely useful and flexible way to input and display date information such as bookings. Its many facilities and options make it complex to configure and program, but the result is well worth the effort.

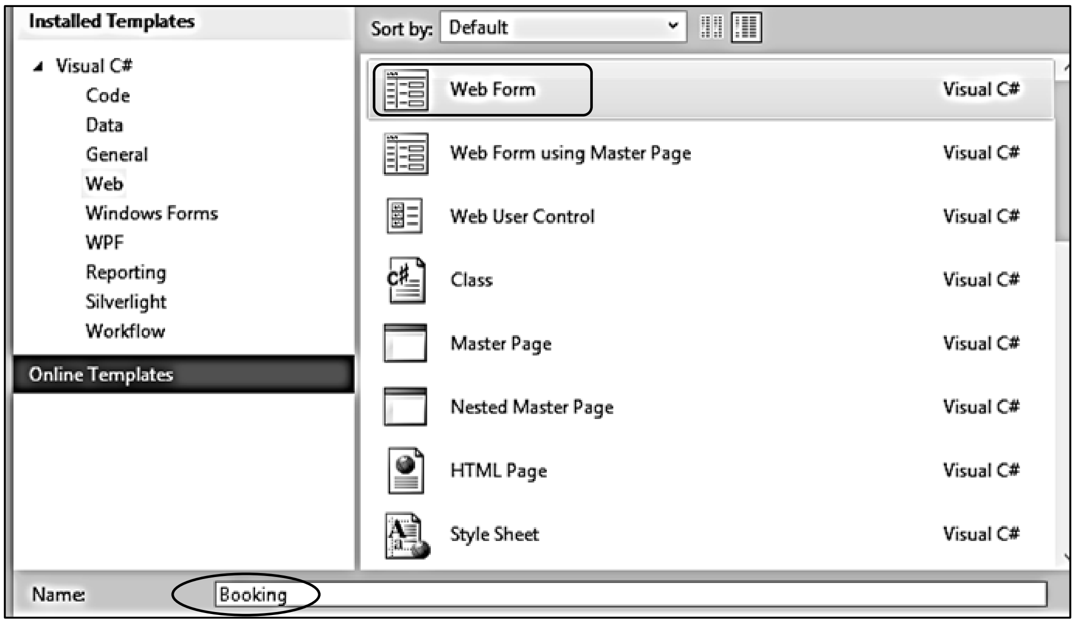
We will set up a web page which will form part of a booking system for a holiday cottage. We will concentrate on using the ***Calendar*** function to display the dates when the cottage is available, and to allow the customer to interactively select a week when they wish to stay in the cottage.

Open ***Visual Studio*** and select ***New Project***. Choose ***Visual C# / Web*** as the application type, and click on '***ASP.NET Empty Web Application***'. Give the project name '**Holiday cottage**' and select a folder location for the project.


Recent Templates
Installed Templates
Visual C#
Windows
Web
Office
Cloud
Reporting
SharePoint
Silverlight
Test
WCF
Online Templates
.NET Framework 4
Sort by: Default
ASP.NET Web Application Visual C#
ASP.NET MVC 2 Web Application Visual C#
ASP.NET Empty Web Application Visual C#
ASP.NET MVC 2 Empty Web Application Visual C#
ASP.NET Dynamic Data Entities Web Application Visual C#
ASP.NET Dynamic Data Linq to SQL Web Application Visual C#

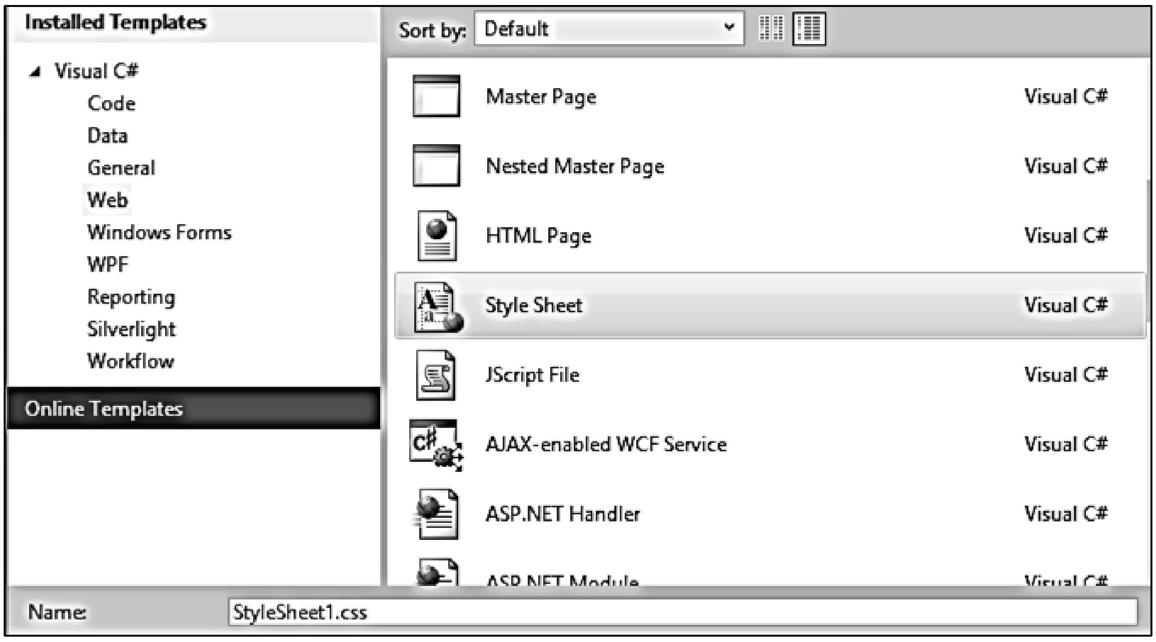
Name: Holiday cottage
Location: C:\WEB APPLICATIONS\PROGRAMS
Solution name: Holiday cottage

Go to the Solution Explorer window and right click the **Holiday cottage** project icon and select **Add / New item**. Click on the ***Web Form*** option, and give the name '**Booking**'.



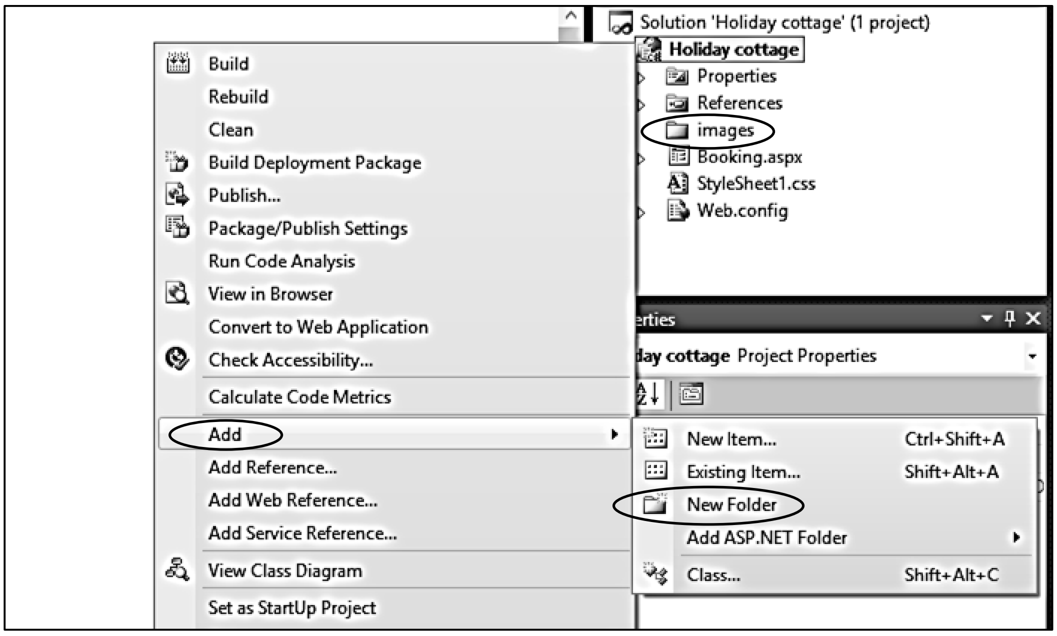
Name: Booking

A style sheet will also be needed to apply formatting to the page. Again right click the **Holiday cottage** project icon and select **Add / New item**. Click on ***Style Sheet***, and accept the name '***StyleSheet1***'.



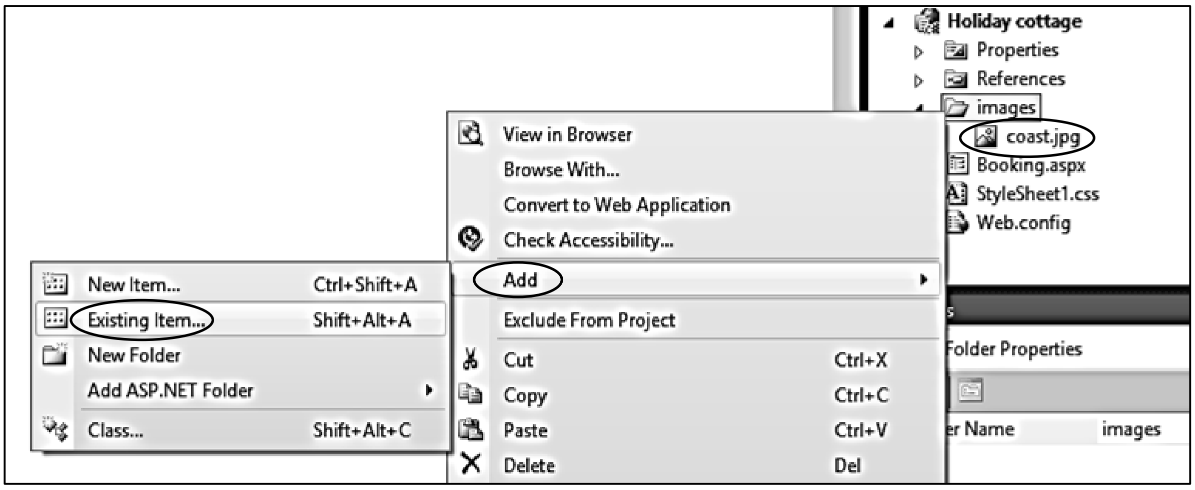
We will use several picture images on the page, so it will be convenient to create a folder in the project to store these. Right click the ***Holiday cottage*** program icon and select ***Add / New Folder***.

A folder will be created in the Solution Explorer window. Rename this as '***images***'.



It would be good to use a coastal view, perhaps of Pembrokeshire or the Lleyn Peninsula, as a background to the page. Use the Internet to find a suitable picture.

Right click the ***images*** folder icon. Select ***Add / Existing Item***, and upload the photograph file.



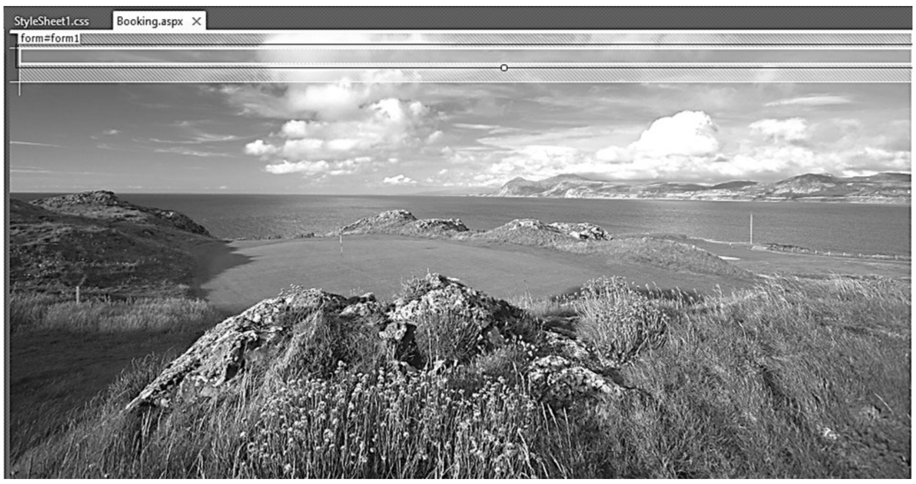
Open the ***StyleSheet1*** page and add code to the ***body*** tag to set the font style and display the background image.

```
body
{
    font-family: Arial, Helvetica, sans-serif;
    background-image:url('images/coast.jpg');
    background-repeat:no-repeat;
    background-size: 100%;
}
```

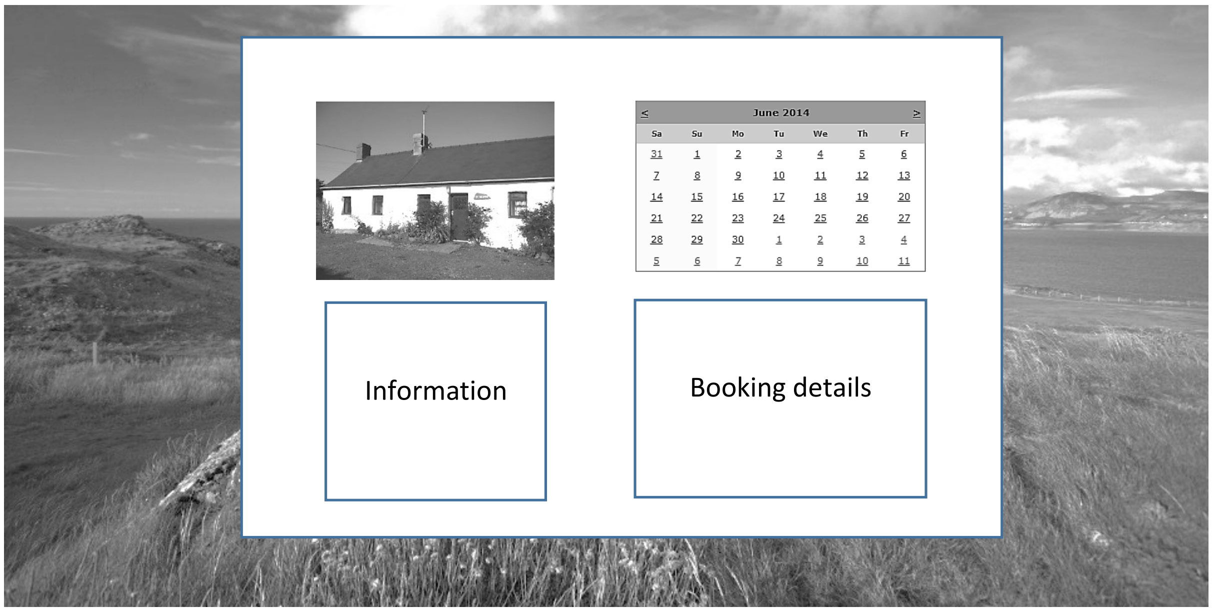
Go now to the Booking.aspx page and add lines to the <head> section of the HTML code to display a title on the page tag and link to the style sheet.

```
<head runat="server">
    <link rel="Stylesheet" type="text/css" href="StyleSheet1.css" />
    <title>Bay View Cottage</title>
</head>
<body>
```

Use the Design button to go to the page layout screen. Check that the background image is displayed.



We can now turn our attention to the layout of the web page. This will contain a panel divided into two columns. The left column will display a photograph of the cottage and text information, whilst the right column will have a calendar and text boxes for the customer to enter details of their booking.



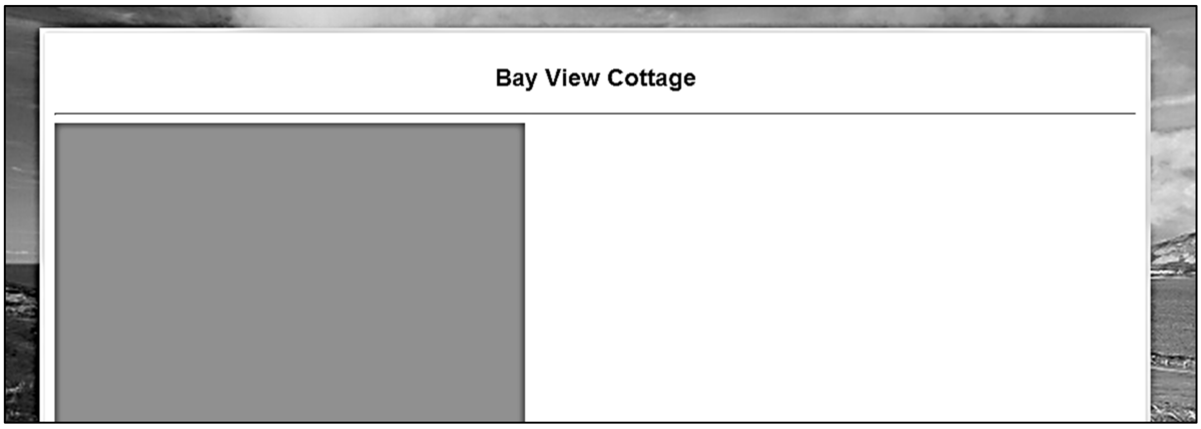
We will begin by creating a ***division*** for the panel, with two further ***divisions*** inside this to represent the two columns of the page layout. Go to the HTML code screen ***Booking.aspx*** and add the divisions within the ***<body>*** section. We will also add a heading '**Bay View Cottag**e'. The tag ***<hr>*** creates a horizontal ruled line below the heading.

```
<body>
   <form id="form1" runat="server">
   <div id="content">
       <center>
          <h2> Bay View Cottage</h2>
          <hr />
       </center>
       <div id="information">

       </div >
       <div id="calendar">

       </div>
   </div>
   </form>
</body>
```

Go to the style sheet and add formatting code for the ***content***, ***information*** and ***calendar*** divisions.

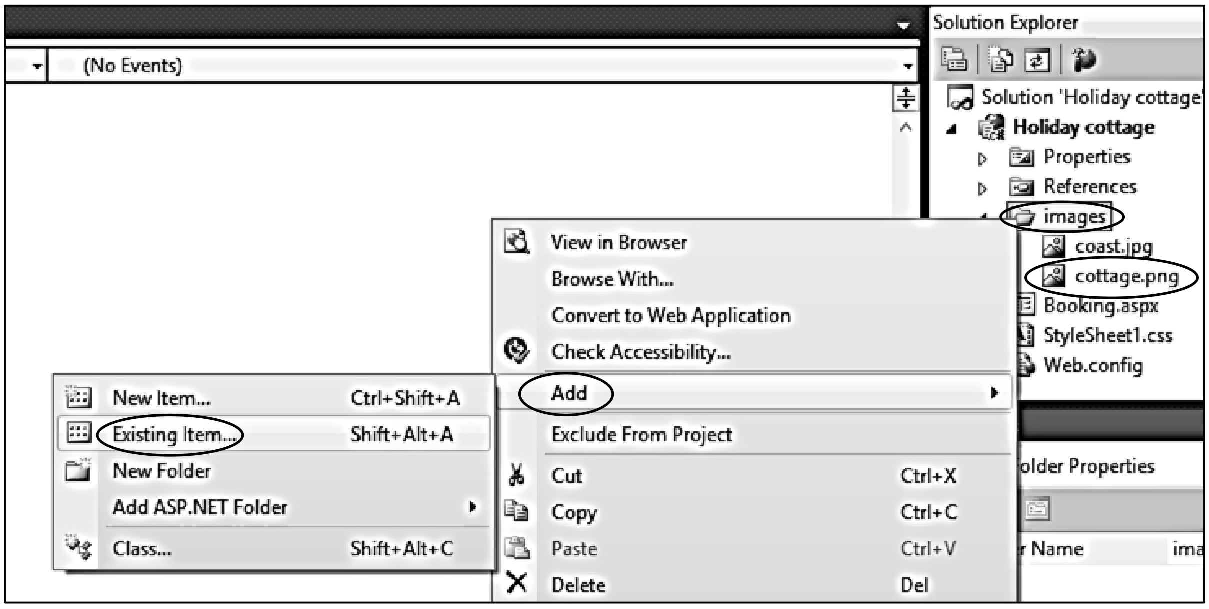
```
body
{
    . . . . .
}
#content
{
    width: 1080px;
    height: 600px;
    margin: 0 auto;
    padding: 10px;
    background-color: #FFFFFF;
    border: 5px solid #DEDEDE;
    color: Black;
    top: 40px;
    position: relative;
}
#information
{
    width: 450px;
    height: 500px;
    margin: 0 auto;
    padding: 10px;
    background-color: CornflowerBlue;
    float: left;
    color: White;
}
#calendar
{
    float: right;
    width: 550px;
    height: 500px;
    margin: 0 auto;
    padding: 10px;
}
p.indent
{
    margin: 25px;
}
```

Build and run the web page. Check that the basic column layout appears correctly.



Use the Internet to obtain a suitable photograph of a cottage. Resize this to be 400 pixels wide.

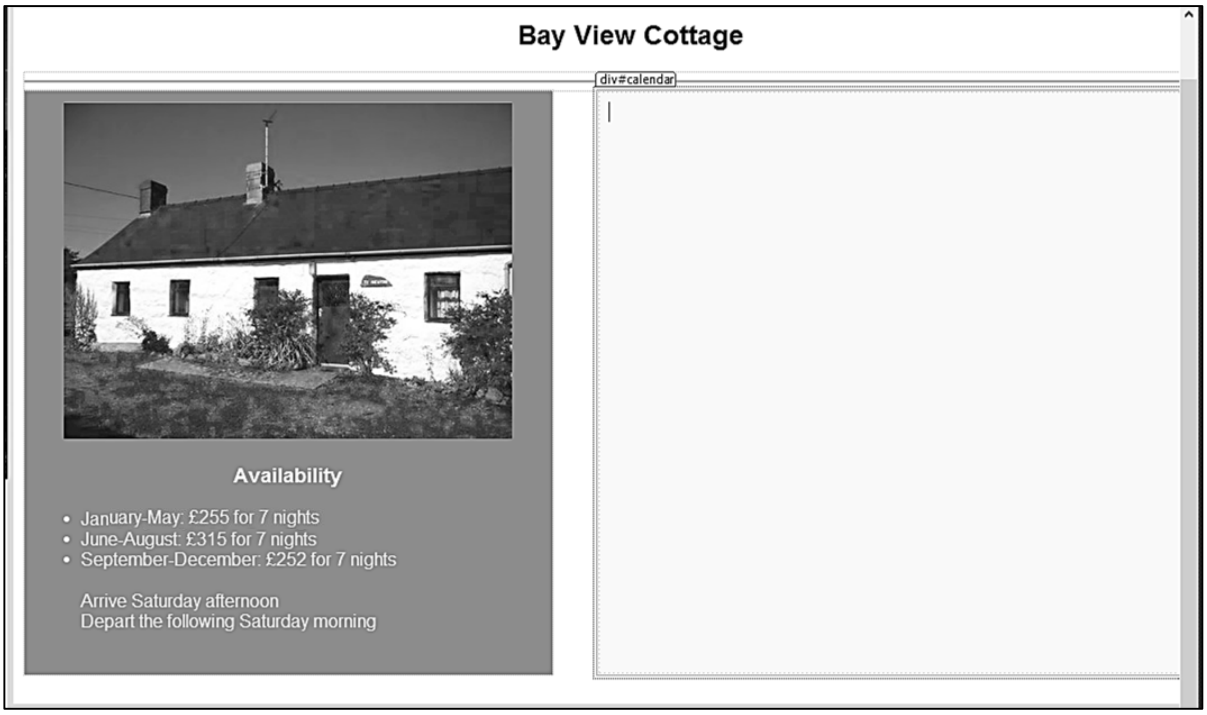
Close the web browser, return to the Solution Explorer and stop debugging. Upload the cottage photograph to the ***images*** folder by right clicking the ***images*** icon and selecting ***Add / Existing Item***.



Go to the HTML page and add code to the information division. This will display the photograph of the cottage, and display the weekly prices as a bullet point list.

```
<div id="information">
   <center>
      <img src="images/cottage.png" />
      <h3>
         Availability
      </h3>
   </center>
   <ul>
       <li>
          January-May: £255 for 7 nights
       </li>
       <li>
          June-August: £315 for 7 nights
       </li>
       <li>
          September-December: £252 for 7 nights
       </li>
   </ul>
   <p class="indent">
      Arrive Saturday afternoon
      <br />
      Depart the following Saturday morning
   </p>
</div >
<div id="calendar">
```

Use the Design button to go to the page preview screen. Check that the cottage photograph and prices are displayed correctly in the left column. Close the browser and stop debugging.



Bay View Cottage

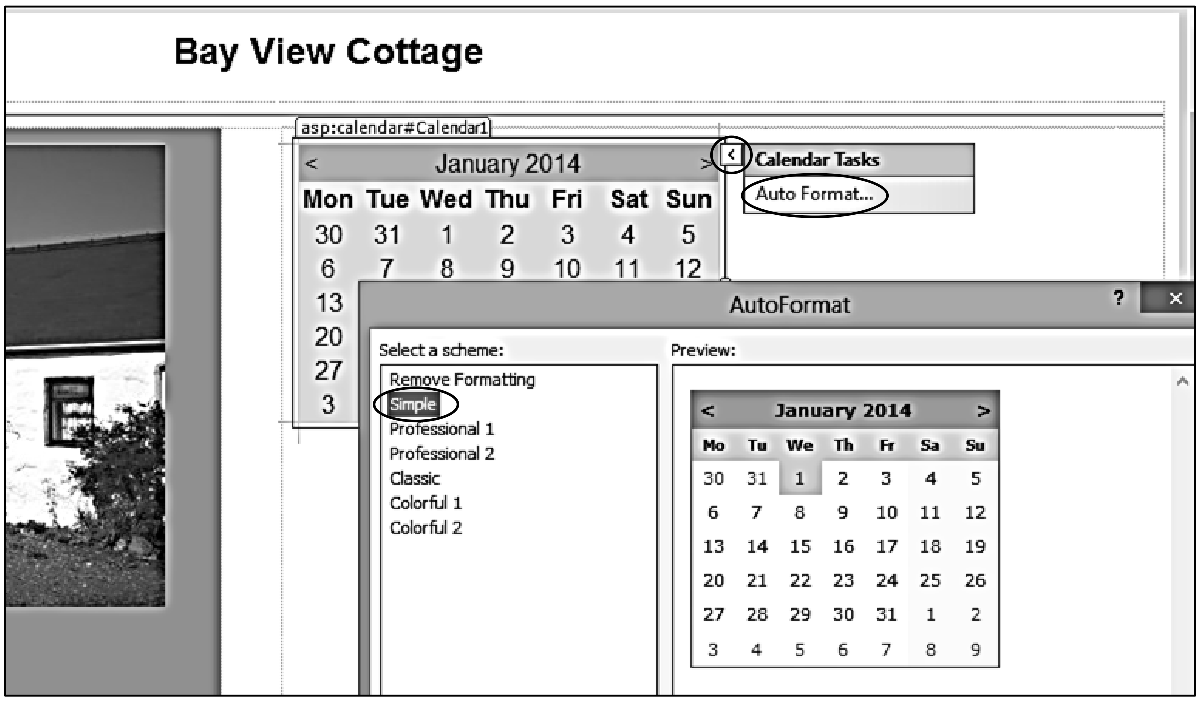
div#calendar

Availability

- January-May: £255 for 7 nights
- June-August: £315 for 7 nights
- September-December: £252 for 7 nights

Arrive Saturday afternoon
Depart the following Saturday morning

Go to the ***View*** option on the main menu and open the ***Toolbox.*** Select the ***Calendar*** component. Drag and drop this in the right column. Click the small arrow icon at the top right corner of the calendar, then open the ***AutoFormat*** window. This allows various styles of calendar to be easily created. We will choose the ***Simple*** calendar option.



Bay View Cottage

asp:calendar#Calendar1

Click the Source button at the bottom of the window to return to the HTML page. Notice that Visual Studio has already added many lines of formatting code for the Calendar component. We will now make some changes to this formatting:

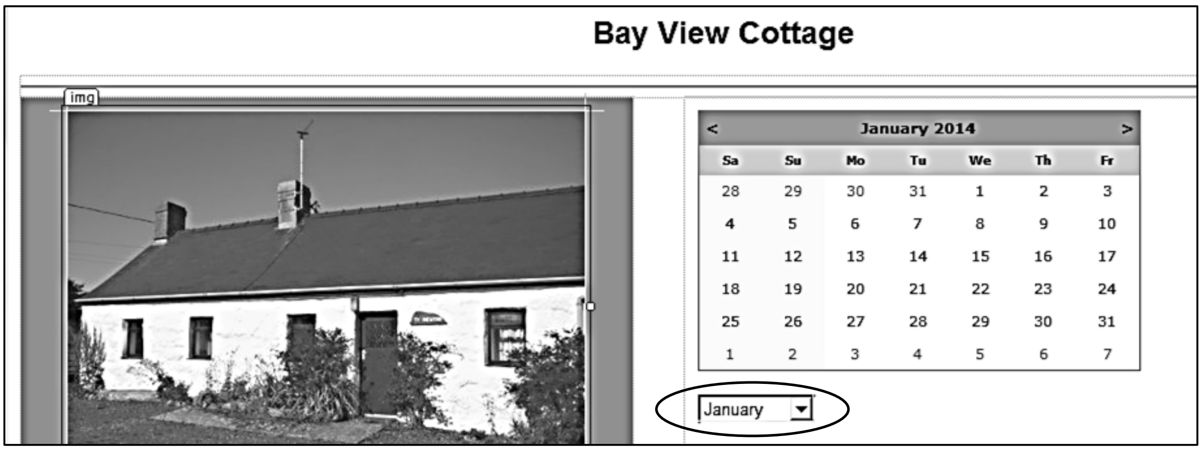
- The calendar currently shows today's date as highlighted. There is no need for this so remove the line of code marked <TodayDayStyle>
- Set the height of the calendar to 200 pixels, and its width to 340 pixels.
- Week bookings of the cottage start on Saturdays, so we will restructure the calendar to also start each week on a Saturday. Add the line of code to do this.

```
<div id="calendar">
    <asp:Calendar ID="Calendar1" runat="server" BackColor="White"
        BorderColor="#999999" CellPadding="4" DayNameFormat="Shortest"
        Font-Names="Verdana" Font-Size="8pt" ForeColor="Black"

        Height="180px"
        Width="340px"
        FirstDayOfWeek="Saturday" >

        <DayHeaderStyle BackColor="#CCCCCC" Font-Bold="True" Font-Size="7pt" />
        <NextPrevStyle VerticalAlign="Bottom" />
        <OtherMonthDayStyle ForeColor="#808080" />
        <SelectedDayStyle BackColor="#666666" Font-Bold="True"
                                              ForeColor="White" />
        <SelectorStyle BackColor="#CCCCCC" />
        <TitleStyle BackColor="#999999" BorderColor="Black" Font-Bold="True" />

        <TodayDayStyle BackColor="#CCCCCC" ForeColor="Black" />    remove

        <WeekendDayStyle BackColor="#FFFFCC" />
    </asp:Calendar>
</div>
```

The next step in adding functionality to the calendar is to allow the user to easily select a month. Add code below the calendar to create a drop down list of months.

```
        </asp:Calendar>
        <br />
        <asp:DropDownList ID="DropDownList1" runat="server" AutoPostBack="True" >
            <asp:ListItem Value="1">January</asp:ListItem>
            <asp:ListItem Value="2">February</asp:ListItem>
            <asp:ListItem Value="3">March</asp:ListItem>
            <asp:ListItem Value="4">April</asp:ListItem>
            <asp:ListItem Value="5">May</asp:ListItem>
            <asp:ListItem Value="6">June</asp:ListItem>
            <asp:ListItem Value="7">July</asp:ListItem>
            <asp:ListItem Value="8">August</asp:ListItem>
            <asp:ListItem Value="9">September</asp:ListItem>
            <asp:ListItem Value="10">October</asp:ListItem>
            <asp:ListItem Value="11">November</asp:ListItem>
            <asp:ListItem Value="12">December</asp:ListItem>
        </asp:DropDownList>
    </div>
```

Click the Design button. The drop down list should have been created below the calendar.

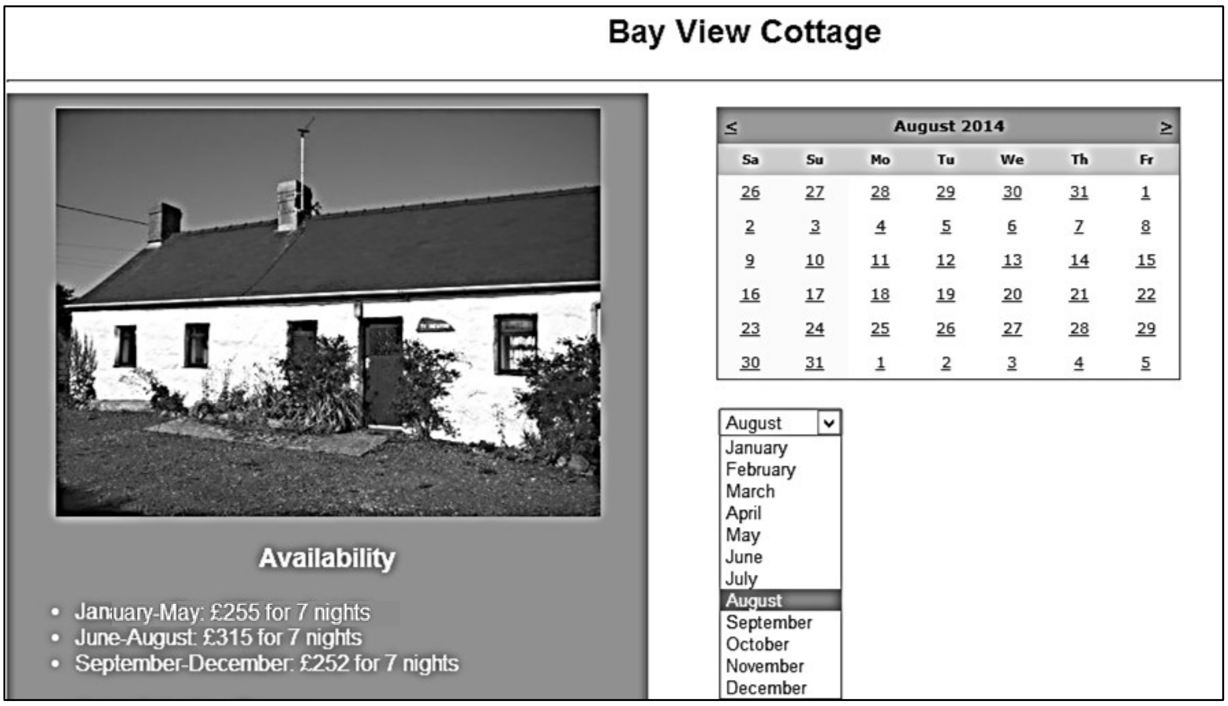


Double click the drop down list component to create a C# IndexChanged method. This will be activated when a month is seleted by the user. Add code to carry out several tasks:

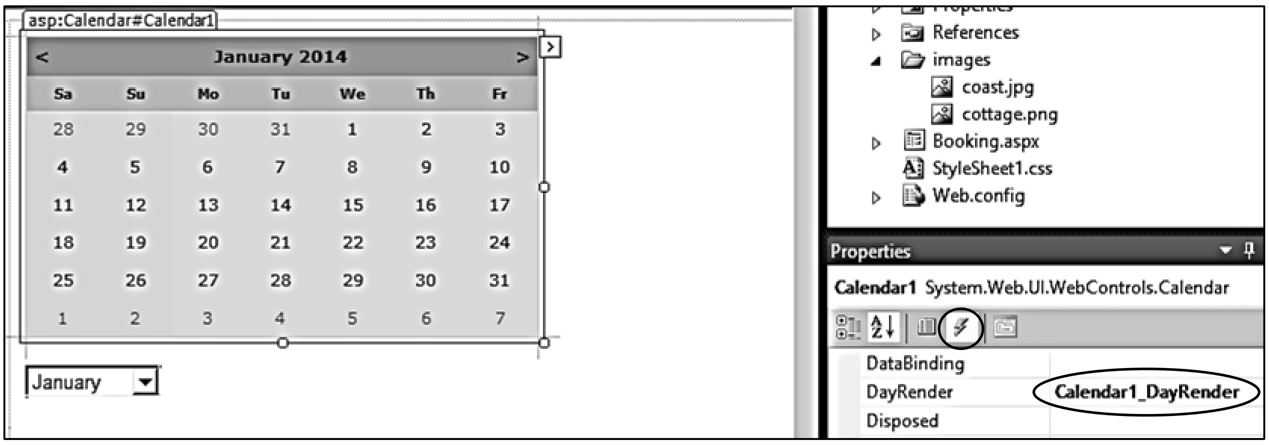
- We obtain the number of the month selected in the drop down list (e.g. March = 3).
- The current year number is found from the computer's clock (e.g. 2014).
- The current month number is also found from the computer's clock.
- If the month selected from the drop down list is earlier in the year than the current month, we will assume that the customer wishes to book for next year – for example: if in August 2014 a customer selects 'April', we will assume this to mean April 2015. The program then increases the year number by one.
- A date is then created to reset the calendar display, using day 01 of the selected month.

```
public partial class Booking : System.Web.UI.Page
{
    protected void Page_Load(object sender, EventArgs e)
    {

    }

    protected void DropDownList1_SelectedIndexChanged(object sender, EventArgs e)
    {
        int month = Convert.ToInt16(DropDownList1.SelectedValue);
        int thisMonth = DateTime.Today.Month;
        int thisYear = DateTime.Today.Year;
        if (month < thisMonth)
            thisYear++;
        string dateString = "01-" + month + "-" + thisYear;
        Calendar1.VisibleDate = Convert.ToDateTime(dateString);
    }
}
```

Run the web page and check that months can be selected correctly using the drop down list.



Close the web browser. Return to Visual Studio and stop the debugging session.
Go to the ***Design*** screen and select the ***Calendar*** component. In the ***Properties*** window, click on the ***lightening flash*** icon to open the list of methods. Double click in the white cell to the right of ***DayRender*** to create a method.



The C# code page opens with an empty ***DayRender*** method created. This method operates whenever the calendar is redrawn, and allows us to add add formatting to particular dates.

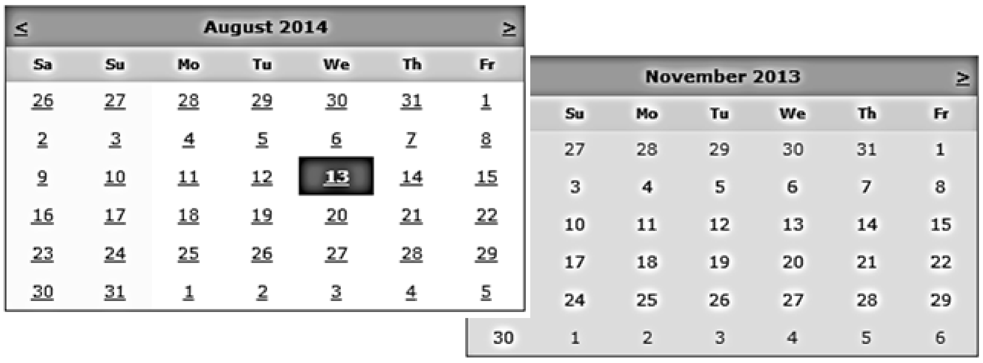
Begin by adding a ***using Drawing*** directive to the top of the page.

```
using System.Web.UI;
using System.Web.UI.WebControls;

using System.Drawing;

namespace Holiday_cottage
{
```

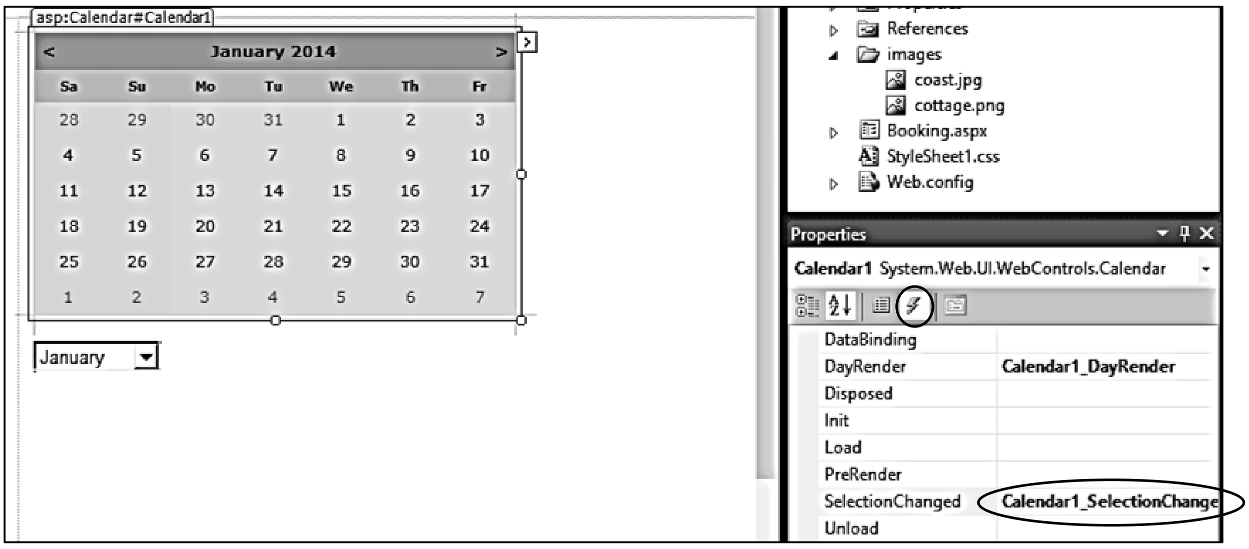
Move down to the ***DayRender*** method and add lines of code. The program obtains today's date, then compares this against each day as it is being created on the calendar display. For any days before today's date, the date number is deactivated so that it cannot be selected by the user, and the date cell is shaded in grey.

```
protected void Calendar1_DayRender(object sender, DayRenderEventArgs e)
{
    DateTime currentDate = Calendar1.TodaysDate;

    //show days before todays date in grey
    if (e.Day.Date < currentDate)
    {
        e.Day.IsSelectable = false;
        e.Cell.BackColor = Color.Gainsboro;
    }
}
```

Build and run the web page. If a future date is clicked, a highlight appears. However, if the calendar is scrolled back to a month in the past then days are shaded out and cannot be selected.



Close the browser, stop debugging and return to the ***Design*** screen. Go to the ***Properties*** window and again select the method list by clicking the ***lightening flash*** icon. Double click to the right of ***SelectionChanged*** to create a method.



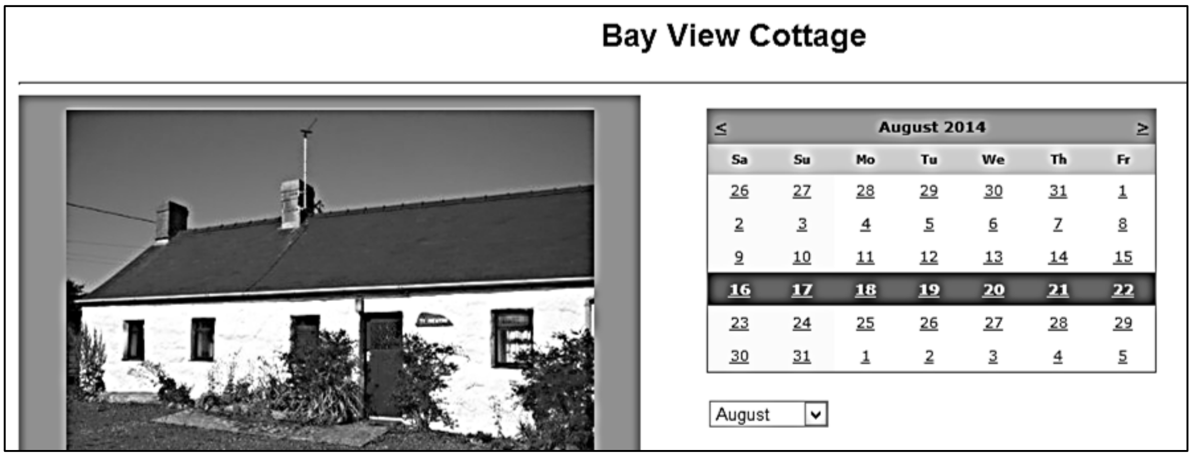
Calendar1 System.Web.UI.WebControls.Calendar

| | |
|---|---|
| DataBinding | |
| DayRender | Calendar1_DayRender |
| Disposed | |
| Init | |
| Load | |
| PreRender | |
| SelectionChanged | Calendar1_SelectionChange |
| Unload | |

Add code to the ***SelectionChanged*** method. The purpose of this code is to allow the user to select a complete week from Saturday to the following Friday by clicking on any of the day numbers within that week.
The method begins by getting the date which has been clicked, which is stored as the variable ***currentDate***. The name of this day is obtained and stored as ***currentDay***.
We now use a loop to count backwards, one day at a time, until we reach the Saturday at the start of the week. As we pass each date, it is marked as '***selected***' so that it will appear highlighted on the calendar.
We reurn to the original day selected, then use a similar loop to count forward until the Friday at the end of the week is reached. Again, each day passed is added to the list of '***selected***' dates.

```
protected void Calendar1_SelectionChanged(object sender, EventArgs e)
{
    DateTime currentDate= Calendar1.SelectedDate;
    string currentDay = Convert.ToString(currentDate.DayOfWeek);
    while (currentDay != "Saturday")
    {
        currentDate = currentDate.AddDays(-1);
        currentDay = Convert.ToString(currentDate.DayOfWeek);
        Calendar1.SelectedDates.Add(currentDate);
    }

    currentDate = Calendar1.SelectedDate;
    currentDay = Convert.ToString(currentDate.DayOfWeek);
    while (currentDay != "Friday")
    {
        currentDate = currentDate.AddDays(1);
        currentDay = Convert.ToString(currentDate.DayOfWeek);
        Calendar1.SelectedDates.Add(currentDate);
    }
}
```

Build and run the web page. Select a month, then click on any date. All days for the selected week should be shown as highlighted.



Bay View Cottage

< August 2014 >

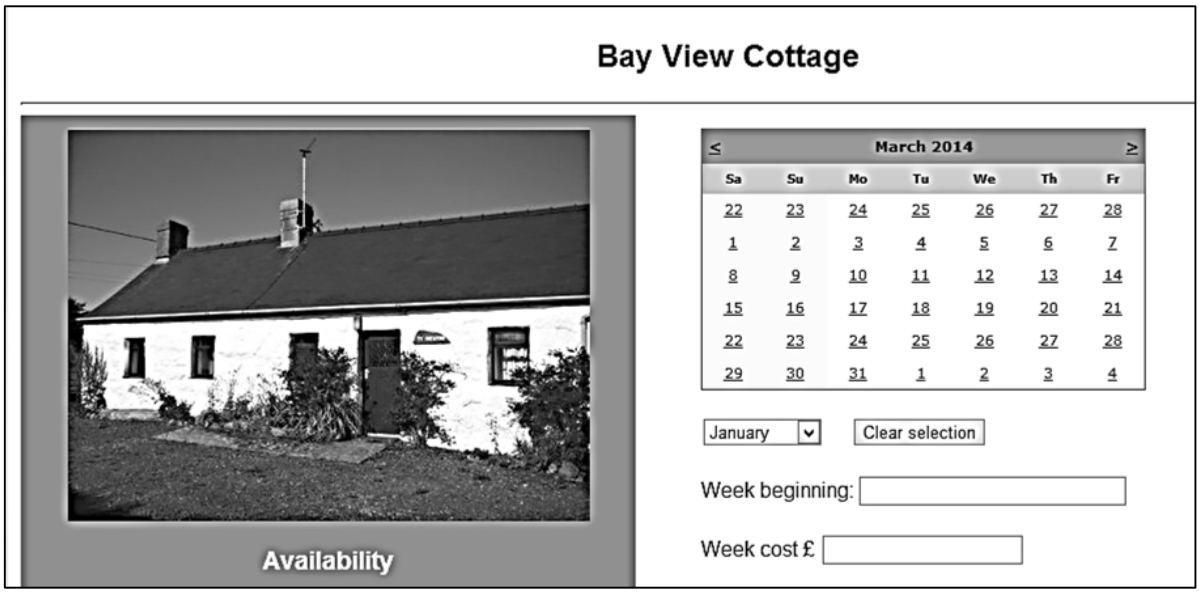
| Sa | Su | Mo | Tu | We | Th | Fr |
|---|---|---|---|---|---|---|
| 26 | 27 | 28 | 29 | 30 | 31 | 1 |
| 2 | 3 | 4 | 5 | 6 | 7 | 8 |
| 9 | 10 | 11 | 12 | 13 | 14 | 15 |
| 16 | 17 | 18 | 19 | 20 | 21 | 22 |
| 23 | 24 | 25 | 26 | 27 | 28 | 29 |
| 30 | 31 | 1 | 2 | 3 | 4 | 5 |

August

We can now work on the section of the page below the calendar where the customer will confirm the booking date and enter their contact details.
Close the web browser and stop debugging. Go to the HTML page beneath the drop down list of months, and add a button, text boxes and captions.

```
            <asp:ListItem Value="10">October</asp:ListItem>
            <asp:ListItem Value="11">November</asp:ListItem>
            <asp:ListItem Value="12">December</asp:ListItem>
        </asp:DropDownList>
        &nbsp;&nbsp;&nbsp;
        <asp:Button ID="btnClear" runat="server" Text="Clear selection" />
        <br />
        <br />
        Week beginning:
        <asp:TextBox runat="server" ID="txtWeekBeginning" Width="200px">
        </asp:TextBox>
        <br />
        <br />
        Week cost £
        <asp:TextBox runat="server" ID="txtWeekCost"></asp:TextBox>
    </div>
```

Build and run the web page. Check that the additional components have been created correctly.



| Sa | Su | Mo | Tu | We | Th | Fr |
|---|---|---|---|---|---|---|
| 22 | 23 | 24 | 25 | 26 | 27 | 28 |
| 1 | 2 | 3 | 4 | 5 | 6 | 7 |
| 8 | 9 | 10 | 11 | 12 | 13 | 14 |
| 15 | 16 | 17 | 18 | 19 | 20 | 21 |
| 22 | 23 | 24 | 25 | 26 | 27 | 28 |
| 29 | 30 | 31 | 1 | 2 | 3 | 4 |

Close the web browser and stop debugging. Go to the C# code page ***Booking.aspx.cs***.
We will add code to calculate the holiday cost. This is simple if the holiday week lies entirely within a single month, but is more complicated if the week spans a month end where the price changes. For example, if the customer booked the week from 30 August to 5 September 2014, they would expect to pay for two days at the higher rate of £315 per week, and the remaining five days at the lower rate of £255 per week. To cope with this possibility, we will calacualte and add the charges for each individual day of the booking.

We will begin by inserting a small method to calculate the charge for one day, depending on the number of the month. This caries out various actions:

- The method accepts the month number as an ***input parameter***.
- The charges for a single day are calculated, for both the lower and higher cost months.
- The method checks the month number supplied. If the month is May or earlier, or September or later, then the lower day cost is selected. Otherwise, the higher day cost is selected.
- The method returns the appropriate day cost as the ***output parameter***.

```
        if (e.Day.Date < currentDate)
        {
            e.Day.IsSelectable = false;
            e.Cell.BackColor = Color.Gainsboro;
        }
    }

    protected double addDayCost(string month)
    {
        double lowerCost = (double)255 / 7;
        double higherCost = (double)315 / 7;
        double dayCost;
        int monthnumber = Convert.ToInt16(month);
        if (monthnumber <= 5 || monthnumber >= 9)
        {
            dayCost = lowerCost;
        }
        else
        {
            dayCost = higherCost;
        }
        return dayCost;
    }

    protected void Calendar1_SelectionChanged(object sender, EventArgs e)
    {
        DateTime currentDate= Calendar1.SelectedDate;
```

Go now to the ***Calendar_SelectionChanged*** method. Add code near the beginning to initialise the ***week cost*** to zero, then add the cost for the date which the user has clicked on the calendar. This is done by extracting the ***month*** from the date, then calling the ***addDayCost*** method we wrote earlier to find the day cost during that month.

```
    protected void Calendar1_SelectionChanged(object sender, EventArgs e)
    {
        DateTime currentDate= Calendar1.SelectedDate;
        string currentDay = Convert.ToString(currentDate.DayOfWeek);

        weekCost = 0.00;
        string monthWanted = currentDate.Month.ToString("00");
        weekCost = weekCost + addDayCost(monthWanted);

        while (currentDay != "Saturday")
```

Add weekCost as a variable at the start of the program.

```
public partial class Booking : System.Web.UI.Page
{
    public static double weekCost;

    protected void Page_Load(object sender, EventArgs e)
    {
```

We also need to include lines of code to carry out the following actions:

- Add the cost for each of the days earlier or later in the week selected.
- Record the start date of the week.
- Finally, transfer the week cost and start date to the text boxes on the web page.

The lines to be added are indicated below.

```
protected void Calendar1_SelectionChanged(object sender, EventArgs e)
{
    DateTime currentDate= Calendar1.SelectedDate;
    string currentDay = Convert.ToString(currentDate.DayOfWeek);
    weekCost = 0.00;
    string monthWanted = currentDate.Month.ToString("00");
    weekCost = weekCost + addDayCost(monthWanted);

    while (currentDay != "Saturday")
    {
        currentDate = currentDate.AddDays(-1);
        currentDay = Convert.ToString(currentDate.DayOfWeek);
        Calendar1.SelectedDates.Add(currentDate);
        monthWanted = currentDate.Month.ToString("00");
        weekCost = weekCost + addDayCost(monthWanted)
    }

    weekStart = currentDate;
    currentDate = Calendar1.SelectedDate;
    currentDay = Convert.ToString(currentDate.DayOfWeek);

    while (currentDay != "Friday")
    {
        currentDate = currentDate.AddDays(1);
        currentDay = Convert.ToString(currentDate.DayOfWeek);
        Calendar1.SelectedDates.Add(currentDate);
        monthWanted = currentDate.Month.ToString("00");
        weekCost = weekCost + addDayCost(monthWanted);
    }

    string format = "dddd MMMM d, yyyy";
    txtWeekBeginning.Text = weekStart.ToString(format);
    format = "0.00";
    txtWeekCost.Text = weekCost.ToString(format);
}
```

The variable ***weekStart*** should also be added at the start of the program.

```
public partial class Booking : System.Web.UI.Page
{
    public static DateTime weekStart;

    public static double weekCost;

    protected void Page_Load(object sender, EventArgs e)
    {
```

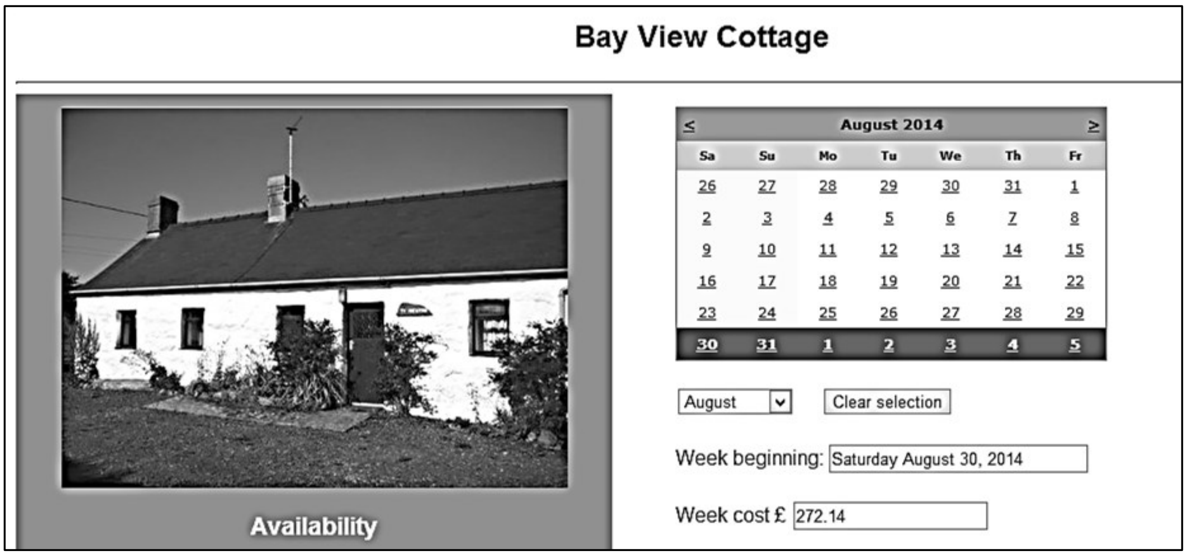
Return to the ***Booking.aspx*** design page and double click the 'Clear selection' button. Add code to the button_click method.

```
protected void btnClear_Click(object sender, EventArgs e)
{
    Calendar1.SelectedDates.Clear();
    txtWeekBeginning.Text = "";
    txtWeekCost.Text = "";
}
```

Build and run the web page. Check that the correct week start date is displayed when any day is selected, and that selections can be cleared by clicking the button.

Check also that the correct weekly charge is displayed:

- Higher rate of £315 from June to August
- Lower rate of £255 in other months
- An intermediate amount if the selected week spans a month end where the rate changes.


Bay View Cottage

| < | | | August 2014 | | | > |
|---|---|---|---|---|---|---|
| Sa | Su | Mo | Tu | We | Th | Fr |
| 26 | 27 | 28 | 29 | 30 | 31 | 1 |
| 2 | 3 | 4 | 5 | 6 | 7 | 8 |
| 9 | 10 | 11 | 12 | 13 | 14 | 15 |
| 16 | 17 | 18 | 19 | 20 | 21 | 22 |
| 23 | 24 | 25 | 26 | 27 | 28 | 29 |
| 30 | 31 | 1 | 2 | 3 | 4 | 5 |

Week beginning: Saturday August 30, 2014

Week cost £ 272.14

Return to Visual Studio and stop debugging.

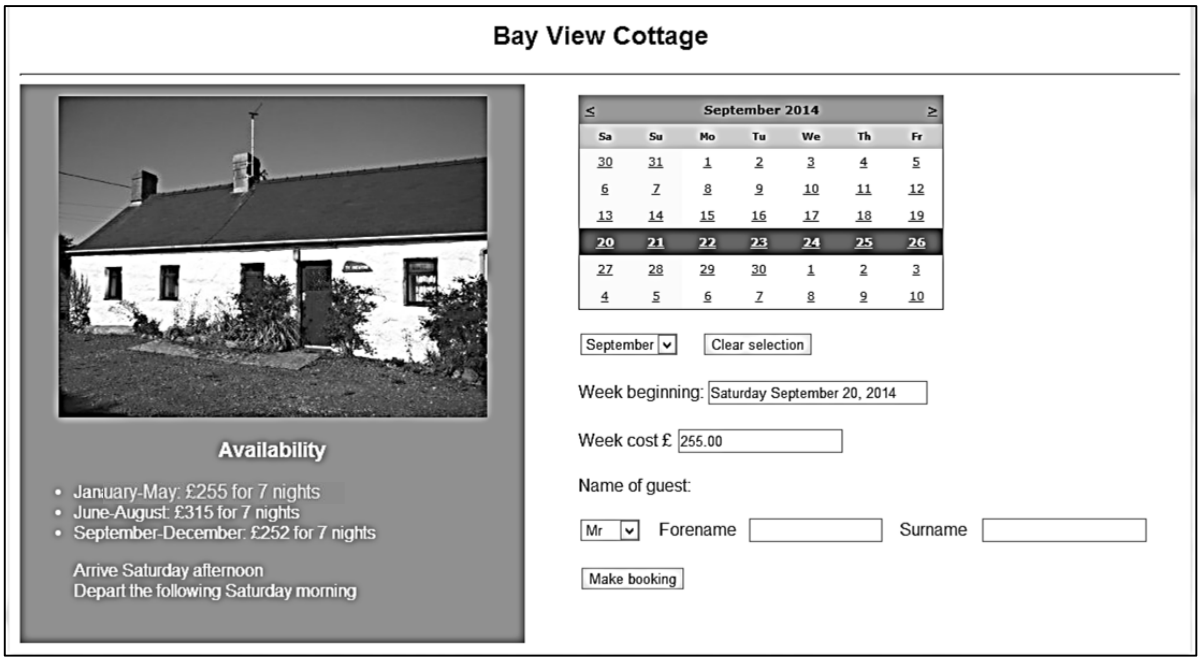
We will add text boxes for the customer to enter their forename and surname, plus a drop down list from which they can select a title: Mr, Mrs, Miss, Ms.   A button will also be needed to make the booking.

In a real booking system we would ask for further contact information, along with payment details. A full booking procedure will be carried out in the ***Canal Boat Holidays*** program in a later chapter.

Go to the ***Booking.aspx*** HTML page and add the components shown below.

```
        Week beginning:
        <asp:TextBox runat="server" ID="txtWeekBeginning" Width="200px">
        </asp:TextBox>
        <br />
        <br />
        Week cost £
        <asp:TextBox runat="server" ID="txtWeekCost"></asp:TextBox>
        <br />
        <br />
        Name of guest:
        <br />
        <br />
        <asp:DropDownList ID="ddlTitle" runat="server">
            <asp:ListItem Value="Mr"></asp:ListItem>
            <asp:ListItem Value="Mrs"></asp:ListItem>
            <asp:ListItem Value="Miss"></asp:ListItem>
            <asp:ListItem Value="Ms"></asp:ListItem>
        </asp:DropDownList>
        &nbsp;&nbsp;
        Forename
        &nbsp;
        <asp:TextBox ID="txtForename" runat="server" Width="120px">
        </asp:TextBox>
        &nbsp;&nbsp;
        Surname
        &nbsp;
        <asp:TextBox ID="txtSurname" runat="server"></asp:TextBox>
        <br />
        <br />
        <asp:Button ID="btnBook" runat="server" Text="Make booking" />
    </div>
```

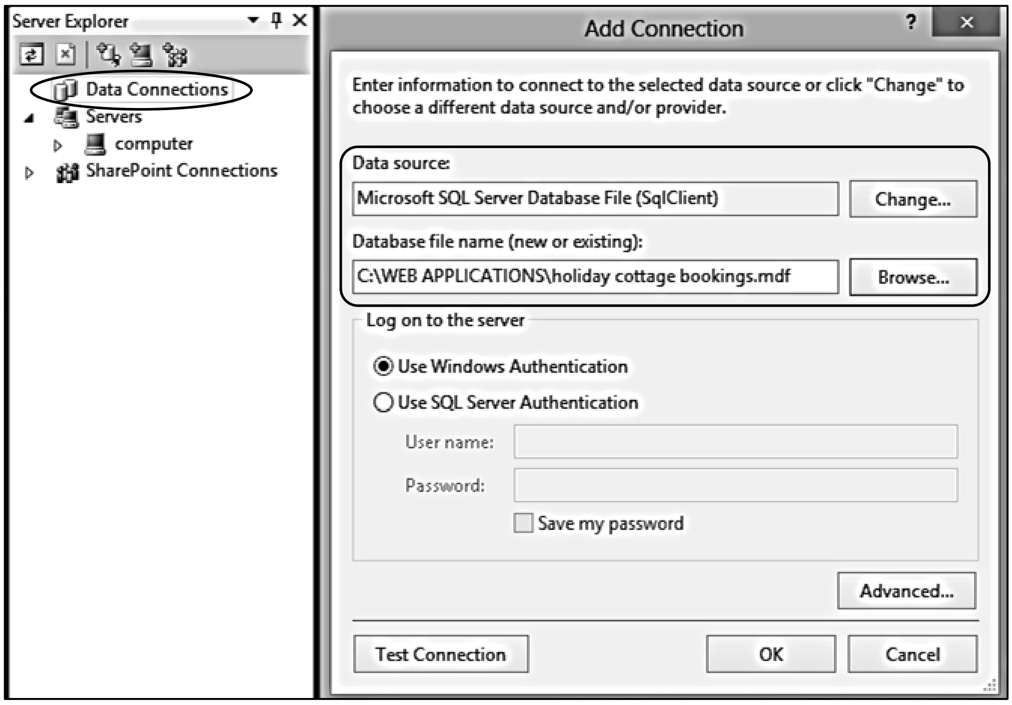
Build and run the web page to check that the additional components are positioned correctly.


Bay View Cottage
Availability
January-May: £255 for 7 nights
June-August: £315 for 7 nights
September-December: £252 for 7 nights
Arrive Saturday afternoon
Depart the following Saturday morning
September 2014
September
Clear selection
Week beginning: Saturday September 20, 2014
Week cost £ 255.00
Name of guest:
Mr
Forename
Surname
Make booking

Close the browser and stop debugging. For an on-line booking system, it is necessary to store customer bookings in a database table. We will set up a database now.

Go to the ***View*** option on the main menu and select ***Server Explorer***. Right click the ***Data Connections*** icon. Select ***Add Connection***.

Check that the ***Data source*** is set to ***Microsoft SQL Server Database File***, and select a folder in which to store the database. Give the database the name '***holiday cottage bookings***'.

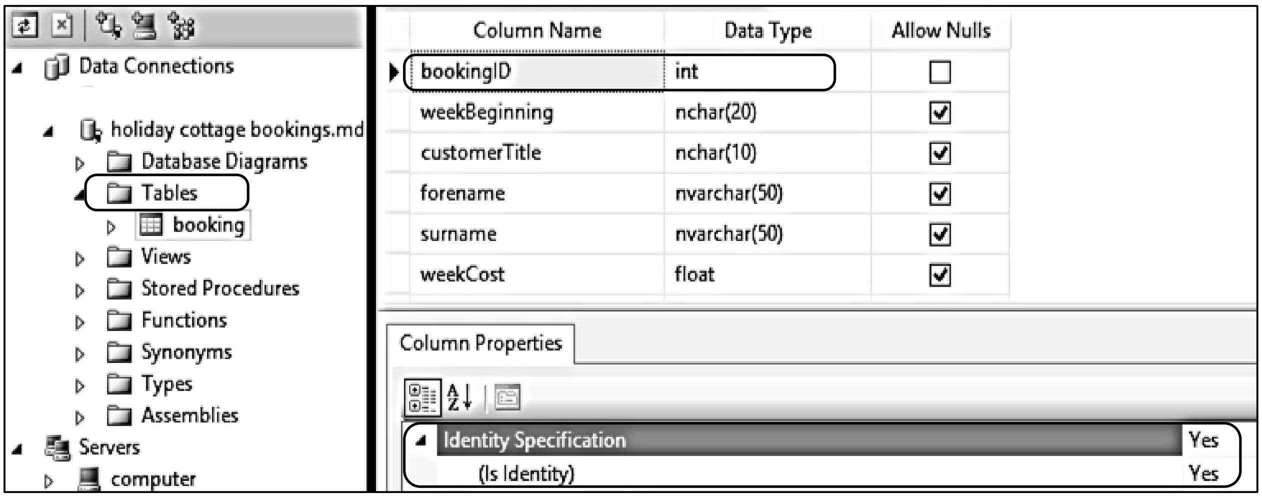


Enter information to connect to the selected data source or click "Change" to choose a different data source and/or provider.

Data source: Microsoft SQL Server Database File (SqlClient)

Database file name (new or existing): C:\WEB APPLICATIONS\holiday cottage bookings.mdf

Click OK to continue, and accept to create a new database. The ***holiday cottage bookings*** database should appear in the ***Server Explorer*** window.
Click the small arrow icon to open the database. Right click on ***Tables***, then select the option to ***Add New Table***.
Add fields and data types for the table as shown. Note that the ***bookingID*** should be set as an auto number. To do this, select the ***bookingID*** field, go to the list of properties at the bottom of the page and find the ***Identity Specification*** section. Click the small arrow icon to open further options in this section, then set '***Is Identity***' to '***yes***'.



| Column Name | Data Type | Allow Nulls |
|---|---|---|
| bookingID | int | ☐ |
| weekBeginning | nchar(20) | ✓ |
| customerTitle | nchar(10) | ✓ |
| forename | nvarchar(50) | ✓ |
| surname | nvarchar(50) | ✓ |
| weekCost | float | ✓ |

Column Properties

| | |
|---|---|
| Identity Specification | Yes |
| (Is Identity) | Yes |

When you have completed the list of fields, click the cross on the table tab. Accept to save the table, then give this name '***booking***'. Return to the ***Booking.aspx.cs*** C# code page. Add '***using System.Data.SqlClient***' and '***using System.Data***' directives, and give the location of the database.

```
using System.Web.UI.WebControls;
using System.Drawing;

using System.Data.SqlClient;
using System.Data;

namespace Holiday_cottage
{
    public partial class Booking : System.Web.UI.Page
    {
            public static DateTime weekStart;
            public static double weekCost;

            string databaseLocation =
                    "C:\\WEB APPLICATIONS\\holiday cottage bookings.mdf;";

        protected void Page_Load(object sender, EventArgs e)
        {
```

Before setting up lines of program code to access the database, it will be useful to add a label to the web page which can be used to display messages to the user. Go to the HTML code page and add a label component below the text boxes, as shown below.

```
        <asp:TextBox ID="txtSurname" runat="server"></asp:TextBox>
        <br />
        <br />
        <asp:Button ID="btnBook" runat="server" Text="Make booking" />
        <br />
        <br />
        <center>
             <asp:Label ID="lblMessage" runat="server"></asp:Label>
        </center>
    </div>
```

Return to the C# code page. We will first write a method to add a booking to the database. Insert this after the ***btnClear_Click*** method.

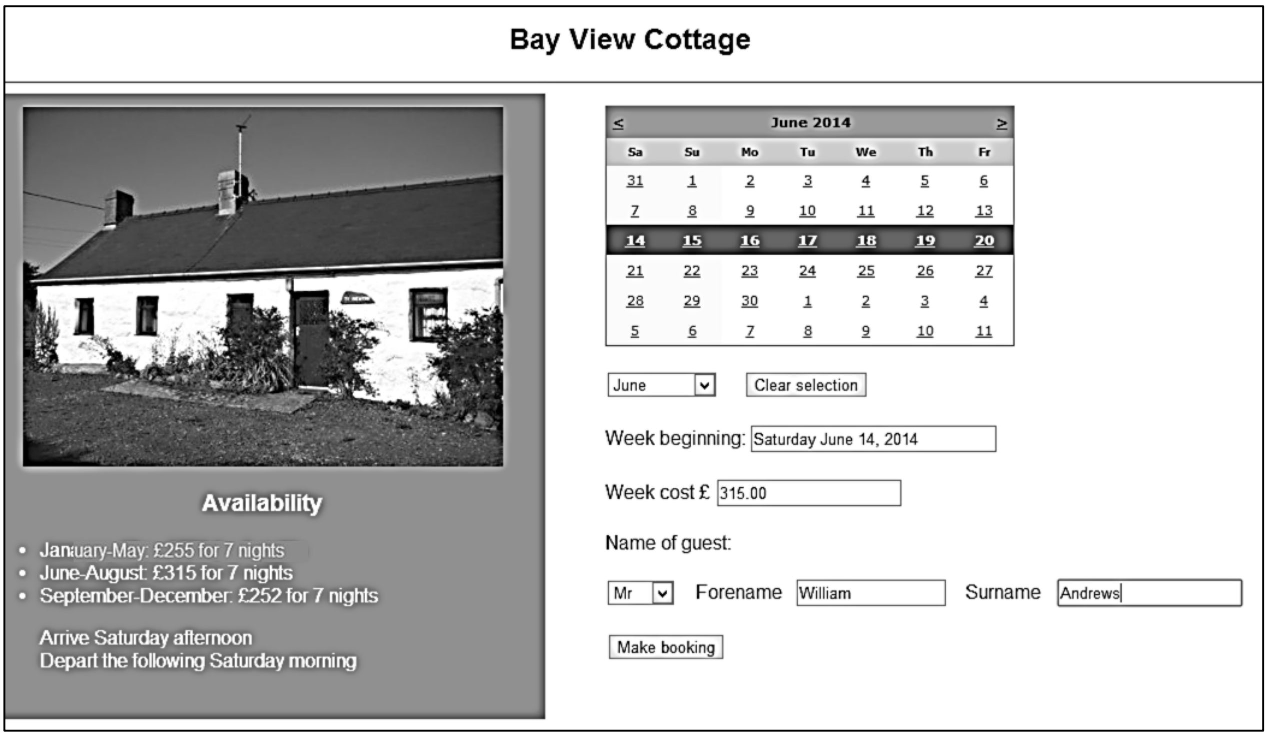
```
    protected void btnClear_Click(object sender, EventArgs e)
    {
        Calendar1.SelectedDates.Clear();
        txtWeekBeginning.Text = "";
        txtWeekCost.Text = "";
    }

    protected void makebooking(DateTime weekStart, string title, string forename,
                                                   string surname, double weekCost)
    {
        SqlConnection cnTB = new SqlConnection(@"Data Source=.\SQLEXPRESS;
            AttachDbFilename=" + databaseLocation + "Integrated Security=True;
            Connect Timeout=30; User Instance=True");
        try
        {
            cnTB.Open();
            SqlCommand cmBooking = new SqlCommand();
            cmBooking.Connection = cnTB;
            cmBooking.CommandType = CommandType.Text;
            cmBooking.CommandText = "INSERT INTO booking(weekBeginning, customerTitle,
                 forename, surname, weekCost) VALUES ('"+ Convert.ToString(weekStart)
                 + "','" + title + "','" + forename + "','" + surname + "','"
                 + Convert.ToString(weekCost) + "')";
            lblMessage.Text = cmBooking.CommandText;
            cmBooking.ExecuteNonQuery();
            cnTB.Close();
            lblMessage.Text = "Record saved";
        }
        catch
        {
            lblMessage.Text = "File error";
        }
    }
```

This method must now be linked to the ***Make booking*** button on the web page. Go to the ***Design*** view and double click the button to create a method. Add code to the method which will:

- Obtain the customer details from the text boxes and drop down list.
- Only continue if a name has actually been entered…
- Save the booking to the database table.
- Clear the calendar and text boxes.

```
protected void btnBook_Click(object sender, EventArgs e)
{
    string title = ddlTitle.Text;
    string forename = txtForename.Text;
    string surname = txtSurname.Text;
    if (forename.Length > 0 && surname.Length > 0)
    {
        makebooking(weekStart, title, forename, surname, weekCost);
        txtWeekBeginning.Text = "";
        txtWeekCost.Text = "";
        txtForename.Text = "";
        txtSurname.Text = "";
        Calendar1.SelectedDates.Clear();
    }
}
```

Build and run the web page. Enter a booking, then click the ***Make booking*** button.



Bay View Cottage

Availability

- January-May: £255 for 7 nights
- June-August: £315 for 7 nights
- September-December: £252 for 7 nights

Arrive Saturday afternoon
Depart the following Saturday morning

| < | | | June 2014 | | | > |
|---|---|---|---|---|---|---|
| Sa | Su | Mo | Tu | We | Th | Fr |
| 31 | 1 | 2 | 3 | 4 | 5 | 6 |
| 7 | 8 | 9 | 10 | 11 | 12 | 13 |
| 14 | 15 | 16 | 17 | 18 | 19 | 20 |
| 21 | 22 | 23 | 24 | 25 | 26 | 27 |
| 28 | 29 | 30 | 1 | 2 | 3 | 4 |
| 5 | 6 | 7 | 8 | 9 | 10 | 11 |

June | Clear selection

Week beginning: Saturday June 14, 2014

Week cost £ 315.00

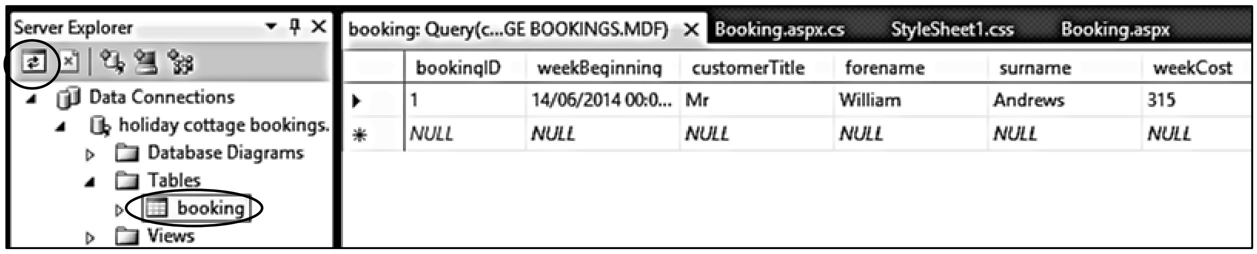
Name of guest:

Mr | Forename William | Surname Andrews

Make booking

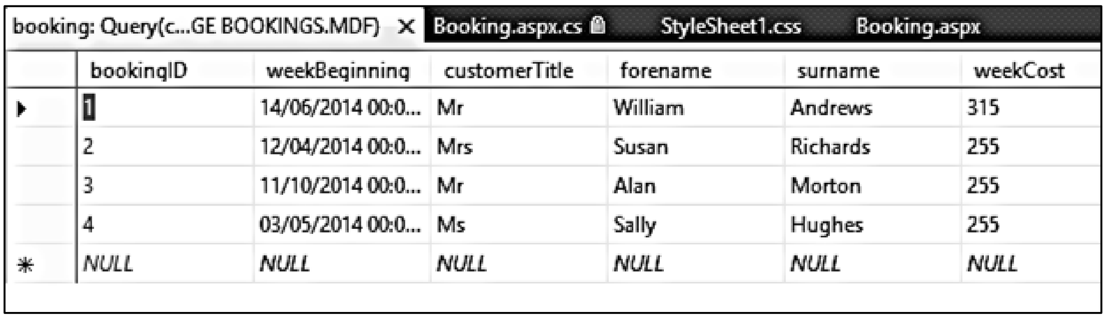
Close the web browser, return to Visual Studio and stop debugging.

Open the ***Server Explorer*** window and click the ***Refresh*** icon. Right click the ***booking*** table. Select ***Show Table Data***. The booking which you entered should have been uploaded to the database table.



| | bookingID | weekBeginning | customerTitle | forename | surname | weekCost |
|---|---|---|---|---|---|---|
| ▶ | 1 | 14/06/2014 00:0... | Mr | William | Andrews | 315 |
| * | NULL | NULL | NULL | NULL | NULL | NULL |

If the record upload is working correctly, re-run the web page and enter several more bookings for different dates. Return to Visual Studio and stop debugging. Close the ***Bookings*** table window, refresh the database, then reopen the table. Check that the bookings have been added correctly.



| | bookingID | weekBeginning | customerTitle | forename | surname | weekCost |
|---|---|---|---|---|---|---|
| ▶ | 1 | 14/06/2014 00:0... | Mr | William | Andrews | 315 |
| | 2 | 12/04/2014 00:0... | Mrs | Susan | Richards | 255 |
| | 3 | 11/10/2014 00:0... | Mr | Alan | Morton | 255 |
| | 4 | 03/05/2014 00:0... | Ms | Sally | Hughes | 255 |
| * | NULL | NULL | NULL | NULL | NULL | NULL |

We have one further action to add to the program, which is to show previously booked weeks as no longer available. This is essential to avoid double bookings. The strategy we will use is:

- Open the database and load all current bookings.
- Obtain the week start date for each booking.
- Use a loop to generate the dates of the other six days of the booked week.
- Store all these booked dates in a list which can then be accessed by the DayRender method. Dates found to be booked can be shaded and deactivated to prevent their selection again.

Go to the C# code page and create a new method with the name ***loadBookings( )***. This will access the database and load all the booking records.

```
protected void loadBookings()
{
   DataSet dsBooking = new DataSet();
   SqlConnection cnTB = new SqlConnection(@"Data Source=.\SQLEXPRESS;
       AttachDbFilename=" + databaseLocation + "Integrated Security=True;
       Connect Timeout=30; User Instance=True");
    cnTB.Open();
    SqlCommand cmBooking = new SqlCommand();
    cmBooking.Connection = cnTB;
    cmBooking.CommandType = CommandType.Text;
    cmBooking.CommandText = "SELECT * FROM booking";
    SqlDataAdapter daBooking = new SqlDataAdapter(cmBooking);
    daBooking.Fill(dsBooking);
    cnTB.Close();
    daBooking.Fill(dsBooking);
 }
```

Go to the start of the C# code and add a ***List*** variable which will be used to store the booked dates.

```
public partial class Booking : System.Web.UI.Page
{
    public static DateTime weekStart;
    public static double weekCost;
    string databaseLocation =
            "C:\\WEB APPLICATIONS\\holiday cottage bookings.mdf;";
    public static List<DateTime> bookedList = new List<DateTime>();
```

Return to the ***loadBookings*** method. Add code which uses an outer loop to access each booking, then an inner loop to add each day of the booking to the ***bookedList***.

```
        daBooking.Fill(dsBooking);
        cnTB.Close();
        daBooking.Fill(dsBooking);

        int countRecords = dsBooking.Tables[0].Rows.Count;
        string weekBeginning;
        bookedList = new List<DateTime>();
        for (int i = 0; i < countRecords; i++)
        {
            DataRow drBooking = dsBooking.Tables[0].Rows[i];
            weekBeginning = Convert.ToString(drBooking[1]);
            DateTime today = Convert.ToDateTime(weekBeginning);
            for (int d = 0; d < 7; d++)
            {
                bookedList.Add(today);
                today = today.AddDays(1);
            }
        }
    }
```

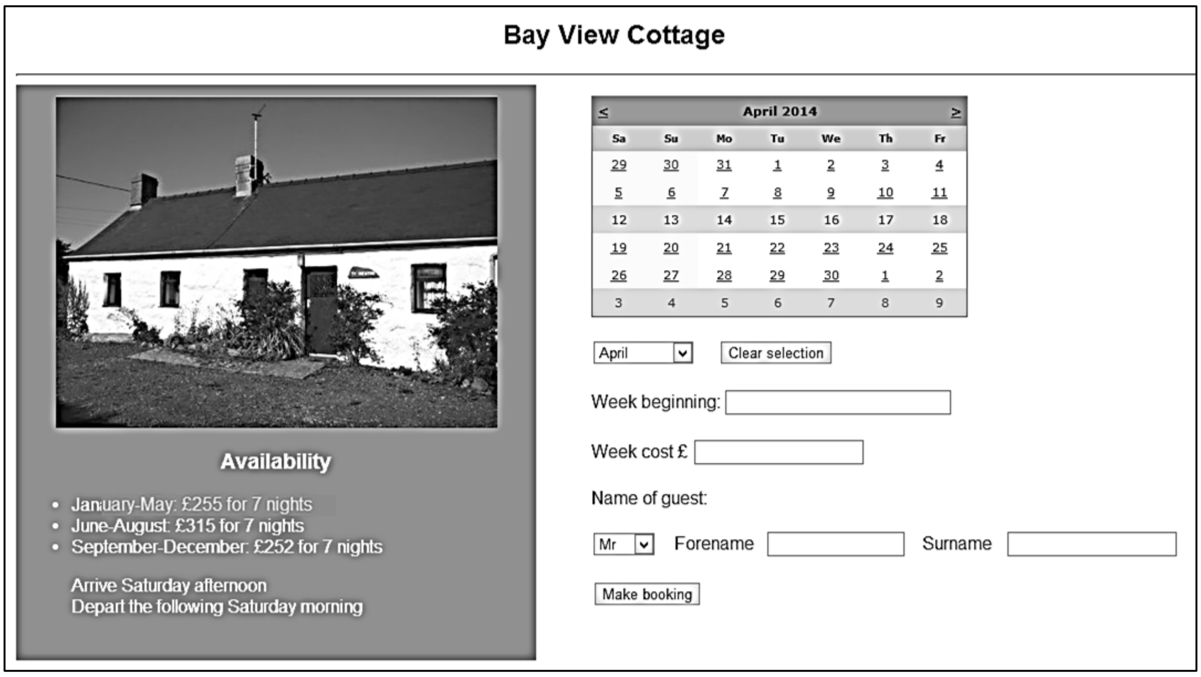
The ***loadBookings*** method needs to be called when the web page first opens. Do this by adding a line of code to the ***Page_load*** method.

```
public partial class Booking : System.Web.UI.Page
{
    public static DateTime weekStart;
    public static double weekCost;
    string databaseLocation =
              "C:\\WEB APPLICATIONS\\holiday cottage bookings.mdf;";
    public static List<DateTime> bookedList = new List<DateTime>();

    protected void Page_Load(object sender, EventArgs e)
    {
        loadBookings();
    }
```

We must update the ***DayRender*** method, so that the list of booked dates are shaded and made inactive. Locate the ***DayRender*** method and add lines of code as shown.

```
protected void Calendar1_DayRender(object sender, DayRenderEventArgs e)
{
    DateTime currentDate = Calendar1.TodaysDate;

    //show days before todays date in grey
    if (e.Day.Date < currentDate)
    {
        e.Day.IsSelectable = false;
        e.Cell.BackColor = Color.Gainsboro;
    }

    foreach (DateTime dt in bookedList)
    {
        if (e.Day.Date == dt.Date)
        {
            e.Day.IsSelectable = false;
            e.Cell.BackColor = Color.Gainsboro;
        }
    }
}
```

Build and run the web page. Check that the weeks for which you entered bookings are now shaded in grey and cannot be selected.



Bay View Cottage

Availability

- January-May: £255 for 7 nights
- June-August: £315 for 7 nights
- September-December: £252 for 7 nights

Arrive Saturday afternoon
Depart the following Saturday morning

| < | | | April 2014 | | | > |
|---|---|---|---|---|---|---|
| Sa | Su | Mo | Tu | We | Th | Fr |
| 29 | 30 | 31 | 1 | 2 | 3 | 4 |
| 5 | 6 | 7 | 8 | 9 | 10 | 11 |
| 12 | 13 | 14 | 15 | 16 | 17 | 18 |
| 19 | 20 | 21 | 22 | 23 | 24 | 25 |
| 26 | 27 | 28 | 29 | 30 | 1 | 2 |
| 3 | 4 | 5 | 6 | 7 | 8 | 9 |

April  Clear selection

Week beginning:

Week cost £

Name of guest:

Mr  Forename  Surname

Make booking

One final small task is to call the ***loadBookings( )*** method after a new booking has been made, so that the week of the booking is also shown in grey and cannot be double booked.

Locate the ***makebooking( )*** method, and add the extra line of code after the 'Record saved' message is displayed.

```
protected void makebooking(DateTime weekStart, string title, string forename,
              string surname, double weekCost)
{
    SqlConnection cnTB = new SqlConnection(@"Data Source=.\SQLEXPRESS;
        AttachDbFilename=" + databaseLocation+ "Integrated Security=True;
        Connect Timeout=30; User Instance=True");
    try
    {
        cnTB.Open();
        SqlCommand cmBooking = new SqlCommand();
        cmBooking.Connection = cnTB;
        cmBooking.CommandType = CommandType.Text;
        cmBooking.CommandText = "INSERT INTO booking(weekBeginning,
            customerTitle, forename, surname, weekCost) VALUES ('"
            + Convert.ToString(weekStart) + "','" + title + "','" + forename
            + "','" + surname + "','" + Convert.ToString(weekCost) + "')";
        lblMessage.Text = cmBooking.CommandText;
        cmBooking.ExecuteNonQuery();
        cnTB.Close();
        lblMessage.Text = "Record saved";

        loadBookings();

    }
    catch
    {
        lblMessage.Text = "File error";
    }
}
```

Build and run the web page. Add bookings, and check that these also appear as no longer available on the calendar.