



أساسيات ASP .Net 2013

الصفحة	العنوان
3	1. أساسيات ASP.NET 2013
4	1.1 مقدمة
4	2.1 لغة التنفيذ المشتركة CLR
4	3.1 لغات .NET
5	4.1 أساسيات ASP.NET
6	5.1 وثائق ASP.NET
8	6.1 الكود الخلفي
9	2. عناصر التحكم HTML Controls
10	1.2 عناصر التحكم ASP.NET
10	2.2 عناصر التحكم HTML Controls
14	3.2 دورة حياة وثيقة ASP.NET Life Cycle
18	4.2 أحداث الصفحة Page-Level Events
19	5.2 أحداث عناصر التحكم Control Events
22	3. عناصر التحكم Web Controls
23	1.5 عناصر تحكم الويب Web Controls
36	4. عناصر التحقق من الصحة
37	1.4 عناصر التحقق من الصحة

الكلمات المفتاحية

ASPX، لغة التنفيذ المشتركة CLR، وثيقة ASPX، الكود الخلفي، عناصر التحكم HTML، عناصر التحكم Web Controls.

الملخص

نستعرض في هذا الفصل أساسيات صفحات ASPX. ثم نُبين استخدام عناصر التحكم من النوع HTML ومن ثم من النوع Web Controls.

الأهداف التعليمية

يتعرف الطالب في هذا الفصل على:

- مفهوم ASPX.
- وثيقة ASPX.
- عناصر التحكم HTML.
- أحداث عناصر التحكم HTML.
- عناصر التحكم Web Controls.
- عناصر التحكم من الصحة.

المخطط

يضم فصل أساسيات ASP.NET 2013 4 وحدات (Learning Objects) هي:

- أساسيات ASP.NET 2013
- عناصر التحكم HTML Controls
- عناصر التحكم Web Controls
- عناصر التحقق من الصحة

أساسيات ASP.NET 2013

الأهداف التعليمية

- ASPX
- لغة التنفيذ المشتركة CLR
- وثيقة ASPX
- الكود الخلفي

مقدمة

تُعتبر .NET. المظلة التي تجمع مجموعة من التقانات التي أطلقتها Microsoft في عام 2002 انطلاقةً من حقيقة توزع التطبيقات على مكونات برمجية components تعمل على حواسيب مختلفة في أمكنة متعددة من العالم.

يكون المكون البرمجي component عبارة عن كبسلة لبرمجيات يُمكن أن تعمل بمفردها، أو أن تُستخدم من قبل مكونات أخرى وبدون أن تعرف هذه المكونات كيفية التحقيق البرمجي لوظائف المكونة المستخدمة. يُشكّل الإطار .NET. الإطار الذي يسمح بتطوير مكونات برمجية باستخدام لغات برمجية مختلفة، أما أدوات التطوير والنشر فتكون مستقلة عن لغة البرمجة.

لغة التنفيذ المشتركة CLR

تُعتبر لغة التنفيذ المشتركة Common Language Runtime التقانة الأساسية لـ .NET، والتي تؤمن خدمات لغة حيادية language-neutral لمعالجة وتنفيذ برمجيات .NET. ومن أهم هذه الخدمات:

- التحقق من أنماط البيانات type checking
- تتبع التنفيذ debugging
- التنظيف garbage collection
- معالجة الاستثناءات exception handling

يتوفر للغة التنفيذ المشتركة، ومن أجل أي لغة برمجة، مترجم يقوم بترجمة البرنامج المصدر إلى لغة مشتركة وسيطة تُدعى (Intermediate Language (IL. يكون لجميع البرامج IL نفس الشكل مهما كانت اللغة المصدر. يقوم المترجم Just-In Time (JIT compiler (والذي هو جزء من CLR) بترجمة طريقة method إلى لغة الآلة عند استدعاء هذه الطريقة.

لغات .NET

حوت .NET. في البداية على خمسة لغات:

- VB .NET
- C++ .NET
- JScript .NET
- J# .NET
- C#

أما اليوم فيوجد حوالي أكثر من عشرين لغة مثل: COBOL, Fortran, Perl, Python (تحتوي كل لغة على مترجم إلى IL).

توفر ميزة وجود هذا العدد من اللغات سهولة تهجير البرامج المكتوبة سابقاً إلى الإطار .NET. كما أن المبرمجين الخبراء في لغة معينة يمكن أن يستمروا باستخدام لغتهم المفضلة. وبالطبع فلا يُفضل كتابة منظومة باستخدام لغات مختلفة.

أساسيات ASP.NET

تُعتبر ASP.NET (Active Server Pages) تقانة لبناء وثائق الويب الديناميكية. تدعم هذه الوثائق تنفيذ الكود على مخدم الويب. كما يمكن أن تحوي بالطبع على خطاطات تُنفذ من جهة الزبون. يمكن كتابة الكود بأي لغة من لغات .NET. (ستتم ترجمة هذا الكود). يمكن وضع الكود ضمن الوثيقة نفسها أو في ملف مستقل يُدعى عادةً ملف الكود الخلفي code-behind file. تُتيح هذه الميزة الفصل بين مصمم الصفحات وبين المبرمج.

تتم ترجمة كل وثيقة إلى صف class يُخزن في مجمع Assembly. يكون هذا الصف مشتق من الصف System.Web.UI.Page، وبالتالي فهو يرث مجموعة من الأعضاء members التي يوفرها هذا الصف. من أكثر هذه الأعضاء استخداماً الأغراض Request و Response، والصفوف HTMLControls و WebControls، والخاصية IsPostBack.

- تُستخدم مثلاً الطريقة Write للغرض Response (من الصف Page) لكتابة خرج في وثيقة.
- يُعرّف الصفان HTMLControls و WebControls مجموعة من عناصر التحكم التي يمكن استخدامها في الوثائق.
- أما الخاصية IsPostBack فتُستخدم لمعرفة فيما إذا كان الطلب الحالي للوثيقة هو نتيجة تفاعل المستخدم مع نموذج أو (على عكس ذلك) الحالة الابتدائية لوثيقة.

تتم ترجمة الوثائق ASP.NET التي لا تحوي ملفات كود خلفي مباشرةً إلى صف مشتق من الصف System.Web.UI.Page.

بينما يكون صف الوثيقة التي تستخدم ملفات كود خلفي صف مشتق من الصف الموجود في الكود الخلفي والذي هو بدوره صف مشتق من الصف System.Web.UI.Page.² (إشارة تولتيب)

1 ندعو عادةً الصف المولد من ترجمة وثيقة بصف الوثيقة document class

2 وبهذا يكون البرنامج في هذه الحالة مشتق من صف الكود الخلفي ومن الصف System.Web.UI.Page

وثائق ASP.NET

يُمكن لوثيقة ASP.NET أن تحوي عدد من العناصر المختلفة:

1. مؤثرات XHTML

تُعرّف هذه المؤثرات عناصر الوثيقة. يُمكن جعل هذه العناصر ديناميكية باستخدام خطاطات من جهة الزبون أو كود من جهة المخدم.

2. تعليقات من جهة الزبون

حيث يتم استخدام `<!--` `-->`

3. تعليمات Directives

من الواصفات attributes. وهي تعليمة مطلوبة في كل وثيقة ASP.NET تحوي على كود برمجي. كما أن الواصفة language لهذه التعليمة مطلوبة لتحديد لغة البرمجة المستخدمة.

4. كتل تعريف Declaration blocks

يوضع كود هذه الكتل ضمن المؤثر script مع إسناد قيمة الواصفة runat إلى server:

```
<script runat = "server">
. . . . .
</script>
```

لا يتم تنفيذ كود هذه الكتل بشكل مباشر. إذ تحوي هذه الكتل عادةً على تعريف البرامج الفرعية (بما فيها معالجات الأحداث). يتم وضع هذه الكتل ضمن صف الوثيقة مباشرة عند الترجمة.

5. كتل الإعادة البرمجية Render blocks

يُكتب كود هذه الكتل ضمن `<% ... %>`. لا يُمكن أن يحوي هذا الكود على تعريف برامج فرعية. يتم وضع هذا الكود ضمن وظيفة في صف الوثيقة عند ترجمة الوثيقة إلى صف الوثيقة. تحوي هذه الكتل عادةً على استدعاءات لإجرائيات فرعية وعلى تعليمات إظهار. يتم تنفيذ هذه الوظيفة عندما يتم تنفيذ صف الوثيقة (والذي يولد وثيقة XHTML التي يتم إرجاعها إلى المتصفح الطالب).

6. تعليقات من جهة المخدم

تُستخدم `<!-- ... -->` لوضع التعليقات من جهة المخدم.

مثال:

يُبين المثال التالي استخدام مختلف العناصر السابقة. حيث تُعرّف في كتلة التعريف الغرض randomGen من الصف Random، والمتحول msg من النمط string، والمصفوفة myArray من 10 عناصر من النمط int كما تُعرّف الوظيفة fillArray() والتي تقوم بملء المصفوفة myArray بعشرة أرقام عشوائية. نقوم في كتلة الإعادة البرمجية باستدعاء الوظيفة fillArray() وإظهار قيم المصفوفة myArray.

```
<!-- ex1.aspx
A simple example of an ASP.NET document
It uses a function to fill an array with pseudorandom numbers,
which are then displayed
-->
<%@ Page language="c#" %>
<html xmlns="http://www.w3.org/1999/xhtml">
  <head> <title> Ex1 </title>
  <script runat = "server">
// Build a pseudorandom number method
    Random randomGen = new Random();
    string msg;
    private int [ ] myArray = new int[10];
// A method to fill the array with pseudorandom numbers
    public void fillArray() {
        for (int index = 0; index < 10; index++)
// Generate a pseudorandom number in the range of 0 to 100
            myArray[index] = randomGen.Next(0, 100);
    }
  </script>
</head>
<body>
<%-- Code to call the fillArray method and display the array--%>
<% fillArray();
    Response.Write(
        "<br /> <b>The array's contents are: </b><br /><br />");
    for (int index = 0; index < 10; index++) {
        msg = string.Format("The element at {0} is: {1} <br />",
                            index, myArray[index]);
        Response.Write(msg);
    }
%>
</body> </html>
```

حيث يكون الخرج:

The array's contents are:

The element at 0 is: 69
The element at 1 is: 31
The element at 2 is: 91
The element at 3 is: 34
The element at 4 is: 71
The element at 5 is: 99
The element at 6 is: 21
The element at 7 is: 28
The element at 8 is: 9
The element at 9 is: 65

الكود الخلفي

سيكون من الأفضل فصل التصريحات البرمجية عن الوثيقة، مما يسمح بفصل منطق البرنامج عن مسائل الإظهار. يُمكن مثلاً إعادة المثال السابق كما يلي:

يتم كتابة كتلة التعريف ضمن صف (ندعوه ex2) مشتق من الصف Page:

```
// ex2.aspx.cs
// The code behind file for ex2.aspx
// Includes a function to build an array of ten pseudorandom
// numbers

using System;
using System.Web;
using System.Web.UI;

public class ex2 : Page {
    // Build a pseudorandom number method
    Random randomGen = new Random();
    protected int [] myArray = new int[10];
    // A method to fill the array with pseudorandom numbers
    public void fillArray() {
        for (int index = 0; index < 10; index++)
            // Generate a pseudorandom number in the range of 0 to 100
            myArray[index] = randomGen.Next(0, 100);
    }
}
```

أما في الملف aspx فيتم في التعليمة @Page التصريح عن الوراثة من الصف السابق باستخدام الوصفة Inherits وتحديد اسم الملف الذي يحوي الصف باستخدام الوصفة Src أو CodeFile:

```
<!-- ex2.aspx
A simple example of an ASP.NET document with a code-behind file
It has the same functionality as ex1.aspx
-->
<%@ Page language="C#" Inherits = "ex2" Src = "ex2.aspx.cs" %>
<html xmlns="http://www.w3.org/1999/xhtml">
  <head> <title> Ex2 </title>
  </head>
  <body>

    <!-- Code to call the fillArray method and display the array-->

    <%
      string msg;
      fillArray();
      Response.Write(
        "<br /> <b>The array's contents are: </b><br /><br />");
      for (int index = 0; index < 10; index++) {
        msg = string.Format("The element at {0} is: {1} <br />",
          index, myArray[index]);
        Response.Write(msg);
      }
    %>
  </body>
</html>
```

عناصر التحكم HTML Controls

الأهداف التعليمية

- التعامل مع عناصر التحكم HTML في صفحات ASPX.

عناصر التحكم ASP.NET

تُدعى عناصر النموذج widgets في الوثائق ASP.NET بعناصر التحكم controls. كما ندعو الكود المرتبط مع هذه العناصر والذي يُنفذ على المخدم بكود تحكم المخدم server controls. يُعالج هذا الكود الأحداث المرتبطة مع عناصر التحكم. يوجد نوعين من عناصر التحكم:

- عناصر التحكم HTML Controls
- عناصر الويب Web Controls

عناصر التحكم HTML Controls

يتضمن الفضاء (namespace) System.Web.UI.HTML.Controls عناصر التحكم HTML. تُشابه هذه العناصر العناصر XHTML إلا أنها تتميز بأنها تتفاعل مع الكود من جهة المخدم والذي يُمكن أن يُغير ديناميكياً من محتوى وسلوك هذه العناصر، حيث يُمكن الوصول للواصفات التي تتحكم بهذه العناصر عن طريق أغراض في صف الوثيقة. يُمكن لعناصر التحكم في أغلب الأحيان أن ترفع أحد الحدثين على المخدم ServerClick أو ServerChange. يُبين الجدول التالي أنماط أكثر عناصر التحكم HTML استخداماً وعناصر XHTML الموافقة والأحداث التي ترفعها على المخدم:

Control Type	XHTML Element	Event
HtmlHidden	<input type="hidden">	ServerChange
HtmlInputButton	<input type="button">	ServerClick
	<input type="submit">	ServerClick
	<input type="reset">	ServerClick
HtmlInputCheckBox	<input type="checkbox">	ServerChange
HtmlInputRadioButton	<input type="radio">	ServerChange
HtmlInputText	<input type="text">	ServerChange
	<input type="password">	ServerChange
HtmlAnchor	<a>	ServerClick
HtmlButton	<button>	ServerClick
HtmlSelect	<select>	ServerChange
HtmlTextArea	<textarea>	ServerChange
HtmlForm	<form>	None
HtmlTable	<table>	None
HtmlTableCell	<td> and <th>	None
HtmlTableRow	<tr>	None

لاحظ أن بعض أنماط عناصر التحكم HTML ليست عناصر نموذج، أي لا يستطيع المستخدم التفاعل معها (مثل HTMLTableCell)، إلا أن وجود الأنماط يسمح بالوصول إلى واصفاتها من خلال التعليمات البرمجية مما يُمكن من تغييرها ديناميكياً.

يتم ترجمة عناصر التحكم من جهة المخدم إلى حقول fields في صف الوثيقة. ويكون معرف العنصر id اسم الحقل. ولهذا فيجب وضع معرف لعناصر التحكم حين استخدامها من جهة المخدم. كما يتم تحديد العنصر بأنه من جهة المخدم عن طريق إسناد القيمة server إلى الوصفة runat (بدون هذا يصبح العنصر عبارة عن عنصر XHTML ساكن لا يصل له المخدم). لنفرض أن لدينا مثلاً:

```
<form runat="server">
  <input type="text" id="address"
        runat="server" />
  . . .
</form>
```

يُمكن في هذه الحالة أن يحوي صف الوثيقة:

```
protected HtmlInputText address;
```

لاحظ أن المؤثر <form> لا يحوي الوصفة action وذلك لأن جميع الأحداث البرمجية التي تنجم عن تفاعل المستخدم مع عناصر التحكم تُعرف في الوثائق نفسها أو في ملف الكود الخلفي. يكون صف أي عنصر تحكم HTML مشتق من الصف HtmlControl وبالتالي فهو يرث خصائص وطرق هذا الصف. أما الخصائص والطرق الخاصة بعنصر التحكم فتكون معرفة في صف العنصر الموافق. فمثلاً، يحوي الصف HtmlAnchor الخاصية Href.

تختلف عناصر التحكم HTML عن عناصر XHTML العادية في أنها تُحول إلى أغراض في صف الوثيقة. مما يعني أن هذه العناصر وواصفاتها يُمكن الوصول لها وتغييرها من خلال التعليمات البرمجية.

مثال: استخدام عناصر HTML Controls من جهة المخدم

نقوم في المثال التالي ببناء صفحة تستخدم تستخدم عناصر HTML. إنما ستكون البرمجة من جهة المخدم أي باستخدام C#.

تهدف الصفحة إلى حساب سعر الإقامة في فندق وذلك وفق مجموعة من الخيارات التي يُمكن أن يقوم بها المستخدم:

- تصنيف الفندق: *, **, ***, ****, *****.
- إطلالة الفندق: البحر، الغابة، الجبل.
- الوجبات المطلوبة: فطور، غداء، عشاء.
- خدمات إضافية: تلفزيون، مسبح، مطبخ.

Hotel Club (HTML Controls/ Server)		
Country:	Italy ▼	
Stars:	*** ▼	
Where:	☒ Sea ☐ Forest ☐ Mountain	
Meals:	☒ Breakfast ☒ Lunch ☒ Dinner	
Extra:	TV Pool Kitchen ▼	
How Much	925	

ولتكن مثلاً الأسعار التالية المعتمدة:

Stars		Where		Food		Extra	
Stars	Price	Sea	+50%	Breakfast	50	TV	50
*	100	Forest	+30%	Lunch	100	Pool	100
**	200	Mountain	+20%	Dinner	200	Kitchen	150
***	300						
****	400						
*****	500						

يُمكن في هذه المسألة البسيطة تثبيت الأسعار في قيم عناصر HTML للرجوع إليها في إجرائية #C المستخدمة لحساب سعر الليلة:

```
<html xmlns="http://www.w3.org/1999/xhtml">
<head><title>HTML Server</title></head>
<body runat="server" bgcolor="papayawhip">
<form id="form1" runat="server">
<table border="1">
<caption>Hotel Club (HTML Controls/ Server)</caption>
<tr>
<td>Country:</td>
<td>
<select name="SelectFlag" id="SelectFlag" runat="server"
onserverchange="SelectFlag_ServerChange">
<option value="images\usa.png">USA</option>
<option value="images\france.png">France</option>
<option value="images\italy.png">Italy</option>
```

```

<option value="images\germany.png">Germany</option></select></td>
<td>
</td></tr>
<tr><td>Stars:</td>
<td>
<select id="SelectStars" name="SelectStars" runat="server">
<option value="100">*</option>
<option value="200">**</option>
<option value="300">***</option>
<option value="400">****</option>
<option value="500">*****</option></select></td><td></td></tr>
<tr><td>Where:</td>
<td>
<input id="RadWhere1" name="RadWhere" checked type="radio" runat="server"
value="1.5" />Sea<br />
<input id="RadWhere2" type="radio" name="RadWhere" runat="server"
value="1.3" />Forest<nbsp;   <br />
<input id="RadWhere3" type="radio" name="RadWhere" runat="server"
value="1.2" />Mountain<nbsp;   <br /></td>
<td></td></tr>
<tr><td>Meals:</td>
<td>
<input id="chkBreak" name="chkBreak" type="checkbox" runat="server"
value="50" />Breakfast<br />
<input id="chkLunch" name="chkLunch" type="checkbox" runat="server"
value="100" />Lunch<br />
<input id="chkDinner" name="chkDinner" type="checkbox" runat="server"
checked="CHECKED" value="200" />Dinner</td><td></td></tr>
<tr><td>Extra:</td><td>
<select id="SelectExtra" multiple name="SelectExtra" runat="server"
dir="ltr" size="3">
<option selected="selected" value="50">TV</option>
<option value="75">Pool</option>
<option value="100">Kitchen</option></select><br /></td>
<td></td></tr>
<tr>
<td>
<input id="Submit1" type="button" runat="server" value="How Much"
onserverclick="Submit1_ServerClick" /></td>
<td>
<input id="txtOut" name="txtOut" runat="server" type="text" />
</td><td></td></tr>
</table>
</form>
</body>
</html>

```

لاحظ إسناد الخاصية `runat=server` إلى عناصر التحكم HTML. كي تستطيع إجراءات #C التالية الوصول إلى هذه العناصر.

توجد أعلام الدول في المجلد `images`. تكون قيمة كل خيار من قائمة البلدان مسار ملف صورة علم البلد. تقوم إجراءات المخدم التالية والمكتوبة على حدث تغيير العنصر المحدد من قائمة البلاد بإسناد قيمة عنصر القائمة (والذي يحوي اسم ملف العلم) إلى الخاصية `src` لعنصر الصورة.

```

protected void SelectFlag_ServerChange(object sender, EventArgs e)
{
    this.imgFlag.Src = this.SelectFlag.Value;
}

```

تقوم إجرائية المخدم التالية بحساب سعر الإقامة استناداً لخيارات المستخدم. تعود هذه الوظيفة إلى قيمة كل عنصر لمعرفة سعر الخيار المحدد.

```
protected void Submit1_ServerClick(object sender, EventArgs e)
{
    double x = 0;
    x = double.Parse(this.SelectStars.Value);

    if (this.RadWhere1.Checked)
        x = x * double.Parse(this.RadWhere1.Value);
    else if (this.RadWhere2.Checked)
        x = x * double.Parse(this.RadWhere2.Value);
    else if (this.RadWhere3.Checked)
        x = x * double.Parse(this.RadWhere3.Value);

    if (this.chkBreak.Checked)
        x = x + double.Parse(this.chkBreak.Value);
    if (this.chkLunch.Checked)
        x = x + double.Parse(this.chkLunch.Value);
    if (this.chkDinner.Checked)
        x = x + double.Parse(this.chkDinner.Value);

    for (int i = 0; i < this.SelectExtra.Items.Count; i++)
        if (this.SelectExtra.Items[i].Selected)
            x = x +
                double.Parse(this.SelectExtra.Items[i].Value);

    this.txtOut.Value = x.ToString();
}
```

دورة حياة وثيقة ASP.NET Life Cycle

يكون لكل وثيقة ASP.NET نوعين من الطلب request:

- الطلب الأول initial request والذي ينتج عن طلب الوثيقة وإظهار نماذجها للمستخدم.
 - الطلب اللاحق postback request والذي يكون بعد تغيير قيم عناصر النموذج من قبل المستخدم.
- يُدعى هذا الطلب بالطلب اللاحق postback لأن قيم عناصر النموذج تُعاد إلى الوثيقة على المخدم.

لتوضيح تسلسل الأحداث لوثيقة ASP.NET تحوي نموذج لنأخذ المثال التالي:

```
<!-- ex3.aspx
A simple example of an ASP.NET document with HTML controls.
It uses textboxes to get the name and age of the client,
which are then displayed.
-->
<%@ Page language="c#" %>

<html xmlns="http://www.w3.org/1999/xhtml" >
  <head> <title> Ex3 </title>
  </head>
  <body>
    <form runat = "server">
      <p>
        Your Name:
        <input type = "text" id = "name" runat = "server" />
      </p>
    </form>
  </body>
</html>
```

```

<br />
Your age:
<input type = "text" id = "age" runat = "server" />
<br />
<input type = "submit" value = "submit" />
<br />
<% if (IsPostBack) { %>
    Hello <%= name.Value %> <br />
    You are <%= age.Value %> years old <br />
<% } %>
</p>
</form>
</body>
</html>

```

لاحظ أن كل من النموذج وعناصر التحكم يجب أن تحوي القيمة server للوصفة runat.

يقوم ASP.NET ضمناً بتخزين حالة كل عنصر تحكم لمنتسخ صف الوثيقة document class instance قبل أن يُعيد المخدم خرج المنتسخ إلى الزبون. تُخزن هذه المعلومات في عنصر تحكم مخفي على الصفحة يُدعى ViewState:

```

<input type="hidden"
name="__VIEWSTATE" id="__VIEWSTATE"

```

عندما تُعاد الوثيقة إلى المخدم تُستخدم بيانات ViewState ضمناً لإعطاء قيم ابتدائية للمنتسخ الجديد عند إعادة توليده. تسمح ViewState إذاً بالمحافظة على الحالة (قيم عناصر النموذج) بين الطلبات المتتالية لنفس الصفحة.

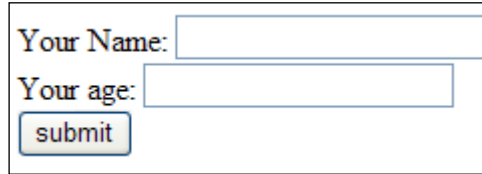
تُبين القائمة التالية الأمور التي تحدث إذا طُلبت الوثيقة السابقة، فُقدت للمتصفح، ثم قام المستخدم بتعبئة صناديق النص فيها، ثم أعادها للمخدم، ثم أُعيدت مرة أخرى للمتصفح:

1. يطلب المستخدم الوثيقة ex3.aspx.
2. تتم ترجمة الوثيقة المطلوبة في المخدم فيُنشئ صف الوثيقة ويُستدعى باني الصف.
3. يتم إعطاء قيم ابتدائية لحالة عناصر التحكم باستخدام ViewState (في الطلب الأول لا تحوي ViewState أية بيانات).
4. تُستخدم بيانات النموذج للطلب لإسناد حالة عناصر التحكم لمنتسخ صف الوثيقة (لا يوجد بيانات نموذج في الطلب الأول).
5. تُسجل الحالة الحالية للمنتسخ في الحقل المخفي ViewState.
6. يُنفذ المنتسخ وتُعاد النتيجة إلى الزبون.
7. يُمسح كل من صف الوثيقة والمنتسخ من المخدم.
8. يقوم المستخدم بالتفاعل مع النموذج في الوثيقة (تغيير قيم بعض عناصر النموذج).
9. يقوم المستخدم بطلب إعادة postback إلى المخدم (عند النقر على زر الإرسال submit مثلاً).
10. تتم ترجمة الوثيقة المطلوبة في المخدم فيُنشئ صف الوثيقة ويُستدعى باني الصف.
11. يتم إسناد حالة عناصر التحكم في المنتسخ باستخدام ViewState.
12. تُستخدم بيانات نموذج الطالب لإسناد قيم لحالة عناصر التحكم لمنتسخ صف الوثيقة.

13. تُسجل الحالة الحالية للصف في ViewState.

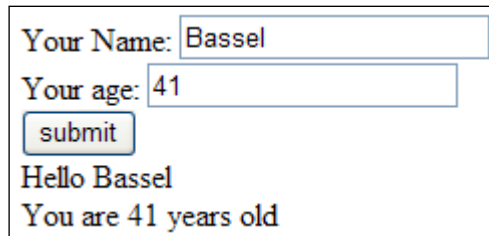
14. يتم تنفيذ المنتسخ وتُعاد النتيجة إلى الزبون. ويُمسح الصف والمنتسخ من المخدم.

تكون الوثيقة المنشأة من صف الوثيقة المترجم من الوثيقة ex3.aspx كما يلي:



```
<html xmlns="http://www.w3.org/1999/xhtml" >
  <head> <title> Ex3 </title>
  </head>
  <body>
    <form name="ctl00" method="post"
      action="ex3.aspx" id="ctl00">
    <div>
    <input type="hidden"
name="__VIEWSTATE"
id="__VIEWSTATE"
value="/wEPDwULLTEzNTg2NzExNzZkZPMc7/+Qlyt5as8R30c8Ya5ME5fW" />
    </div>
    <p>
      Your Name:
      <input name="name" type="text" id="name" />
      <br />
      Your age:
      <input name="age" type="text" id="age" />
      <br />
      <input type = "submit" value = "submit" />
    <br />
    </p>
    <div>
      <input type="hidden"
name="__EVENTVALIDATION"
id="__EVENTVALIDATION"
value="/wEWAwKL1uvQCgL7uPQdAtCCr6oGMa08C59NSjolaHXJlcGerJ0RLfK=" />
    </div></form>
  </body>
</html>
```

وبعد أن تُعبأ من قبل المستخدم وترسل للمخدم (بالنقر على submit) كما يلي:



```

<html xmlns="http://www.w3.org/1999/xhtml">
<head>
  <title>Ex3 </title>
</head>
<body>
  <form name="ctl00" method="post"
    action="ex3.aspx" id="ctl00">
    <div>
      <input type="hidden" name="__VIEWSTATE" id="__VIEWSTATE"
        value="/wEPDwULLTEzNTg2NzExNzZkZPMc7/+Q1yt5as8R30c8Ya5ME5fW"
      />
    </div>
    <p>
      Your Name:
      <input name="name" type="text" id="name" value="Bassel" />
      <br />
      Your age:
      <input name="age" type="text" id="age" value="40" />
      <br />
      <input type="submit" value="submit" />
      <br />
      Hello Bassel
      <br />
      You are 40 years old
      <br />
    </p>
    <div>
      <input type="hidden" name="__EVENTVALIDATION"
        id="__EVENTVALIDATION"
        value="/wEWAwKL1uvQCgL7uPQdAtCCr6oGMA08C59NSjo1aHXJ1cGerJ0RLfk=" />
    </div>
  </form>
</body></html>

```

تختلف هذه الوثيقة عن النسخة الأصلية من ex3.aspx في ثلاثة مواضع:

1. تحوي عنصر التحكم المخفي ViewState والذي يحوي بيانات النموذج بشكل مشفر.
2. يكون للنموذج اسم ومعرف (ctl00).
3. تم استبدال كتلة الإعادة البرمجية بخرجها.

يُمكن طلب الإرجاع (أي إرسال قيم عناصر النموذج إلى المخدم) postback من قبل المستخدم بعدة طرق. فبالطبع، يحدث الإرجاع عند نقر المستخدم على زر الإرسال submit. كما أنه يحدث عند النقر على أي زر أمر. كذلك يُمكن للمبرمج تحديد طلب الإرجاع لمختلف عناصر التحكم كصناديق الخيار مثلاً وذلك بإسناد القيمة true إلى الخاصية AutoPostBack، وعندها يتم الإرجاع عند تغيير قيمة العنصر.

أحداث الصفحة Page-Level Events

يُنشئ الصف Page أربعة أحداث للصفحة:

- الحدث Init والذي يُرفع مباشرةً بعد إنشاء المنتسخ صف الوثيقة.
- الحدث Load والذي يُرفع بعد إسناد حالة المنتسخ من بيانات النموذج و ViewState.
- الحدث PreRender والذي يُرفع قبل تنفيذ المنتسخ لبناء الوثيقة الجواب للزيون.
- الحدث Unload والذي يُرفع قبل مسح المنتسخ.

يكفي لكتابة معالجات هذه الأحداث التصريح عن الإجراءات باستخدام أسمائها المعرفة مسبقاً والمسجلة ضمناً عند إنشاء صف الوثيقة.

مثال:

يُبين المثال التالي التصريح عن الأحداث الأربع السابقة وتسلسل وقوعها:

```
<%@ Page language="c#" %>

<script runat="server">

    protected void Page_Load(object sender, EventArgs e)
    {
        T.Value = T.Value + " Load ";
    }
    protected void Page_Unload(object sender, EventArgs e)
    {
        T.Value = T.Value + " Unload ";
    }
    protected void Page_PreRender(object sender, EventArgs e)
    {
        T.Value = T.Value + " PreRender ";
    }
    protected void Page_Init(object sender, EventArgs e)
    {
        T.Value = T.Value + " Init ";
    }
}
</script>

<html xmlns="http://www.w3.org/1999/xhtml" >
    <head> <title> Page Events </title>
    </head>
    <body>
        <form runat = "server" id="MyForm" >
            Page Events Sequence:
            <input id="T" runat="server" />
        </form>
    </body>
</html>
```

حيث يكون الخرج:

Page Events Sequence: Init Load PreRender

- يوجد طريقتين لإنشاء وتسجيل معالج حدث لعنصر HTML. تُسند في الطريقة الأولى اسم إجرائية معالج الحدث إلى الوصفة OnServerClick أو الوصفة OnServerChange. يكون لإجرائية معالج الحدث **البروتوكول** وهو البروتوكول للتفويض (delegate) EventHandler والذي يُعرّف المنهج القياسي لمعالجة الأحداث في لغة التنفيذ الموحدة CLR التالي:
- تُعيد void.
 - يوجد لها معاملي دخل الأول من النوع object، والثاني من النوع System.EventArgs.

أحداث عناصر التحكم Control Events

مثال:

يُبين المثال التالي الطريقة الأولى لتسجيل معالج الحدث.

```
protected void TextHandler (object sender, EventArgs e)
{
    . . .
}
<input type="text" id="Name"
    OnServerChange="TextHandler"
    Runat="server" />
```

تعتمد الطريقة الثانية على كتابة وتسجيل الحدث باستخدام المنهج القياسي في CLR، والذي يستخدم التفويض لتسجيل معالج الحدث. حيث نتبع الخطوات الثلاث التالية:

- كتابة معالج الحدث والذي هو عبارة عن وظيفة تُعيد void، ولها معاملي دخل من النوع object و EventArgs.
- إنشاء منتسخ جديد من النمط EventHandler باستخدام new وتمرير اسم الوظيفة السابقة للبان.
- تسجيل منتسخ التفويض السابق إلى خاصية حدث عنصر التحكم بإضافته إلى المنتسخات المسجلة سابقاً.

يتم عادة دمج الخطوتين السابقتين في تعليمة واحدة توضع في معالج الحدث Page_Init كما يُبين المثال التالي:

```
protected void TextboxHandler(object sender, EventArgs e)
{
    . . .
}
```

```
protected void Page_Init(object sender, EventArgs e)
{
    Name.ServerChange +=
        New EventHandler(TextBoxHandler);
}
...
<input type="text" id="Name" runat="server" />
```

مثال:

يُبين المثال التالي استخدام الطريقتين السابقتين وتسلسل وقوع الأحداث:

```
<%@ Page language="c#" %>
<script runat="server">

    protected void Page_Load(object sender, EventArgs e)
    {
        T.Value = T.Value + " Load ";
    }
    protected void Page_Unload(object sender, EventArgs e)
    {
        T.Value = T.Value + " Unload ";
    }
    protected void Page_PreRender(object sender, EventArgs e)
    {
        T.Value = T.Value + " PreRender ";
    }
    protected void Page_Init(object sender, EventArgs e)
    {
        T.Value = T.Value + " Init ";
        T2.ServerChange += new EventHandler(T2Handler);
    }

    protected void T1Handler(object sender, EventArgs e)
    {
        T.Value = T.Value + " T1 Handler ";
        T1.Style.Add(HtmlTextWriterStyle.BackgroundColor, "red");
    }
    protected void T2Handler(object sender, EventArgs e)
    {
        T.Value = T.Value + " T2 Handler ";
        T2.Style.Add(HtmlTextWriterStyle.BackgroundColor, "blue");
    }
}
</script>

<html xmlns="http://www.w3.org/1999/xhtml" >
    <head> <title> Control Events </title>
    </head>
    <body>
        <form runat = "server" id="MyForm" >
```

```

        <br/>
        <input type="text" id="T1" onserverchange="T1Handler"
runat="server" />
        <br/>

        <input type="text" id="T2" runat="server" />
        <br/>
        <input type = "submit" value = "submit" />
        <br/>
        Events Sequence:
        <input id="T" runat="server" style="width: 424px" />
    </form>
</body>
</html>

```

حيث يظهر بعد النقر على الزر :submit

T1

T2

submit

Events Sequence: Init Load PreRender Load T1 Handler T2 Handler PreRender

عناصر التحكم Web Controls

الأهداف التعليمية

- التعرف على عناصر التحكم Web Controls.

عناصر تحكم الويب Web Controls

يتضمن الفضاء System.Web.UI.WebControls عناصر تحكم الويب وهي مجموعة أكبر وأغنى من عناصر التحكم HTML. فبالإضافة إلى عناصر التحكم التي توافق العناصر العادية XHTML، يوجد عناصر عديدة أخرى.

فمثلاً، يوجد عناصر تحكم لقوائم صناديق الاختيار checkbox list، قوائم أزرار الخيار radio button lists، القوائم المنسدلة drop-down lists، القوائم list boxes. كذلك يوجد عناصر تحكم خاصة للربط مع قواعد البيانات وللتحقق من صحة القيم المدخلة.

يُبين الجدول التالي أهم عناصر تحكم الويب والتقابل، إن وجد، مع عناصر XHTML:

Web Control	XHTML Element
<ASP:HyperLink>	<a>...
<ASP:LinkButton>	<a>
<ASP:Image>	
<ASP:Panel>	...
<ASP:Label>	...
<ASP:Button>	<input type="submit"/> or or <input type="reset"/>
<ASP:TextBox>	or or <textarea>...</textarea>
<ASP:CheckBox>	☐
<ASP:RadioButton>	☐
<ASP:ImageButton>	<input type="image"/>
<ASP:Table>	<table>...</table>
<ASP:TableRow>	<tr>...</tr>
<ASP:TableCell>	<td>...</td>
AdRotator	None
Calender	None
CheckListBox	None
RadioButtonList	None

توجد جميع عناصر تحكم الويب ضمن الفضاء asp، وبالتالي فيجب أن تبدأ جميع أسماء المؤثرات بـ asp.:
فمثلاً، لتعريف مربع نص نكتب:

```
<asp:textbox id="phone" runat="server" />
```

توفر عناصر تحكم الويب أسلوب برمجي أكثر تجانساً من عناصر HTML، وهي تُعاد render للزبون كتركيب من عناصر التحكم XHTML.

تملك عناصر تحكم الويب مجموعة من الخصائص والطرق والأحداث. يُبين الجدول التالي أهم الخصائص المشتركة والموروثة من الصف WebControls:

Property	Description
Attributes	إعادة جميع الثنائيات (خاصية، قيمة الخاصية)
AccessKey	إعادة مفتاح الاختصار الذي يساعد على اختيار العنصر
BackColor	لون الخلفية
Border Color	لون الإطار
BorderStyle	نمط الإطار
BorderWidth	عرض الإطار
ClientID	معرف العنصر المعطى
Controls	عناصر التحكم الأبناء التابعة للعنصر الحالي
Enabled	مفعّل أم لا
EnableViewState	قيمة منطقية تُحدد فيما إذا العنصر سيُحافظ على قيمة ViewState لهذا العنصر ولجميع أبنائه
Font	الخط
ForeColor	اللون
Height	الارتفاع

Id	معرف العنصر
Page	مؤشر إلى الصفحة التي تحوي العنصر
Parent	مؤشر لأب العنصر
Style	نمط العنصر
TabIndex	تسلسل ترتيب العنصر
ToolTip	نص المساعدة
Visible	مرئي
Width	العرض

يُبين الجدول التالي أهم الطرق المشتركة والموروثة من الصف WebControls:

Method	Description
DataBind	تفعيل الربط بمصدر البيانات
FindControl	البحث ضمن الكائن الحالي عن عنصر تحكم معين
HasControls	قيمة منطقية فيما إذا كان لعنصر التحكم أبناء

كذلك تملك عناصر تحكم الويب خصائص وطرق مميزة لكل عنصر. يُبين الجدول التالي أهمها:

Control	Properties	Events
HyperLink	ImageUrl, NavigateUrl, Target, Text	-
LinkButton	CommandArgument, CommandName, Text, CausesValidation	OnClick, OnCommand
Image	AlternateText, ImageAlign, ImageUrl	-
Panel	BackImageUrl, HorizontalAlign, Wrap	-
Label	Text	-
Button	CommandArgument, CommandName, Text, CausesValidation	OnClick, OnCommand
TextBox	AutoPostBack, Columns, MaxLength, ReadOnly, Rows, Text, TextMode, Wrap	OnTextChanged
CheckBox	AutoPostBack, Checked, Text, TextAlign	OnCheckChanged
RadioButton	AutoPostBack, Checked, GroupName, Text, TextAlign	OnCheckChanged
ImageButton	CommandArgument, CommandName, CausesValidation	OnClick, OnCommand
Table	BackImageUrl, CellPadding, CellSpacing, GridLines, HorizontalAlign, Rows	-
TableRow	Cells, HorizontalAlign, VerticalAlign	-
TableCell	ColumnSpan, HorizontalAlign, RowSpan, Text, VerticalAlign, Wrap	-
Literal	Text	-
Placeholder	-	-

إنشاء عناصر التحكم ضمن الكود

يُمكن إنشاء عناصر تحكم الويب إما بتعريفها مباشرة:

```
<asp:button id="helpButton" Text="help"
    OnClick="OnClickHandler"
    Runat="server" />
```

أو بإنشائها ضمن كود C#:

```
protected Button helpButton= new Button();
helpButton.Text="help";
helpButton.id=" helpButton";
helpButton.OnClick="OnClickHandler ";
helpButton.runat=" server";
```

يُمكن عند إنشاء عناصر الويب برمجياً استخدام عنصر التحكم placeholder وذلك لتحديد مكان توضع عناصر التحكم. فمثلاً، يُمكن أولاً إنشاء عنصر تحكم placeholder:

```
<asp:placeholder id="buttonPlace" runat="server" />
```

ووضع الزر المنشئ داخله باستخدام التعليمة:

```
buttonPlace.Controls.Add(helpButton);
```

خرج عناصر التحكم

يُمكن استخدام الطريقة Response.Write لإظهار خرج. إلا أنه من الأفضل، وللتحكم بموضع الإخراج، استخدام عنصر التحكم Label للإظهار ضمنه وذلك بإسناد الخرج إلى الخاصية Text. لنفرض مثلاً أن الوثيقة تحوي العنصر Label التالي في الموضع الذي نريد إظهار الخرج فيه:

```
<asp:label id="output" runat="server" />
```

تقوم التعليمات التالية بوضع نص معين في العنصر السابق:

```
<% string msg= string.Format(
    "The result is {0} <br />", result);
output.Text=msg ; %>
```

مثال 1:

يقوم المثال التالي بإنشاء صندوق نص TextBox وقائمة منسدلة Drop-Down List وزر أمر Button في وثيقة ASP.NET. كما يستخدم الكود الخلفي لملء عناصر القائمة المنسدلة. تحوي الوثيقة أيضاً عنصر تحكم Label لتأمين مكان وضع الرسالة الصادرة عن الكود الخلفي. يحوي الكود الخلفي معالج لحدث النقر على زر الأمر لكتابة نص معين حسب العنصر المختار من القائمة.

تتم إضافة عنصر إلى القائمة المنسدلة باستخدام الطريقة Add على الخاصية Items للقائمة. كما يتم إنشاء كل عنصر باستدعاء باني الصف ListItem مع تمرير قيمة العنصر. فمثلاً، لإضافة عنصر إلى القائمة mySelect مع القيمة "red":

```
mySelect.Items.Add(new ListItem("red"))
```

```
<!-- ex4.aspx
An example of an ASP.NET document that creates a textbox,
a drop-down list, a submit button, and a label.
A code-behind file is used to populate the drop-down list and
handle the button clicks. The label is used for the return
message
-->
<%@ Page language="c#" Inherits = "ex4" Src = "ex4.aspx.cs" %>

<html xmlns="http://www.w3.org/1999/xhtml">
<head> <title> Ex4 </title>
</head>
<body>
  <form id="Form1" runat = "server">
    Name: <asp:TextBox runat = "server" id = "name" />
    <br /><br />
    Favorite Color:<asp:DropDownList runat = "server"
                                id = "color" />

    <br /><br />
    <asp:button runat = "server" id = "submit"
               text = "Submit" OnClick = "OnClickHandler" />

    <br /><br />
    <asp:label id = "message" runat = "server" />
  </form>
</body>
</html>
```

وملف الكود الخلفي .aspx.cs:

```
// The code behind file for ex4.aspx.
// In an OnLoad handler, it populates the drop-down
// list created in the associated ASP.NET document.
// It also includes a handler for the button, which
// produces a message to the client, including the
// client's name and the chosen item from the drop
// down list

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

```

public class ex4 : System.Web.UI.Page
{
    protected DropDownList color;
    protected TextBox name;
    protected Button submit;
    protected Label message;

    // OnLoad handler to populate the dropdownlist

    override protected void OnLoad(EventArgs e)
    {
        if (!IsPostBack)
        {
            color.Items.Add(new ListItem("blue"));
            color.Items.Add(new ListItem("red"));
            color.Items.Add(new ListItem("green"));
            color.Items.Add(new ListItem("yellow"));
        }
    }

    // Handler for the button

    protected void OnClickHandler(object src, EventArgs e)
    {
        string newMsg = string.Format(
            "Hi {0}, your favorite color is {1}",
            name.Text, color.SelectedItem);
        message.Text = newMsg;
    }
}

```

حيث يُظهر المتصفح:

وبعد اختيار عنصر من القائمة المنسدلة والنقر على زر الأمر:

مثال 2:

يستخدم المثال التالي مجموعة من عناصر التحكم الأساسية لتحقيق نموذج تسجيل بيانات.

```
<%@ Page Language="C#" %>
<!DOCTYPE html >
<html xmlns="http://www.w3.org/1999/xhtml" >
  <head id="Head1" runat="server">
    <title>Web Controls Demonstration</title>
  </head>
  <body>
    <form id="form1" runat="server">
      <div>
        <h3>This is a sample registration form.</h3>
        <p><em>Please fill in all fields and click Register.</em></p>
        <p>
          <asp:Image ID="UserInformationImage" runat="server"
            ImageUrl="~/Images/user.png"
            EnableViewState="False" />
          &nbsp;
          <span style="color: teal">
            Please fill out the fields below.</span>
        </p>
        <table>
          <tr>
            <td style="width: 230px; height: 21px"
              valign="top">
              <asp:Image ID="FirstNameImage" runat="server"
                ImageUrl="~/Images/fname.png"
                EnableViewState="False" />
              <asp:TextBox ID="FirstNameTextBox"
                runat="server"
                EnableViewState="False"></asp:TextBox>
            </td>
            <td style="width: 231px; height: 21px"
              valign="top">
              <asp:Image ID="LastNameImage" runat="server"
                ImageUrl="~/Images/lname.png"
                EnableViewState="False" />
              <asp:TextBox ID="LastNameTextBox" runat="server"
                EnableViewState="False"></asp:TextBox>
            </td>
          </tr>
          <tr>
            <td style="width: 230px; height: 21px"
              valign="top">
              <asp:Image ID="EmailImage" runat="server"
                ImageUrl="~/Images/email.png"
                EnableViewState="False" />
              <asp:TextBox ID="EmailTextBox" runat="server"
                EnableViewState="False"></asp:TextBox>
            </td>
            <td style="width: 231px; height: 21px"
              valign="top">
              <asp:Image ID="PhoneImage" runat="server"
                ImageUrl="~/Images/phone.png"
                EnableViewState="False" />
              <asp:TextBox ID="PhoneTextBox" runat="server"
                EnableViewState="False"></asp:TextBox>
              Must be in the form (555) 555-5555.
            </td>
          </tr>
        </table>
      </div>
    </form>
  </body>
</html>
```

```

<p>
    <asp:Image ID="PublicationsImage" runat="server"
        ImageUrl="~/Images/publications.png"
        EnableViewState="False" />
    &nbsp;
    <span style="color: teal">
        Which book would you like information about?</span>
</p>
<p>
    <asp:DropDownList ID="BooksDropDownList"
        runat="server"
        EnableViewState="False">
        <asp:ListItem>Visual Basic 2013 How to Program 3e
        </asp:ListItem>
        <asp:ListItem>Visual C# 2013 How to Program 2e
        </asp:ListItem>
        <asp:ListItem>Java How to Program 6e</asp:ListItem>
        <asp:ListItem>C++ How to Program 5e</asp:ListItem>
        <asp:ListItem>XML How to Program 1e</asp:ListItem>
    </asp:DropDownList>
</p>
<p>
    <asp:HyperLink ID="BooksHyperLink" runat="server"
        NavigateUrl="http://www.deitel.com" Target="_blank"
        EnableViewState="False">
        Click here to view more information about our books
    </asp:HyperLink>
</p>
<p>
    <asp:Image ID="OSImage" runat="server"
        ImageUrl="~/Images/os.png"
        EnableViewState="False"/>
    &nbsp;
    <span style="color: teal">
        Which operating system are you using?</span>
</p>
<p>
    <asp:RadioButtonList
        ID="OperatingSystemRadioButtonList"
        runat="server" EnableViewState="False">

        <asp:ListItem>Windows 8</asp:ListItem>
        <asp:ListItem>Windows 7</asp:ListItem>
        <asp:ListItem>Windows XP</asp:ListItem>
        <asp:ListItem>Linux</asp:ListItem>
        <asp:ListItem>Other</asp:ListItem>
    </asp:RadioButtonList>
</p>
<p>
    <asp:Button ID="RegisterButton" runat="server"
        Text="Register" EnableViewState="False" />
</p>
</div>
</form>
</body>
</html>

```

This is a sample registration form.

Please fill in all fields and click Register.

User Information

Please fill out the fields below.

First Name

Bassel

Last Name

ALKHATIB

Email

t_balkhatib@svuonline.org

Phone

Must be in the form (555) 555-5555.

Publications

Which book would you like information about?

Visual Basic 2013 How to Program 3e

[Click here to view more information about our books](#)

Operating System

Which operating system are you using?

☒ Windows 8

☐ Windows 7

☐ Windows XP

☐ Linux

☐ Other

Register

مثال 3:

نقوم في المثال التالي ببناء صفحة تستخدم تستخدم عناصر Web Controls. ستكون البرمجة طبعاً من جهة المخدم أي باستخدام C#. تهدف الصفحة التالية إلى حساب سعر الإقامة في فندق وذلك وفق مجموعة من الخيارات التي يُمكن أن يقوم بها المستخدم:

- تصنيف الفندق: *, **, ***, ****, *****.
- إطلالة الفندق: البحر، الغابة، الجبل.
- الوجبات المطلوبة: فطور، غداء، عشاء.
- خدمات إضافية: تلفزيون، مسبح، مطبخ.
- خدمات إضافية: تلفزيون، مسبح، مطبخ.

نتكن مثلاً الأسعار التالية المعتمدة:

Stars		Where		Food		Extra	
Stars	Price	Sea	+50%	Breakfast	50	TV	50
*	100	Forest	+30%	Lunch	100	Pool	100
**	200	Mountain	+20%	Dinner	200	Kitchen	150
***	300						
****	400						
*****	500						

نقوم في هذه المسألة بتهيئة الأسعار في قيم عناصر Web Controls للرجوع إليها في إجرائية C# المستخدمة لحساب سعر الليلة.

```
<html xmlns="http://www.w3.org/1999/xhtml" >
<head runat="server"><title>Web Controls</title></head>
<body bgcolor="#ccffcc">
<form id="form1" runat="server">
<div>
<table border="1">
<caption>Hotel Club (Web Controls)</caption>
<tr>
<td style="width: 100px">Country:</td>
<td style="width: 100px">
<asp:DropDownList ID="lstCty" runat="server" AutoPostBack="True"
OnSelectedIndexChanged="lstCty_SelectedIndexChanged"
Style="position: relative">
<asp:ListItem Value="images\usa.png">USA</asp:ListItem>
<asp:ListItem Value="images\france.png">France</asp:ListItem>
<asp:ListItem Value="images\germany.png">Germany</asp:ListItem>
<asp:ListItem Value="images\italy.png">Italy</asp:ListItem>
</asp:DropDownList></td><td style="width: 100px">
<asp:Image ID="imgFlag" runat="server" ImageUrl="~/images/usa.png"
Style="position: relative" /></td></tr><tr>
<td style="width: 100px">Stars:</td>
<td style="width: 100px">
<asp:DropDownList ID="lstStars" runat="server" Style="position: relative">
<asp:ListItem Value="100">*</asp:ListItem>
<asp:ListItem Value="200">**</asp:ListItem>
<asp:ListItem Value="300">***</asp:ListItem>
<asp:ListItem Value="400">****</asp:ListItem>
<asp:ListItem Value="500">*****</asp:ListItem>
</asp:DropDownList></td></tr>
<tr>
<td style="width: 100px">Where:</td>
<td style="width: 100px">
<asp:RadioButtonList ID="lstWhere" runat="server" Style="position:
relative">
<asp:ListItem Value="1.5" Selected="True">Sea</asp:ListItem>
<asp:ListItem Value="1.3">Mountain</asp:ListItem>
<asp:ListItem Value="1.2">Forest</asp:ListItem>
</asp:RadioButtonList></td><td style="width: 100px"></td></tr>
<tr><td style="width: 100px">Meals:</td>
<td style="width: 100px">
```

```

<asp:CheckBoxList ID="lstFood" runat="server" Style="position: relative">
<asp:ListItem Value="50" Selected="True">Breakfast</asp:ListItem>
<asp:ListItem Value="100" Selected="True">Lunch</asp:ListItem>
<asp:ListItem Value="200">Dinner</asp:ListItem>
</asp:CheckBoxList></td><td style="width: 100px"></td></tr>
<tr>
<td style="width: 100px">Extra:</td>
<td style="width: 100px">
<asp:ListBox ID="lstExtra" runat="server" Style="position: relative"
SelectionMode="Multiple">
<asp:ListItem Value="50">TV</asp:ListItem>
<asp:ListItem Value="100">Pool</asp:ListItem>
<asp:ListItem Value="150">Kitchen</asp:ListItem></asp:ListBox></td>
<td style="width: 100px"></td></tr>
<tr>
<td style="width: 100px">
<asp:Button ID="Button1" runat="server" OnClick="Button1_Click"
Style="position: relative" Text="How Much" /></td>
<td style="width: 100px">
<asp:TextBox ID="txtOut" runat="server" Style="position:
relative">0</asp:TextBox></td>
<td style="width: 100px"></td></tr></table></div></form>
</body>
</html>

```

توجد أعلام الدول في المجلد images. تكون قيمة كل خيار من قائمة البلدان مسار ملف صورة علم البلد. تقوم إجرائية المخدم التالية والمكتوبة على حدث تغيير العنصر المحدد من قائمة البلاد بإسناد قيمة عنصر القائمة (والذي يحوي اسم ملف العلم) إلى الخاصية imageUrl لعنصر الصورة.

```

protected void lstCty_SelectedIndexChanged(object sender,EventArgs e)
{
    this.imgFlag.imageUrl = this.lstCty.SelectedValue;
}

```

تقوم إجرائية المخدم التالية بحساب سعر الإقامة استناداً لخيارات المستخدم. تعود هذه الوظيفة إلى قيمة كل عنصر لمعرفة سعر الخيار المحدد.

```

protected void Button1_Click(object sender, EventArgs e)
{
    double x = 0;

    x = double.Parse(this.lstStars.SelectedValue);

    x = double.Parse(this.lstWhere.SelectedValue) * x;

    for (int i = 0; i < this.lstFood.Items.Count; i++)
        if (this.lstFood.Items[i].Selected)
            x = x + double.Parse(this.lstFood.Items[i].Value);
    for (int i = 0; i < this.lstExtra.Items.Count; i++)
        if (this.lstExtra.Items[i].Selected)
            x = x + double.Parse(this.lstExtra.Items[i].Value);

    this.txtOut.Text = x.ToString();
}

```

Hotel Club (Web Controls)

Country:	Italy ▼	
Stars:	*** ▼	
Where:	☒ Sea ☐ Mountain ☐ Forest	
Meals:	☒ Breakfast ☒ Lunch ☒ Dinner	
Extra:	TV Pool Kitchen	
How Much	950	

عناصر التحقق من الصحة

الأهداف التعليمية

- استخدام عناصر التحقق من الصحة.

عناصر التحقق من الصحة

استعرضنا سابقاً كيفية كتابة تعليمات JavaScript للتحقق من صحة القيم المدخلة على الزبون. يوجد العديد من الأسباب التي تستدعي إعادة التحقق من صحة القيم المدخلة على المخدم لاسيما أنه يُمكن لمستخدم مراوغ تجنب عمليات التحقق من جهة الزبون. يوجد ستة عناصر تحكم ويب للتحقق من صحة القيم المدخلة. يُبين الجدول التالي العناصر الأربعة الأكثر استخداماً:

Control	Properties	Values
RequiredFieldValidator	None	None
CompareValidator	Operator	Equal, NotEqual, GreaterThan, GreaterThanEqual, LessThan, LessThanEqual, DataTypeCheck
	Type	String, Currency, Date, Double, Integer
	ValueToCompare	Constant
	ControlToCompare	Another Control
RangeValidator	MaximumValue	Constant
	MinimumValue	Constant
	Type	String, Currency, Date, Double, Integer
RegularExpressionValidator	ValidationExpression	Regular Expression

يتمّ وضع عناصر التحقق من الصحة مباشرةً بعد العناصر التي نريد التحقق من قيمها، وذلك كي تظهر رسائل الخطأ الموافقة جانبها.

يتمّ كتابة رسالة الخطأ في الخاصية ErrorMessage لعنصر التحقق من الصحة. كما يتمّ الربط بين عنصر التحقق من الصحة وعنصر التحكم المطلوب التحقق من قيمه بوضع معرفه في الخاصية ControlToValidate لعنصر التحقق.

مثال:

يُبين المثال التالي استخدام بعض عناصر التحقق من الصحة حيث يكون حقل الاسم في النموذج واجب الإدخال، ورقم الهاتف من الشكل ddd-ddd-dddd، والعمر محصور بين 10 و100.

```
<!-- ex5.aspx
An example of an ASP.NET document to illustrate server-side
validation Web controls.
It uses Web control textboxes to get the name, phone number,
and age of the client. These three are validated on the
server
1. The name must be present
2. The phone number must be in the form ddd-ddd-dddd
3. The range of the age must be 10 to 110
-->
<%@ Page language="c#" %>
<html>
<head> <title> Ex5 </title>
</head>
<body>
<form id="Form1" runat = "server">
<p>
Your name:
<asp:TextBox id = "name" runat = "server" />
<asp:RequiredFieldValidator ID="RequiredFieldValidator1"
ControlToValidate = "name"
Display = "Static"
runat = "server"
ErrorMessage = "Please enter your name">
</asp:RequiredFieldValidator>
<br />
Your phone number:
<asp:TextBox id = "phone" runat = "server" />
<asp:RegularExpressionValidator
ID="RegularExpressionValidator1"
ControlToValidate = "phone"
Display = "Static"
runat = "server"
ErrorMessage = "Phone number form must be ddd-ddd-dddd"
ValidationExpression = "\d{3}-\d{3}-\d{4}">
</asp:RegularExpressionValidator>
<br />
Your age:
<asp:TextBox id = "age" runat = "server" />
<asp:RangeValidator ID="RangeValidator1"
ControlToValidate = "age"
Display = "Static"
runat = "server"
MaximumValue = "110"
MinimumValue = "10"
Type = "Integer"
ErrorMessage = "Age must be in the range of 10 to 110">
</asp:RangeValidator>
<br />
<input type = "submit" value = "Submit" />
</p>
</form>
</body>
</html>
```

عند إدخال قيم غير محققة لأحد الشروط تظهر رسالة خطأ موافقة بجانب عنصر التحكم المعني:

Your name:		Please enter your name
Your phone number:	2222	Phone number form must be ddd-ddd-dddd
Your age:	140	Age must be in the range of 10 to 110
Submit		

ولا تظهر رسائل خطأ عند إدخال قيم موافقة للشروط:

Your name:	Bassel
Your phone number:	111-222-3333
Your age:	41
Submit	