

الفصل الثاني

سلسلة

ASP.NET

خطوة بخطوة حتى الاحتراف

C# & VB

اعداد المهندس

محمد عمر الحاج خلف

في هذا الفصل

- ✓
- ✓ عرض التقويم
- ✓
- ✓ تقسيم الصفحة لمناظير عديدة
- ✓ تقسيم العمل لمجموعة خطوات

استخدام أدوات التحكم الغنية

في الفصل السابق قمنا بدراسة الأدوات القياسية ASP.Net ، سنتناول في هذا الفصل دراسة أدوات تحكم أكثر تخصصاً وتعرف مجموعة الأدوات هذه باسم Rich controls .

في القسم الأول من هذا الفصل سنتعلم كيف تسمح للمتصفحين بأن يرفعوا ملفاتهم كأن يقوموا مثلاً برفع ملفات فيديو ... إلخ ، بعد ذلك سنتعلم استخدام أداة التقويم كطريقة تسهل على إدخال التواريخ ، كما نستعرض في هذا الفصل كيفية التعامل مع أداة التحكم AdRotator وسنناقش حالتين : أولاً أن تكون

في قاعدة بيانات ، ثانياً أن تكون مخزنة في ملف XML MultiView قوم بإظهار وإخفاء مناطق من الصفحة ، وتقسيم الصفحة إلى ألسنة عديدة (taps) ، وفي نهاية الفصل نعرض كيفية التعامل مع أداة التحكم Wizard من تقسيم عمل ما إلى مجموعة من الخطوات وعرضها على المستخدم كمجموعة واجهات متتالية .

FileUpload حديد المكان الذي سيتم تخزين مین فيه ، إما مجلد داخل الموقع و ضمن قاعدة بيانات الموقع وسنستعرض كلا الحالتين .

FileUpload

- ✓ Enabled : تفعيل أو إلغاء تفعيل هذه الأداة .
- ✓ FileBytes : للحصول على محتويات الملف كمصفوفة بايتات .
- ✓ FileContent : للحصول على محتويات الملف كمجرى stream .
- ✓ FileName : .
- ✓ HasFile : تعيد True عندما يتم رفع الملف .
- ✓ PostedFile : تعيد الملف المرفوع مغلف بغرض HttpPostedFile .

FileUpload تدعم الطرائق التالية :

- ✓ Focus : تسمح بوضع التركيز على هذه الأداة .
- ✓ SaveAs : تقوم بتخزين الملف (المراد رفعه) على الموقع .

تقوم الخاصية PostedFile بتغليف الملف المرفوع بـ HttpPostedFile ، هذا الأمر يسمح بالحصول على معلومات إضافية حول الملف .

HttpPostedFile

✓ ContentLength : لمعرفة حجم الملف بالبايتات .
✓ ContentType : () .
✓ FileName : .
✓ InputStream : stream .

HttpPostedFile يدعم الطريقة SaveAs والتي تقوم بتخزين الملف على الموقع .

HttpPostedFile يقدم بعض الخصائص المتوفرة مسبقاً م
FileUpload والتي يمكن التعامل معها دون الحاجة للصف HttpPostedFile ، على سبيل
المثال للحصول على اسم الملف يمكن اتباع أحد الأسلوبين :

- ✓ FileUpload.FileName
- ✓ HttpPostedFile.FileName

كما أن الطريقة SaveAs على HttpPostedFile على الرغم من توافرها مع أداة
FileUpload بشكل صريح .

أنشئ صفحة جديدة وأضف عليها الأدوات FileUpload1 , Button1, Lab
جديد داخل ملفات الموقع باسم uploads لنقوم برفع الملفات إليه ، في حدث الضغط على الزر
:

C#

```
protected void Button1_Click(object sender, EventArgs e)
{
    try
    {
        if (FileUpload1.HasFile)
        {
            string path = "~/uploads/" + FileUpload1.FileName;
            FileUpload1.SaveAs(MapPath(path));
            Label1.Text = "File Uploaded successfully...";
        }
    }
    catch (Exception ex) { }
}
```

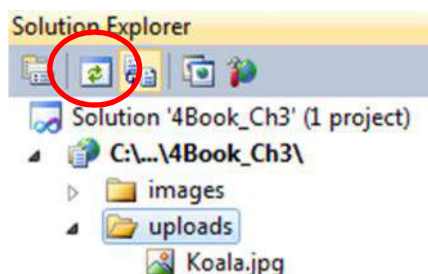
```
Protected Sub Button1_Click(ByVal sender As Object, ByVal e As
System.EventArgs) Handles Button1.Click

    If (FileUpload1.HasFile) Then
        Dim filePath As String = "~/uploads/" & FileUpload1.FileName
        FileUpload1.SaveAs(MapPath(filePath))
    End If
End Sub
```

يجب أن يكون كالتالي :

```
<div>
    <asp:FileUpload ID="FileUpload1" runat="server" />
    <br />
    <asp:Button ID="Button1" runat="server" Text="Upload"
        onclick="Button1_Click" />
    <br />
    <asp:Label ID="Label1" runat="server" Text=""/>
</div>
```

نفذ التطبيق السابق وحاول رفع ملف ما (صورة مثلاً) ، أغلق التنفيذ ، يجب أن يكون الملف قد
أضيف للمجلد Upload



في المثال السابق يستطيع المستخدم أن يرفع
بتحديد الأنماط التي نسمح للمستخدم بأن يقوم برفعها (فقط بعض أنواع الصور).

أنشئ صفحة جديدة وأضف عليها الأدوات FileUpload1 , Button1, Lab
جديد داخل ملفات الموقع باسم uploads لنقوم برفع الملفات إليه ، في حدث الضغط على الزر
:

C#

```
protected void Button1_Click(object sender, EventArgs e)
{
    try
    {
        if (FileUpload1.HasFile)
        {
            if (CheckFileType(FileUpload1.FileName))
            {
                string path = "~/uploads/" + FileUpload1.FileName;
                FileUpload1.SaveAs(MapPath(path));
                Label1.Text = "File Uploaded successfully...";
            }
        }
    }
    catch (Exception ex) { }
}
```

VB

```
Protected Sub Button1_Click(ByVal sender As Object, ByVal e As
System.EventArgs) Handles Button1.Click

    If (FileUpload1.HasFile) Then
        If (CheckFileType(FileUpload1.FileName)) Then
            Dim filePath As String = "~/uploads/" & FileUpload1.FileName
            FileUpload1.SaveAs(MapPath(filePath))
        End If
    End If
End Sub
```

IO ، حيث نضيف الكود التالي أعلى الصفحة:

C#

```
using System.IO;
```

VB

```
Imports System.IO
```

CheckFileType سؤلة عن فحص نوع الملف فهي :

C#

```
bool CheckFileType(string fileName)
{
    string ext = Path.GetExtension(fileName);
    switch (ext.ToLower())
    {
        case ".gif":
            return true;
        case ".png":
            return true;
        case ".jpg":
            return true;
        case ".jpeg":
            return true;
        default:
            return false;
    }
}
```

VB

```
Function CheckFileType(ByVal fileName As String) As Boolean
    Dim ext As String = Path.GetExtension(fileName)
    Select Case ext.ToLower()
        Case ".gif"
            Return True
        Case ".png"
            Return True
        Case ".jpg"
            Return True
        Case ".jpeg"
            Return True
        Case Else
            Return False
    End Select
End Function
```

:

ASP.net

```
<div>
    <asp:FileUpload ID="FileUpload1" runat="server" />
    <br />
    <asp:Button ID="Button1" runat="server" Text="Upload"
        onclick="Button1_Click" />
</div>
```

```
<br />
<asp:Label ID="Label1" runat="server" Text=""/>
</div>
```

يق السابق وحاول أن ترفع ملف ما مغاير للأنواع المحددة ، ستلاحظ عدم رفع الملف ، الصور السابقة لتري قبول عملية الرفع .

رفع الملفات إلى قاعدة بيانات

حيث نقوم بتخزين الملفات في قاعدة بيانات عوضاً عن مجلد عادي ، هذا الأسلوب يسبب كبر هائل بحجم قاعدة البيانات وبالتالي صعوبة وبطء في نقلها والتعامل معها ، كما أننا نحتاج لتحويل بيانات ثنائية (,) لنتمكن من تخزينها في الجدول ، الأسلوب الأفضل والمتبع في المواقع هو رفع الملفات إلى مجلد عادي وتخزين أسماء الملفات المرفوعة ضمن قاعدة البيانات مع تخزين معلومات أخرى عنها كتاريخ الرفع مثلاً ، على العموم لن ندخل بهذه التفاصيل الآن على تبار أننا لم نتطرق لموضوع قواعد البيانات ، سيكون لنا عودة لمناقشة كيفية رفع الملفات بأسلوب احترافي في فصل المواضيع المتقدمة .

يتم قبول رفع الملفات ذات الأحجام أصغر من ميغابايت (عد للمثال الأول وحاول رفع ملف أكبر من هذا الحجم لتري عدم قبول العملية) ، ولتغيير الحجم الأقصى المسموح به Web.config ونضيف كود السطر الرابع :

```
XML

<configuration>
  <system.web>
    <compilation debug="true" targetFramework="4.0"/>
    <httpRuntime maxRequestLength="10240"/>
  </system.web>
</configuration>
```

حيث تم تحديد الحجم الأعظمي ب كيلو بايت (أي ميغا بايت) .

عرض التقويم (Calendar)

Calendar والتي تقوم بتمكين المستخدم من تحديد التواريخ ASP.Net بسهولة ويسر .

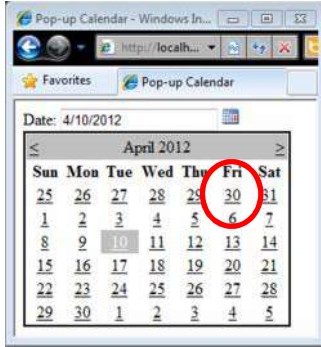
Calendar

- ✓ DayNameFormat : تحديد أسلوب ظهور أسماء أيام الأسبوع (اسم اليوم بالكامل ، اسم مختصر ...) ويمكن أن يأخذ إحدى القيم التالية : FirstLetter, FirstTwoLetters, Full, Short, and Shortest.
- ✓ NextMonthText : تحديد الذي ينقلنا للشهر عند الضغط عليه
- ✓ NextPrevFormat : تحديد تنسيق نصي الشهر (مختصرة) وتأخذ إحدى القيم : CustomText, FullMonth, and ShortMonth.
- ✓ PrevMonthText : تحديد الذي ينقلنا للشهر عند الضغط عليه
- ✓ SelectedDate : لتحديد يوم يوم محدد .
- ✓ SelectedDates : لتحديد مجموعة من الأيام أو الحصول عليها .
- ✓ SelectionMode : تحديد أسلوب التواريخ ، نختار يوم أو أسبوع كامل أو شهر بالكامل ويأخذ القيم : Day, DayWeek, DayWeekMonth, and None.
- ✓ SelectMonthText : تحديد نص الرابط الذي يقوم بتحديد الشهر كاملاً عند الضغط عليه
- ✓ SelectWeekText : تحديد نص الرابط الذي يقوم بتحديد الأسبوع كاملاً عليه
- ✓ ShowDayHeader : عرض أو إخفاء أسماء الأيام
- ✓ ShowNextPrevMonth : عرض أو إخفاء روابط الأشهر اللاحقة والسابقة
- ✓ ShowTitle : عرض أو إخفاء شريط العنوان الخاص بالأداة Calendar
- ✓ TitleFormat : لتحديد تنسيق شريط العنوان حيث يمكن أن يظهر اسم الشهر فقط أو يظهر معه العام الحالي أيضاً ويأخذ : Month ,MonthYear .
- ✓ TodaysDate : لتحديد تاريخ اليوم الحالي ، بشكل افتراضي تأخذ تاريخ اليوم الحالي من السيرفر .
- ✓ VisibleDate : لتحديد الشهر الذي سيتم عرضه ، بشكل افتراضي يتم عرض الشهر الذي يحتوي على تاريخ اليوم الحالي المحدد بالخاصية السابقة.

Calendar تدعم الأحداث التالية :

- ✓ SelectionChanged : يتم إطلاقه عند اختيار تاريخ جديد .
- ✓ VisibleMonthChanged : يتم إطلاقه عند الضغط على رابط الشهر التالي أو السابق .

عرض أداة التقويم بشكل منبثق



سنقوم في هذه الفقرة بتطبيق الأسلوب المتبع في المواقع والمنشآت الاحترافية ، وهو أن تكون أداة التقويم بشكل افتراضي غير ظاهرة للمستخدم ، إنما يدل على وجودها رمز صغير يعبر عن التقويم فإن شاء المستخدم يضغط على هذا الرمز لتنبثق له أداة Calendar فيختار التاريخ منها ، وإن شاء كتبه بشكل يدوي دون الاستعانة بهذه الأداة ، يتم أداء العمل السابق بالاستعانة بلغة الجافا سكربت وتقنية . CSS

أنشئ صفحة جديدة ، وأضف عليها أدوات التحـ Label1 , TextBox1 , Calendar1
html
img واربطها مع صورة مناسبة كما هو موضح بالأعلى
النقر عليها سنقوم باستدعاء كود جافاسكربت المسؤول عن إظهار الأداة Calendar1
CSS فهي ليست موضوعنا إنما سأكتفي بوضع كود الصفحة كاملاً :

ASP.net

```
<%@ Page Language="C#" AutoEventWireup="true" CodeFile="Default4.aspx.cs"
    Inherits="Default4" %>

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

<html xmlns="http://www.w3.org/1999/xhtml">
<head runat="server">

<script type="text/javascript">
    function displayCalendar() {
        var datePicker = document.getElementById('datePicker');
        datePicker.style.display = 'block';
    }
</script>

<style type="text/css">
#datePicker
{
    display:none;
    position:absolute;
    border:solid 2px black;
    background-color:white;
}
</style>

<title>Pop-up Calendar </title>
</head>
<body>
    <form id="form1" runat="server">
```

```

<div>
<asp:Label id="Label1" Text=" Date:" AssociatedControlID="TextBox1"
Runat="server" />
<asp:TextBox id="TextBox1" Runat="server" />

<div id="datePicker">
<asp:Calendar ID="Calendar1" runat="server"
onselectionchanged="Calendar1_SelectionChanged"></asp:Calendar>
</div>
</div>
</form>
</body>
</html>

```

Calendar1 SelectionChanged
فيقوم بإسناد التاريخ : TextBox1

C#

```

protected void Calendar1_SelectionChanged(object sender, EventArgs e)
{
    TextBox1.Text = Calendar1.SelectedDate.ToString("d");
}

```

VB

```

Protected Sub Calendar1_SelectionChanged(ByVal sender As Object, ByVal e As
System.EventArgs) Handles Calendar1.SelectionChanged
    TextBox1.Text = Calendar1.SelectedDate.ToString("d")
End Sub

```

(AdRotator)

يتم اختيار الإعلان الذي سيعرض بشكل عشوائي حيث
XML أو في جدول ضمن قاعدة بيانات . AdRotator

AdRotator

XML الذي يحتوي على الإعلانات . AdvertisementFile : لتحديد ✓

- ✓ AlternateTextField : لتحديد اسم حقل النص البديل (XML)
(ظهر محتوياته في حال فشل عرض صورة الإعلان ، القيمة الافتراضية لهذا الحقل هي : AlternateText .
 - ✓ DataMember : لربط عضو بيانات محدد ضمن مصدر البيانات DataSource .
 - ✓ DataSource : لتحديد مصدر بيانات الإعلانات بشكل برمجي .
 - ✓ DataSourceID : لتحديد مصدر بيانات الإعلانات بشكل تصريحي .
 - ✓ ImageUrlField : لتحديد اسم حقل صورة الإعلان (اسم وسم XML)
(، القيمة الافتراضية ImageUrl .
 - ✓ KeywordFilter : تحديد فلتر للإعلانات عبر كلمة مفتاحية وحيدة (نستعرضها لاحقا بالتفصيل) .
 - ✓ NavigateUrlField : لتحديد اسم حقل روابط التنقل (اسم وسم XML)
(جدول ، القيمة الافتراضية NavigateUrl .
 - ✓ Target : تمكنا من فتح نافذة جديدة .
- AdRotator AdCreated عند اختيارها لإعلان ما وقبل أن يتم عرضه .

نشئ صفحة جديدة وأضف عليها أداة AdRotator XML وسمه Ads.xml
(XML : Website Add New Item XML File)
مجلد لنخزن صور الإعلانات داخله وليكن اسمه images وضع فيه ثلاث صور ولتكن اسمائها pic1 , pic2 , pic3
Ads.xml واكتب فيه الكود التالي :

XML

```
<?xml version="1.0" encoding="utf-8" ?>
<Advertisements>

  <Ad>
    <ImageUrl>~/images/pic1.png</ImageUrl>
    <Width>300</Width>
    <Height>50</Height>
    <NavigateUrl>http://www.vb4arab.com</NavigateUrl>
    <AlternateText>Advertisement 1</AlternateText>
    <Impressions>50</Impressions>
  </Ad>
  <Ad>
    <ImageUrl>~/images/pic2.png</ImageUrl>
    <Width>300</Width>
    <Height>50</Height>
```

```

<NavigateUrl>http://www.yahoo.com</NavigateUrl>
<AlternateText>Advertisement 2</AlternateText>
<Impressions>25</Impressions>
</Ad>
<Ad>
<ImageUrl>~/images/pic3.png</ImageUrl>
<Width>300</Width>
<Height>50</Height>
<NavigateUrl>http://www.google.com</NavigateUrl>
<AlternateText>Advertisement 3</AlternateText>
<Impressions>25</Impressions>
</Ad>
</Advertisements>

```

وتحديد صورة كل إعلان وقياساتها *

البديل في حال عدم عرض الصورة والرابط الذي سيتم الانتقال إليه عند الضغط على الإعلان ، أما الخاصية Impressions فهي تحدد أهمية الإعلان كنسبة مئوية فالإعلان الأول أولوية عرضه هي % أما الثاني والثالث فكل منهما % أسماء الوسوم التي تمت كتابتها في ملف Ads.xml

هي نفس الأسماء الافتراضية للخصائص AlternateTextField ImageUrlField NavigateUrlField ، فلو قمت بتغيير القيم الافتراضية لهذه الخصائص فعليك تعديل أسماء وسوم xml .

بعد أن انتهينا من تجهيز ملف الإعلانات Ads.xml ، نعود إلى الصفحة التي أنشأناها AdRotator1 xml السابق عبر الخاصية AdvertisementFile ، ليصبح كود الصفحة :

ASP.net

```

<div>
<asp:AdRotator ID="AdRotator1" runat="server"
AdvertisementFile="~/Ads.xml" />
</div>

```

قم بتحديث الصفحة عدة مرات لترى كيف يتم عرض الإعلانات بشكل عشوائي مع الأخذ بعين الاعتبار الأولوية المحددة لكل إعلان عبر الخاصية Impressions .

لنفرض السيناريو التالي : لديك أداتين AdRotator1, AdRotator2 xml يحتوي على بيانات خمسة إعلانات، أول ثلاثة إعلانات لمواقع تعليمية ، الإعلان الرابع والخامس لمواقع

رياضية ، وتريد أن يتم عرض يتم عرض الإعلانات التعليمية AdRotator1 الرياضية تعرض عبر الأداة AdRotator2 ، فكيف يتم ذلك ؟ ببساطة نقوم بعمل فترة للإعلانات ويتم هذا على مرحلتين :

xml بحيث تأخذ الإعلانات التعليمية قيمة : <Keyword> :
(Teaching) الرياضية تأخذ قيمة أخرى (Sporting)
Ads2.xml يكون بالشكل التالي :

XML

```
<?xml version="1.0" encoding="utf-8" ?>
<Advertisements>
<!-- Teaching Advertisements -->
  <Ad>
    <ImageUrl>~/images/pic1.png</ImageUrl>
    <Width>200</Width>
    <Height>200</Height>
    <NavigateUrl>http://www.vb4arab.com</NavigateUrl>
    <AlternateText> vb4arab </AlternateText>
    <Keyword>Teaching</Keyword>
  </Ad>
  <Ad>
    <ImageUrl>~/images/pic2.png</ImageUrl>
    <Width>200</Width>
    <Height>200</Height>
    <NavigateUrl>http://www.arabTeam2000.com</NavigateUrl>
    <AlternateText> arabTeam2000 </AlternateText>
    <Keyword>Teaching</Keyword>
  </Ad>
  <Ad>
    <ImageUrl>~/images/pic3.png</ImageUrl>
    <Width>200</Width>
    <Height>200</Height>
    <NavigateUrl>http://www.AspWorkShopes.com</NavigateUrl>
    <AlternateText> AspWorkShopes </AlternateText>
    <Keyword>Teaching</Keyword>
  </Ad>
<!-- Sporting Advertisements -->
  <Ad>
    <ImageUrl>~/images/pic4.png</ImageUrl>
    <Width>200</Width>
    <Height>200</Height>
    <NavigateUrl>http://www.kooora.com</NavigateUrl>
    <AlternateText> kooora </AlternateText>
    <Keyword>Sporting</Keyword>
  </Ad>
  <Ad>
    <ImageUrl>~/images/pic5.png</ImageUrl>
    <Width>200</Width>
    <Height>200</Height>
    <NavigateUrl>http://www.hihi2.com</NavigateUrl>
```

```

<AlternateText> hihi2 </AlternateText>
<Keyword>Sporting</Keyword>
</Ad>
</Advertisements>

```

ثانياً : تحديد الفلتر الذي ستطبقه كلاً من الأدوات AdRotator1, AdRotator2 ويتم هذا عبر الخاصية KeywordFilter بحيث نسند لكل قيمة أحد الفلاتر السابقة المحددة في ملف xml وبالتالي يكون كود الصفحة كما يلي :

ASP.net

```

<div>
  <asp:AdRotator ID="AdRotator1" runat="server"
    AdvertisementFile="~/Ads2.xml" KeywordFilter="Teaching" />
  <br />
  <asp:AdRotator ID="AdRotator2" runat="server"
    AdvertisementFile="~/Ads2.xml" KeywordFilter="Sporting" />
</div>

```

قم بتنفيذ الصفحة السابقة وأعد تحميلها عدة
الإعلانات التعليمية ، في حين AdRotator2 تعرض فقط الإعلانات الرياضية .

في الفقرات السابقة تعلمنا كيف نتعامل مع الإعلانات XML ، يوجد أسلوب آخر وهو تخزين بيانات الإعلانات في جدول ضمن قاعدة البيانات ، سنقوم بعرض تلك الطريقة في فصل المواضيع المتقدمة ، بالإضافة لموضوع تتبع الإعلانات ومعرفة عدد مرات مشاهدة كل إعلان وغيرها من الأمور التي سنناقشها بالتفصيل في ذلك الفصل .

عرض مناظير مختلفة ضمن الصفحة (MultiView , View)

MuliView اء وإظهار مناطق مختلفة من الصفحة ، هذه الأداة مفيدة عندما تحتاج لإنشاء صفحة بالسنة عديدة (tabs) ، كما أنها تستخدم لتقسيم نموذج طويل إلى عدة نماذج أصغر ، هذه الأداة تقوم باحتواء مجموعة أدوات من النوع View ، بحيث تعرض View فقط بحسب ما يحدده المستخدم و (View) الذي يتم عرضه يدعى ActiveView حين أن محتويات باقي المناظير تبقى مخفية ، إذاً يتم عرض View .

MultiView

✓ ActiveViewIndex : لتحديد رقم المنظور View الذي سيتم عرضه ، يبدأ الترقيم من

✓ Views : للحصول على مجموعة المناظير المحتواة ضمن الأداة MultiView .

MultiView تدعم الطرائق التالية :

✓ GetActiveView : للحصول على المنظور المعروف حالياً .

✓ SetActiveView : لتحديد المنظور الذي سيتم عرضه .

أخيراً ، الأداة MultiView والـ ActiveViewChanged والذي يطلق عند اختيار منظور ما من المناظير المحتواة داخلها .

View لا تدعم أي خصائص أو طرائق خاصة بها ، وذلك لأنها لا تستخدم إلا ضمن أداة أخرى تحتويها ، على كلٍ فإن الأداة View تدعم الأحداث التالية :

✓ Activate : والذي يتم إطلاقه عندما نختار هذا المنظور من مجموعة مناظير الأداة MultiView ليتم عرضه .

✓ Deactivate : يطلق عندما نختار منظور آخر ، أي عندما يلغى عرض هذا المنظور .

لنقم بتطبيق المثال التالي لتوضيح مبدأ عمل أدوات التحكم السابقة ، أنشئ صفحة جديدة وأضف ليها ثلاثة أزرار (Buttons) وضع داخلها ثلاثة أدوات

View عبارة تدل عليه وذلك لتمييزه عند اختياره ، وبالتالي سيصبح

:

ASP.net

```
<div>
  <asp:Button ID="Button1" runat="server" Text="View1"
    onclick="Button1_Click" />
  <asp:Button ID="Button2" runat="server" Text="View2"
    onclick="Button2_Click" />
  <asp:Button ID="Button3" runat="server" Text="View3"
    onclick="Button3_Click" />
<br />
<asp:MultiView ID="MultiView1" runat="server">
  <asp:View ID="View1" runat="server">
    Hi , i'm View1 .
  </asp:View>
  <asp:View ID="View2" runat="server">
    Hi , i'm View2 .
  </asp:View>
</asp:MultiView>
```



```

        <asp:View ID="View3" runat="server">
            Hi , i'm View3 .
        </asp:View>
    </asp:MultiView>
</div>

```

(لزر على الترتيب) :

C#

```

protected void Button1_Click(object sender, EventArgs e)
{
    MultiView1.ActiveViewIndex = 0;
}

protected void Button2_Click(object sender, EventArgs e)
{
    MultiView1.ActiveViewIndex = 1;
}

protected void Button3_Click(object sender, EventArgs e)
{
    MultiView1.ActiveViewIndex = 2;
}

```

VB

```

Protected Sub Button1_Click(ByVal sender As Object, ByVal e As
System.EventArgs) Handles Button1.Click

    MultiView1.ActiveViewIndex = 0
End Sub

Protected Sub Button2_Click(ByVal sender As Object, ByVal e As
System.EventArgs) Handles Button2.Click

    MultiView1.ActiveViewIndex = 1
End Sub

Protected Sub Button3_Click(ByVal sender As Object, ByVal e As
System.EventArgs) Handles Button3.Click

    MultiView1.ActiveViewIndex = 2
End Sub

```

View مختلف ليتم عرضه عند النقر على أحد الأزرار السابقة وذلك عبر الخاصية `ActiveViewIndex` كما رأينا مسبقاً ، نفذ

قمنا هنا بالإكتفاء بكتابة نص بسيط لتطبيق المثال بسهولة ويسر .

`MultiView` سيم نموذج كبير إلى عدة نماذج فرعية أصغر ، بإمكاننا `Button` داخل كل منظور يبحث ينقلنا إلى منظور آخر عند الضغط عليه ، وذلك بالاستفادة من الأوامر التالية التي تدعمها الأداة `MultiView` :

- ✓ `NextView` : لعرض المنظور التالي للمنظور المعروض حالياً ضمن `MultiView`
- ✓ `PrevView` : لعرض المنظور السابق للمنظور المعروض حالياً ضمن `MultiView`
- ✓ `SwitchViewByID` : لتحديد المنظور الذي سيتم عرضه ، ويتم تحديده من خلال `Button` `CommandArgument`
- ✓ `SwitchViewByIndex` : لتحديد المنظور الذي سيتم عرضه ، ويتم تحديده من خلال `Button` `CommandArgument`

إذاً ، وبمنتهى البساطة إن أردت الانتقال إلى المنظور التالي فعليك إضافة `Button` الحالي واسند للخاصية `CommandName` القيمة `NextView` إلى منظور سابق حيث تستبدل القيمة السابقة بالقيمة `PrevView` .

في المثال التالي سنقوم بتصميم نموذج لإدخال البيانات وهو مقسم إلى ثلاثة مناظير (واجهات - `views`) ، المنظور الأول يطلب إدخال الاسم ، المنظور الثاني يطلب إدخال الثالث فيقوم بعرض المعلومات التي أدخلتها في الواجهتين السابقتين . أنشئ صفحة جديدة وأضف عليها أداة `MultiView` وضع داخلها ثلاث أدوات `View` `Label` `View1` واجعل قيمة الخاصية `Text` "your Name" `Button` `TextBox1` واجعل قيمة الخاصية `CommandName` `NextView` `Label2` واجعل قيمة الخاصية `Text` "your Age" `Button` `TextBox2` واجعل قيمة الخاصية `CommandName` `NextView` ، نأتي للمنظور الثالث ، أضف داخله أداتين `Label3`, `Label4` `Activate` الخاص بهذا المنظور واكتب الكود التالي :

C#

```
protected void View3_Activate(object sender, EventArgs e)
{
    Label13.Text = TextBox1.Text;
    Label14.Text = TextBox2.Text;
}
```

VB

```
Protected Sub View3_Activate(ByVal sender As Object, ByVal e As
System.EventArgs) Handles View3.Activate

    Label13.Text = TextBox1.Text
    Label14.Text = TextBox2.Text
End Sub
```

:

ASP.net

```
<div style="border-style: solid; border-width: thin">

    <asp:MultiView ID="MultiView1" runat="server" ActiveViewIndex="0">
        <asp:View ID="View1" runat="server">
            <asp:Label ID="Label1" runat="server" Text="Your Name : " />
            <asp:TextBox ID="TextBox1" runat="server"></asp:TextBox>
            <br />
            <asp:Button ID="Button1" runat="server" Text="next"
                CommandName="NextView" />
        </asp:View>
        <asp:View ID="View2" runat="server">
            <asp:Label ID="Label2" runat="server" Text="Your Age : " />

            <asp:TextBox ID="TextBox2" runat="server"></asp:TextBox>
            <br />
            <asp:Button ID="Button2" runat="server" Text="next"
                CommandName="NextView" />
        </asp:View>
        <asp:View ID="View3" runat="server" onactivate="View3_Activate">

            your name :
            <asp:Label ID="Label13" runat="server" Text="Label"></asp:Label>
            <br />
            your age :
```

```

        <asp:Label ID="Label1" runat="server" Text="Label"></asp:Label>
    </asp:View>
</asp:MultiView>

</div>

```

حاول التعديل على المثال السابق بإضافة أزرار تسمح بالانتقال إلى منازير سابقة .

Wizard

Wizard تشبه من حيث المبدأ الأداة MultiView حيث تقوم أيضاً بتقسيم نموذج كبير إلى عدة نماذج فرعية أصغر ، بالإضافة إلى أن هذه الأداة تدعم مزايا عديدة غير متوفرة بنظيرتها MultiView Wizard تحتوي داخلها على مجموعة من الأدوات WizardStep يمكن عرض WizardStep .

Wizard

- WizardStep : ActiveStep ✓
- ActiveStepIndex : لتحديد أداة WizardStep التي نريد تفعيلها ، ويمكن استخدامها WizardStep الفعّال حالياً . ✓
- CancelDestinationPageUrl : لتحديد عنوان لصفحة يتم نقل المستخدم إليها إذا ضغط (Cancel) . ✓
- DisplayCancelButton : إظهار أو إخفاء الزر Cancel . ✓
- DisplaySideBar : إظهار أو إخفاء الشريط الجانبي والذي يقوم بعرض قائمة بخطوات Wizard . ✓
- FinishDestinationPageUrl : لتحديد عنوان لصفحة يتم نقل المستخدم إليها عند ضغطه على الزر إلغاء الأمر Finish . ✓
- HeaderText : لتحديد العنوان الذي يظهر أعلى الأداة Wizard . ✓
- WizardSteps : للحصول على جميع الأدوات WizardStep الموجودين ضمن الأداة Wizard . ✓

Wizard :

- ✓ المنطقة العلوية وندعوها بالترويصة Header
- ✓ المنطقة الجانبية وندعوها بالشريط الجانبي SideBar
- ✓ قبة والتي تحتوي على الخطوات ندعوها بمنطقة الإبحار Navigation Area

Wizard تدعم القوالب التالية :

- ✓ HeaderTemplate : لتحديد مظهر منطقة الترويسة
- ✓ SideBarTemplate : لتحديد مظهر منطقة الشريط الجانبي
- ✓ StartNavigationTemplate : لتحديد مظهر منطقة الإبحار لأول خطوة فقط
- ✓ FinishNavigationTemplate : لتحديد مظهر منطقة الإبحار لآخر خطوة فقط
- ✓ StepNavigationTemplate : لتحديد مظهر منطقة الإبحار لجميع الأولى والأخيرة .

Wizard تدعم الطرائق التالية :

- ✓ GetHistory() : WizardStep
اجتزناها.
- ✓ GetStepType() : للحصول على نوع خطوة معينة ، حيث توجد الأنواع التالية
WizardStep : Start, Finish, Step, Auot, Complete .
- ✓ MoveTo() : WizardStep معينة .

Wizard تدعم الأحداث التالية :

- ✓ ActiveStepChanged : يتم إطلاقه عندما تتغير الخطوة الفعالة .
 - ✓ CancelButtonClick : يتم إطلاقه عند الضغط على الزر Cancel .
 - ✓ FinishButtonClick : يتم إطلاقه عند الضغط على الزر Finish .
 - ✓ NextButtonClick : يتم إطلاقه عند الضغط على الزر Next .
 - ✓ PreviousButtonClick : يتم إطلاقه عند الضغط على الزر Previous .
 - ✓ SideBarButtonClick : يتم إطلاقه عند الضغط على زر SideBar .
- إذاً ، وكما رأينا فإن أداة التحكم Wizard تحتوي داخلها على مجموعة خطوات WizardStep .

WizardStep تمتلك الخصائص التالية :

- ✓ AllowReturn : السماح للمستخدم بالعودة لهذه الخطوة من خطوة لاحقة ، أو منعه من .
- ✓ Name : للحصول على اسم الخطوة الحالية .
- ✓ StepType : لتعيين نوع الخطوة الحالية أو استعادته ، حيث يمكن أن تكون الخطوة من
Start, Finish, Step, Auot, Complete :
- Start : إذا تم تعيين هذه القيمة ، عندئذٍ الزر Previous لن يظهر في هذه الخطوة .

- Step : يظهر كلا الزرين Next , Previous في هذه الخطوة .
- Finish : يظهر الزرين Previous , Finish .
- Complete : لن يظهر أي زر في الخطوة .
- Auto : وهو الخيار الأنسب ، حيث يتم إظهار الأزرار المناسبة لكل خطوة بحسب موقعها من مجموعة الخطوات .
- ✓ Title : لتعيين عنوان للخطوة أو لاستعادته ، عنوان الخطوة يتم عرضه في منطقة الشريط
- SideBar
- ✓ Wizard : Wizard الحاوية لهذه الخطوة .

WizardStep تطلق الحدثين التاليين :

- ✓ Activate : يُطلق عند تفعيل هذه الخطوة (أي عندما يتم عرضها) .
- ✓ Deactivate : يُطلق عند إلغاء تفعيل هذه الخطوة (أي عند الانتقال) .

Wizard يتألف من أربع

، الخطوتين الثانية والثالثة لإدخال بيانات المستخدم ، الخطوة الرابعة تعرض البيانات التي قام المستخدم بإدخالها ، فيما يلي عرض للكود النهائي للصفحة لإعطاء منظر أكثر جمالية لأداة Wizard (يمكن حذف هذه التنسيقات بدون مشك) :

ASP.net

```
<%@ Page Language="C#" AutoEventWireup="true" CodeFile="Default9.aspx.cs"
Inherits="Default9" %>

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

<html xmlns="http://www.w3.org/1999/xhtml">
<head id="Head1" runat="server">
<style type="text/css">
.wizard
{
border:solid 1px black;

font:14px Verdana,Sans-Serif;
width:400px;
height:300px;
}
.header
```

```

{
    color:gray;
    font:bold 18px Verdana,Sans-Serif;
}
.sidebar
{
    background-color:#eeeeee;
    padding:10px;
    width:100px;
}
.sidebar a
{
    text-decoration:none;
}
.step
{
    padding:10px;
}
</style>
<title>Show Wizard</title>
</head>
<body>
    <form id="form1" runat="server">
        <div>
            <asp:Wizard id="Wizard1" HeaderText="Product Survey"
                OnFinishButtonClick="Wizard1_FinishButtonClick"
                CssClass="wizard" HeaderStyle-CssClass="header"
                SideBarStyle-CssClass="sidebar"
                StepStyle-CssClass="step"
                Runat="server">
                <WizardSteps>
                    <asp:WizardStep ID="WizardStep1" Title="Introduction">
                        Please complete our survey so that we can improve our
                        products.
                    </asp:WizardStep>
                    <asp:WizardStep ID="WizardStep2" Title="Step 1">
                        <asp:Label id="lblSSN" Text="Social Security Number:"
                            AssociatedControlID="txtSSN" Runat="server" />
                        <br />
                        <asp:TextBox id="txtSSN" Runat="server" />
                    </asp:WizardStep>
                    <asp:WizardStep ID="WizardStep3" Title="Step 2" StepType="Finish">
                        <asp:Label id="lblPhone" Text="Phone Number:"
                            AssociatedControlID="txtPhone" Runat="server" />
                        <br />
                        <asp:TextBox id="txtPhone" Runat="server" />
                    </asp:WizardStep>
                    <asp:WizardStep ID="WizardStep4" Title="Summary" StepType="Complete">
                        <h1>Summary</h1>
                        Social Security Number:
                        <asp:Label id="lblSSNResult" Runat="server" />
                        <br /><br />
                        Phone Number:
                        <asp:Label id="lblPhoneResult" Runat="server" />
                    </asp:WizardStep>
                </WizardSteps>
            </div>
        </form>
    </body>
</html>

```

```

        </asp:Wizard>
    </div>
</form>
</body>
</html>

```

Wizard1 فنكتب الكود التالي (وهو FinishButtonClick لإظهار البيانات التي أدخلها المستخدم في خطوات سابقة) :

C#

```

protected void Wizard1_FinishButtonClick(object sender,
    WizardNavigationEventArgs e)
{
    lblSSNResult.Text = txtSSN.Text;
    lblPhoneResult.Text = txtPhone.Text;
}

```

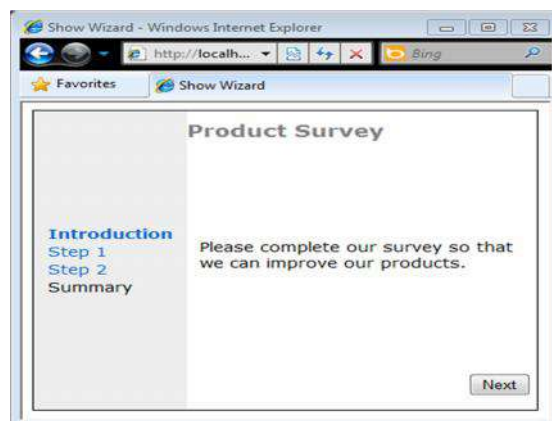
VB

```

Protected Sub Wizard1_FinishButtonClick(ByVal sender As Object, ByVal e As
    System.Web.UI.WebControls.WizardNavigationEventArgs) Handles
    Wizard1.FinishButtonClick
    lblSSNResult.Text = txtSSN.Text
    lblPhoneResult.Text = txtPhone.Text
End Sub

```

نتيجة تنفيذ الصفحة السابقة ستكون بالشكل التالي :



إلى هنا نأتي لنهاية هذا الفصل ، حيث استعرضنا فيه كيفية رفع الملفات إلى الموقع ، وكيفية التعامل مع أداة التقويم واستخدامها كأداة منبثقة بالاستعانة بأكواد جافا سكريبت ، ثم رأينا معاً مبدأ الإعلانات في المواقع وتعاملنا معها من خلال ملفات XML ، ثم توجهنا إلى تقسيم نماذج العمل حيث كانت لنا وقفة مع أداة التحكم MultiView وأسلوب عرضها لمناظير عديدة بغية تقسيم النماذج الطويلة إلى نماذج فرعية أصغر وأبسط ، وفي نهاية الفصل تعلمنا معاً كيفية العمل بأداة Wizard

أعيد وأذكر بأن هناك العديد من الفقرات والأمثلة الهامة جداً والمتعلقة بهذا الفصل والتي لا مجال لعرضها الآن بسبب احتوائها على معلومات لم ندرسها بعد ، إنما سنناقشها بالتفصيل في فصول

m-hajjkhalf@hotmail.com

<http://www.facebook.com/mohammed.hajkhalaf>

— سوريا