

الواجب الأول

برمجة الويب

١ . إنشاء تطبيقات الويب باستخدام *ASP.NET*

٢ . إدارة تطبيقات الويب

obeyikanda.com

obeikandi.com

الفصل الأول إنشاء تطبيقات الويب باستخدام ASP.NET

سنقوم في هذا الفصل بالتعرف على السمات الأساسية لصفحات الخادم النشطة الموجودة داخل نطاق **.NET**. والمسماة **ASP.NET** وكذلك كيفية استخدام هذه الصفحات وعرضها من خلال مستعرض الويب.

بانتها هذا الفصل، ستتعرف على:

- مفهوم الاتصال بالويب.
- استخدام **HTML** مع **ASP.NET**.
- عناصر وخصائص صفحات **ASP.NET**.
- استخدام تصنيفات **ASP.NET**.
- مثال تطبيقي على كيفية إنشاء صفحات **ASP.NET** خارج بيئة تطوير **Visual Studio 2008**.

من السهل أن يقوم المستخدم النهائي بعرض أي من صفحات الويب المتاحة على الإنترنت بمجرد كتابته لعنوان الصفحة داخل نافذة المستعرض. أما بالنسبة لك كمبرمج لتطبيقات ومواقع الويب، فالوضع يختلف كثيراً. فلابد أن تعرف جيداً ما يحدث على الجانب الآخر حتى يحصل المستخدم على هذه الصفحات داخل مستعرض الويب.

سنقوم في هذا الفصل بالتعرف على مفهوم صفحات الخادم النشطة أو **Active Server Pages.NET (ASP.NET)** وهي إحدى مكونات نطاق **.NET** التي تتيح للمطورين إنشاء صفحات ومواقع الويب الديناميكية أي التي تتغير محتوياتها باستمرار.

مفهوم الاتصال بالويب

يطلق استعراض الويب **Web Browsing** على عملية الاتصال بين أجهزة العميل وأجهزة الخادم من خلال شبكة من الشبكات. وجهاز العميل عبارة عن مستخدم يقوم بفتح أحد مستعرضات الويب على حاسبه مثل **Internet Explorer** أو **Netscape Navigator** بينما جهاز الخادم عبارة عن أي خادم مثل **Internet Information Server (IIS)** أما الشبكة فقد تكون الشبكة العنكبوتية الأم (الإنترنت) أو أي شبكة داخلية صغيرة داخل مؤسسة من المؤسسات والتي يطلق عليها **Intranet**. ولكي تتمكن من استخدام **ASP.NET**، يجب أن يكون **IIS** وكذلك نطاق **.NET** مثبتين على حاسبك.

إنشاء تركيبات دليل الويب

بعد أن تقوم بالوصول إلى خادم الويب، فمن السهل إتاحة الصفحة للآخرين بوضع ملف الصفحة على هذا الخادم. فإذا أراد المستخدم مشاهدة محتويات إحدى الصفحات، يقوم مباشرةً بإدخال عنوان الصفحة **URL** داخل مربع العنوان الموجود بمستعرض الويب المثبت عليه كما في العنوان التالي:

<http://www.compuscience.com/Books/Programming/index.htm>

والذي يتم فيه تحديد اسم خادم الويب وهو **www.compuscience.com** في هذه الحالة بالإضافة إلى اسم الصفحة وموقعها على الخادم

index.htm /Books/Programming/index.htm وكما ترى فإن الصفحة index.htm تقع في المستوى الثاني داخل تركيب خادم الويب الذى يأخذ نفس سمات تركيب المجلدات داخل نظام التشغيل Windows عدا استثناءين هامين وهما:

- تحتوى خادمات الويب على أدلة تخيلية Virtual Directories وهو ما يعنى إمكانية اختلاف اسم مجلد الويب عن موقعه على مساحة التخزين.
- تعتبر الأدلة التخيلية والصفحات الموجودة بداخلها تطبيقات ASP.NET على خادم IIS.

إنشاء دليل تخيلى داخل IIS

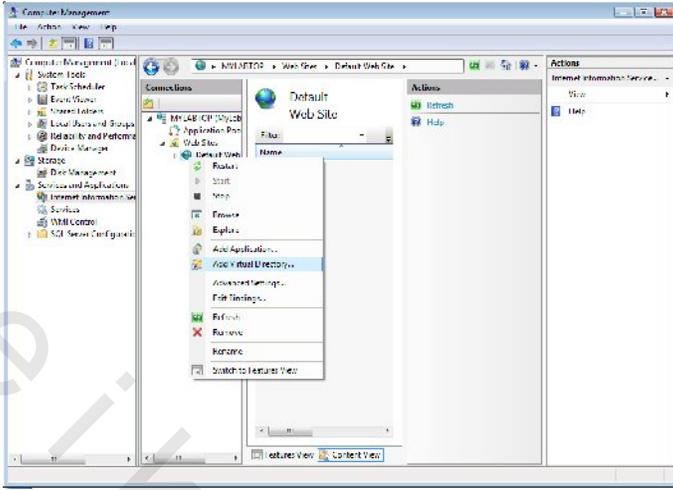
يعتبر إنشاء الدليل التخيلى Virtual Directory الخطوة الأولى فى إعداد تطبيق ASP.NET جديد على خادم الويب الخاص بك. لتوضيح ذلك، سنقوم بإنشاء دليل لاحتواء الأمثلة التى نقوم بإنشائها فى هذا الفصل. تابع معنا الخطوات الآتية:

١. افتح نافذة مستكشف Windows أو نافذة My Computer ثم انتقل إلى المجلد C:\inetpub\wwwroot (على فرض أنك قمت بتنحيم Windows على القرص C) وهذا هو الدليل الرئيسى لخادم الويب الموجود لديك والذى تم إنشاؤه بمجرد تثبيت IIS.

٢. انقر بزر الفأرة الأيمن أى مكان خالى بالمجلد wwwroot ثم اختر New Folder من القائمة الموضوعية وأعد تسمية المجلد الجديد إلى اسم معبر وليكن ASPTest.

٣. من سطح المكتب انقر رمز My Computer بزر الفأرة الأيمن ثم اختر Manage من القائمة الموضوعية، تظهر نافذة Computer Management (انظر شكل ١-١).

٤. من العمود الأيسر بالنافذة، انتقل إلى المجلد Services and Applications, ثم Web Sites Internet Information Services.



شكل ١-١ نافذة Computer Management

٥. انقر رمز **Default Web Site** بالجانب الأيمن من النافذة بزر الفأرة الأيمن ثم اختر **Add Virtual Directory** من القائمة الموضعية، يظهر المربع الحوارى **Add Virtual Directory**.
٦. قم بكتابة أى اسم مميز لموقع الويب التخيلى الجديد بمربع النص **Alias** وليكن نفس الاسم الذى قمنا بتعريفه للمجلد **ASPTTest** ثم قم بتحديد مكان المجلد على القرص الصلب بمربع النص التالى **Physical path** وهو المسار **C:\inetpub\wwwroot\ASPTTest** أو انقر الزر المجاور **...** ثم قم بتحديد المجلد من النافذة الناتجة.
٧. انقر زر **Ok** لإغلاق المربع الحوارى **Add Virtual Directory** وإتمام عملية إنشاء الدليل التخيلى الجديد.
٨. قم بإغلاق نافذة **Computer Management** وبذلك تكون جاهزاً بوضع ملفات موقعك داخل المجلد **ASPTTest**.

استخدام HTML مع ASP.NET

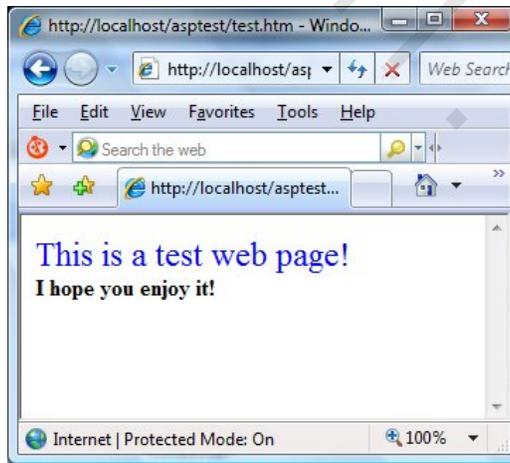
لإضافة صفحات ويب جديدة إلى التطبيق ASPTest، كل ما تحتاج إليه هو وضع ملفات هذه الصفحات داخل المجلد ASPTest. وحينما يقوم مستعرض الويب لديك بطلب صفحة ويب من هذا المجلد، يتولى برنامج IIS مسئولية معالجة الملف الصحيح والاستجابة مباشرة لطلب المستعرض. لتوضيح ذلك، استخدم أحد محررات النصوص لإنشاء ملف HTML وليكن Test.htm وقم بوضعه داخل المجلد ASPTest حيث يحتوي الملف على الكود التالي:

```
<html>
<body>
<font size ="+2" color="blue"> This is a test web page! </font>
</br>
<b> I hope you enjoy it!</b> </br>
</body>
</html>
```

والآن قم بفتح مستعرض الويب المثبت لديك ثم قم بإدخال العنوان التالي:

<http://localhost/asptest/test.htm>

اضغط زر الإدخال، تلاحظ عرض صفحة الويب داخل المستعرض (انظر شكل ٢-١).



شكل ٢-١ تظهر صفحة الويب داخل المستعرض بمجرد كتابة عنوان الصفحة



من الطبيعي أن يكون كود HTML واضح لك تماماً في هذا المستوى من الدراسة. لمعرفة المزيد عن HTML، راجع أحد كتبنا المخصصة لهذا الغرض مثل كتاب "المرجع الأساسي لمستخدمي HTML" وكتاب "برمجة الإنترنت باستخدام XHTML".

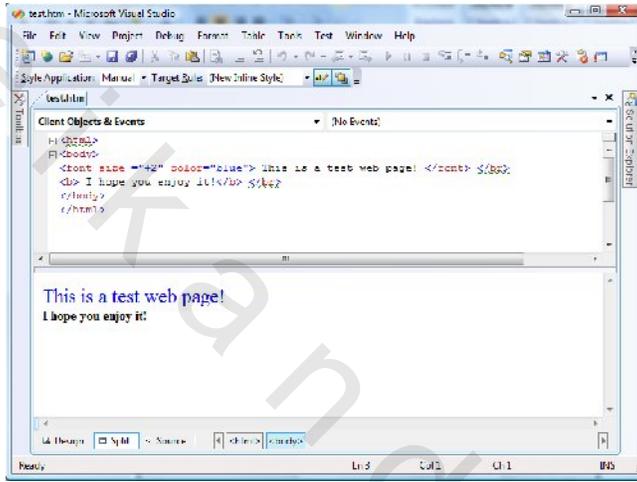
تحرير HTML باستخدام Visual Studio

على الرغم من أن كتابة كود HTML من خلال Notepad طريقة بدائية إلا أنها توضح مدى سهولة إنشاء صفحات الويب. لكن لأن Visual Studio 2008 يحتوي على محرر HTML داخلي، يكون من المفيد إنشاء هذه الصفحات أو حتى تحريرها من داخل هذا المحرر لما يحتويه من أدوات مساعدة متنوعة. للتعرف على كيفية استخدام محرر HTML الموجود داخل Visual Studio 2008، قم بفتح الملف Test.htm الذي أنشأناه منذ قليل وذلك باتباع الخطوات التالية:

١. قم بتشغيل Visual Studio 2008.
٢. افتح قائمة File بشريط القوائم ثم اختر Open ثم File من القوائم المنسدلة الناتجة.
٣. انتقل إلى المجلد C:\inetpub\wwwroot\ASPTTest.
٤. انقر الملف Test.htm نقراً مزدوجاً لفتحه داخل نافذة التحرير، تظهر محتويات الصفحة كما لو كانت داخل مستعرض الويب.
٥. يمكنك كتابة النصوص وتعيين خصائصها المختلفة داخل الصفحة كما لو كنت تستخدم أحد برامج تحرير النصوص العادية وسوف ينعكس هذا التغيير على الكود الحقيقي للصفحة.
٦. يمكنك العمل مع الصفحة في ثلاثة أنماط مختلفة، الأول هو نمط التصميم Design الذي تظهر فيه الصفحة كما لو كانت داخل مستعرض الويب، والثاني نمط المصدر Source الذي يظهر به الكود الحقيقي للصفحة والثالث Split هو مزيج

للمنطين السابقين، حيث يتم فيه تقسيم الصفحة إلى جزئين، يحتوى العلوى على كود الصفحة (النمط Source) بينما يحتوى السفلى على الصفحة كما تظهر بنافذة المستعرض (النمط Design).

٧. انقر زر Split أسفل الصفحة لفتح نافذتى التصميم والعرض فى نفس الوقت (انظر شكل ٣-١).



شكل ٣-١ فتح ملف HTML داخل محرر بيئة تطوير Visual Studio 2008

صفحات الخادم النشطة ASP.NET

منذ الوهلة الأولى لظهور الويب، احتوت صفحات الويب على بيانات HTML ثابتة لا تتغير إلا بتغير الصفحة نفسها. ولكن دعت الحاجة بعد ذلك إلى التغيير المستمر للبيانات لعرض محتويات المكتبات والمخازن من خلال الويب، وهذه المحتويات بطبيعتها قابلة للتغيير، لا أقول كل دقيقة وإنما ربما كل جزء من الثانية. ويتم هذا التغيير التلقائى للبيانات حقيقةً من خلال خادم الويب الذى يقوم باستخدام برنامج آخر تم إنشاؤه بإحدى اللغات الأخرى كلغة C مثلاً لفتح قاعدة البيانات والحصول على محتوياتها أو حتى تحديثها.

ومنذ عدة سنوات، قامت مايكروسوفت بطرح تقنية تسمى صفحات الخادم النشطة أو (ASP) Active Server Pages والتي من خلالها يتم مزج كود Visual Basic

مع كود HTML في ملف واحد على خادم الويب. فحينما يقوم برنامج IIS بمعالجة ملف ASP، يقوم بإرجاع كود HTML إلى مستعرض الويب الذي يجيد التعامل معه وإظهار محتوياته بينما يقوم بتنفيذ كود Visual Basic على الخادم نفسه.

حينما يطلب المستخدم من مستعرض الويب إظهار ملف بالامتداد .htm أو .html، يقوم خادم الويب ببساطة شديدة بقراءة محتويات الملف من القرص الصلب وتحويلها إلى مستعرض الويب لعرضها من خلاله. أما حينما يطلب المستخدم من المستعرض إظهار ملف بالامتداد .aspx (وهو امتداد ملفات ASP.NET)، يقوم IIS بتنفيذ الكود المضمن بهذا الملف والذي يتضمن العديد من العمليات، ومنها إرجاع كود HTML الخاص بالصفحة إلى مستعرض الويب.



للتعرف عن قرب على كيفية عمل صفحات الخادم النشطة ASP.NET، تابع معنا الخطوات الآتية:

١. افتح برنامج Notepad أو أى محرر نصوص آخر.
٢. قم بفتح ملف Test.htm الذى قمنا بإنشائه من قبل.
٣. قم بتغيير كود الملف كى يظهر كما يلى:

```
<SCRIPT LANGUAGE="VB" RUNAT="SERVER">
</SCRIPT>
<HTML>
<BODY>
<FONT SIZE="+2" COLOR="BLUE"> This is a test web page!
</FONT> <BR>
<%
Dim intCounter As Integer
For intCounter = 1 to 10
    Response.Write("<B> I hope you enjoy it!</B><BR>")
Next
Response.Write("Page Created: " & DateTime.Now)
%>
</BODY>
```

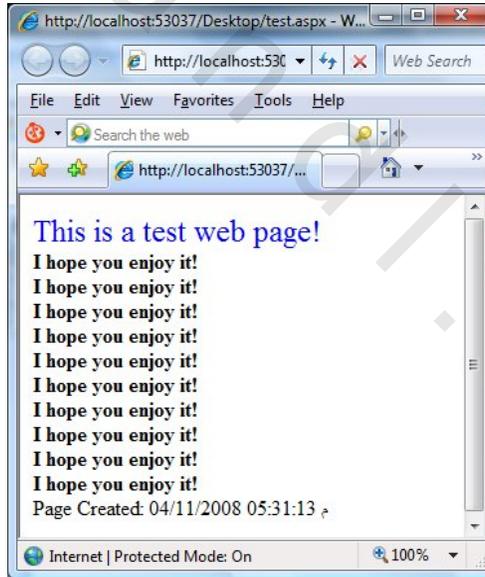
</HTML>

وكما ترى فإن هذا الكود قريب الشبه إلى حدٍ كبير بالكود السابق والفرق الوحيد هو إضافة الكود الخاص بلغة **Visual Basic** بين الرمزین %< و %> كما سنرى بعد قليل.

٤ . قم بحفظ الملف بنفس الاسم ولكن مع استخدام الامتداد **.aspx**. أى يصبح **.Test.aspx**.

٥ . لمشاهدة الصفحة الجديدة داخل نافذة مستعرض الويب، قم بإدخال العنوان **http://localhost/ASPTest/Test.aspx** داخل شريط عنوان المستعرض ثم اضغط مفتاح الإدخال (انظر شكل ١-٤).

٦ . انقر زر **Refresh** من شريط أدوات المستعرض (أو اضغط مفتاح F5)، تلاحظ تغيير محتويات الصفحة ودلالة ذلك تغيير الوقت الذى يظهر أمام عبارة **Page Created** أسفل الصفحة.



شكل ١-٤ انقر زر **Refresh** لتغيير محتويات التاريخ

يحتوى الكود السابق رغم بساطته الشديدة على العديد من سمات وخصائص صفحة **ASP.NET** وعلى رأسها ما يلي:

المختلفة. ومن خلال عملك مع صفحات ASP.NET ستلاحظ ظهور عدد من السمات والخصائص الجديدة التي لم تكن موجودة في ASP والتي يمكن إجمال أهمها فيما يلي:

- الكود المترجم *Compiled Code* حيث يتميز كود صفحات ASP.NET بأنه مترجم *Compiled* بعكس كود صفحات ASP الذى يتم تفسيره *Interpreted* مما يؤدي كما تعلم إلى سرعة تنفيذ صفحات ASP.NET مقارنةً بصفحات ASP. فإذا قمت بإجراء أى تعديلات على ملف ASP.NET ذى الامتداد *.aspx*، يقوم IIS بإعادة ترجمة كود الملف (الصفحة) مع أول طلب للصفحة من قبل مستعرض الويب.

- لغة قوية *Robust Language* حيث تكمن قوة لغة Visual Basic 2008 أو أي من اللغات الأخرى التي تنتمي إلى عائلة Visual Studio 2008 في القدرة على استخدام ASP.NET بعكس الإصدار القديم ASP الذى يستخدم من قبل اللغات *VBScript* و *JScript*.

- البرمجة المبنية على النماذج *Forms-based Programming* حيث تم إنشاء نموذج برمجة جديد يسمى *WebForms* يتم من خلاله إنشاء صفحات ويب قوية بسهولة تامة كما لو كنت تتعامل مع النماذج العادية مع فصل واجهة المستخدم عن الكود الحقيقية لصفحة ASP.NET. ولعل هذه السمة أهم السمات الثلاثة على الإطلاق.

التعرف على رموز البرمجة

لفصل كود Visual Basic عن رموز HTML داخل صفحة ASP.NET، نلجأ لاستخدام بعض الرموز الخاصة والتي يطلق عليها رموز البرمجة *Script Tags*. فأنشاء معالجة الصفحة يقوم IIS بتنفيذ الكود الموجود داخل هذه الرموز فقط حيث يوجد شكلان لهذه الرموز يمكن بياهما كما يلي:

- يحدد الرمز `<SCRIPT>` بمصاحبة الصفة `RUNAT="SERVER"` الكود الذى يتم تنفيذه من قبل الخادم.
- يحدد الرمز `<%>` و `>%` بداية ونهاية الكود الذى يتم تنفيذه من قبل الخادم. ولعل الشكل الثانى هو الأيسر والمعتاد لمن استخدموا `ASP` من قبل. إلا أن هناك بعض القواعد المختلفة لاستخدام رموز البرمجة فى `ASP.NET` عنها فى `ASP`. فبدءاً من `ASP.NET` لا يمكنك استخدام الصيغة الثانية المختصرة إذا احتوى الكود على تعريف دالة مخصصة. فعلى سبيل المثال رغم أن الكود التالى يعمل بشكل صحيح داخل `ASP` إلا أنه لا يعمل داخل `ASP.NET`:

```
<%
Sub Test()
    Response.Write("Hi")
End Sub
Call Test()
%>
```

حيث يجب مع `ASP.NET` استخدام الرمز `<Script>` مع إعلانات الدوال واستخدام الرموز `<%>` و `>%` مع العبارات التى يتم تنفيذها أثناء معالجة الصفحة. وبذلك يمكنك إعادة كتابة الكود السابق كما يلى:

```
<SCRIPT language = "VB" RUNAT="SERVER">
Sub Test()
    Response.Write("Hi")
End Sub
</SCRIPT>
<%
Call Test()
%>
```

إحضار المسميات *Importing Namespaces*

تعرفنا فيما سبق على مفهوم المسميات `Namespaces` وأوضحنا أنها تحتوى على مجموعة من التصنيفات المرتبطة والتى تؤدى وظائف متشابهة كما تعرفنا أيضاً على كيفية استخدام عبارة `Imports` لإحضار المسميات إلى تطبيقك كى تتمكن من استخدام

محتوياتها مباشرةً كما لو كانت معرفة داخل التطبيق. يمكنك بنفس الطريقة إحضار المسميات إلى صفحة ASP.NET باستخدام صيغة مختلفة قليلاً عن الصيغة السابقة كما يلي:

```
<% @Import Namespace = "System.IO" %>
```

حيث قمنا في هذه العبارة بإحضار المسمى System.IO لاستخدامه داخل صفحة ASP.NET الحالية.

يجب أن تسبق عبارة إحضار المسمى أى تعريف للمتغيرات داخل صفحة ASP.NET.



لأن ASP.NET تحتوى على القوة الكاملة للغة Visual Basic 2008، يمكنك بالطبع بنفس الطريقة السابقة تعريف المسميات الخاصة بك وإحضارها إلى تطبيقات ASP.NET.

استخدام تصنيفات ASP.NET

تحتوى ASP.NET على العديد من التصنيفات الخاصة بالويب والمثلة في صورة كائنات Objects مثل الكائن Response الذى تعرضنا له من قبل والمستخدم لتوجيه استجابة الخادم لطلب مستعرض الويب. ولعل العديد من هذه الكائنات مألوفاً لك إذا كنت ممن استخدموا ASP من قبل مع وجود بعض الاختلافات الطفيفة. سنقوم فيما يلي بالتعرف على أهم هذه الكائنات.

الكائن Session

تشابه الدورة أو الجلسة Session إلى حد كبير بالزيارة التى تجريها لطبيك والتي تبدأ بمجرد دخولك إلى عيادته وتنتهى بمجرد مغادرتك لها. وأثناء هذه الزيارة ربما تناقش العديد من المواضيع ولكن فى حالتنا تكون هذه المناقشة مع صفحات الويب المختلفة التى تزورها على موقع الويب. ولعل متغيرات الدورة Session Variables أحد السمات المحببة لدى مستخدمى ASP والتي لم تزل موجودة داخل ASP.NET حيث يتم تخزين هذه المتغيرات على خادم الويب ويتم الوصول إليها من خلال الكود الموجود على الخادم.

يمكنك تعيين متغير دورة **Session Variable** بسهولة شديدة عن طريق تعيين اسم للمتغير وتخصيصه بقيمة كما في الأمثلة التالية:

```
Session("UserId") = "Waleed"  
Session("CurrentPurchases") = 1000  
Session.Add("YearsWarranty",5)
```

وكما ترى يمكنك تعيين المتغيرات مباشرةً أو من خلال الوظيفة **Add** الجديدة في **ASP.NET**. وبمجرد إنشائك لمثل هذه المتغيرات، يحتفظ بها الخادم إلى أن تنتهي الدورة الحالية **Current Session**، حيث يتم إنهاء الدورة الحالية لأحد الأسباب والتي من أهمها مايلي:

- إذا قام المستخدم بإغلاق مستعرض الويب.
- إذا انتهى وقت الدورة المحدد داخل الخاصية **Session.Timeout** الخاصة بهذه الدورة، فطالما استمر المستخدم في التفاعل مع الخادم، يكون الخادم نشطاً.
- إذا قام الكود باستدعاء الوظيفة **Session.Abandon** لقطع الدورة لسبب من الأسباب.

وبمجرد إنهاء المرحلة الحالية، يتم تدمير جميع محتويات المتغيرات المعرفة داخل هذه المرحلة

توفير الأمان مع المتغيرات

يعتبر توفير الأمان أثناء استخدام الصفحات المتعددة أحد الاستخدامات الهامة لمتغيرات المرحلة **Session Variables**. لتوضيح ذلك، تخيل أن لديك موقعاً يحتوي على صفحة دخول **Login Page** التي من الممكن أن تستخدم متغير مرحلة **Session Variable** يحتوي على اسم المستخدم الحالي والذي يكون بدوره متاحاً للصفحات الأخرى. افترض أيضاً أن الموقع يحتوي على الصفحات الثلاثة التالية:

- الصفحة **usermenu.aspx** والتي تحتوي على قائمة بارتباطات الصفحات الأخرى تبعاً لاسم المستخدم الحالي.
- الصفحة **cssalary.aspx** والتي تقوم بعرض تقارير رواتب الموظفين.

- الصفحة `csmenu.aspx` والتي تحتوى على عروض الشركة من أجهزة وكتب وبرامج.

وغالباً في مثل هذه المواقع، يرى المستخدم الصفحة `usermenu.aspx` بمجرد إدخال بيانات الدخول الخاصة به والتي تتمثل في اسم المستخدم `UserID` وكلمة المرور `Password`، حيث تقوم هذه الصفحة بقراءة قاعدة البيانات ثم عرض قائمة بارتباطات الصفحات الأخرى. ويعتمد هذا بالطبع على ما إذا كان المستخدم مخولاً بالدخول أم لا. فمثلاً يستطيع أى زائر مشاهدة عروض الشركة بينما لا يتمكن من رؤية تقارير المرتبات إلا أناس بعينهم من خلال اسم المستخدم وكلمة المرور. ولكن يظهر هنا سؤال غاية في الأهمية، كيف يتسنى لك منع المستخدم من الدخول إلى الصفحة (صفحة تقارير المرتبات مثلاً) من خلال كتابة عنوان هذه الصفحة مباشرة داخل شريط عنوان مستعرض الويب متخطياً بذلك صفحة الدخول برمتها؟ يتمثل أحد الحلول حقيقةً في اختبار أمان الدخول في بداية كل صفحة من الصفحات كما يلي:

```
<%  
If Not CheckValidUser("cs",Session("UserID")) Then  
    Response.Write("Access denied")  
    Response.End  
End If  
>%
```

وفي هذا الكود يتم تمرير اسم التطبيق (`cs`) وقيمة متغير المرحلة `UserID` إلى الدالة المخصصة `CheckValidUser`. فإذا قامت هذه الدالة بإرجاع القيمة `False`، لا يتم تنفيذ كود `HTML` الموجود أسفل الكود السابق الخاص باختبار الأمان نتيجةً لإنهاء الاستجابة من خلال الوظيفة `Response.End()` كما سنرى بعد قليل.

يتم في التطبيقات الفعلية توجيه المستخدم إلى صفحة الدخول مرةً أخرى إذا لم يتم بإدخال البيانات الصحيحة في صفحة الدخول الأولى أو إذا تخطاها كليةً كما ذكرنا.



الكائن Response

في كل مرة يقوم فيها مستعرض الويب بطلب صفحة من خادم الويب، يقوم الخادم بالاستجابة من خلال الكائن Response وهو أحد حالات التصنيف بالـ **HttpResponse** المستخدم في إدارة استجابة الخادم. وقد تعرفنا بالفعل على أحد الاستخدامات الشهيرة لهذا الكائن من خلال الوظيفة **Write()** المستخدمة في إرسال البيانات من الخادم إلى مستعرض الويب. يوضح جدول ١-١ التالي مجموعة من الوظائف والخصائص الهامة داخل الكائن **Response**.

جدول ١-١ الوظائف والخصائص الهامة داخل الكائن Response

الاسم	الوظيفة
Write()	إرسال النصوص إلى المستعرض
End()	إنهاء الاستجابة الحالية
Redirect()	توجيه المستعرض إلى صفحة أخرى
IsClientConnected	اختبار اتصال المستخدم من عدمه
Cookies	توفير الوصول إلى بيانات المستخدم المسماة Cookies
Cache	التحكم في تخزين بيانات الصفحة Caching

توجيه المستخدم لصفحة أخرى

رأينا منذ قليل كيفية استخدام الوظيفتين **Response.Write()** لإرسال البيانات من الخادم إلى المستعرض و **Response.End()** لإنهاء عملية الاستجابة الحالية بين الخادم والمستعرض. أما الوظيفة **Response.Redirect()** فتستخدم لإخبار المستعرض بطلب صفحة أخرى ومن ثم توجيه المستخدم (المستعرض) إلى صفحة أخرى غير الصفحة الحالية كما في الكود التالي:

<%

```
If Not CheckValidUser("cs",Session("UserID")) Then
    Response.Redirect("/Security/login.htm")
Response.End
```

End If

%>

وهو نفس الكود الذى استخدمناه من قبل لتحقيق أمان الصفحة، عدا أنه بدلاً من أن يقوم الخادم بإظهار رسالة خطأ بالمستعرض، يقوم بتوجيه المستخدم إلى صفحة الدخول الموجودة على العنوان /Security/login/ الموجود على الخادم.

يمكنك استخدام الوظيفة (`Response.Redirect()`) في حالة عدم توجيه كود HTML إلى المستعرض. فإذا احتوى الملف على كود HTML أو الوظيفة (`Response.Write()`) قبل استخدام عبارة `Response.Redirect`، تظهر على الفور رسالة خطأ.

يمكنك استخدام الوظيفة (`Server.Transfer.Response.Redirect()`) أو الدالة (`Server.Transfer()`) بدلاً من الوظيفة (`Response.Redirect()`).



إنهاء الاستجابة الحالية

تستخدم الخاصية `IsClientConnected` لاختبار مدى اتصال المستخدم بالاستجابة من عدمه، حيث تستغرق عملية الطلب من المستعرض والاستجابة من الخادم غالباً ثوانى معدودة، ولكن مع الاستجابات الطويلة ربما أردت إيقاف عملية الاستجابة إذا ما قام المستخدم بقطع المرحلة `Session`. لتوضيح ذلك، دعنا نرى الكود التالى:

<%

Call LongProcessNumberOne()

If Not Response.IsClientConnected Then

 Response.End

End If

Call LongProcessNumberTwo()

Response.Write("Done!")

%>

في هذا الكود اعتبرنا أن عمليات استدعاء الدوال تستغرق القليل من الوقت، وباختبار الخاصية `IsClientConnected` يمكننا تجنب المعالجة التى تستغرق وقتاً كبيراً إذا أعرض المستخدم عن الانتظار.

إرسال Cookies إلى المستخدم

كلمة Cookies لا تعنى بالطبع أجزاء الكيك أو البسكويت فنحن لا نتناول كتاب لتعلم فن الطهي، ولا أدرى حقيقةً لماذا اختارت مايكروسوفت هذا المصطلح الغريب. دعنا من كل ذلك، فكلمة Cookies عبارة عن أجزاء صغيرة من البيانات يتم تخزينها لدى المستخدم وتعبر عن خصائص عمليات الاتصال بين المستخدم والمواقع المختلفة. لتوضيح ذلك بمثال بسيط، حينما تفتح إحدى الصفحات داخل مستعرض الويب الخاص بك ثم تقوم بنقر أى من الارتباطات الموجودة بهذه الصفحة، ماذا يحدث؟ يتحول لون هذا الارتباط إلى لون مخالف للارتباطات الأخرى الموجودة بالصفحة دلالةً على أنك قمت باختياره من قبل. وإذا قمت بإغلاق هذه الصفحة ثم فتحها مرةً أخرى في أى وقتٍ آخر، تلاحظ الإبقاء على اللون المخالف للارتباط الذى قمت بنقره من قبل. فأنى للمستعرض أن يعرف بأنك قمت بنقر هذا الارتباط من قبل؟ يتم ذلك حقيقةً من خلال جزء من البيانات يسمى Cookie. أرجو أن يكون مفهوم Cookies بات واضحاً لديك الآن.

مع كل طلب من مستعرض الويب إلى الخادم يتم إرسال Cookies إلى الخادم بحيث يسهل التعامل معها من خلال كود ASP.NET والذي يتيح لك أداء المهام المختلفة مثل إضفاء الطابع الشخصى على الموقع كما ذكرنا منذ لحظات. ففي الكود التالى على سبيل المثال يتم إرسال Cookie باسم Zipcode مرةً أخرى إلى المستخدم:

```
Dim ckZip As New HttpCookie("Zipcode")
ckZip.Value = "38138"
ckZip.Expires = DateTime.Now.AddDays(30)
Response.Cookies.Add(ckZip)
```

وعلى ذلك فحينما يقوم المستخدم باستخدام هذا الموقع مرةً أخرى، يتم إرسال ZipCode إلى الخادم مما يتيح إظهار معلومات ZIP Code المرتبطة تلقائياً. كما يعنى استخدام الخاصية Expires إلغاء تأثير Cookie بعد ٣٠ يوم.

التحكم فى تخزين الاستجابة Response Caching

من الخصائص المفيدة المصاحبة للكائن Response الخاصية Cache، حيث يقوم

المستعرض بتخزين كود HTML والصور المصاحبة لاستجابة معينة داخل مجلده المؤقت لفترة من الوقت وذلك لزيادة سرعة التنفيذ. يمكنك تحديد الوقت اللازم لبقاء الصفحة داخل المجلد المؤقت من خلال الخاصية Cache كما في الكود التالي الذي يحدد زمن التخزين بخمس وعشرين دقيقة:

```
Dim ts As New TimeSpan(0,25,0)
Response.Cache.SetMaxAge(ts)
```

الكائن Request

يتم استخدام الكائن Request للتعرف على المعلومات المصاحبة لطلب صفحة الويب. ومن استخدامك للإنترنت لعلك لاحظت احتواء عنوان URL لأحدى الصفحات على العديد من المعاملات الأخرى كما في العنوان التالي:

```
http://cshome/report.aspx?title=payroll&dept=23&format=PDF
```

يقوم هذا العنوان بطلب الصفحة report.aspx مع تمرير ثلاثة معاملات هي: title و dept و format حيث يمكن الوصول إلى هذه المعاملات من خلال الوظيفة Request.QueryString. فعلى سبيل المثال، يقوم الكود التالي باسترجاع قيم المعاملات الموجودة في العنوان السابق وتخزينها داخل متغيرات:

```
Dim strTitle As String
Dim intDeptNumber As Integer
Dim strFormat AS String
```

```
strTitle = Request.QueryString("title")
intDetNumber = Request.QueryString("dept")
strFormat = Request.QueryString("format")
```

وسوف نرى فيما بعد كيفية استخدام عبارة Response.Write في إنشاء ارتباط تشعبي يحتوي على معاملات URL.

عند استخدام المعاملات داخل URL، يتم فصل العنوان عن المعامل الأول بعلامة الاستفهام "?" كما يتم فصل المعامل الأول عن المعاملات التالية من خلال الرمز "&".



استرجاع البيانات من نماذج HTML

تعتبر النماذج ومعاملات العنوان إحدى الطرق الشهيرة في تمرير القيم البسيطة من صفحة إلى أخرى بدون استخدام متغيرات المرحلة **Session Variables** طالما أنك لا تمنع مشاهدة المستخدم لهذه القيم. يمكنك استخدام نموذج يحتوى على كود HTML لنقل البيانات من العميل إلى الخادم حيث تتكون النماذج أساساً من الرمز **<FORM>** وعدد من الحقول **<INPUT>** وحينما يتم إرسال النموذج إلى خادم الويب، تكون قيم هذه الحقول متاحة من خلال **Request.Form**. لترى كيفية أداء ذلك عملياً، قم بإنشاء ملف جديد داخل المجلد **ASPTest** باسم **default.htm** ثم قم بإدخال الكود التالي:

```
<HTML>
<BODY>
<H1>Security Check</H1>
<HR>
<FORM ACTION="checkuser.aspx" METHOD="POST">
User ID: <INPUT TYPE="TEXT" NAME="txtUserID"><BR>
Password:<INPUT TYPE="PASSWORD" NAME="txtPassword">
<BR>
<INPUT TYPE="SUBMIT" VALUE="LOGIN">
</FORM>
</BODY>
</HTML>
```

لمشاهدة محتويات هذا النموذج، قم بكتابة العنوان **http://localhost/asptest/** داخل مربع عنوان مستعرض الويب، تحصل على نموذج يحتوى على حقل إدخال وزر **Submit** بعنوان **Login** (انظر شكل ١-٦).



شكل ٦-١ محتويات النموذج داخل مستعرض الويب

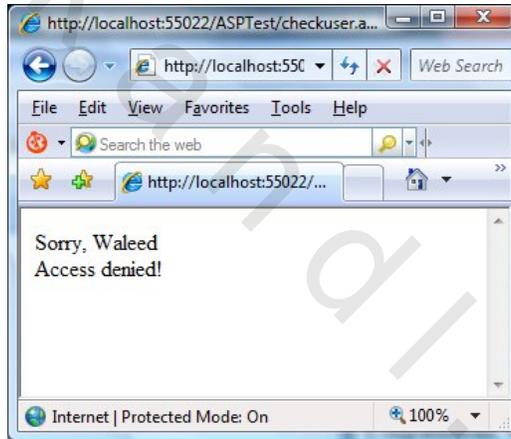
وكي يتم معالجة النموذج السابق، قم بإنشاء ملف جديد باسم `checkuser.aspx` داخل المجلد `ASPTTest` ثم قم بإدخال الكود التالي إلى هذا الملف:

```
<%  
If Request.Form("txtPassword").ToLower = "comp" Then  
    Response.Write ("Congratulations, " &  
                    Request.Form("txtUserId"))  
    Response.Write ("<BR>You entered the right password!")  
Else  
    Response.Write ("Sorry, " & Request.Form("txtUserId"))  
    Response.Write ("<BR>Access denied!")  
End If  
>%
```

في هذا الكود يتم مقارنة قيمة `txtPassword` التي قام المستخدم بإدخالها داخل الحقل `Password` في النموذج السابق. فإذا كانت `comp` يتم إظهار تهنئة للمستخدم مع إظهار اسمه من خلال `Request.Form("txtUserId")` (انظر شكل ٧-١) وإلا يتم رسالة أخرى بعدم صحة كلمة المرور (انظر شكل ٨-١).



شكل ٧-١ نتيجة تنفيذ الكود عند إدخال كلمة مرور صحيحة (comp)



شكل ٨-١ نتيجة تنفيذ الكود عند عدم إدخال كلمة المرور الصحيحة

التعرف على قيم Cookies

كما رأينا من قبل فإن الكائن Response يحتوي على الوظائف اللازمة لإرسال أجزاء البيانات الصغيرة المسماة Cookies إلى العميل، وكذلك الحال بالنسبة للكائن Request الذى يحتوي على الوظائف اللازمة لاسترجاع هذه القيم من العميل إلى الخادم وذلك كما فى الكود التالى:

```
Dim strZipCode As String  
If Not IsNothing(Request.Cookies("ZipCode")) then
```

```
strZipCode = Request.Cookies("ZipCode").Value  
End If
```

استرجاع القيم عن طريق الاسم

من خلال الشرح السابق تعرفنا على عدد من التجمعات **Collections** للقيم المخزنة داخل الكائن **Response** مثل التجمع **QueryString**. وأحد هذه التجمعات الجديدة داخل **ASP.NET** هو التجمع **params** الذى يعمل كمجموعة من **Cookies** وقيم **QueryString** ومتغيرات الخادم وحقول النماذج. يوضح الكود التالى كيفية الوصول إلى بعض القيم التى ذكرناها فى الشرح السابق من خلال التجمع **params**:

```
strZipCode = Request.params("ZipCode")  
strPassword = Request.params("txtPassword")  
intDeptNumber = Request.params("dept")
```

مفهوم الكائنات Server و Application

يتيح لك الكائن **Server** التحكم فى أداء العديد من الأحداث على الخادم مثل إنشاء الكائنات وتنفيذ الكود. ففي الإصدارات السابقة من **ASP** كان الكائن **Server** هو الكائن الوحيد المستخدم فى الوصول إلى الكائنات الخارجية من خلال الوظيفة **CreateObject**، أما فى عالم **.NET**، فيمكنك استخدام القوة الكاملة للغة **Visual Basic 2008** بما فى ذلك إحضار المسميات كما أوضحنا من قبل. وللحفاظ على التوافق مع كائنات **COM**، أبقت مايكروسوفت على الوظيفة **CreateObject()** للوصول إلى هذه الكائنات كما فى الكود التالى:

```
<%  
Dim cn As Object  
cn = Server.CreateObject("MyComObj.MyComClass")  
>%
```

كما يستخدم الكائن **Server** أيضاً للاستيعاض عن إحدى الصفحات بصفحة أخرى على مستوى الخادم وذلك من خلال الوظيفة **Server.Transfer()** التى تأخذ عنوان الصفحة البديلة كعامل لها كما فى العبارة التالية التى تتسبب فى نقل التحكم إلى الصفحة

:default.aspx

Server.Transfer("default.aspx")

لا يمكن استخدام الوظيفة `Server.Transfer()` داخل ملفات `.htm` أو `.asp`. وإنما يتم استخدامها فقط داخل ملفات `.aspx`.



من الكائنات الهامة الموجودة داخل `ASP.NET` أيضاً الكائن `Application` والذي يتيح لك تعيين المتغيرات المتاحة لجميع مستخدمي التطبيق كما في العبارة التالية:

```
Application("MyPrice") = 55
```

وتعمل هذه المتغيرات بنفس طريقة عمل متغيرات المرحلة `Session Variables` حيث تظل قيمة هذه المتغيرات بالذاكرة طالما كان التطبيق نشطاً (يوجد دائماً مرحلة واحدة نشطة على الأقل).

مثال تطبيقي

الآن وبعد أن تعلمنا أساسيات `ASP.NET` وحتى لا يذهب كلامنا أدراج الرياح، سنقوم فيما يلي بشرح مثال تطبيقي على كيفية إنشاء موقع ويب كامل عبارة عن ألبوم من صور لأغلفة الكتب التي تنشرها شركة كمبيوساينس، حيث سنقوم بإنشاء أربعة ملفات `ASP.NET` كما يلي:

- الملف `category.aspx` ومن خلاله يستطيع المستخدم تحديد المجموعة التي يرغب في عرض صورها أو حتى إنشاء مجموعة جديدة
- الملف `pictures.aspx` ويقوم بإظهار جميع الصور الموجودة في مجموعة معينة
- الملف `thumbnail.aspx` يقوم بإنشاء عينة صغيرة من الصورة لتسريع عملية التحميل
- الملف `upload.aspx` ويتيح للمستخدمين إضافة صور جديدة إلى مجموعة الصور الموجودة

وأثناء إنشائها لهذا التطبيق، سنتعرض لمعظم المبادئ والأساسيات التي شرحناها كما سنتعرض لبعض تقنيات معالجة الرسوم والملفات التي يتم شرحها بالتفصيل في فصولٍ أخرى من الكتاب.

إنشاء مجلد التطبيق

سنقوم فيما يلي بإنشاء مجلد خاص بالتطبيق بنفس طريقة إنشاء المجلد **ASPTest** في بداية هذا الفصل. تابع معنا الخطوات الآتية:

١. باستخدام مستكشف **Windows**، قم بإنشاء المجلد **Books** أسفل المجلد **C:\inetpub\wwwroot**.

٢. من سطح المكتب انقر رمز **My Computer** بزر الفأرة الأيمن ثم اختر **Manage** من القائمة الموضوعية، تظهر نافذة **Computer Management** (راجع شكل ١-١).

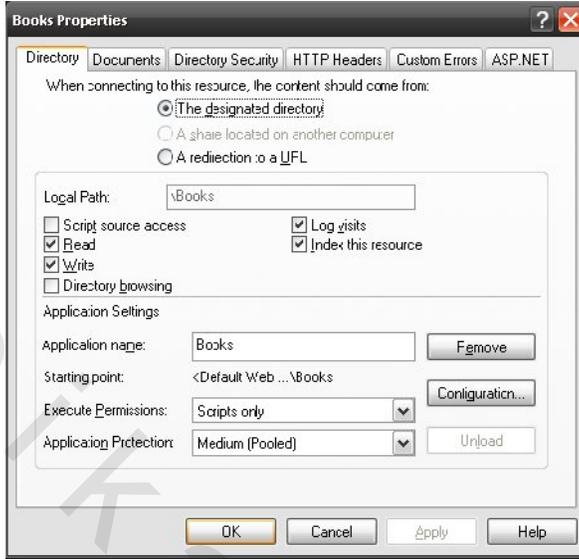
٣. من العمود الأيسر بالنافذة، انتقل إلى المجلد **Services and Applications**، **Internet Information Services** ثم **Web Sites**.

٤. انقر رمز **Default Web Site**، تلاحظ ظهور المجلد **Books** بالعمود الأيمن من النافذة على الرغم من أننا لم نقوم بعد بإنشاء الدليل التخيلي.

٥. انقر المجلد **Books** بزر الفأرة الأيمن ثم اختر **Properties** من القائمة الموضوعية، تظهر نافذة **Books Properties**. انقر زر **Create** لإنشاء التطبيق الجديد.

٦. تأكد من تنشيط مربع الاختيار **Write** وذلك لتمكين المستخدم من إضافة الصور على التطبيق (انظر شكل ١-٩).

٧. انقر زر **Ok** لإغلاق مربع الحوار **Books Properties** ثم قم بإغلاق نافذة **Computer Management**.



شكل ٩-١ مربع الخصائص الخاص بالتطبيق Books

إدارة مجموعات الصور

لتنظيم الصور الموجودة بالتطبيق، سنتيح للمستخدم وضع الصورة داخل مجموعة معينة وكذلك عرض الصور التي تنتمي إلى هذه المجموعة، حيث يتم تمثيل هذه المجموعات كمجلدات فرعية أسفل المجلد الرئيسي Books. كما سنتيح للمستخدم تسمية هذه المجموعات بنفسه واستخدام كود Visual Basic 2008 لإنشاء مجلدات (أدلة) فرعية بنفس الاسم، وفي هذه الحالة يعتبر IIS هذه المجلدات الفرعية جزءاً من التطبيق Books. قم بإنشاء ملف جديد باسم category.aspx أسفل المجلد Books ثم قم بإدخال الكود التالي إلى هذا الملف:

```
<% @Import Namespace = "System.IO" %>
<SCRIPT LANGUAGE="VB" RUNAT="Server">
Private Sub DisplayCategorylist()
'Writes links for each subdirectory in /Books
Dim strAppRoot As String
Dim strCatList() As String
Dim strCatLink AS String
Dim strShortName AS String
```

```
Dim intPos AS Integer
Dim i As Integer
'Get an array of subdirectory names
strAppRoot = Server.MapPath("/Books")
strCatList = Directory.GetDirectories(strAppRoot)
For i = LBound(strCatList) to UBound(strCatList)
    'Extract the directory name from the full path
    intPos = strCatList(i).ToUpper.IndexOf("BOOKS") + 7
    strShortName = strCatList(i).Substring(intPos)
    'Build a hyperlink and send it to the browser
    strCatLink = "<A HREF=""pictures.aspx?category="
    strCatLink &= strShortName & """">" & strShortName
    strCatLink &= "</A><BR>"
    Response.Write(strCatLink)
Next
End Sub

Private Sub CreateNewCategory(strName As String)
    'Creates a new subdirectory
    Dim strAppRoot As String
    Dim strNewFolder As String
    strAppRoot = Server.MapPath("/Books")
    strNewFolder = strAppRoot & "\" & strName
    Call Directory.CreateDirectory(strNewFolder)
End Sub

Private Sub Page_Load(ByVal sender As System.Object, ByVal e
As System.EventArgs) Handles MyBase.Load
    If Not (Request.Form("txtNewCatName") Is Nothing) Then
        Call CreateNewCategory(Request.Form ("txtNewCatName")
        .ToString)
    End If
End Sub
</SCRIPT>
<HTML>
<BODY>
<H1>Books Archive</H1>
```

<HR>

Click a category name below to view Compuscience Books:

<%

Call DisplayCategoryList

%>

<HR>

Enter a new category name to create:

```
<FORM ACTION="category.aspx" METHOD="POST"
NAME="frmTest">
```

```
<INPUT TYPE="TEXT" NAME="txtNewCatName">
```

```
<INPUT TYPE="SUBMIT" VALUE="Create">
```

```
</FORM>
```

```
</BODY>
```

```
</HTML>
```

يحتوى هذا الكود على دالتين مخصصتين هما الدالة `DisplayCategoryList()` والدالة `CreateNewCategory()`. تقوم الدالة `DisplayCategoryList()` بطباعة الارتباطات التشعبية لمجموعات الكتب، فإذا وجد على سبيل المثال مجموعة باسم `Programming`، تقوم الدوارة `For` في الكود السابق بإنشاء السطر التالى من كود `HTML` والخاص بهذه المجموعة:

```
<A HREF="pictures.aspx"?category=Programming">
```

```
Programming</A>
```

كما تقوم الدالة `CreateNewCategory()` بإنشاء مجلد جديد للمجموعة الجديدة على القرص الصلب حيث يتم استدعائها من خلال الحدث `Page_Load` إذا قام المستخدم بتمرير اسم مجموعة جديدة داخل نموذج الإدخال.

إنشاء صفحة عرض الكتب

والآن بعد أن أتقنا للمستخدم إمكانية اختيار مجموعة من الكتب أو حتى تعيين مجموعة جديدة، نحتاج إلى إنشاء صفحة الويب التى تمكنه من عرض الكتب الموجودة فى المجموعة التى قام باختيارها. كما سنقوم بإنشاء صفحة أخرى لعرض صور هذه الكتب فى أحجام

صغيرة لتسريع عملية التحميل وهو ما نطلق عليه **thumbnails**، حيث تكون كل صورة صغيرة عبارة عن ارتباط للصورة الكبيرة المصاحبة. قم بإنشاء ملف جديد باسم **pictures.aspx** داخل المجلد **Books** ثم قم بإدخال الكود التالي إلى هذا الملف:

```
<% @Import Namespace = "System.IO" %>
<HTML>
<BODY>
<%
    Dim strAppRoot As String
    Dim strFileList() As String
    Dim strCategory AS String
    Dim strPhotoPath As String
    Dim strShortName As String
    Dim strCurrentPic As String
    Dim intPos AS Integer
    Dim i As Integer

'Get path to physical directory
    strCategory = "/books/" & Request.Params("Category")
    strPhotoPath = Server.MapPath(strCategory)
'Get List of files
    strFileList = Directory.GetFiles(strPhotoPath)
'Create photo thumbnail page, 5 pictures per table row
    Response.Write("<H1>" & Request.Params("Category") &
        "</H1>")

    Response.Write("<HR><TABLE>")
    Response.Write("<TR>")
intPos = strPhotoPath.Length + 1
For i = LBound(strFileList) to UBound(strFileList)
    strShortName = strFileList(i).SubString(intPos)
    strCurrentPic = strCategory & "/" & strShortName
    Response.Write("<TD><A HREF='<''& strCurrentPic &
        '<''>")

    Response.Write("<IMG SRC='<''thumbnail.aspx?
        FileName=" & strCurrentPic & '<''>")
    Response.Write("</A><TD>")
If (i+1) mod 5 = 0 Then
```

```

                Response.Write("</TR><TR>")
            End If
        Next
    %>
</TR>
</TABLE>
<BR>
<A HREF="category.aspx">Back to category list</A><BR>
<%
Response.Write("<A HREF=""upload.aspx?category=")
Response.Write(Request.Params("Category") & """">Upload new
Book image</A>")
%>
</BODY>
</HTML>

```

قم أيضاً بإنشاء ملف جديد باسم thumbnail.aspx داخل المجلد Books ثم قم بإدخال الكود التالي إلى الملف:

```

<% @Import Namespace = "System.IO" %>
<% @Import Namespace = "System.Drawing" %>
<%
'This ASPX page generates a thumbnail of an image
'It can be used in an <IMG> tag
Dim strWebPath As string
Dim strFileName As String
Dim intLastSlash As string
Dim BytesRead As Integer
Dim ByteValues() As Byte
Dim intFileLength As Integer
Dim imgFullSize As System.Drawing.Image
Dim imgThumbnail As System.Drawing.Image
Dim stmThumbnail As MemoryStream
'Figure out the physical file name
strWebPath = Request.Params("FileName")
intLastSlash = strWebPath.LastIndexOf("/")
strFileName = Server.MapPath(strWebPath.SubString
                                (0,intLastSlash))
strFileName &= "\" & strWebPath.SubString(intLastSlash + 1)

```

```
'Load the image into memory
  imgFullSize = System.Drawing.Image.FromFile(strFileName)
'Get a Thumbnail image
  imgThumbnail = imgFullSize.GetThumbnailImage(100, 100,
                                                Nothing, Nothing)
'Save the image as a Memory Stream
  stmThumbnail = New MemoryStream()
  imgThumbnail.Save(stmThumbnail, Imaging.ImageFormat.Jpeg)
'Send the image back to the client browser
  intFileLength = stmThumbnail.Length
  ReDim ByteValues(intFileLength)
  stmThumbnail.Position=0
  BytesRead = stmThumbnail.Read(ByteValues, 0, intFileLength)
  If BytesRead > 0 Then
    Response.BinaryWrite(ByteValues)
  End If
  stmThumbnail.Close()
%>
```

إضافة الصور إلى التطبيق

الخطوة الأخيرة لإتمام هذا التطبيق هي إنشاء صفحة ASP.NET التي تمكن المستخدم من إضافة صور جديدة إلى الموقع. قم بإنشاء ملف جديد باسم `upload.aspx` أسفل المجلد `Books` ثم قم بإدخال الكود التالي إلى هذا الملف:

```
<HTML>
<SCRIPT LANGUAGE="VB" RUNAT="SERVER">
Sub btnUpload_Click(sender As Object, e As EventArgs)
  Dim strCatFolder As String
  Dim strUserFileName As String
  Dim strFileName As String
  Dim intLastBackslash As Integer

  'Get the file name without the user's local path
  strUserFileName = filePicture.PostedFile.FileName
  intLastBackslash = strUserFileName.LastIndexOf("\")
  strFileName=strUserFileName.SubString(intLastBackslash + 1)
  'Get the physical path to the category folder
```

```
strCatFolder = Server.MapPath("/books") & "\" &
txtCategory.Value
'Save the file
filePicture.PostedFile.SaveAs(strCatFolder & "\" &
strFileName)
'Display the pictures page again
Server.Transfer("pictures.aspx?category=" &
txtCategory.Value)
End Sub

Private Sub Page_Load(ByVal sender As System.Object, ByVal e
As System.EventArgs) Handles MyBase.Load
'Loads the category name into the hidden text field
txtCategory.Value=Request.Params("Category")
End Sub
</SCRIPT>
<BODY>
<H1>Upload a new Book Image</H1>
<HR>
<FORM ENCTYPE="multipart/form-data" RUNAT="SERVER"
ACTION="UPLOAD.ASPX">
Select image File:
<INPUT ID="filePicture" TYPE="FILE" RUNAT="SERVER">
<INPUT ID="txtCategory" TYPE="HIDDEN" RUNAT="SERVER">
<INPUT ID="btnUpload" TYPE="BUTTON" RUNAT="SERVER"
VALUE="Upload" OnServerClick="btnUpload_Click">
</FORM>
</BODY>
</HTML>
```

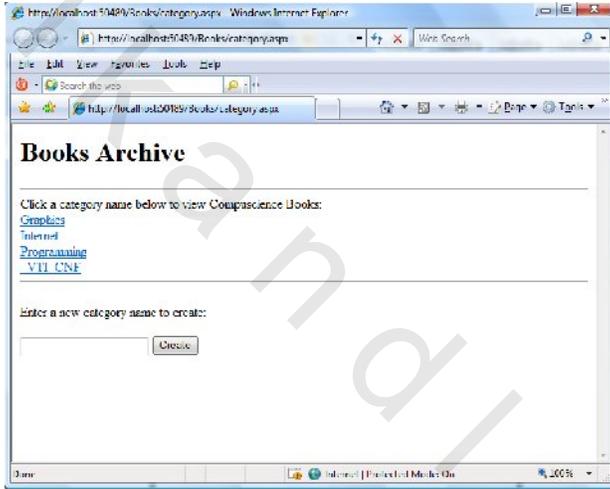
اختبار التطبيق

لتجربة عمل التطبيق الآن بعد الانتهاء من إنشائه، تابع معنا الخطوات الآتية:

١. افتح مستعرض الويب ثم قم بفتح الصفحة

<http://localhost/books/category.aspx>

٢. قم بإدخال اسم مجموعة الكتب ثم انقر زر **Create**، تقوم الصفحة بإظهار هذه المجموعة كما يتم إنشاء مجلد باسم المجموعة أسفل الجلد الرئيسي للتطبيق **.Books**.
٣. قم بتكرار الخطوة السابقة لإضافة مجموعات جديدة (انظر شكل ١-١٠).
٤. قم بنسخ الصور إلى المجموعات المختلفة على القرص الصلب.
٥. من صفحة الويب، انقر اسم المجموعة لمشاهدة الصور التي بداخلها (انظر شكل ١-١١).



شكل ١-١٠ عرض مجموعات الكتب داخل الصفحة



شكل ١-١١ عرض صور المجموعة المختارة

٦. انقر أي من الصور لتكبيرها وعرضها في نافذة مستقلة.
٧. لإضافة صورة جديدة إلى المجموعة من خلال المستعرض، انقر الارتباط **Upload a new Book image** حيث تظهر نافذة جديدة لتحديد الصورة التي ترغب في إضافتها.
٨. انقر زر **Browse** ثم قم بتحديد الصورة المطلوبة.
٩. انقر زر **Upload** لإضافة الصورة المحددة إلى قائمة صور المجموعة.

