



From Technologies to Solutions

SharePoint

Designer Tutorial

Working with SharePoint Websites

Get started with SharePoint Designer to put together a business site with SharePoint

Mike Poole

[PACKT]
PUBLISHING

SharePoint Designer Tutorial

Working with SharePoint Websites

Get started with SharePoint Designer to put together a business site with SharePoint

Mike Poole



This material is copyright and is licensed for the sole use by George McCAVITT on 7th March 2009
Corndon, Manor Road, Stourpaine, BLANDFORD FORUM, Dorset, DT11 8TQ

SharePoint Designer Tutorial

Working with SharePoint Websites

Copyright © 2008 Packt Publishing

All rights reserved. No part of this book may be reproduced, stored in a retrieval system, or transmitted in any form or by any means, without the prior written permission of the publisher, except in the case of brief quotations embedded in critical articles or reviews.

Every effort has been made in the preparation of this book to ensure the accuracy of the information presented. However, the information contained in this book is sold without warranty, either express or implied. Neither the author, Packt Publishing, nor its dealers or distributors will be held liable for any damages caused or alleged to be caused directly or indirectly by this book.

Packt Publishing has endeavored to provide trademark information about all the companies and products mentioned in this book by the appropriate use of capitals. However, Packt Publishing cannot guarantee the accuracy of this information.

First published: July 2008

Production Reference: 1100708

Published by Packt Publishing Ltd.
32 Lincoln Road
Olton
Birmingham, B27 6PA, UK.

ISBN 978-1-847194-42-8

www.packtpub.com

Cover Image by Vinayak Chittar (vinayak.chittar@gmail.com)



This material is copyright and is licensed for the sole use by George McCAVITT on 7th March 2009
Corndon, Manor Road, Stourpaine, BLANDFORD FORUM, Dorset, DT11 8TQ

Credits

Author

Mike Poole

Project Manager

Abhijeet Deobhakta

Reviewer

John Jansen

Project Coordinator

Lata Basantani

Senior Acquisition Editor

Douglas Paterson

Indexer

Hemangini Bari

Development Editor

Ved Prakash Jha

Proofreader

Chris Smith

Technical Editor

Dhiraj Bellani

Production Coordinator

Aparna Bhagat

Editorial Team Leader

Mithil Kulkarni

Cover Work

Aparna Bhagat

About the Author

Mike Poole owns and runs 2F3 Internet, which he established in 1997 to specialize in providing Microsoft-based web development and IT training.

He has received a much acclaim for his web development from many satisfied customers including the BBC, British Medical Association, Microsoft, and six different agencies of the Scottish Government. His work has won him awards from Scottish Enterprise (winner of best e-commerce site) and Microsoft (IM bot competition finalist and winner in both UK and international phases).

Mike has also written and taught courses in Microsoft Excel and Web Development at Edinburgh University.

His current technological interests include creating highly efficient/scalable database solutions and integrating online solutions into virtual environments.

About the Reviewer

John Jansen is a Test Lead for the SharePoint Designer product group in Microsoft. He has been working at Microsoft for the past nine years, and before that, was a freelance website designer, as well as database administrator and teacher.

*Dedicated to my parents for generously allowing me to squander their money
on computers at university, my wife for sitting patiently at dinner parties while
friends badger me with their IT questions, and my God who still loves me despite having
to compete with technology for my attention.*

Table of Contents

Preface	1
Chapter 1: Introduction to SharePoint	7
What is SharePoint?	7
Why Choose SharePoint Designer?	9
What is SharePoint Designer?	10
Installing SharePoint Designer	10
Connecting to an Existing SharePoint Site	13
Further Information	15
Summary	15
Chapter 2: SharePoint Designer and its Environment	17
Development Tool Options	18
Tour of the Environment	18
Task Panes	19
Organizing Our Task Panes	25
Code View	26
The Button Bar	27
The Status Bar	28
Summary	31
Chapter 3: Adding Content and Tables	33
The Wine Company	33
Creating a New Site	33
Creating Our First Page	35
Adding and Formatting Text	37
Previewing Our Page	38
Creating Hyperlinks	39
Adding Images	41
Creating Tables	42
Table and Cell Properties	43
Layout Tables	47

Organizing Our Files	49
Publishing Our Site	49
Viewing Our Page	51
Authorization	53
Editing Existing Sites	53
Summary	54
Chapter 4: Formatting Pages	55
The Wine Company Website	55
Creating Our Site	56
Creating a New Site	56
Creating Our Homepage	57
Publishing Our Site	58
Formatting Our Homepage	58
Using Layers	61
Adding an Image to Our Layer	64
Publishing Images	65
Renaming Our Styles	67
Cascading Style Sheets	67
Editing Styles	69
Master Pages	70
Where Are Our Master Pages Stored?	71
Creating a Master Page	71
Editing Our Master Page	71
Adding a Content Region	73
Saving Our Master Page	74
Attaching Our Master Page to an Existing Page	74
Creating a New Page Using a Master Page	74
Modifying the Master Page	76
Attaching Our StyleSheet to Our Master Page	77
Summary	78
Chapter 5: Collaborating with Other Contributors	79
Contributor Mode	79
Server-Based Sites versus Disk-Based Sites	80
Enabling Contributor Settings	81
Contributor Groups	82
Region Types	83
Setting Up Contribution on Our Master Page	85
The Contributor's Experience	86
Workflows	87
Workflow Designer	87

Workflows and Lists	88
Defining New Workflows	90
Summary	92
Chapter 6: Collecting Data	93
Data Sources	93
Creating Our XML Data Source	94
Creating a Data View	95
Adding and Deleting Records	99
InfoPath	100
Summary	101
Chapter 7: Displaying Data	103
Formatting the Data View	103
Direct Formatting	103
CSS Formatting	103
Conditional Formatting	107
Formatting Numbers	109
Filtering Data	110
Using Formulae	110
Sorting Data	113
Allowing Users to Sort the Data	114
Paging	114
Summary	115
Chapter 8: Adding Web Parts	117
What Are Web Parts?	117
Web Part Zones	118
Inserting a Web Part Zone	119
Inserting a Web Part	120
Adding Graphs	122
Summary	127
Chapter 9: Using ASP.NET Controls	129
ASP.NET Controls	129
Standard Controls	130
Data Controls	130
Validation Controls	131
Navigation Controls	131
Login Controls	131
Adding a Simple Control	132
The Menu Control	133
The Calendar Control	134
Editing the web.config File	136

Validating Our Forms	137
Creating a Login Feature	137
Configuring SQL Server	139
Adding Our First User	140
Adding a New Virtual Server	141
Adding a Host Header	142
Adding an A Record	142
Extending the Virtual Server	142
Using Visual Studio's Web Site Administration Tool	144
Changing the Authentication Provider	146
Return to the web.config File	148
Additional Configuration Tweaks	148
Summary	149
Chapter 10: Integrating with Exchange	151
Introduction to Outlook Web Access Web Parts	151
Viewing the Outlook Web Access Web Parts	152
Enabling Outlook Web Access	153
Enabling Forms-Based Authentication	154
Integrating the My Tasks Web Part	155
Troubleshooting the Error Messages	156
Summary	158
Chapter 11: Search Tools	159
Federated Searching	160
Search Web Parts	160
Using the Search Web Parts	162
Hard-Coding Results	164
Other SharePoint Search Solutions	165
Search Center	165
Search Server 2008	165
Search Term Vocabulary	166
Summary	166
Index	167

Preface

SharePoint is a web-based collaboration and document management platform from Microsoft. Microsoft Office SharePoint Designer (SPD) is a WYSIWYG HTML editor and web design program, which has replaced FrontPage, and is the ideal environment for working with pages on a SharePoint site.

This book is ideal for people new to SharePoint Designer who need to put together a working SharePoint site as quickly as possible. If you want to get started, and finished, as quickly as possible, this book is for you. You won't just learn how to use SharePoint Designer; you'll see how to use it to put together a SharePoint site.

This book will introduce you to the SharePoint Designer environment, and lead you through the key features as you complete important SharePoint customization activities. Throughout the book, you will be developing an example site for a wine business, and you will see what help SharePoint Designer offers, and step through clear instructions to get things done.

The book begins by familiarizing you with the Designer environment and helping you to connect to your SharePoint site. You will then learn how to add and format content, and use SharePoint's workflow tools to collaborate with other content creators before learning how to connect to different SharePoint data sources. You will also learn to use ASP.NET Web Parts in your SharePoint site to create calendars, graphs, integrate with Exchange Server, and add powerful search tools to your site.

What This Book Covers

In *Chapter 1*, we will learn what SharePoint is and why we should choose SharePoint Designer for developing SharePoint sites. Then, we will learn how to install SharePoint Designer and connect to an existing SharePoint site.

In *Chapter 2*, we will learn what SharePoint Designer does and how to get the most out of the interface. We will learn how to arrange the IDE's task panes , and about the benefits of switching between Design view and Code view. We will also get familiar with the features discretely tucked away on the Status bar to ensure that our pages conform to agreed standards and will render correctly in the browsers visitors use.

In *Chapter 3*, we will learn how to create a new site, add pages to that site, add text and graphics to our page, and preview the site in our web browser. We will also learn how to create tables and will learn about the benefits of layout tables. We will learn to organize our files and publish our site. We will also learn about the methods we can use to create a consistent style for our site.

In *Chapter 4*, we will learn to create new pages that follow a consistent theme and allow other users to contribute towards our site while still keeping our styles protected.

In *Chapter 5*, we will see that SharePoint Designer is more than just a tool to allow us to design pretty pages. When used properly, it becomes an integral part of a company's business processes.

In *Chapter 6*, we will collect data in our SharePoint site and examine the many methods that SharePoint has for allowing us to display that data.

In *Chapter 7*, we will discover how easy it is to interrogate a whole range of data sources and display information from them in our SharePoint site in an attractive and useful manner.

In *Chapter 8*, we will learn what Web Parts are and how they can be added to our site. We will also learn about the benefits of grouping our Web Parts within Web Part Zones.

In *Chapter 9*, we will learn about what ASP.NET controls are and which ones are available to us. We will also see how to implement simple controls, menu controls, calendar controls, validation controls, and login controls into our pages.

In *Chapter 10*, we will witness the power of the ready-made tools that SharePoint makes available to us. We will learn how to use OWA Web Parts to display our Exchange information in our SharePoint site and will learn about the configuration changes that are required to do so successfully and how to troubleshoot any error messages that we may come across.

In *Chapter 11*, we will learn to use the search capabilities that SharePoint provides for us. We will also see how to add search forms and results lists to our pages.

What You Need for This Book

In order to work through the examples in this book you will need to have Microsoft Office SharePoint Designer 2007 installed on your computer (no surprise there!). You should also ensure that you have access to a SharePoint site, either on your local network or across the Internet.

It would also be beneficial to have administrative access to Windows Server 2003 or 2008, Microsoft Exchange Server 2007 and Microsoft SQL Server 2005 for several of the more advanced examples.

Who This Book is For

This book is ideal for people new to SharePoint Designer who need to put together a working SharePoint site as quickly as possible.

No experience of SharePoint Designer is expected, and no skill with creating SharePoint sites is assumed.

Conventions

In this book, you will find a number of styles of text that distinguish between different kinds of information. Here are some examples of these styles, and an explanation of their meaning.

Code words in text are shown as follows: "We could have left the `VirtualPath` blank so that server-side scripts are allowed in all SharePoint sites, but limiting it in this way ensures that only our birthday page has script access."

A block of code will be set as follows:

```
a:link, a:visited, a:active {
    color:#903;
    padding-top:1px;
    padding-bottom:1px;
    border-color:#903;
    border-width:1px;
    border-style:solid;
    text-decoration:none;
}
a:hover {
    background-color:#EEE;
}
```

New terms and **important words** are introduced in a bold-type font. Words that you see on the screen, in menus or dialog boxes for example, appear in our text like this: "We then click the **Create** button in our **Conditional Formatting** task pane and select **Apply Formatting**".

 Important notes appear in a box like this.

 Tips and tricks appear like this.

Reader Feedback

Feedback from our readers is always welcome. Let us know what you think about this book, what you liked or may have disliked. Reader feedback is important for us to develop titles that you really get the most out of.

To send us general feedback, simply drop an email to feedback@packtpub.com, making sure to mention the book title in the subject of your message.

If there is a book that you need and would like to see us publish, please send us a note in the **SUGGEST A TITLE** form on www.packtpub.com or email suggest@packtpub.com. If there is a topic that you have expertise in and you are interested in either writing or contributing to a book, see our author guide on www.packtpub.com/authors.

Customer Support

Now that you are the proud owner of a Packt book, we have a number of things to help you to get the most from your purchase.

Downloading the Example Code for the Book

Visit http://www.packtpub.com/files/code/4428_Code.zip to directly download the example code.

The downloadable files contain instructions on how to use them.

Errata

Although we have taken every care to ensure the accuracy of our contents, mistakes do happen. If you find a mistake in one of our books – maybe a mistake in text or code – we would be grateful if you would report this to us. By doing this you can save other readers from frustration, and help to improve subsequent versions of this book. If you find any errata, report them by visiting <http://www.packtpub.com/support>, selecting your book, clicking on the **let us know** link, and entering the details of your errata. Once your errata are verified, your submission will be accepted and the errata added to the list of existing errata. The existing errata can be viewed by selecting your title from <http://www.packtpub.com/support>.

Piracy

Piracy of copyright material on the Internet is an ongoing problem across all media. At Packt, we take the protection of our copyright and licenses very seriously. If you come across any illegal copies of our works in any form on the Internet, please provide the location address or the website name immediately, so we can pursue a remedy.

Please contact us at copyright@packtpub.com with a link to the suspected pirated material.

We appreciate your help in protecting our authors, and our ability to bring you valuable content.

Questions

You can contact us at questions@packtpub.com if you are having a problem with some aspect of the book, and we will do our best to address it.

1

Introduction to SharePoint

"Collaboration", "empowerment", and "information worker" are common buzzwords that are never far away in the world of SharePoint promotion, and which accurately convey the essence of SharePoint.

In this book, we will delve into the matters that are beyond the marketing speak and gain a deeper understanding of what SharePoint is. We will also learn to use SharePoint Designer to create and maintain sites that run on the platform.

My hope is that as you come to understand SharePoint better, you will associate your own adjectives with SharePoint such as "cool", "powerful", and "straightforward".

What is SharePoint?

"So, what is SharePoint then?" is a question that I hear frequently, not just from computer users but also, surprisingly, from savvy IT professionals. With over 24 different Microsoft Server products, it is understandable that not everyone knows what SharePoint is.

The aim of SharePoint is to improve team productivity by allowing staff to collaborate efficiently and providing them with the information they require. The information workers (i.e. staff) are being empowered!

SharePoint is a web-based collaboration, document management, and process management product that allows us to build an enterprise portal. It provides us with the framework to create websites that not only provide access to documents and shared workspaces but also allow other web-based applications such as wikis and blogs to be created. It also allows elaborate workflows to be created, allowing business processes to be monitored and actioned.

SharePoint makes this possible by pulling together the following existing Microsoft technologies and making them available to us for use:

- ASP.NET (including Web Parts)
- Internet Information Services (IIS)
- Active Directory
- SQL Server

The title, SharePoint, actually refers to two different Microsoft products:

- **Windows SharePoint Services 3.0 (WSS)** is a free add-on for Windows Server 2003 and 2008, which provides us with the following SharePoint basics:
 - Basic document management with version control
 - Wiki
 - Blog
 - RSS support
 - Workflows
 - Meeting workspaces
 - Team sites
 - Form library
 - Discussion lists
 - Web part customization
- **Microsoft Office SharePoint Server 2007 (MOSS)** must be purchased separately, and adds lots of additional functionality to the basic functions already provided by the WSS Platform:
 - Improved document management
 - Enterprise search
 - Project management (by integrating with Microsoft Project Server)
 - Excel services (only available in the enterprise edition of MOSS 2007)

In addition to these, SharePoint is a great way to share and exchange information such as calendars and to-do lists.

Although SharePoint is of benefit to small businesses, which can use it to develop sites without the need of much customization, the cost of a MOSS implementation can be prohibitive. Many of the server applications that SharePoint likes to interact with, such as Active Directory and Exchange Server, are absent from small business environments. This makes SharePoint a less natural choice for such companies.

SharePoint is more commonly found in medium-sized companies and large enterprises where the requirement for collaboration is greater (due to the larger workforce). Larger companies often find it easier and more cost-effective to implement SharePoint because most of the IT infrastructure is already in place.

Why Choose SharePoint Designer?

I am sure you will already have guessed that Microsoft Office SharePoint Designer 2007 is a web design tool that allows us to edit and deploy SharePoint websites. In this book, we will be using Microsoft Office SharePoint Designer 2007 to build SharePoint sites.

Microsoft Office SharePoint Designer 2007 is often referred to simply as SPD. I will refer to it as SharePoint Designer from now on in an attempt to save a few trees as well as your precious time.

Why choose SharePoint Designer? My reason is this: there is pretty much nothing that you would want to do to your SharePoint site that you cannot do with SharePoint Designer. If, on the other hand, you were to design your site using a product other than SharePoint Designer or Visual Studio, then you would soon find that you are missing out on being able to use many of SharePoint's built-in features (workflow for example).

With copies selling for US\$275 (or a pricier £230 on the other side of the Pond) from your favorite online store, it is a little more than you would pay for a copy of Microsoft Word, but you are buying yourself a lot of power.

Unfortunately, although SharePoint Designer is a part of the Office family, it is not included in any of the Office system suites.

If you don't already own a copy of SharePoint Designer, it is worth checking if you are eligible to subscribe to Microsoft's Action Pack that includes annual licenses for not just SharePoint Designer but also Windows Server 2003, SharePoint Server Enterprise 2007, and SQL Server 2005 Standard Edition, which are useful if you would like to build a test SharePoint network.

What is SharePoint Designer?

Even very competent and technically-minded people ask me what SharePoint is or what SharePoint Designer is. It is not an easy topic to explain. The following paragraph is how I would sum up SharePoint Designer:

SharePoint Designer fills in the space in the Office family that was vacated by Microsoft FrontPage (Microsoft's previous HTML editing program). It combines the familiar Office user interface, which was popular with users of FrontPage, with the power of Windows SharePoint Services and Microsoft Office SharePoint Server, allowing data and reports to be built into your SharePoint site easily with the help of task panes and templates.

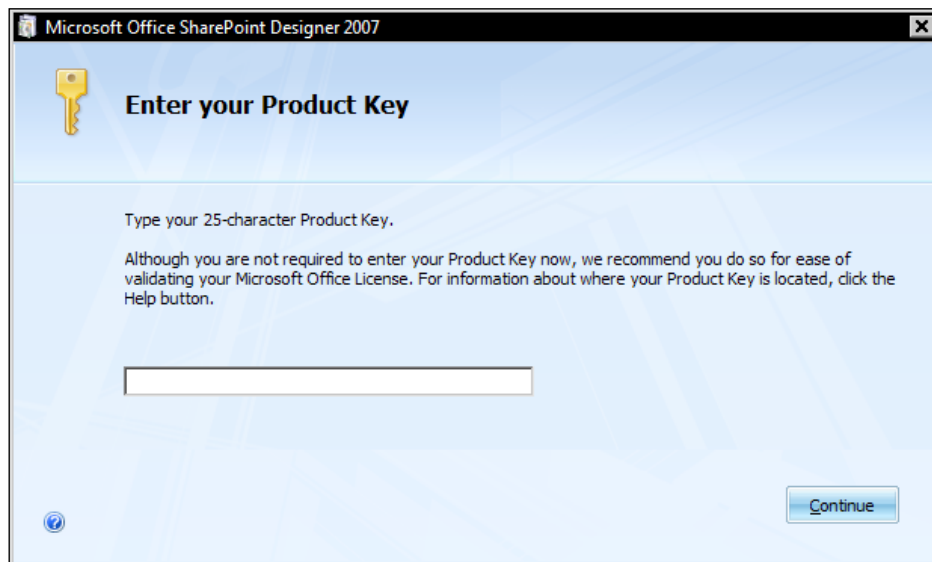
This may sound a little obscure or like marketing speak and not really put you in the picture. Thankfully, like many things in life, it is easier to understand the concept when we see it in action, so let's dive into it and take a look.

Installing SharePoint Designer

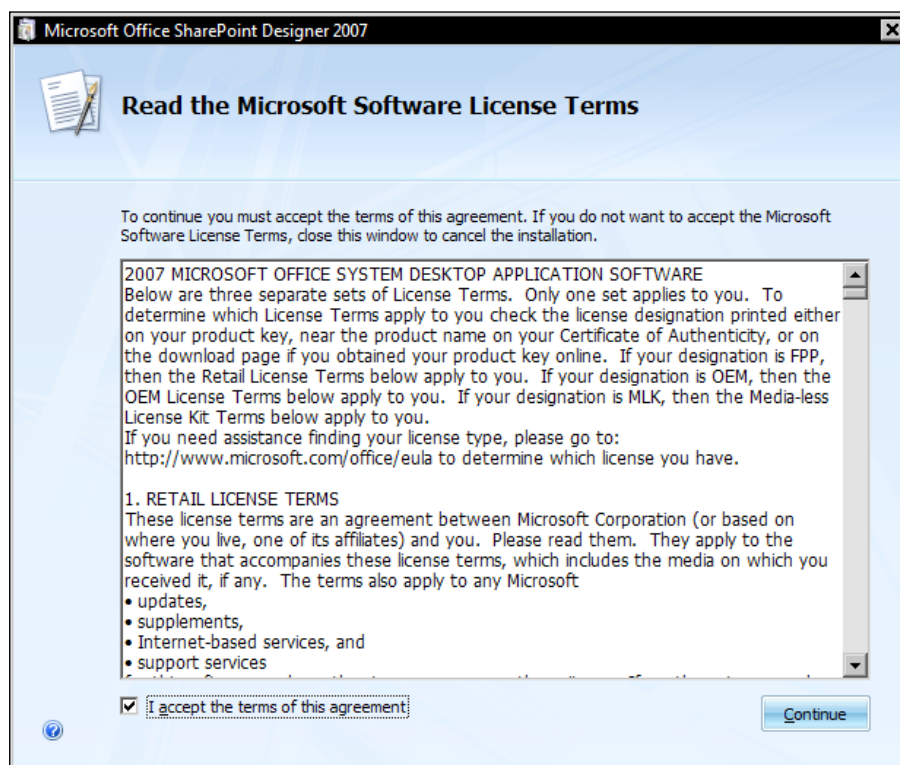
If you have not installed SharePoint Designer already, let's kick off by installing it on our computer. We begin by inserting the installation media (e.g. DVD). We then enter our product key and click the **Continue** button.



If our installation does not autorun, then we will need to start the installation by double-clicking on the setup icon.



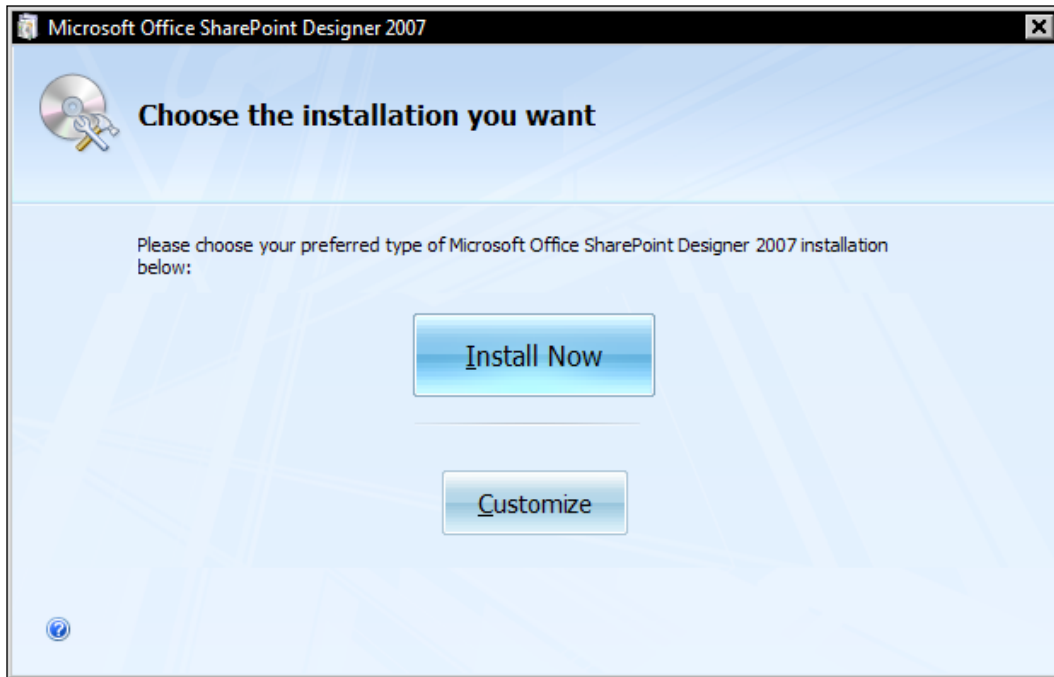
Next, we carefully read all ten thousand words of the license agreement, tick the checkbox to say that we accept the terms of the agreement, and click the **Continue** button.



On the next screen, we have the option of installing the default configuration by clicking the **Install Now** button or of customizing our installation.

Virtually all of SharePoint Designer is installed by default. If we were to click on the **Customize** button, we would be able to install additional Office features such as Japanese font support and proofing tools. We would also be able to specify an alternative file location and provide different user information.

We will opt for the default setup by clicking the **Install Now** button (it is, after all, larger than the other button, so would seem to be the one that Microsoft would like us to opt for).

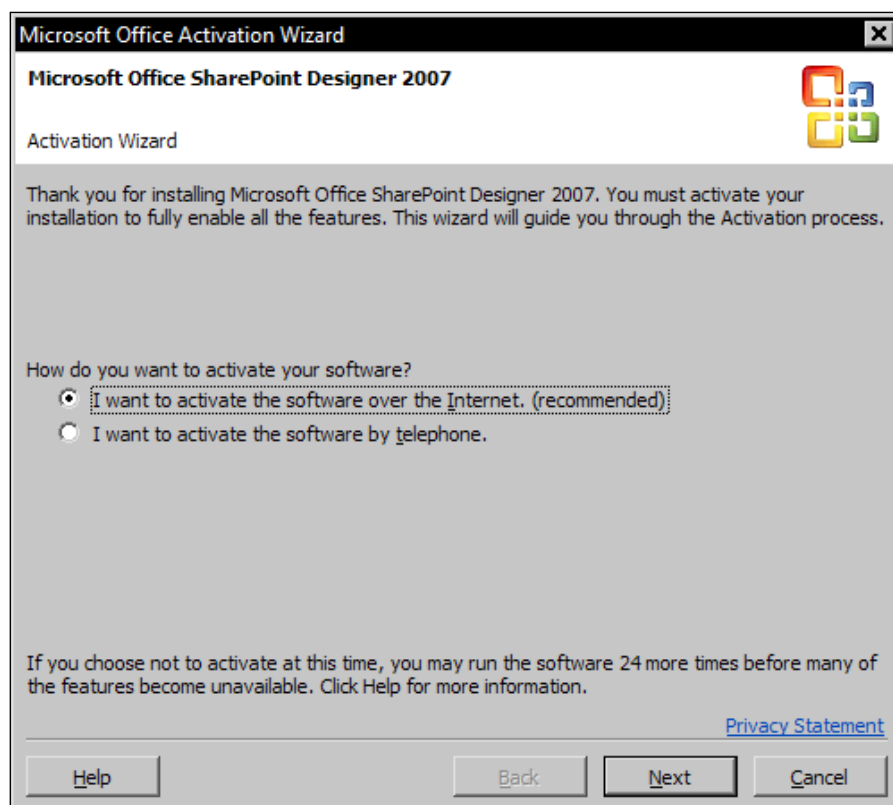


The installation program then spends about three minutes (or longer if we have a slow computer), installing SharePoint Designer on our system. Once it is finished, we can click on the **Close** button.

We can then start SharePoint Designer by going to **Start | All Programs | Microsoft Office | Microsoft Office SharePoint Designer 2007**.

If we are using another website editing program on our computer, SharePoint Designer will ask us if we would like to make SharePoint Designer our default editor.

SharePoint Designer will then present us with the activation wizard. Let's go ahead and activate our product. We are now ready to use SharePoint Designer.



Connecting to an Existing SharePoint Site

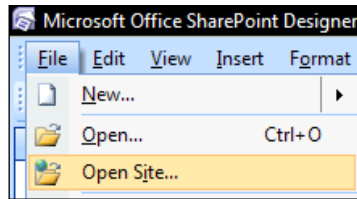
Now that we have successfully installed SharePoint Designer on our machine, we can open up an existing SharePoint site from anywhere on our network, the Internet, or our local computer.



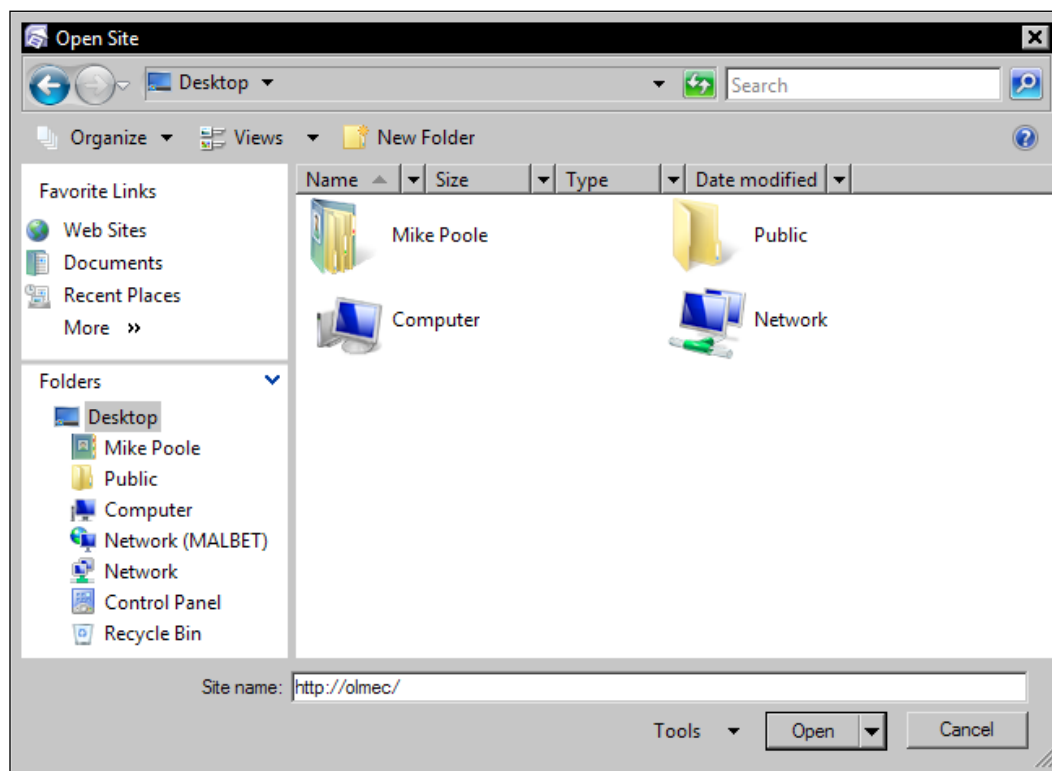
Virtually, all SharePoint sites are hosted on remote web servers on networks or on the Internet, but it is also possible to create simpler sites locally and edit the pages there (e.g. on your laptop when you are on the move). So, we will also cover that in this book.

We will be using a variety of examples in this book to demonstrate various concepts. In Chapter 3, we will be introduced to the Wine Company and create a basic site. We will also create other small sites to illustrate different features.

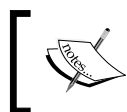
To open a site, we go to **File | Open Site**.



This will present us with the **Open Site** dialog. We simply type the name of our site (e.g. <http://olmec/>) into the **Site name** field and click **Open**. Note that by default, the default SharePoint site name will be the server address (i.e. <http://servername/>).



SharePoint Designer should then connect to our SharePoint site and list the files and folders in the **Web Site** pane in the center of the designer and in the **Folder List** task pane on the left of the designer.



We can only connect to our site if our network login has the necessary permissions to open the SharePoint site. If it does not, then we will be prompted to use a login with the necessary permissions.

Further Information

Further information about SharePoint Server is available from Microsoft on its main website at <http://www.microsoft.com/sharepoint/> and on its Office Online website at <http://office.microsoft.com/sharepoint/>.

The SharePoint Community Portal is also a good resource, not only because it provides lots of useful information but also because it is powered by MOSS 2007, allowing us to see what is possible by taking a peek at the site. Although the main framework of the community portal is built using MOSS 2007, disappointingly, many of the sections such as the blog, the forums, and the QnA do not use MOSS 2007 functionality. You can see the community portal at <http://sharepoint.microsoft.com/sharepoint/>.

Summary

In this first chapter, we learned what SharePoint is and why we should choose SharePoint Designer for developing SharePoint sites.

We also installed SharePoint Designer and learned to connect to an existing SharePoint site. We are now nicely set to continue with the rest of the book.

In the next three chapters, we will become familiar with adding basic content to our site. We will then move on to discover how to collaborate with contributors and connect with databases before spending time on the final four chapters of the book, using Web Parts to extend the functionality of our sites.

2

SharePoint Designer and its Environment

Now that we have been introduced to SharePoint's powerful capabilities and know what we would like to build using them, let's find out more about the tool that would be best to build our masterpiece.

In this chapter, we will cover the following:

- We will examine if there is other software that we can use to develop our SharePoint site.
- We will take a tour of the integrated development environment (IDE) and familiarize ourselves with its elements including:
 - The many task panes that give us access to SharePoint's most powerful controls. We will learn what they are used for and how to arrange them neatly on our screen.
 - The benefits of switching between Design and Code view.
 - The command bar.
 - The Status bar.

Development Tool Options

So, we want to create and maintain our SharePoint site. Which piece of software do we choose to do this with?

- How about **Notepad**? For half of my first decade as a programmer, I wrote my HTML and ASP using nothing but Notepad. Even nowadays, I do the majority of my work in the Code view of whichever IDE I am using. There is nothing wrong with this; it is often the only way to create truly optimal code. It would be feasible to code pages using your favorite text editor, but there are many tasks, such as laying out your Master Pages, Web Part Pages, or Layout pages, where a WYSIWYG editor is necessary. In any case, much of the *raison d'être* for SharePoint is to allow content creators to easily add pages to their site, so let's not make things difficult for ourselves by choosing a tool that is too basic.
- **Visual Studio 2005** then? It is brimming with features and can be a good option if you are a developer, as you almost certainly already have this product and will be familiar with how it operates. It is an expensive option, although you can download the perfectly adequate Visual Web Developer 2005 Express Edition for free from <http://msdn.microsoft.com/vstudio/express/vwd/>. It is also worth noting that Visual Studio and Office SharePoint Designer can be used together by different team members to build a site. Visual Studio also allows us to perform complex tasks, such as modifying `onet.xml`, building application pages on the server file system, writing custom Web Parts, workflow actions, and event receivers. The downside of using Visual Studio is that it has no knowledge of the SharePoint platform, and hence it does not offer a rich design experience (e.g. property grids, WYSIWYG, Web Part customization).
- FrontPage's successor aimed at designers, **Expression Web**, allows excellent control of the look of our sites, but would not be a suitable choice for developing SharePoint sites because it does not allow SharePoint sites to be opened.

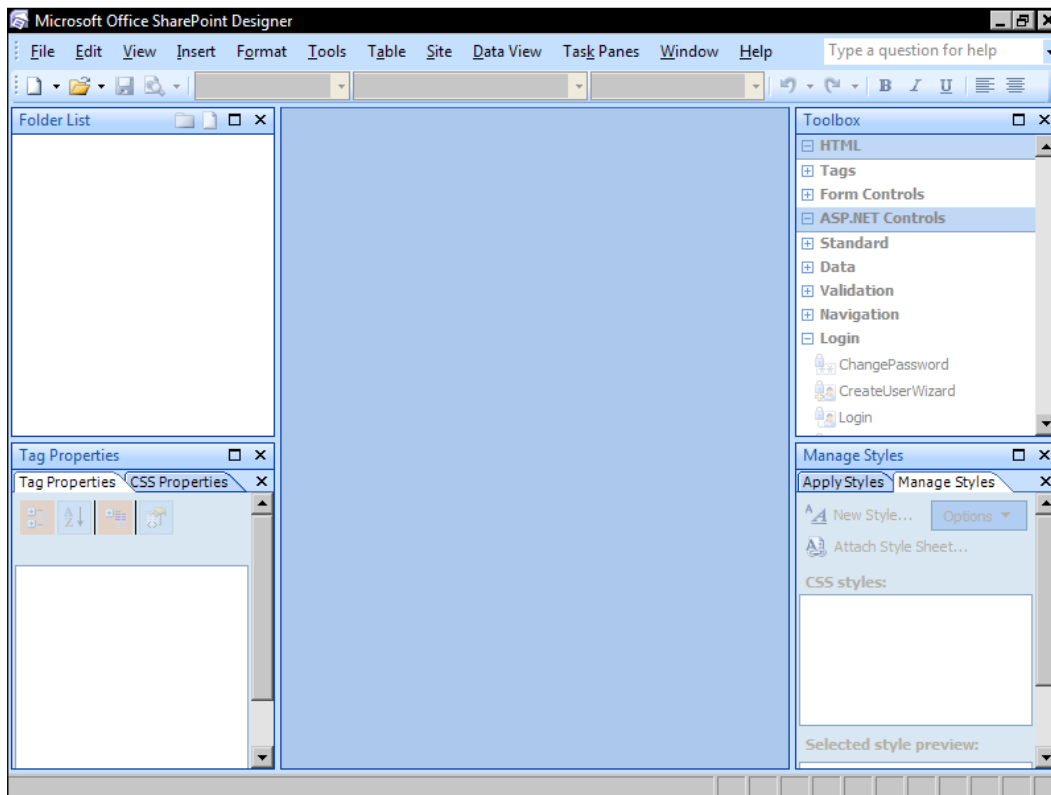
As you will no doubt have guessed from the title of the book, there is another product we should consider – Microsoft Office SharePoint Designer 2007.

Tour of the Environment

Let's fire up SharePoint Designer and take a look around.

If you are familiar with Visual Studio, the first thing you will notice about the design environment is that SharePoint Designer looks remarkably similar to Visual Studio. The other thing that may catch your eye (especially, if you are using a large monitor)

is that there is a big expanse of nothingness in the middle of the screen. This blank canvas is where we will build our creation.



Task Panes

Before we begin tinkering with SharePoint Designer, notice that by default four task panes are open for us to use:

- **Folder List**
- **Tag Properties**
- **Toolbox**
- **Manage Styles**

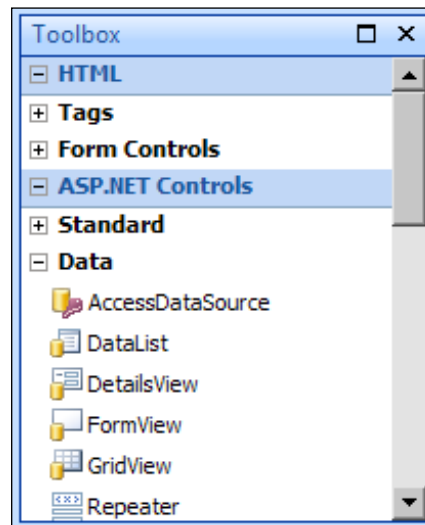


If we have already altered the task panes that are open in the designer, we can reset the task panes to the default view by clicking on **Task Panes | Reset Workspace Layout**.

Because we are not currently editing a page, the contents of these task panes are either empty or grayed out.

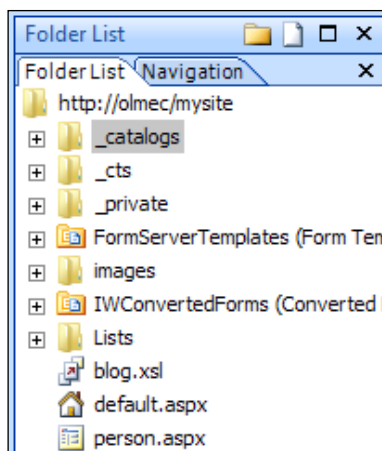
We can open any of the 24 task panes that SharePoint Designer provides us with by selecting them from **Task Panes** on the menu bar. As we do this, notice that some of the task panes add extra tabs to existing task panes, forming groups of task panes. For example, when we go to **Task Panes** and click on **Navigation**, it groups the new **Navigation** task pane alongside the existing **Folder List** task pane. It is possible to "tear off" tabs and re-group them in a different pane, if we decide that the default groupings do not suit our design style. SharePoint Designer will save any changes to the environment to the registry when we close the program to ensure that our settings remain the same the next time we open the designer.

We can have up to six groups of task panes on our page and could in theory have all our task panes open simultaneously.

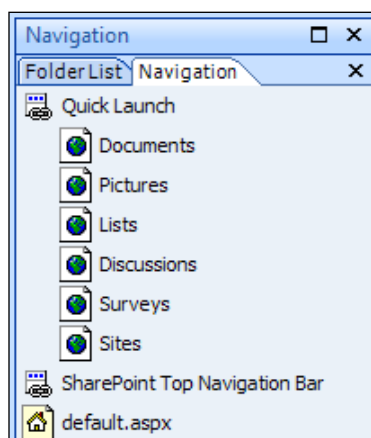


Let's take a look at what each of the task panes does:

- **Folder List** displays all the folders and files in our SharePoint site. As we have not created or opened a site, the list is currently empty. When we create a new site, this pane fills with all the content within the site, as in the following example of a newly created **mysite** (**mysite** contains the personal pages, which are displayed in a SharePoint site at <http://<SERVERNAME>/personal/<USERNAME>/>).



- **Navigation** is similarly empty until we open a site. Once our site is open, we can use it to configure our site's navigational items, such as the navigation bar. We can see an example of a populated **Navigation** task pane in the image below. The **Quick Launch** items that you see in the image populate the Quick Launch bar that appears down the left-hand side of a SharePoint site.



- **Tag Properties** allows us to specify all manner of properties for the entity we have selected on our page. The options that are available vary depending on what type of item we are editing. If, for example, we have inserted an ASP.NET button control onto our page, we can specify formatting details (e.g. the background color or text that appears on the button), or we can specify actions that the button will perform (e.g. the tooltip text which appears that the user hovers their cursor over the button).

- **CSS Properties** allows us to see the style information for the element that we have selected on our web page and to edit that style. This task pane gives us a huge degree of control over the styles on our page, not only allowing us to specify one style but also to change the order of precedence of many different styles. In addition, there is a **Summary** button that allows us to easily see exactly what CSS is cascading to the selected element and understand exactly how to change its appearance.
- **Layout Tables** offers some very powerful features to allow us to instantly create the layout for our new page by creating a table from ten pre-specified table designs and specifying the dimensions of the table and its cells.
- **Apply Styles** allows us to choose a style that we will give to our selected entity. These styles might simply specify the font size and color, but can also be used to give us precise control of entity positioning and borders. This task pane can be set to show all styles used in our site, or just the styles used on the current page. In addition, we can create new styles or attach previously created CSS pages.
- **Manage Styles** is where we specify the styles that are used in the **Apply Styles** task pane. We can either create a new style or select an existing CSS style sheet. We can categorize our styles by CSS order, by element, or by type. We can also create new styles or attach CSS using this task pane. On the left side of the list of selectors, there is an icon indicating whether the selector is a class, an HTML tag, or an ID.
- **Behaviors** will be very familiar to the users of Adobe Dreamweaver. This task pane allows us to specify events that take place when an action is performed at run time. For example, a sound can play or a rollover effect can be displayed on an image.
- **Layers** will be familiar to those of us that have used DHTML in the past. It allows content to be placed in front of or behind other content to create effects such as those annoying adverts we see, obscuring the content of web pages. Fortunately, they can also be used to enrich our sites.
- **Toolbox** is our indispensable task pane. If we could take one task pane with us onto a desert island (along with our computer, power supply, broadband, etc., etc.), this would be the one to choose. We can drag HTML, ASP.NET, and SharePoint elements from here and drop them onto our page. The ASP.NET Web Parts in this task pane are not suitable for use in SharePoint sites. SharePoint Web Parts are displayed in the **Web Parts** task pane, when we are connected to a SharePoint server.

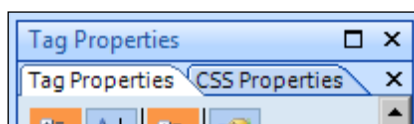
- **Data Source Library** controls the connections to the many different types of data sources that SharePoint allows us to use (e.g. ODBC database connections, XML files and web services, SharePoint lists, and libraries). The task pane displays the data sources for our current site, and can also display data sources for other sites (as long as we have the 'open web' permission on those sites).
- **Data Source Details** displays the information stored in our selected data source. It displays the contents of every field and allows us to page through the data, one record at a time. When we drag a data source from this task pane onto our web page, SharePoint Designer automatically creates a Data View, displaying the first five records contained in the data source on our page. This task pane allows us to see all of the data inside the data source of a currently selected Data View, or inside a data source we chose to view by clicking **Show Data** in the **Data Source Library**. We can also modify the source properties from this task pane, scroll through the records to see the hierarchy of our data source (this is particularly useful when working with XML data), view the XPath expression for each field, and to save space, can choose not to show the actual data, but rather just the schema for our source. When we drag a data source from this task pane onto our web page, SharePoint Designer creates a Data View, displaying all the records for that data source, unless there are more than 24 records, in which case, the view is paged to 10. For XML data, we insert the first five columns (if there are five) of the first repeating region. For List data, we insert three columns.
- **Conditional Formatting** is the same principle as the conditional formatting in Microsoft Excel. It allows us to automatically change the style of an HTML tag or data value in our Data View, depending on the values we specify. For example, if we want all account balances of less than \$250 to be in bold text with a red background, this can be set up using **Conditional Formatting**.
- **Find Data Source** allows us to find the data source we would like to use. This is particularly important if our site contains a large number of data sources, or if we are working with multiple sites. To find the data source we would like to use, we simply select the location from the drop-down list, then type a search phrase (e.g. "links") into the **For:** field.
- **Web Parts** allows us to harness the power of the underlying .NET 2.0 framework by dragging and dropping all manner of ready-made controls onto our page. This is great because most, if not all, of the work for common tasks, such as displaying task lists or building forms, has already been done for us. SharePoint Designer regularly queries the server to check which Web Parts are available in order to keep this task pane up to date.

- **Find 1** and **Find 2** allow us to find or find and replace instances of a text string on our page or in the underlying code (depending upon whether **Design** view or **Code** view is selected). The find tools are extremely powerful and allow us to use regular expressions to conduct our search. We are also able to build highly complex HTML rules (e.g. find all instances of the word foo that exist inside table cells, that are inside divs, that have an attribute called bar and a value of baz). These searches can also be saved and then reused or shared.
- **Accessibility** allows us to check that design-time HTML complies with most of the disability legislation, which is in place to help users with visual impairments use our site. SharePoint Designer will not ensure that the run-time HTML is compliant, so it is a good idea to also check the site using an accessibility evaluation tool such as the ones listed at <http://www.w3.org/WAI/ER/tools/>.
- **Compatibility** allows us to ensure that our site conforms to the CSS, HTML 4.01, XHTML 1.0 and 1.1 standards used by standard-compliant browsers such as Firefox and Safari. We can also check that our site will run on various versions of Internet Explorer, which does not adhere to the standards as closely.
- **Hyperlinks** will display a report of the internal and external links on our site and check to see if any of these are broken.
- **CSS Reports** is great for detecting and removing unused styles from our site. It will also display a list of styles that are used on a page.
- **Clip Art** allows us to search our computer and Microsoft's online image bank for clip art and photos and display those on our web page. A word of caution before adding dancing telephones to your company's contact page. Just because you can to do, it does not mean your marketing department is going to approve of it.
- **Clipboard** will store up to 24 of the most recent items we have copied or cut from any of the applications on our computer and allow these to be pasted into our web page.
- **Contributor** shows us the contributor settings for our site. Collaboration is one of the most powerful features of SharePoint. It allows groups of users to be set up, each with different permissions for viewing and editing a site. By default, this task pane will probably show us as members of the **Content Authors** group, which has some power to edit our pages, but does not allow us complete world domination.

Organizing Our Task Panes

Once we have opened our task panes using the **Task Panes** menu, we have complete control over where we position those task panes on the page:

- **Move** a task pane by clicking and dragging on the title bar of our task pane. If we drag it to the edge of the program window and let go, it will **dock** neatly onto the page either vertically (if we drag it to the left or the right of the window) or horizontally (if we drag it to the bottom or the top of the window).
- **Resize** the task panes by hovering the mouse cursor over the edge of the task pane until the cursor changes to a double-headed arrow and drag the border out or up.
- **Close** task panes we are not using by clicking on the **Close** cross in the top-right of the task pane. If our task panes have been grouped together, then we can either click on the upper cross to close the whole group or on the lower cross to close the active task pane.



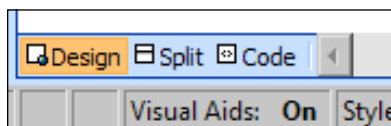
- **Merge** task panes together into groups by clicking on the title bar of a task pane and dragging it onto another task pane. We can even merge groups into other groups by clicking on the title bar of the group and dragging it onto another group.
- **Stack** your task panes or groups of task panes into one column by dragging the title bar of your task pane or group into an existing column of task panes. We can then use the **Maximize** button, when we would like to access the contents of that task pane. If you are using a monitor that has low screen resolution, you may find it helpful to stack your task panes in one column because you will then have enough room in the **Design** view to build your web pages.
- **Float** a task pane anywhere on the screen by right-clicking on the task pane's title bar and selecting **Float** from the very short shortcut menu. If you are fortunate enough to have a computer with multiple monitors, you may like to have SharePoint Designer filling one screen and float all your task panes on the other.
- **Restore** task panes to their default position once you have arranged them in a confusing enough manner by selecting **Reset Workspace Layout** from your **Task Panes** menu.

If you want to be a real whiz, try the keyboard shortcut *Ctrl + Tab* to cycle through the task panes in a group (or *Ctrl + Shift + Tab* to cycle through them backwards).

Another point to note is that in some of the task panes (e.g. **Accessibility**), it may not be immediately obvious that to run a report we must first click on the small green arrow.

Code View

In the center of our design environment, we have the area where we will be building our pages. Clicking in this area allows us to start typing the text that will be on our page. It works in much the same way as Microsoft Word does. We can type some text, highlight it, and format it using different font faces, colors, and sizes. This is of course what is known as the WYSIWYG (What You See Is What You Get) method of working and is nice and straightforward. Microsoft makes a big deal about the fact that we can build SharePoint sites in this way without the need to type any code. That is great for casual users who just want to copy and paste their information onto a web page, but it does not mean that more experienced developers cannot tinker under the bonnet if they want to. This is where the **Code** tab comes in; it allows the view to be flicked from the **Design** view to the **Code** view at the click of a tab. You can see the **Code** tab at the foot of the workspace pane.



When we click on the **Code** tab, the HTML of our page is displayed.

```
Untitled_1.htm
<body>
1 <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
2 <html xmlns="http://www.w3.org/1999/xhtml">
3
4 <head>
5 <meta http-equiv="Content-Type" content="text/html; charset=utf-8" />
6 <title>Untitled 1</title>
7 </head>
8
9 <body>
10
11 </body>
12
13 </html>
14
```

If you think the code that SharePoint Designer creates looks a little confusing, let me assure you that it is far tidier and easier to work with than the code that Microsoft FrontPage used to create.

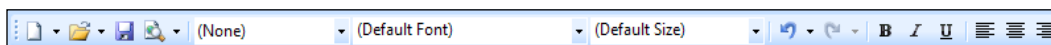
The eagle-eyed among us will have noticed there is also a third tab called **Split**. When we click this, it takes us into **Split** view, which shows us the **Code** view in a pane at the top of the page at the same time as the **Design** view in a pane beneath it. When we work in either of these panes, it moves the other pane up and down, depending on where on the page we are working. Notice that when we highlight an element on the page, it highlights the associated item in the other pane. Nice!

If we are blessed with multiple monitors (and I really do recommend that if you are not, you do everything in your power to ensure that you are, because you will never look back!), SharePoint Designer allows us to make full use of our additional screen area. We can right-click on a page that is in Design view and select the **Open Page in New Window** option. We can move this window to our second monitor and change the view to Code view. This allows us to make changes on one screen and see the changes occur in the other automatically. Now that is slick.

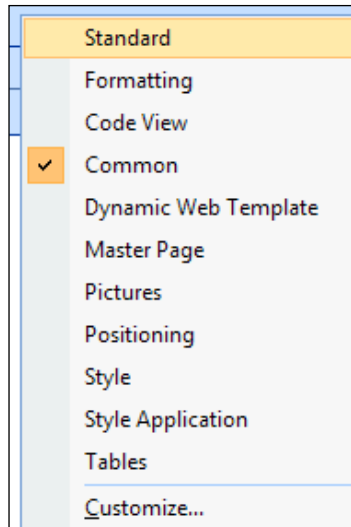
In this book, we will concentrate on designing our site using the **Design** view, but occasionally we will flick into the **Code** view for a brief demonstration of how additional functionality can be added to our site, if we should feel so inclined. You may never use the **Code** view, but it can still be useful to know what can be done should you wish so.

The Button Bar

At the time when the majority of the Microsoft Office suite has upgraded to a shiny new chunky tabbed ribbon bar, SharePoint Designer has retained the traditional command bar at the top of its screen. By default, the Common button bar is displayed, which mainly allows us to format our highlighted text.

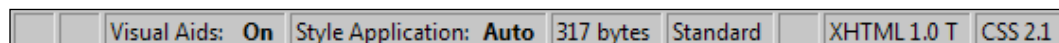


By right-clicking on the empty space to the right of the Common button bar, we bring up a shortcut menu, which allows us to display other button bars that are specific to various tasks, such as inserting tables or selecting which Master Page our page should use.



The Status Bar

The grey Status bar at the foot of the designer displays a host of useful information to help us design pages that not only look good in the designer but also work well in real life.

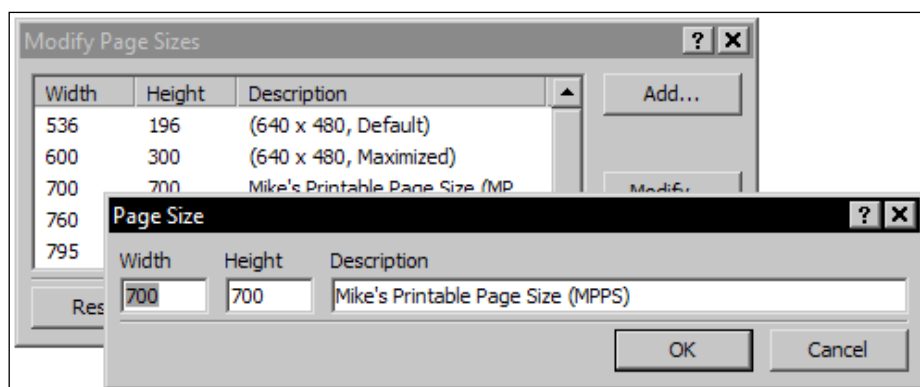


From left to right, the Status bar shows us the following helpful information:

- The **Contextual Message** will usually say **For Help, press F1** (and this is a good tip because I have found the help in SharePoint Designer to be very comprehensive). The message will change occasionally as we develop our site to display useful information. For example, when we click on a hyperlink in the **Design** view, it will display the target address.

- **Visual Aids** are useful markers and error messages, which are displayed on the page at design time to help us compose our page. They do not appear to a visitor when they view our web page. The **Visual Aids** can be toggled on and off by double-clicking on the Status bar cell. Right-clicking on this cell displays a shortcut menu where we can select which visual aids are displayed in the designer, when **Visual Aids** is set to **On**. Personally, I find the **Margins and Padding** visual aid very useful.
- **Style Application** allows us to switch between automatic application of site styles and manual application of styles, which allows us more control of how our page looks. **Manual** style application can be at the expense of a uniform-looking site and should only be used by a developer with a good grasp of CSS. It allows us to render classes and selectors in exactly the way we would like them to appear (e.g. using our corporate colors rather than the default colors provided by SharePoint).
- **Download Statistics** displays the total file size of our HTML and the files embedded on the page. Over time, as the average user's internet connection has become faster, it has become less important that a page is small so that it downloads in good time, but the cost of bandwidth has also risen over the same period. If you pay for the bandwidth used by your web server, then keeping your page sizes small will help keep your bills small. Generally speaking, pages should not need to be more than 60-100KB in size. If the size of our pages is dramatically high, then the first thing we would want to try doing is reduce the quality of our JPEG images, which can halve the overall page size. It is also important to note that the statistics do not include the sizes of any images that are dynamically displayed on the page. For example, if we have a product page that displays information about a bottle of wine (if you like wine, you will love the example site we are going to build in this book), then that page will display one of hundreds of pictures of wine bottles automatically, depending on which product was selected. In this case, SharePoint Designer does not know which image will be selected, so does not include that file information in the download statistics. This is especially important to bear in mind because such dynamic images are cached less frequently and so will result in higher bandwidth usage.
- **Standard Rendering Mode** will display the word **Standard** if our site conforms to an agreed doctype. If we would like to check that our site renders nicely in Internet Explorer 6 (and we should!), then we can double-click on this area and change the **Secondary Schema** to **Internet Explorer 6.0**, thereby enabling the **Quirks** mode. More information about quirks mode can be found at http://en.wikipedia.org/wiki/Quirks_mode.

- **Page Size** displays the current dimensions of our **Design** view. The width of the page in pixels is displayed first, then the height. Right-clicking on this cell brings up a shortcut menu, which allows us to select different dimensions for our page or to create our own page sizes. The dimensions in brackets refer to the screen resolution, whereas the dimensions in front of the brackets refer to the available space within the typical browser window on that screen (given that users seem to collect toolbars, the available vertical space may be less than this guideline predicts). Setting the page dimensions is a feature that we should seriously consider using in all our sites, so that users with smaller screens can view our site without the need to scroll left and right. More importantly, it will mean we don't get an embarrassing phone call asking us why our site does not print off properly.



- **Document Schema** shows the HTML or XHTML doctype declaration that is displayed in the HTML. The doctype declaration tells the browser what type of HTML is being used. By default, the more recent XHTML 1.0 Transitional is used. To change the doctype, double-click on this cell in the Status bar (or if you want to take longer to do exactly the same thing, use **Tools | Page Editor Options** on the menu bar). Choosing the correct doctype allows SharePoint Designer to suggest appropriate options through its built-in **IntelliSense**. SharePoint Designer will alert us to code that does not conform to our chosen doctype by using red squiggles and yellow highlighting. For more information about doctype standards, see the World Wide Web Consortium website at <http://www.w3.org/MarkUp/>.
- **CSS Schema** is similar to the doctype, but tells the user's browser which version of Cascading Style Sheets is being used on the page. By default, it is set to **CSS 2.1**, which is the most recent version. This tool works in a similar way to the **Document Schema** tool, by allowing us to check the validity of our code. For more information about CSS standards see <http://www.w3.org/Style/CSS/>.

Summary

In this chapter, we learned what SharePoint Designer does and how to get the most out of the interface. We learned how to arrange the task panes in the IDE so that we could get access to the controls that we need in a way that suits our working style and the benefits of switching between Design view and Code view. We are also now familiar with the features discretely tucked away on the Status bar, and aware that they are at our disposal, so that we can ensure that our pages conform to agreed standards and will render correctly in the browsers visitors use.

In Chapter 3, we will crack on with creating a new site which we will add content and tables.

3

Adding Content and Tables

Now that we have learned about the foundations of SharePoint and the designer, let's crack on and build and publish our first page.

But first, a few words about wine.

The Wine Company

The Wine Company Limited is a fictitious company that has been selling wines from all over the world via the Internet to customers in the UK, since March 2000. It grew from humble beginnings and now employs 25 people at its office and warehouse complex in the central belt of Scotland. Its staff includes one marketing manager called Gavin and an all-round IT person called Tony, who looks after everything from PC support to database management. Tony is also responsible for maintaining the company websites.

Over the last seven years, the company has had two public facing e-commerce websites developed by local companies, but in the summer of 2007, with the help of a SharePoint consultant, decided that SharePoint was finally mature enough to meet its needs.

Throughout this book, we will be using the Wine Company website as our example.

Creating a New Site

Let us proceed and build our first page. We could just create a page straight away, but because we would like to publish our page to the SharePoint server, once we have done so, we will create a new site, which we will call "hello world". To create a new site, follow these steps:

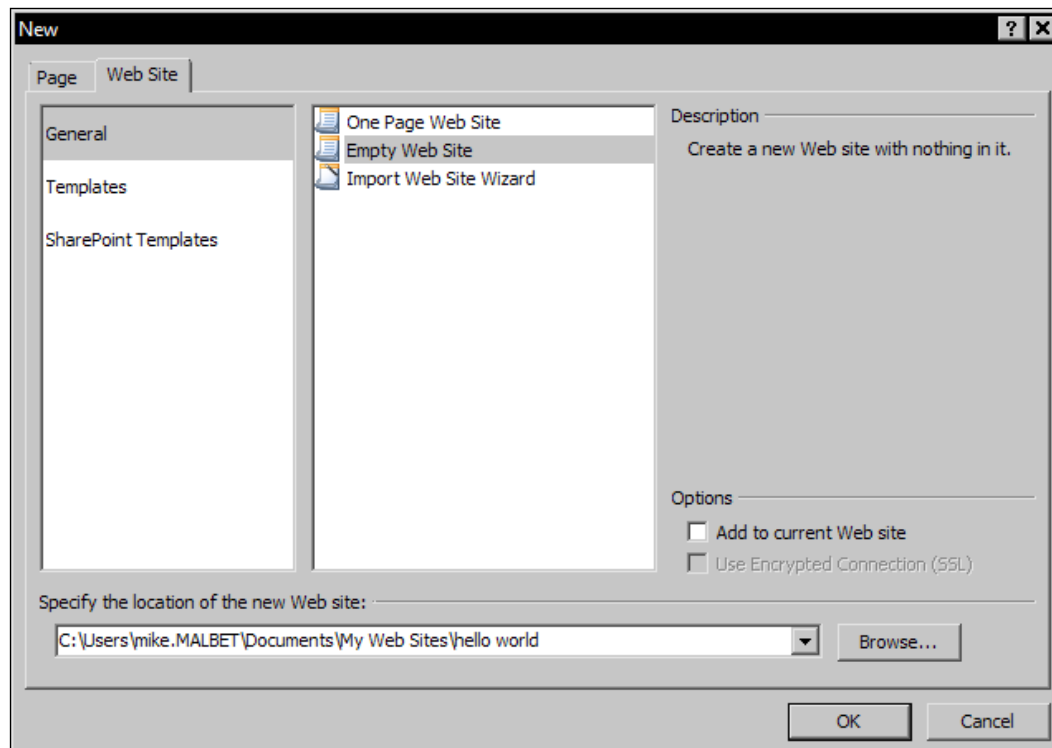
1. From the menu bar in SharePoint Designer, select **File | New | Web Site**.
2. Select **Empty Web Site** from the list of possible pages.

3. We can store our local version of the site anywhere we wish on our computer or network. For now on, we will use the default location inside My Documents because it is as good a place as any. Give your new site a new folder within the My Web Sites folder (e.g., C:\Users\<USERNAME>\Documents\My Web Sites\hello world).
4. Click **OK**.



Note that during this process, we created a new folder for our hello world site. This is important because usually we will create more than one website on our computer. Having our sites in different folders helps keep things tidy (and as you will see, I like tidiness!).

When going through this process, we will see the **New** dialog which will be similar to the one below.



Creating Our First Page

If we had selected the **One Page Web Site** option in the previous section, a blank page called `Untitled_1.htm` would have been displayed in the center of our designer. It would have been possible for us to have used this as our first page, but I recommend we use a different type of page for all the web pages in our site.

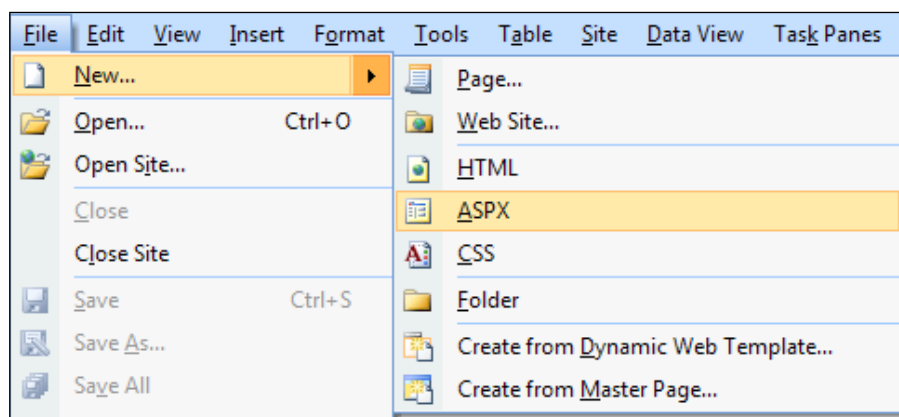
HTML pages (i.e. files with the `.htm` suffix) are simple souls. They will display static information very nicely, but when it comes to actually doing something (such as displaying a list of products contained in our database), they will shrug their shoulders and say they can't do that.

If you want your pages to do more than look pretty (and I am sure you do because you have an interest in SharePoint), then Active Server Pages are for you. The latest generation of ASP pages have `.aspx` as their suffix.

I recommend always creating ASPX pages because they can contain dynamic content, SharePoint-specific controls, and other ASP.NET controls when necessary.

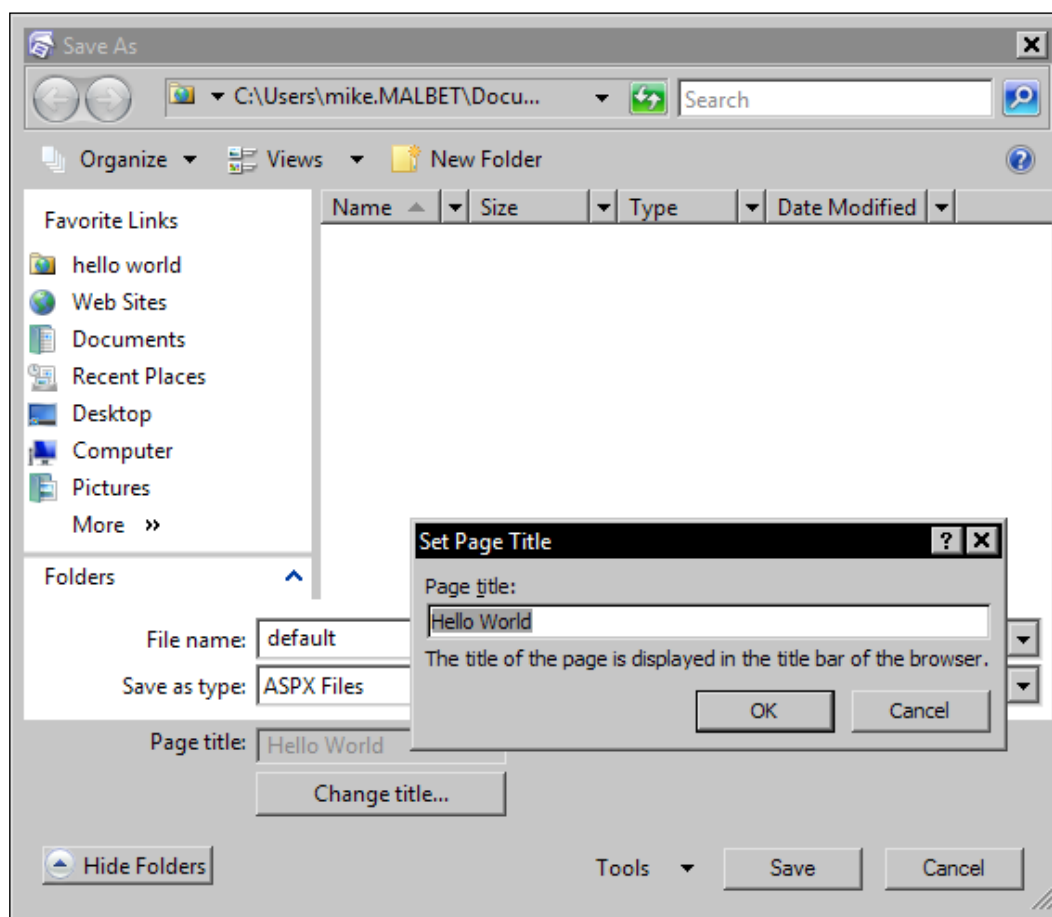
For example, if we start with an HTML page, which only displays a photo and a few paragraphs of text, then the time comes when we decide we want to quickly add a list of the top ten bottles of wine from our database to that page and we will have to tear up the old HTML page and create it as a new ASPX page. When we do this, not only will all the links to the page on our site and other websites need to change, but we will also lose any good search engine rank the page had and any bookmarks that people had pointing to that page. So, I think it is better to just start with ASPX, even if we don't have any dynamic content to start with.

To add a new page, select **File | New | ASPX** from the menu bar.

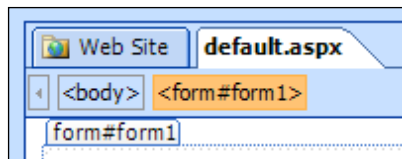


The first thing we should do is give our page a more meaningful filename and save it to our hard drive:

1. Select **File | Save**.
2. Leave the default location as `C:\Users\<USERNAME>\Documents\My Web Sites\hello world`.
3. Change the **File name** to **default**.
4. Save as type **ASPX Files**.
5. Click **Change title...** and type **Hello World** and press **OK**.



This has saved our empty page to our hard drive. You will notice that a couple of things happened when we did this. Firstly, the name of our file changed to `default.aspx`, as we would expect, but it has also added a new **Web Site** tab to our tabs in the workspace.



Adding and Formatting Text

You will notice in the white workspace there is a tag that says **form#form1**. This means that there is a form on the page with an ID of `form1`. All ASPX pages are required to have a `form` tag, so SharePoint Designer inserts one for us.

Now, let's go ahead and add some text to our page. Click inside the white workspace and type the words **Hello World**. For the purpose of this exercise, it does not matter if we type within the **form#form1** area or in the blank space beneath it.

Once we have done that, we hit the *Enter* key and add some more text beneath our **Hello World** text.

We can make our **Hello World** text into a heading by doing the following:

1. Highlight the **Hello World** text on the page.
2. Click on the **Style** drop-down on the Common button bar and select the **Heading 1 <h1>** option.
3. Remove the extra carriage return that SharePoint Designer added in front of your second sentence.

You will notice that so far this has not been any more difficult than using Microsoft Word to add content.



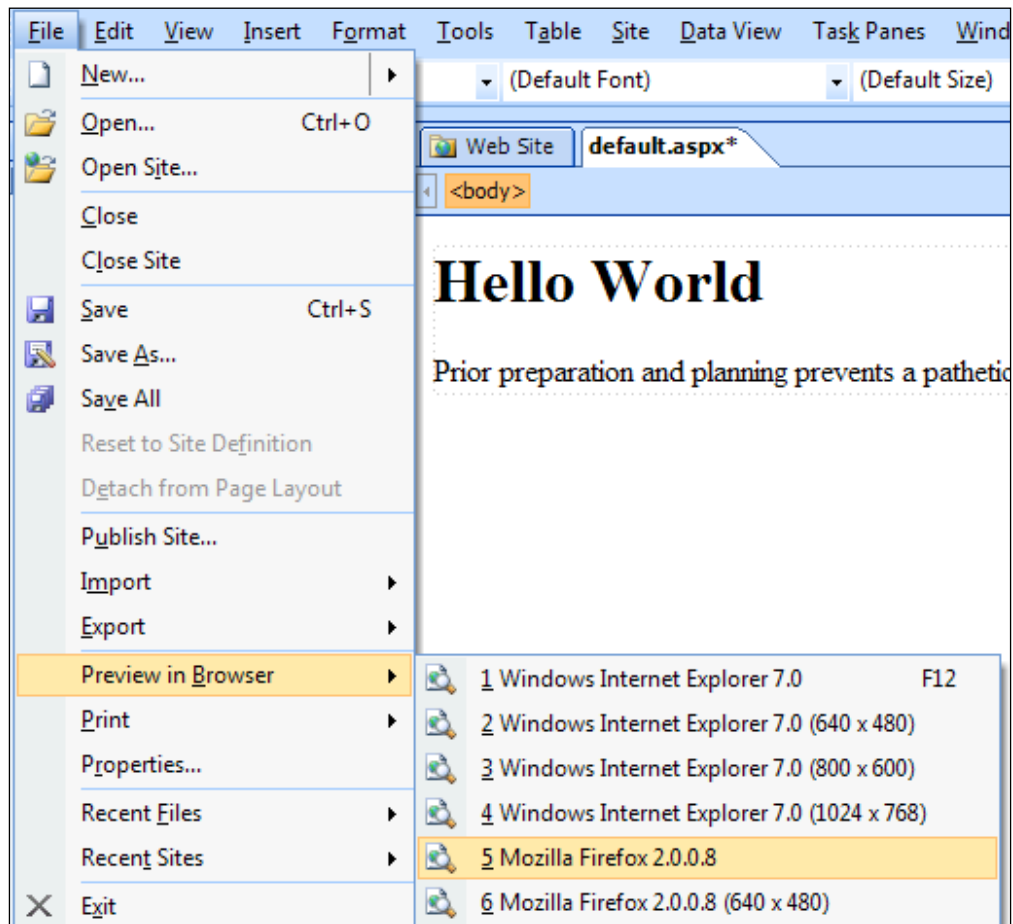
Once we have finished making these changes, we must remember to save our page.

There is plenty more that we can do to format our page nicely, but we will save that for the next chapter.

Previewing Our Page

SharePoint allows us to quickly check what our page looks like inside the browsers that are installed on our computer.

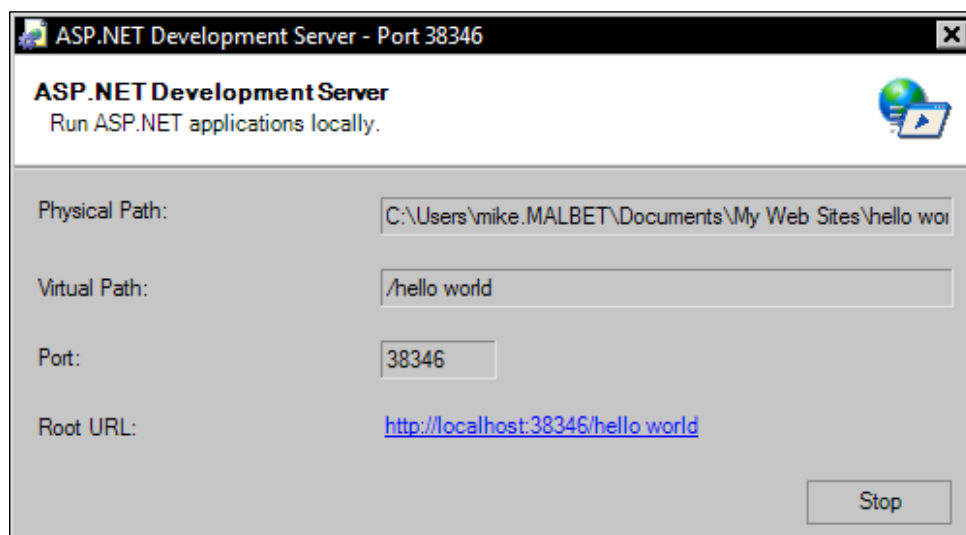
To preview our page, we go to **File | Preview in Browser** and select the browser we would like to use. Once we have previewed our page, for the first time, use anything but the first item in the list; don't be fooled into thinking that our browser is no longer on the list. It has simply moved up to first place on the list.



To quickly preview our page at any time, we can hit *F12*.

You may have noticed that when we preview our pages, a small icon is added to our system tray. This is because our computer requires a web server to display our page in the browser. Rather than forcing us to install and configure Internet Information Server (IIS) and the latest .NET framework on our computer, it quickly starts the ASP.NET Development Server, which serves our page for us. This server will continue to run on our PC until we either double-click on the icon and click **Stop**, or close SharePoint Designer.

Don't worry that the port number you see is different from the one in the graphic below, because a different number is selected each time the Development Server is started.



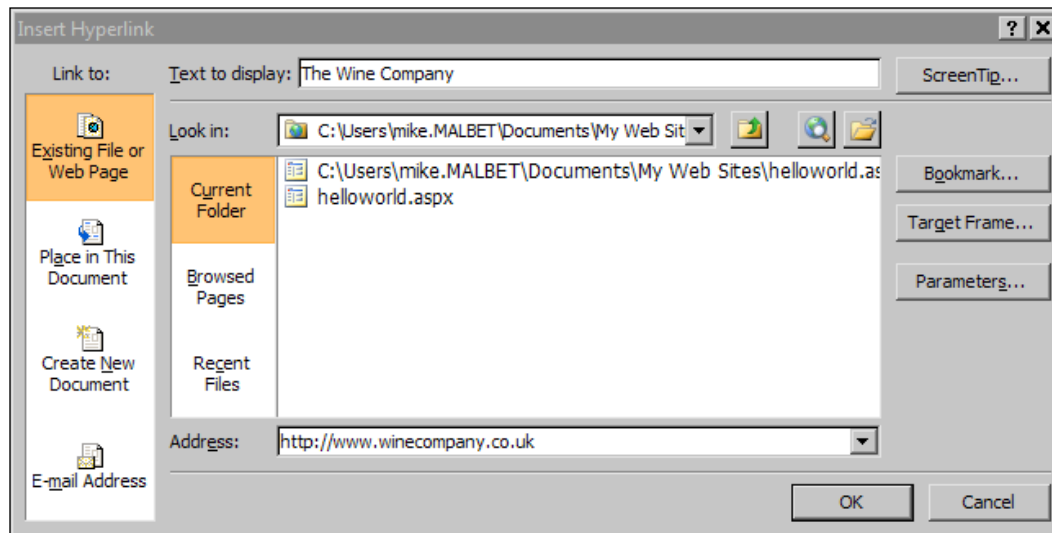
Creating Hyperlinks

Now that we have seen what our page looks like in the browser, let's return to the designer and add a hyperlink.

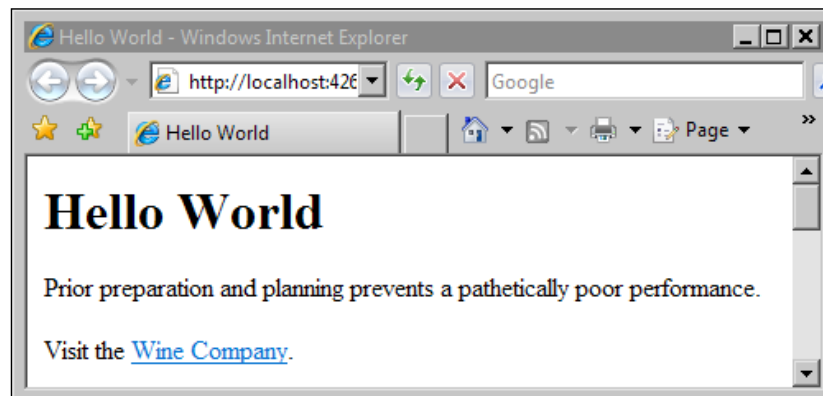
We can make text link to other pages on our site or on the Internet, allowing users to click through to those pages:

1. Type "**Visit the Wine Company.**" onto our page.
2. Highlight the text you would like to use as the linking text (i.e. "Wine Company").
3. Go to **Insert | Hyperlink**.

4. Either browse to a page we would like to link to, if we are linking to another page on our site, or type an address into the **Address** drop-down list (e.g. <http://www.winecompany.co.uk>), if we are linking to an external site.



Notice that we can also use the **Insert Hyperlink** dialog to insert a link to an email address (so that it creates a new message addressed to our chosen recipient in the user's email client). We can also specify the target frame or window that the new web page will open in. In addition, there is a **Parameters** button that allows us to pass information from one page to the other, which is useful when creating dynamic applications.



When choosing suitable text to use for our link, we should bear in mind that it is more useful to users if we use descriptive text for our links. For example, using the text "contact us" or "red wine" is more useful than lots of links dotted around our page saying "click here" or "here".

Adding Images

We will take this opportunity to add the Wine Company logo to our test page. The most common type of image used on web pages for full color images is a JPEG and our logo is no exception.

You can find the logo in the code bundle that you get along with this book. For further details, refer to the *Downloading the Example Code for the Book* section in the Preface.

Before we can add the logo to our page, we must first copy the logo into our website folder, rather than just inserting the logo from wherever it happens to be on our computer. The reason for this is that when we publish our site to the SharePoint server, we also want to make sure that all the files that are used in the site, including images, are published with it. If we were to publish a page that used an image on our C drive, it may display fine to us (because the browser would be able to find the image on our hard disk), but the image would not appear to users of other computers because they would not have the image on their computers.

Go ahead and copy the logo into the `hello world` folder that was automatically created within the `My Web Sites` folder, which is in the `My Documents` (or just `Documents`, if you are using Windows Vista) folder on your computer. You should already see the `default.aspx` file we created earlier in that folder. Because I am an organized soul, I would usually make a folder called `images` within the `My Web Sites` folder and put the image there, but because we are just creating a small test site, let's not worry about that.



It is also possible to import images directly into our site's **Folder List** using SharePoint Designer by selecting **File | Import** from the menu.

Now that we have copied the image into the folder our site uses, let's return to SharePoint Designer and insert the logo into our page by clicking on the spot on the page where we would like our logo and selecting **Insert | Picture | From File...** This allows us to browse to the `hello world` folder, select the image, and insert it to our page. It is also possible to drag and drop the image from our **Folder List** onto the design surface.

By default, we will see an **Accessibility Properties** dialog before the image is added. This allows us to specify **Alternate text**, which will be used by speech browsers used by the blind. Something meaningful, such as "Wine Company logo", should be added here, rather than an unhelpful generic term like "picture". It should also be noted that search engines also use this ALT text in their algorithms, so it is in our best interest to use something meaningful wherever possible (e.g. the name of the product we are selling). The **Long description** is not as necessary.

Creating Tables

Web designers love using tables (often perhaps when they should not, but more about this later in the chapter). It is not uncommon for a single web page to have three or more tables nested inside one another. We will add a table to our test page that will display a list of the company's five bestselling wines, with the help of the following steps:

1. **Go to Table | Insert Table.**
2. Change the **Size** options to **7** for **Rows** and **3** for **Columns**.
3. Set **Alignment** to **Center**.
4. Under **Specify width**, type **450** instead of the default **100**.
5. Click the **In pixels** radio button.
6. Set the **Cell padding** to **5** and the **Cell spacing** to **0**.
7. Click **OK**.

We can now click inside the table, which was created, and type our data like so:

Wine	Variety	Price
Chateau de l'Hospital	Sauvignon Blanc	\$16.99
Chateau le Gay	Cabernet Sauvignon	\$94.99
Goats do Roam Goat Roti	Red	\$14.99
Mad Housewife	Chardonnay	\$10.68
Thirsty Lizard	White Shiraz	\$6.49
Vivacious Vicky	White	\$8.99

Believe it or not, these are the names of real wines (not made-up names such as *Nasti Spumante*).

As we type the data into the cells, notice how the width of the columns automatically resizes to give the text the best fit.

Once we have created our table, we can edit the table properties at any time by right-clicking on the table and selecting **Table Properties...** from the shortcut menu.

Table and Cell Properties

Let's take a more detailed look at the table options:

- **Size** allows us to specify how many rows and columns our table has. A word of caution – if our columns are going to have anything longer than a few words in them, then we will probably want to keep the number of columns to five or less, so that our page does not become too wide.
- **Layout** allows us to control the size and location of our table as well as the size of the gaps in and around the cells:
 - **Alignment** allows our table to be positioned on the left, in the center, or on the right of our page.
 - **Float** can be set so that our table floats to the left or to the right of the page. "So how is **Float** different from **Alignment**?" I hear you ask. Good question. The difference is that when **Float** is enabled, the content on the page outside of the table will wrap around the table nicely.
 - **Cell padding** specifies the size of the margin in pixels, between the content of the cell and the edge of the cell.
 - **Cell spacing** is similar to **Cell padding**, but is the size of the margin between different cells. The choice whether to use **Cell padding** or **Cell spacing** to provide space between the contents of our cells only really needs to be made if our table has a visible border or background color.
 - **Specify width** allows us to enter either a set width for our table (if we click **In pixels**) or a percentage of the page or cell width (if we select **In percent**). A table does not need to have a specified size, but it is a good idea to provide one so that we can be sure about how wide our table will be. If we do not specify a size, the table will become whatever width other content on the page thinks it should be.
 - **Specify height**, on the other hand, is usually left blank, allowing the table to continue down the page for as long as it requires. It should be noted that when specifying height or width, using fixed sizes rather than percentages is preferred by the standards bodies (see <http://www.w3.org/TR/REC-CSS2/tables.html> for more information about table standards).

- **Borders** allows us to specify the width and color of any borders that we might use in our table:
 - **Size** allows us to specify how many pixels wide the outer border of our table is. If we do not want to have a border, then we simply select **0**. If we specify a border of greater than zero, then a border of one pixel wide will also appear between each of the cells in our table.
 - **Color** gives us the opportunity to not only make the border of our table a nice color to match the rest of our page, but also remove the basic 3D effect that some browsers render wider borders with.
 - **Collapse table border** changes the table from using W3C's Separated Borders Model to using their Collapsing Border Model. In the Collapsing model, it is possible to specify borders that surround individual cells, rows, and columns (and indeed row groups and column groups). This model is rather complex and would take a good number of pages to describe, so I won't go into it any deeper here (see <http://www.w3.org/TR/REC-CSS2/tables.html#borders> for further details of these models).
- **Background** allows us to edit the background details of our site with the help of the following options:
 - **Color** allows us to set the background color for the entire table. Clicking on **More colors...** provides us with a web-safe color picker. Web-safe colors are those that will display on any monitor. Because of advancements in screen technology, colors ceased needing to be web safe to display on our computers a good while ago (our monitors now tend to be far better than our eyes!). Interestingly, though, now that mobile devices have become more common, some thought again needs to be given to choosing colors that are web safe.
 - **Use background picture** allows an image on our web server to be used as the background pattern for our table. As with other graphical techniques, this should be used with care, otherwise it can look garish and can make the content of the table more difficult to read. When used sparingly, specifying a background image allows some very nice techniques. For instance, we can use a subtle faded flag image for each country behind the name of the wine, when editing the background images of individual cells.

- **Set as default for new tables** does exactly what it says on the tin. If we would like to create a lot of similar tables, then checking this box will automatically create future tables that are the same as the one we have just created.

In addition to being able to specify properties for the whole table, we can also specify properties for individual cells. To do this, we right-click in the cell we would like to edit and select **Cell Properties...** from the shortcut menu.

Many of the options for the cells are the same as those that we saw for the whole table. The additional options are:

- **Horizontal alignment** allows us to position our content on the left, center, or right of the cell. It also gives us the opportunity to justify our text, which can look smart, if the columns are fairly wide. If our columns are not wide, then selecting the **Justify** option can make the text look silly, because with only a few words per line, there will be large gaps between words.
- **Vertical alignment** is useful when we have very little content in one column but a lot of content in the column beside it (causing the rows to be fairly tall). By default, the user's browser will display the content of these cells in the middle of the cell (i.e. neither at the top nor at the bottom). My own preference in these situations is that the content is displayed at the top of the cell, so I would set the **Vertical alignment** to **Top**. It is also possible to align the content to the bottom of the cell or to the baseline by setting the **Vertical alignment** to **Bottom** or **Baseline**. The **Baseline** command has the same effect as the **Bottom** one, if the same sized fonts are being used, because it lines up the text using an imaginary line that the text rests on. It becomes useful when the fonts in the neighboring cells are of different sizes.
- **Rows spanned** allows us to merge our cell with the cell (or cells, if we select a number greater than two) above. When using this feature, you will notice that other content shifts to the right of the table, poking out the end in an extra column. This is because we still have the same number of cells in our table, but one of them now spans two rows. Once we have used the **Rows spanned** feature, we should usually delete the additional cells that have been moved to the right of our table. We can remove row spanning by returning the number of **Rows spanned** to **1**. We can make life easy for ourselves by selecting the cells that we would like to merge in **Design** view and selecting **Modify | Merge cells** from the context menu.
- **Columns spanned** operates the same as **Rows Spanned**, but merges cells horizontally rather than vertically.

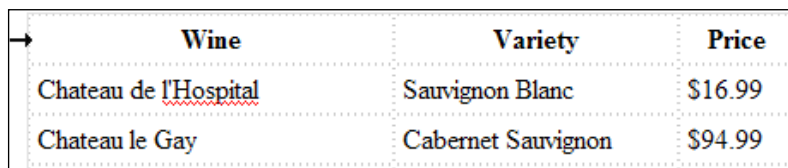
- **Header cell** can be checked to give our cells the special **th** designation. This means that our cells will be recognized as header information and will be rendered appropriately. This feature is usually applied to the first row of a table, as in the example we will see in a few moments.
- **No wrap** allows us to override the default, which automatically wraps the text in our cell. When using the **No wrap** feature, it is recommended that we add line breaks ourselves, so that the column does not become ridiculously wide.

We are also able to specify the width or the height of a cell in the **Cell Properties** dialog. This avoids the cell automatically resizing, depending on the content that is added to it. Notice that when we do this, it will set the width of the whole column and the height of the whole row, rather than just the width and height of our cell. The width of cells in a column and the height of cells in a row cannot differ.

If we would like to change the properties of all the cells in a row in one go, we hover our cursor over the left-hand border of the row we would like to edit until the cursor changes to a right-hand arrow and then left-click. This will highlight the entire row. Now, when we right-click and select **Cell Properties...** from the shortcut menu, any changes we specify will apply to the entire row.

Let's use this technique to make the headings in the first row of our table stand out. We are going to create header cells:

1. Hover the cursor over the left-hand border of the first row in our table.
2. Left-click so that the row is highlighted.
3. Right-click on the highlighted area and select **Cell Properties...** from the shortcut menu.
4. Click in the **Header cell** checkbox so that it becomes checked.
5. Click **OK**.



Wine	Variety	Price
Chateau de l'Hospital	Sauvignon Blanc	\$16.99
Chateau le Gay	Cabernet Sauvignon	\$94.99

You will notice that when we did this, the contents of the first row became bold and were centered. This is the default style for header cells.

If we wished to edit all the cells in a column, we would use the same procedure, but instead of clicking to the left of the row, we would click on the top of the column we would like to change.

We can add or delete rows or columns at any time by right-clicking on a cell in the table and selecting either **Insert** or **Delete** from the shortcut menu.

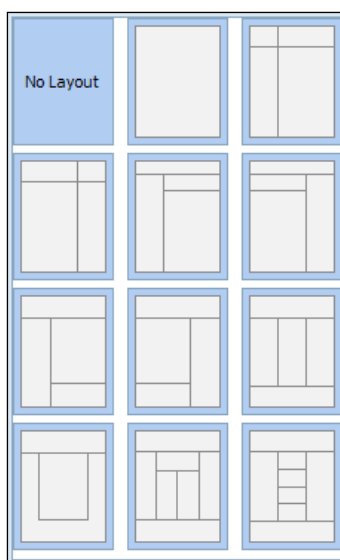
Layout Tables

Tables are not only useful for displaying tabular data. They are also used on the majority of websites to give the page some structure. By using invisible tables content, creators (that's us) can divide their page up into distinct areas.

One of the most common ways in which this is done is to have a main cell for the central content of the page with other smaller cells above and/or beside it for navigation buttons.

Usually, developers would create these tables themselves, but SharePoint Designer has made this easier for us by providing us with a selection of layout tables to choose from.

To view the layout tables that are available to us, go to **Task Panes | Layout Tables**. This adds the **Layout Tables** task pane to our designer. The most useful feature of this task pane is at the bottom, where it lists pre-defined table layouts, which we can use to structure our page.



Let's create a new page by going to **File | New | ASPX**, so we can experiment with these layout tables. Clicking on the image of the layout table immediately adds that table layout to our new page. To remove the layout table, click on the **No Layout** image.

Of the layout tables, the one I find to be the most useful is the one they call **Corner, Header, Left, and Body**, which is the third layout table shown in the task pane. This is a traditional page layout that allows us to have a logo in the top-left of the page, a banner, or navigation at the top of the page, and more navigation down the left of the page. If we also wanted to have some advertising space on our page, **Header, Left, Top Right, and Body** might be a useful one to select.

Besides using **Layout Tables** to structure our page, we can also add layout cells to our page by clicking **Insert Layout Cell....** This adds a table of one column and one row (i.e. just one cell) to our page. We may wonder why someone would want such a small table. The idea is that it gives us greater control over the positioning and formatting of the contents of that cell.

One final point to note is that if you have a page size set (e.g. 760x420), then you may see a small dotted line running from left to right some way down the page. This is "The Fold", which is an imaginary line showing the bottom of the page in the user's browser. To see anything below this line, the users will need to scroll down the page. Generally speaking, we would like to have the most important information on our page above this line, so the user can see it at a glance.

Divs versus Tables

It is worth pointing out that there are many markup purists who consider it an unforgivable sin to use tables for anything other than displaying tabular data. They argue that using tables to structure our pages makes them less accessible, and that we should use style elements such as `div` and `span` instead. In particular, the purists would see the use of layout cells as a shocking crime against web standards.

I won't comment on where I sit in this debate, but would like to warn that if you decide to use only styles to lay out your page, there will be times when you find it difficult to build and maintain more complex pages.

Whatever your preference, you will find that SharePoint Designer is one of the most standard-compliant development tools on the market. It does not force us to use **Layout Tables** instead of **CSS Layouts** (note that the **Layout Tables** task pane is not displayed by default, while the **Manage Styles** task pane is). SharePoint Designer also makes **CSS Layouts** (which is notoriously difficult to do in a way that works in all browsers) easier by providing us with **Visual Aids** to make it easier to see the page elements. We can switch these aids on by selecting **View | Visual Aids** from the menu. We are also provided with a **CSS Layouts** option as soon as we go to create a new page in the **File | New** dialog.

Organizing Our Files

During the few years I was in the army, the mantra "prior preparation and planning prevents a pathetically poor performance" was bashed into my psyche. This is true in other more sedentary walks of life, but especially in setting up a SharePoint site. If you are involved in the administration of SharePoint sites as well as the design of the sites, you will no doubt have noticed that SharePoint allows us to organize a whole hierarchy of containers. We can have server farms, which have collections of web applications within them, which, in turn, have site collections within those and finally individual sites forming those site collections.

Fortunately, we are only dealing with one site at the moment. Even so, if you are a tidy soul like me, then you will want to organize your files neatly within your site. If you are not a naturally organized person, then this is a good chance to take your first steps in that direction. In addition to ensuring that we give our files meaningful names, we can do a lot to make our lives (and the lives of everyone else working on our site) easier by arranging our pages neatly using the **Web Site** tab, the **Folder List** task pane, and the **Navigation** task pane.

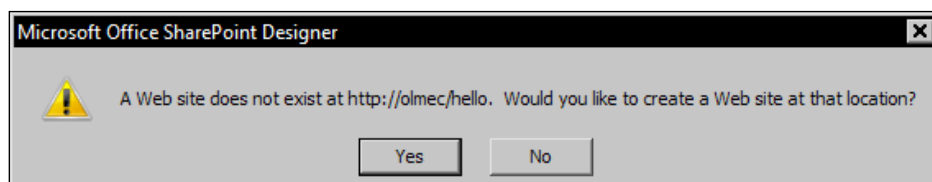
Publishing Our Site

So far, we have been working on our site "locally" (i.e. on our own computer). Let's go ahead and publish our site, so that other users on our network can view it.

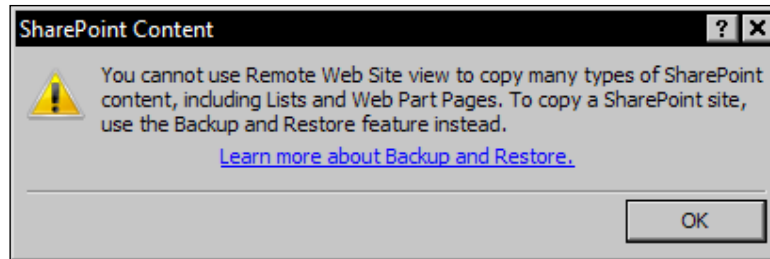
1. Go to **File | Publish Site**.
2. Select the **FrontPage Server Extensions or SharePoint Services** option.
3. Type the address of the site you would like to create in the **Remote Web site location** field. I am going to use `http://olmec/hello`. Note that I chose an address without a space in it.
4. Click **OK**.

We will see the following two dialogs once we have done this, because we are creating a new site.

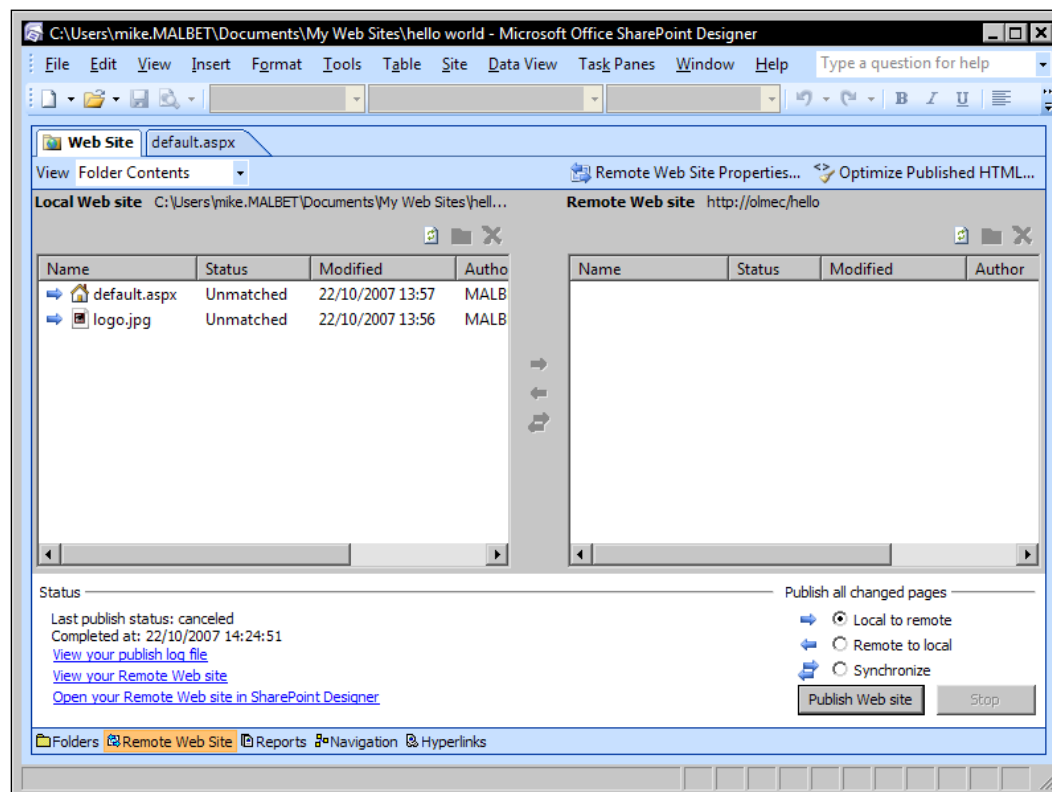
The first dialog asks us whether we would like to create a website at our selected location, if it has not been done so.



The next dialog warns us that publishing our site in this manner (as opposed to working on the remote files directly) has limitations in the functionality that we can use. We have not added any SharePoint content yet, so can simply click the **OK** button.



Once we are past the warning dialog, our **Web Site** pane opens in the **Remote Web Site** view, which was previously empty. If you have used a File Transfer Protocol (FTP) program to upload files to a website before this, screen will look most familiar:



Notice that the icon for our `default.aspx` file is a cute little house. This is because the name "default.aspx" is the one that automatically displays as our homepage when someone visits our website.

To publish our site, we merely need to click the **Publish Web site** button. Our files are then copied onto the SharePoint server.

Viewing Our Page

Users will now be able to see our creation live on the network by typing the URL of our page into their browser's address bar.

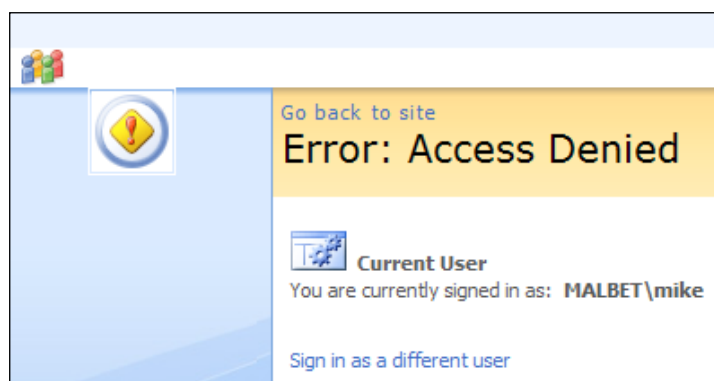


The address of our site is composed of the following four elements:

- **Protocol name.** The site will usually use the hypertext transfer protocol (HTTP) and so will be `http://`, but at times it will be its encrypted relative, which is `https://`.
- **Server address** will vary depending on where the user is viewing the site from:
 - **Server Name.** I will be using the example address `http://olmec/` for the rest of this book, because Olmec is the name of the SharePoint server we have in our server farm. If we are browsing the site using a browser on the SharePoint server itself, then we can also use `http://localhost/` to view our site.
 - **Domain Name.** If your network administrator has set up your SharePoint Site, IIS (Internet Information Server), and any firewalls correctly, then the site can be viewed by users outside your network. The domain name that the Wine Company uses is `www.winecompany.co.uk` (take a look and you will see the example site that we will be building in this book). Notice that the network administrator usually sets up this domain name to point to the site without the need for the user to type in any other items, such as folder or page names, in the address. Often subdomains will be used to point to different SharePoint sites (e.g. `forum.winecompany.co.uk` or `survey.winecompany.co.uk`).
- **Site name** e.g. `hello`. The default name of the site that SharePoint creates on your behalf is `Pages`.
- **Page name** e.g. `pagename.aspx`. If we would like our page to be the first to display in a site, then the name `default.aspx` should be used.

Authorization

In order for users to be able to view our new site in their browser, they will need to have the correct permissions applied within SharePoint. If this has not been done, then they will see a message like the following one displayed in their browser:



Editing Existing Sites

It is possible to edit existing sites on our SharePoint server (although, of course, I would urge you to be careful, if you are unsure about what you are doing to those pages). To open a site, follow this procedure:

1. Close any sites you have opened in your designer using **File | Close Site**.
2. **File | Open Site**.
3. Click on the site you would like to open (e.g. <http://olmec/mysite>).
4. Click **Open**.
5. Enter your username and password into the login dialog.

If we are working with multiple sites, we will also find **File | Recent Sites** useful, because it gives us fast access to open sites that we have been working on recently.

Summary

In this chapter, we learned how to create a new site and to add pages to that site. We then moved on to add text and graphics to our page and learned how to preview the site in our web browser.

We also looked at how to create tables and learned about the benefits of layout tables and were made aware that there is much debate about using div, wherever possible, in order to make our pages as accessible as possible.

Finally, we looked at organizing our files and publishing our site.

We are now equipped to create new pages, to add basic content to those pages, and to publish our pages to the SharePoint server.

All good stuff, but I am sure you will agree that our page lacks visual oomph. Let's carry on and learn about the methods we can use to create a consistent style for our site. In the following chapter, we will learn to format our pages using Cascading Style Sheets. We will also learn about Master Pages and how to edit them to give our site a consistent appearance.

4

Formatting Pages

Now that we have added the most basic content to our site, we will take a more in-depth look at using features, such as Cascading Style Sheets and Master Pages, to format our pages in a consistent and scalable fashion.

The Wine Company Website

Before creating any site, it is a good idea to thoroughly consider which pages the site will contain. Tony and Gavin at the Wine Company have decided that their new SharePoint site will have the following navigation buttons on each of the pages of the site:

- Red Wine
- White Wine
- Contact
- Search

They have deliberately kept the number of links as low as possible, so that visitors to the site are steered towards the most important pages (i.e. the ones selling wine). Tony also suggested to Gavin that they design the menu in a way that allows more buttons to be quickly added if required (he knows from experience that Gavin is liable to ask for new features to be added to their site, often at very short notice).

The Wine Company site will also have a couple of links that will allow users to access password-protected information within the site:

- Register
- Log-In

In addition to this, the site will have a search box that will be visible on every page (thereby providing customers with yet another avenue to find their favorite tipples).

Creating Our Site

Creating their site will be a useful way of recapping on the skills we learned in the previous chapter.

We will create the foundation for the site in three stages:

1. Create a new site on our computer.
2. Add a homepage to this site.
3. Publish the site to our SharePoint server.

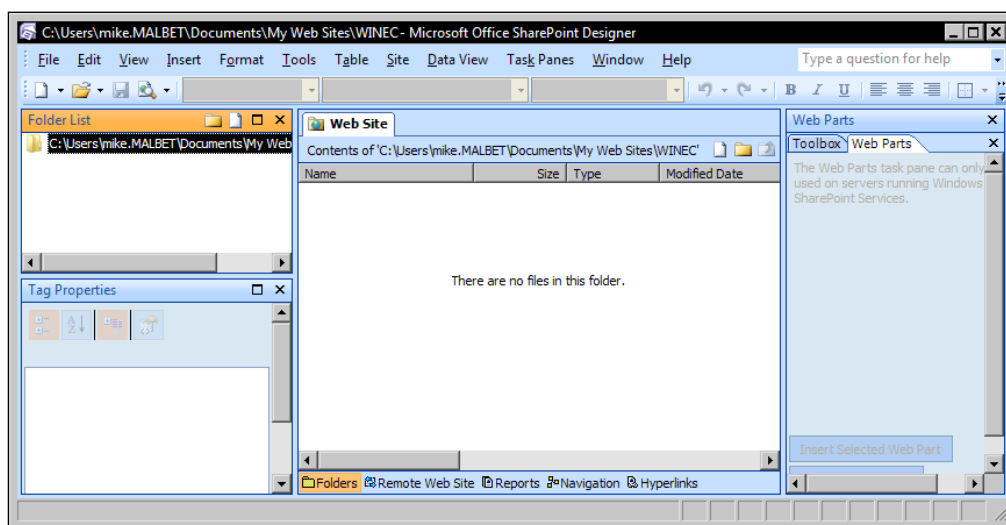
Creating a New Site

The first thing that we are going to do is create a new SharePoint site, which we will call "WINEC". We will be using this site for the rest of the exercises in this book.

Start by creating the site on your computer:

1. Click **File | New | Web Site....**
2. Select **Empty Web Site** from the list of possible pages.
3. Create a new folder within the **My Web Sites** folder for your site and name the new folder **WINEC** (e.g. `C:\Users\<USERNAME>\Documents\My Web Sites\WINEC`).
4. Click **OK**.

SharePoint Designer will then display the Folder view for our new empty site on the screen.



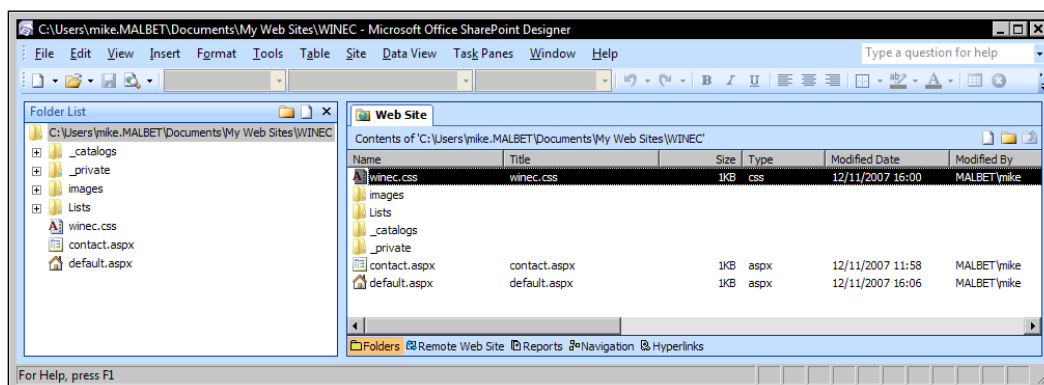
Creating Our Homepage

Let's add a new blank page to our site. Add the new page by selecting **File | New | ASPX**.

The first thing we will do with this page is save it as `default.aspx`, so that it will be used automatically as the homepage on our site.

1. Select **File | Save**.
2. Leave the default location as `C:\Users\<USERNAME>\Documents\My Web Sites\WINEC`.
3. Change the **File name** to **default**.
4. Save as type **ASPX Files**.
5. Click **Change title...** and type **Wine Company** and press **OK**.
6. Click **Save**.

When we click on the **Web Site** tab in the designer, we will see that our new page has been added to the website and that it has been given a cute little house icon to signify that this is our homepage.



Clicking on **default.aspx** returns us to the **Design** view of our homepage. Type the following text inside the **form#form1** container on the page:

Welcome to the Wine Company
The Wine Company is a fictitious company which supplies wines from all over the world to our UK customers.

We will save the page with **File | Save**.

Publishing Our Site

Now that we have created our homepage, we can publish our site on the SharePoint server.

1. Go to **File | Publish Site....**
2. Select the **FrontPage Server Extensions or SharePoint Services** option.
3. Type the address of our new site in the **Remote Web site location:** field as **http://olmec/winec**.
4. Click **OK**.
5. When asked if you would like to create this new website, click **Yes**.
6. Click **OK** on the next information dialog.
7. This time we want to download the default files and folders, which SharePoint server created in our site for us, as well as uploading our site. To do this, click on the **Synchronize** radio button in the bottom-right of the screen.
8. Click **Publish Web site**.
9. You will almost certainly see a further information dialog such as one that tells you that certain components require a server running WSS to function (e.g. `_catalogs/masterpage/default.master`). Click **Ignore and Continue**.

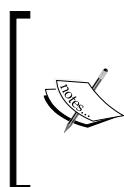
We will now be able to browse to our site from anywhere on the network by typing the address of our site (e.g. `http://olmec/winec/`).

Formatting Our Homepage

Now that we have our page ready, we can jazz it up a little by applying some formatting. We can access the formatting features in two ways.

Most obviously, we can use the buttons on the **Common** button bar, which is displayed at the top of the designer.

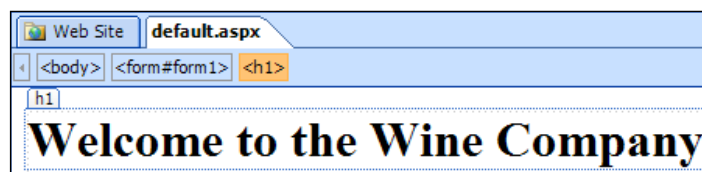




There is also a **Formatting** button bar, which can be used instead of the **Common** button bar, but there is little point in using it because it only includes three commands that are not available on the **Common** button bar (**Manage Styles**, **Increase Font Size**, and **Decrease Font Size**). These three commands can be added to the Common command bar by clicking the **Toolbar Options** button and choosing **Add or Remove Buttons**.

Let's turn our first line of text into a heading by highlighting it and selecting **Heading 1 <h1>** from the **Styles** drop-down (the one that currently says **(None)**) on the **Common** button bar.

Notice that as the text changes to become a large heading, a fine blue dotted border appears around the text with a small **h1** tab on it. This blue line denotes the area that the style applies to and will not display on the web page at run time. If you wish, then visual aids such as this can be turned off by going to **View | Visual Aids**.

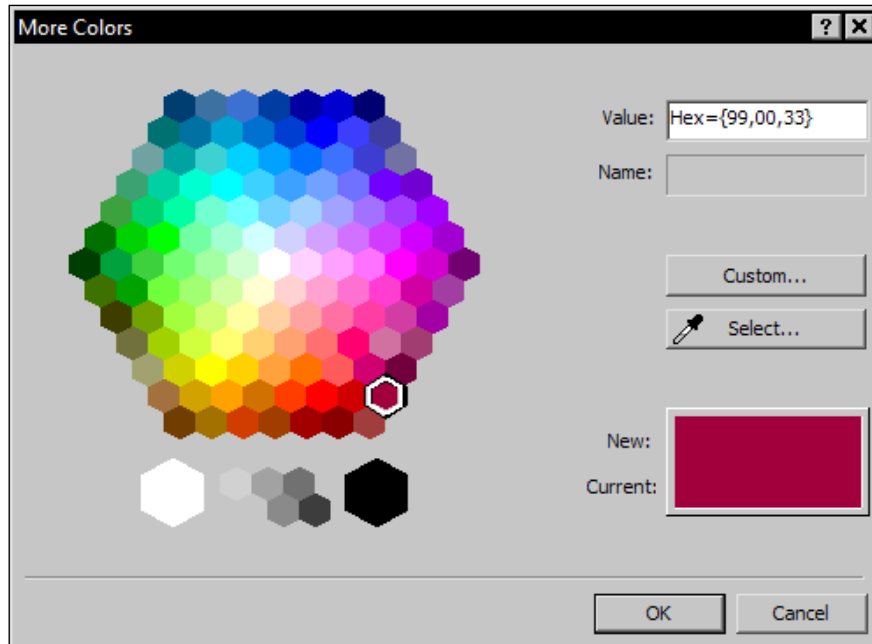


Also notice that our second sentence was pushed further down the page. This is because by default an **h1** tag adds an extra line break after it. Personally, I dislike this behavior and remove these line breaks from the **h1** style as soon as I possibly can (we will see how to do this shortly).

The Wine Company uses a deep red color in its marketing materials. This color has red, green, blue (RGB) values of 153, 0, and 51 respectively. Although web browsers can understand decimal RGB values, it is more common for them to use hexadecimal values. In hexadecimal, this color becomes **#990033**.

We can color the second sentence on our page this deep red color by highlighting the sentence and clicking the small black drop-down arrow to the right of the **Font Color** icon on the **Common** button bar. This gives us a short menu of standard (fairly garish) colors we can use. Clicking on **More Colors...** gives us a nicer selection of colors. Let's click on the Wine Company's standard burgundy color, which is near the bottom-right of the color picker.

We know we have the correct color when the **Value:** field displays **Hex={99,00,33}**. Clicking **OK** applies this color to our text.



The **Format** menu provides us with more in-depth formatting tools. Of particular significance for the moment are the items in the second section of this menu. I will not insult your intelligence by going into minute detail about how to use this to make your text bold, etc., but there are a few items of interest that are different from the other programs in the Office suite:

- **Font** automatically defaults to Times New Roman, but the interesting thing here is that our text can be given more than one font. The reason for this is that for a font to display within the user's browser, it must be installed on their computer. If the font is not installed on their computer, then their browser will use their default font (which is usually Times New Roman). To help ensure that our page displays in a manner which is consistent, we can select the combination of fonts that we would like our page to try to use. As an example, our page can use Arial, if it is installed on the user's PC, otherwise it will use Helvetica, and if it also does not have that, it will use sans-serif. Because all three fonts look vaguely similar, our page should appear roughly in the way we intend it to, irrespective of who is viewing it. It is also possible to select **Blink** in this font dialog to make our text blink incessantly, but the good news is that it will only blink in browsers such as Firefox and will not blink in Internet Explorer 7.

- The **Paragraph** dialog is very straightforward, especially because it provides us with a helpful preview, at the bottom of the dialog, of what each of the features does when we select them. All that should be pointed out is that the items in the **Line spacing** drop-down need to be selected a few times before the preview updates to display what our paragraph will look like.
- **Bullets and Numbering** is equally simple to use. My personal favorite use of this is to specify a square 18 pixel graphic of an arrow, which I use as the bullet for many of my lists.
- **Borders and Shading** is the dream dialog for lovers of divs. This is to purists what **Table | Table Properties | Table...** is to content creators who prefer to use tables. This **Borders and Shading** dialog provides massive amounts of control, allowing the size, margins, border, and colors of the section of the page you are editing to be changed in minute detail.
- **Position** is grayed out on our screen at the moment because the text we have selected is not positionable. Do not fear though, we will return to this in a moment.
- **Behaviors** are JavaScript events that are triggered when a user interacts with an element on our page. They would not be the first thing we would think of when someone mentions formatting. It is included in this menu because behaviors can be applied to span tags (i.e. the HTML tag to which our style relates).
- **Layers**. Ahh yes, **Layers**. Well, they deserve further explanation.

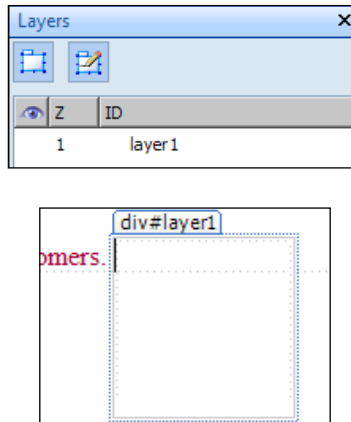
Using Layers

Everything seems to have layers. Onions have layers, people have layers, and even donkeys have layers (according to Donkey in "Shrek", Dreamworks 2001). Web pages are no exception. We can add layers to our page to build it up in three dimensions. Content can be placed to overlap other content on our page.

We will take a moment to add a new layer to our page, which we will use to display a special offer of a fabulous revolutionary (pardon the pun) new corkscrew that customers get for free with any case of wine. By using layers to do this, we can position the special offer exactly where we would like it to appear on our page. Not only that, but using layers also allows us to display or hide the content whenever we wish.

1. Select **Format | Layers**.
2. This will display the **Layers** task pane.
3. Click on the **Insert Layer** icon in the **Layers** task pane.

You will notice that two things happened when we did this. First of all, a new layer called **layer1** was added to our empty list of layers in the **Layers** task pane, and secondly, a new layer with the heading **div#layer1** was placed on our page where our cursor had been.



As you know already, I am an organized kind of chap. I don't have labels on the shelves in my wardrobe at home with the words "socks" and "pants" on them, but I am not far off. In this spirit of being organized, the first thing we are going to do to our new layer is give it a more meaningful name:

1. Double-click on the word **layer1** in our **Layers** task pane so that the text becomes highlighted.
2. Type **specialOffer** and press *Enter*.

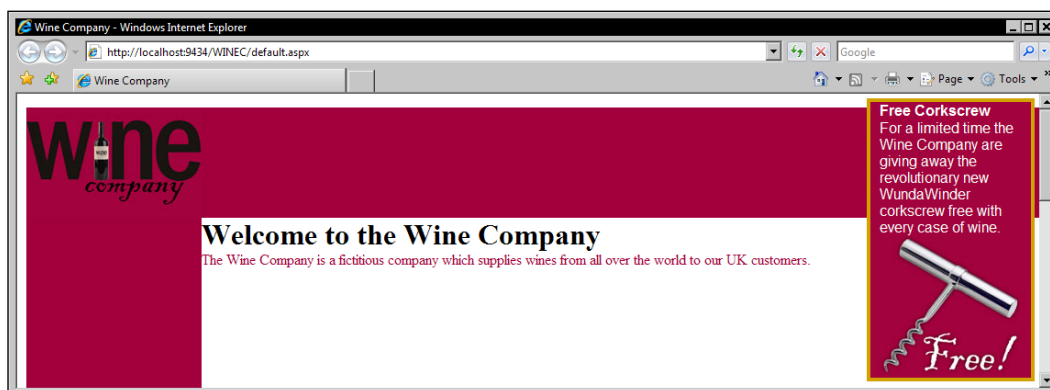
Now that our nicely named layer is on the page, let's change the properties to make it a decent size and position it ready to contain our special offer:

1. Select the **specialOffer** layer.
2. Go to **Format | Position...**, which will open up the **Position** dialog.
3. Set the size to a **Width** of **175px** and a **Height** of **300px**.
4. Set the location to be in from the **Right** by **5px** and down from the **Top** by **5px** (leave the **Positioning style** set to **Absolute** so that this position is measured from the edge of the page rather than being relative to where our cursor is on the page).
5. Click **OK**.

This will place our **specialOffer** layer snugly in the top right-hand corner of our web page. Go ahead and type the following two sentences into the layer:

Free Corkscrew

For a limited time the Wine Company are giving away the revolutionary new WundaWinder corkscrew free with every case of wine.



We would like to format the background and borders of the new layer to help it stand out from the page and catch the eye of visitors to the site:

1. Hover the cursor over the border of the layer (the multi-headed arrow cursor will appear) and left-click to select the entire layer.
2. Go to **Format | Borders and Shading....**
3. Notice that the **Setting** for the border is set to **Default**.
4. Select **solid** from the **Style** list (notice that this will change the **Setting** to **Box**).
5. Click on the **Color** drop-down list and select **More Colors....**
6. Select the gold color with the value **Hex={CC,99,00}** from near the bottom-left corner of the color picker.
7. Click **OK**.
8. Click the **Shading** tab.
9. Click on the **Background color:** drop-down list. You will notice that the deep red and the gold colors that we are using on our page have been made available for us in a new **Document Colors:** section. Click on our deep red.
10. Click **OK**.

We have now successfully colored our layer. If it looks purple instead of red in your **Design** view, this will be because you still have it highlighted. Go ahead and preview the page in your browser (**File | Preview in Browser**). Notice how our layer stays nicely in the right corner of our page, irrespective of the width we resize our browser window to. As we make our browser thinner, we see that the layer covers the other text on our page. This means that care will need to be taken when designing our page to ensure that we are not obscuring important page content with our **specialOffer** layer.

We can certainly make our special offer feature more appealing. Let's start by changing the text. We are going to make the font a sans-serif font and color it white. We will also add some padding around the text and make the first sentence bold:

1. Highlight all the text that is on our layer.
2. Select **Format | Font....**
3. Change the **Font** face to **Arial, Helvetica, sans-serif**.
4. Click on the **Color** drop-down and select **White** from the **Standard Colors**.
5. Click **OK**.
6. Select **Format | Paragraph....**
7. Change the **Indentation** of both the **Left side** and the **Right side** to **10px**.
8. Click **OK**.
9. Highlight the first sentence of our special offers text (i.e. **Free Corkscrew**).
10. Select **Format | Font** once again.
11. Change the **Font style** to **Bold**.
12. Click **OK**.

Adding an Image to Our Layer

We are going to jazz up our feature slightly more by adding a photo, which we created (`wundawinderBackground.jpg`), as a background on our layer:

1. Open the **Layers** task pane.
2. Right-click on **specialOffer**.
3. Select **Borders and Shading**.
4. Click the **Shading** tab.

5. Click on the **Browse...** button and browse your hard drive for the image we will use as the background picture.
6. Once you have located the correct image, click it once and then select **Open**.
7. Click **OK**.

We will see the image embedded into the background of our layer.

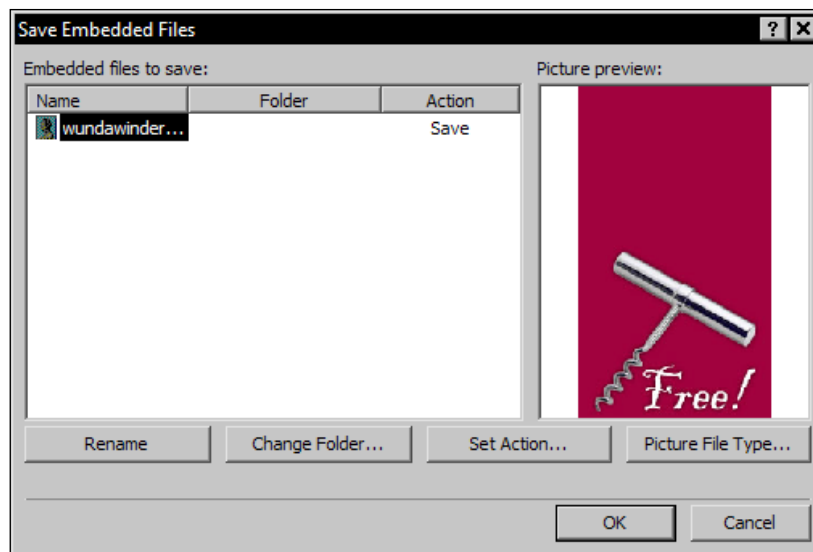


Publishing Images

When building SharePoint sites, it is vital that we understand which files are in which locations. There are three different types of location that our files can reside in:

- **Source Files.** This can be anywhere on our PC, network, or removable media such as CD, DVD, Memory stick, or digital camera. Currently the `wundawinderBackground.jpg` image only exists in the location that you copied the source files for this book to.
- **Local Web Site.** If you take a look in the `WINEC` folder inside `My Web Sites` on your computer, you will notice that the `images` folder is empty. This is the local folder that stores the images used by our website.
- **SharePoint Server.** When we publish our local website to the server, it will upload a copy of the images so they can be seen by visitors to our site.

SharePoint Designer will prompt us to ensure that our images are copied to the correct locations. Go ahead and save the page that we embedded our image in. When we do this, SharePoint Designer prompts us to save the source images to our local website so that they will be there to display at run time.



The reason that I like this dialog so much is that it does not just prompt us to upload the images that the site needs, but it also allows us to optimize and organize our images.

When we are creating large sites, we want to organize our files in a system of sub-folders. There is nothing worse than having to scroll through hundreds of web pages, images, and style sheets, which have all been saved in the root of the website.

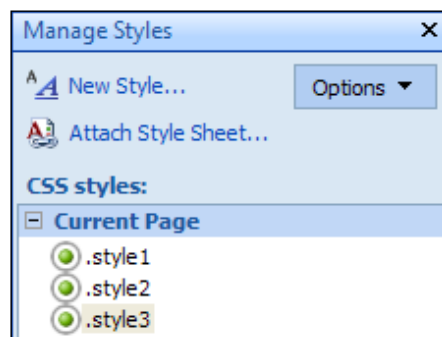
Specify the folder that we would like our image to be placed in by clicking the **Change Folder...** button and browsing into the `images` folder in the view of our website that we are presented with. Once in the `images` folder, click on the **Create New Folder** icon and name the new folder **specialOffers**. Click **OK** to finish this procedure.

We can also optimize our image by clicking on the **Picture File Type...** button. This is particularly useful if our source image has a larger than necessary file size, such as a PNG image or a high quality JPEG. In our case, the image that I created for the **specialOffer** should be pretty optimal so would not benefit from any tinkering.

Once SharePoint Designer has made a copy of our embedded image in our local website, it will be visible when we go to **File | Preview in Browser**. For the image to be visible in our published website, we will need to synchronise with our remote website.

Renaming Our Styles

If we peek into our **Manage Styles** task pane, we can see the styles we have created on our page:



When we click on one of our styles, we see a handy preview of what that style looks like displayed in the preview pane at the foot of the task pane.

As you will no doubt have guessed, I would like to tidy this up a little by giving our styles names that are more meaningful. Right-click on each style in turn and select the **Rename class** option. We will rename our styles to be called **redText**, **specialOfferBox**, and **specialOfferText**. When doing so, ensure that the **Rename class references in this page** check box remains checked to ensure that the class names in our code update as well as the names in our **Manage Styles** pane.

Cascading Style Sheets

Formatting our page directly by using the formatting tools is all very well, but in larger sites or sites that are maintained by more than one person, it can be difficult to maintain a consistent look and feel of all the pages in the site. This is where Cascading Style Sheets (CSS) comes in. It allows us to define the styles we will use in one central place and then to apply those styles to the elements on our pages.

Working with Styles in SharePoint Designer involves three separate task panes:

- CSS Properties
- Manage Styles
- Apply Styles

When formatting our page, the style information was stored within the head section of the HTML of the page. A far better way to store our styles is to have them in one central style sheet so that we can use our styles on any page of our site.

To centralize our styles, we create a style sheet where the styles will be specified. We will do this now by going to **File | New | CSS**.

The first thing we are going to do is save this empty style sheet to the root of our site:

1. Go **File | Save**.
2. Change the **File name** to **winec**.

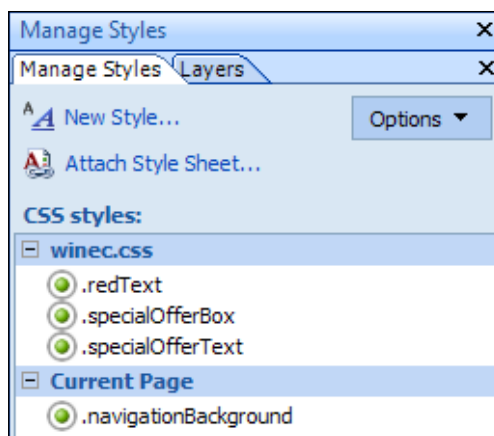
Now that we have our blank CSS, we can move into the **Code** view of **default.aspx** and cut the following style code from it:

```
<style type="text/css">
.redText {
    color: #990033;
}
.specialOfferBox {
    border-style: solid;
    border-color: #CC9900;
    font-family: Arial, Helvetica, sans-serif;
    color: #FFFFFF;
    background-color: #990033;
    background-image: url('images/specialOffers/wundawinderBackground.
jpg');
}
.specialOfferText {
    margin-left: 10px;
    margin-right: 10px;
}
</style>
```

Paste all of this code apart from the opening and closing style tags (`<style type="text/css">` and `</style>`) into **winec.css**.

If we now save **winec.css** and **default.aspx** and return to the **Design** view of **default.aspx**, we will see that **default.aspx** has lost its formatting (because we cut it from the code). No matter, we can reunite it with these lost styles by clicking **Attach Style Sheet** from the **Manage Styles** task pane and browsing for **winec.css**. When we click the **OK** button, all is well once again. Our styles are now being applied from the centralized style sheet. Notice how our styles are now listed in the **winec.css** category in the task pane rather than in the **Current Page** category, where they were previously.

We will need to use **Attach Style Sheet** to attach the style sheet to any of the pages that we would like that style sheet to apply to. We will shortly learn about **Master Pages**. For now, it is good to be aware that attaching the style sheet to a Master Page will ensure that our styles are available to all pages that use that Master Page.



Editing Styles

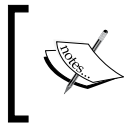
I promised earlier in the chapter that we would remove the automatic line break from the end of our h1 tags. In formatting speak, this is known as using an inline style.

To do this, we could simply add the following line of code to **winec.css**:

```
H1 {display: inline}
```

Doing so would not really teach you to use SharePoint Designer though, so let's do it the menu-driven way.

1. Click on **New Style** in the **Manage Styles** task pane.
2. Select **h1** in the **Selector** drop-down.
3. Change the **Define in** drop-down to **Existing style sheet** so that the style applies to all pages on our site that use the `winec.css` style sheet.
4. Ensure that the **URL** is set to **winec.css**.
5. Click on the **Layout** category.
6. Change **display** to **inline**.
7. Click **OK**.



When working with style sheets, it is helpful to know that when the name of the style begins with a dot, it is referring to a class; when there is no dot, it is referring to an HTML tag (as in the example above); and when it starts with a #, it is referring to an ID.

Now when we flick to **default.aspx**, notice that the extra line break at the end of our heading has been removed. Also notice that h1 has appeared in our list of styles listed for **winec.css** in our **Manage Styles** task pane and that the icon beside the style is blue (denoting it as a known style rather than a custom style we have created).

A word of warning while on the subject of styles: If you do need to copy text from Microsoft Word into SharePoint Designer, be aware that you will also be importing the styles that Word has used to mark up your document. SharePoint Designer makes it easy for us to remove these styles. Simply paste the content into the **Design** view and then select the **Remove Formatting** smart tag.

If you would like to learn more about writing cascading style sheets, then you may find that <http://www.w3.org/Style/CSS/> is a useful resource.

Master Pages

We could just bash ahead and build our first page using the formatting techniques that we have just learned. This is the way that sites were traditionally built. One page was created and once all involved were happy that it looked OK, it was copied many times to create the basis for all the other pages in the site.

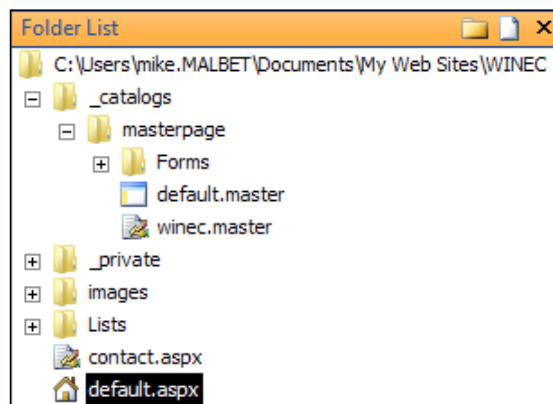
We could do it that way too, but let's wait a moment. What happens when six months down the line we would like to add a new navigation button to the menu that appears on every page? Do we need to change all the pages on our site individually? The clever developers (and I humbly place myself in that category) used Server Side Includes to create the page elements (e.g. the menu bar) in one place, which then propagated throughout their site. They could place some additional code into the include and "hey presto" the new button appeared on every page of their site.

By introducing Master Pages, Microsoft allows us to achieve the same results without the need to learn about Server Side Includes. Master Pages are an ASP.NET 2.0 feature that allows us to give all the pages in our site a consistent look and layout. They are templates that we build and attach to content pages to create the web pages within our site.

Master Pages use Content Placeholders to indicate regions of replaceable content. When we update the information on the Master Page, this automatically updates the content on our content pages.

Where Are Our Master Pages Stored?

If we take a look in our **Folder List** task pane, we will see the `_catalogs/masterpage/` folder tree, which was copied on our local version of the WINEC site when we first synchronized our site. This location is referred to as the "Master Page Gallery" and contains some default forms and a file called `default.master`, which is our Default Master Page.



Creating a Master Page

We will now create the Master Page that we will use throughout our site by doing the following:

1. Select **File | New | Page**.
2. Select **Master Page**.

It is as simple as that. A new Master Page has been created for us. We want to start from a totally clean slate so delete **ContentPlaceHolder1** from the page.

Editing Our Master Page

Now that we have our blank Master Page, we will give it some structure by using the layout tables provided for us. Open the **Layout Tables** task pane and click on the third layout (the one with the title **Corner, Header, Left, and Body**). This will add a blue layout grid to our new Master Page. This layout stretches the full width of the browser window with a 120 pixel tall row at the top of the page and a 191 pixel wide column down the left-hand side.

If we flip into the **Code** view, we can see that SharePoint Designer has automatically designated different CellTypes to the four cells in our layout:

```
<table border="0" cellpadding="0" cellspacing="0" style="width:
1104px; height: 882px">
  <!-- MSTableType="layout" -->
  <tr>
    <td valign="top">
      <!-- MSCellType="DecArea" -->
      &nbsp;</td>
    <td valign="top" style="height: 120px">
      <!-- MSCellType="ContentHead" -->
      &nbsp;</td>
  </tr>
  <tr>
    <td valign="top" style="width: 191px">
      <!-- MSCellType="NavBody" -->
      &nbsp;</td>
    <td valign="top" style="height: 762px; width: 913px">
      <!-- MSCellType="ContentBody" -->
      &nbsp;</td>
  </tr>
</table>
```

This is rather spooky because Microsoft's plan for our page seems to be exactly the same as ours. We will use each of the four cells on our page for the following purpose:

- DecArea in the top-left will be where we place our company logo.
- ContentHead stretching across the top of our page is where we may wish to place banner adverts in future.
- NavBody down the left side of the page will contain our site navigation buttons.
- ContentBody in the center of the page will contain the content that our page has.

The first thing that we are going to do is add the deep red color as a background to the two top cells and the left-hand column. This will allow us to see changes to our content page when we apply our Master Page to it. We will start with the top-left cell:

1. Right-click inside the top-left cell.
2. Select **Cell Properties...** from the shortcut menu.
3. Click on the **Background Color** drop-down and select **More Colors...**

4. Click on our **990033** color.
5. Click **OK**.
6. Click **Apply**.

Now repeat this process for the top-right cell and the left column.

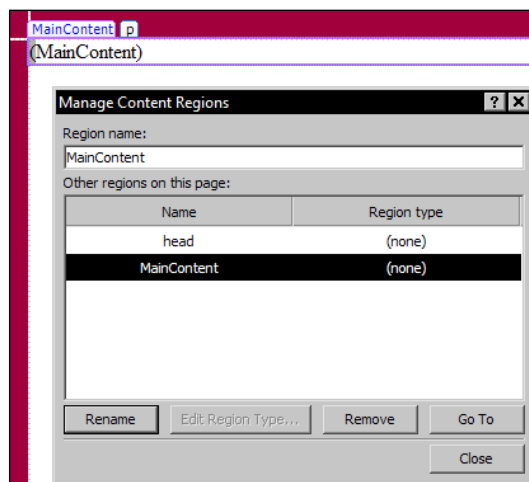
If we peek into the **Manage Styles** task pane, we will see that the red background has been created as a new style called **style1**. Not only is this untidy and unimaginative but it is likely to cause problems if there is also a style called `style1` in one of our content pages. Before we go any further, let's rename **style1** to **navigationBackground**.

Adding a Content Region

We must now specify the part of the Master Page where we would like our content to appear. We do this by adding a Content Region:

1. Right-click anywhere in the bottom-right cell (i.e. the large one with the white background) of our Master Page.
2. Select **Manage Microsoft ASP.NET Content Regions....**
3. The dialog that appears should only contain one Content Region called **head**.
4. Type the name **MainContent** into the **Region name** field.
5. Click **Add**.
6. Click **Close**.

This has added a new Content Placeholder called **MainContent** into the main cell of the layout table on our Master Page.



Saving Our Master Page

Now that we have a Content Region in our Master Page, we can go ahead and save the page. It is possible to save a Master Page that does not contain any Content Regions but there is little point because you will not be able to use it for anything.

1. Go to **File | Save**.
2. Browse into the `_catalogs` folder and then into the `masterpage` folder.
3. Give your file the name **winec**.
4. Click **Save**.

Attaching Our Master Page to an Existing Page

We can now go ahead and use our new Master Page as the template for our homepage layout. Microsoft refers to this as *attaching* the Master Page.

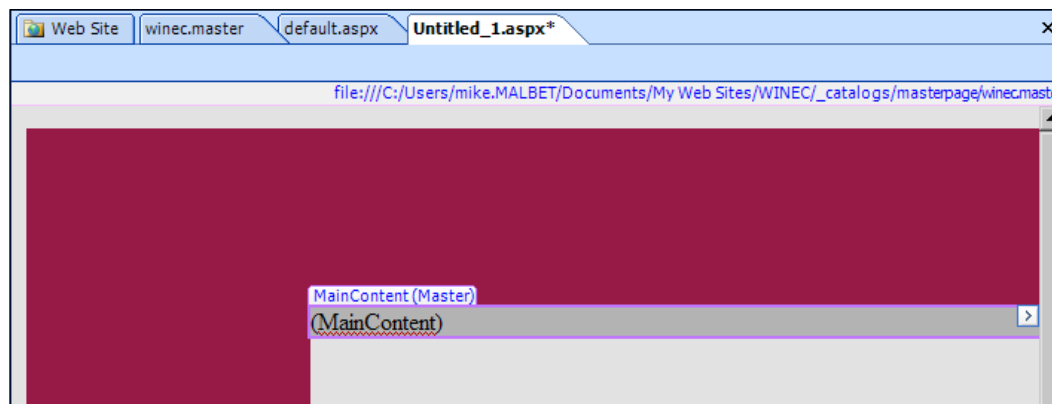
1. Open `default.aspx` in the designer.
2. Go to **Format | Master Page | Attach Master Page....**
3. Browse into `_catalogs` then `masterpage`.
4. Click on the **winec.master** file.
5. Click **Open**.
6. Click **OK**.
7. In the **Match Content Regions** dialog, click **OK** to use SharePoint Designer's suggestions for mapping the content regions between the two pages.

Creating a New Page Using a Master Page

Once we have established our Master Page, the easiest way to create new content pages is to create them directly from the Master Page.

1. Go **File | New | Create from Master Page....**
2. Click **Browse**.
3. Browse into `_catalogs` then `masterpage`.
4. Click on **winec.master**.
5. Click **Open**.
6. Click **OK**.

This will create a new page based on our Master Page called **Untitled_1.aspx**.



Notice that the title of our MainContent Placeholder has the word **Master** in parenthesis after it. This signifies that the Content Placeholder will display whatever the Placeholder in our Master Page displays. In order to add our custom content to this new page, we need to change the Content Placeholder to allow custom content. We can switch to using custom content by clicking on the right-hand arrow button at the right-hand end of the Content Placeholder and selecting **Create Custom Content**. Once this has been clicked, the title changes to **Custom** and we can edit our content as we wish.



Go ahead and replace the text in the Placeholder with the following:

Contact the Wine Company

Let's save our new page as **contact.aspx**.

Take a look into the **Code** view for our new page and you will see that the code is remarkably compact.

```
<%@ Page masterpagefile="_catalogs/masterpage/winec.master"
language="C#" title="Contact the Wine Company" %>
<asp:Content id="Content1" runat="server" contentplaceholderid="MainContent">
    <p>Contact the Wine Company</p>
</asp:Content>
```

Notice three things in this code:

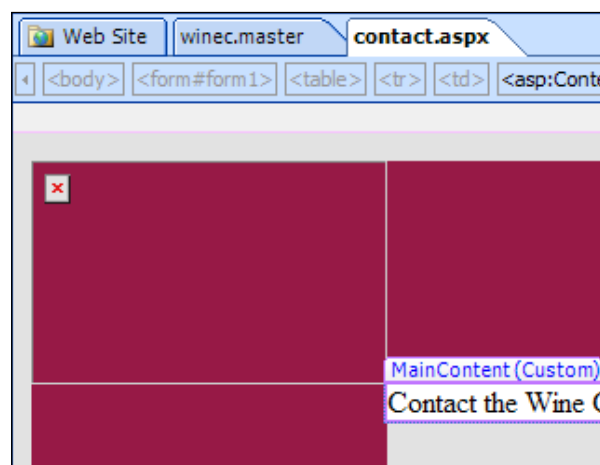
1. The first line of the code is a reference to the Master Page that our content page uses.
2. There is no need for any table code (<table>, <tr>, <td>) because that is all controlled by the Master Page.
3. Our page has handily used our content as the title for our page. No more Untitled Document text appearing in the browser's title bar by default.

Modifying the Master Page

Let's return to our Master Page and pretty it up a little by adding the company logo into the top left-hand corner of the page.

1. **Insert | Picture | From File....**
2. Browse to redlogo.191.jpg and select it.
3. Click **Insert**.
4. Specify **Wine Company** as the **Alternate Text**.
5. Click **OK**.
6. **File | Save**.
7. In the **Save Embedded Files** dialog, click **Change Folder...** and save the logo to a new folder called nav.
8. Click **OK** to close the **Save Embedded Files** dialog.

When we open our **contact.aspx** page in the Designer, we will notice a small red cross signifying a missing image where the logo should be.



When we saved our Master Page, the content page automatically updated to contain our logo but unfortunately the content page is looking in the wrong folder for the logo. The reason it cannot find the folder is that the content page is in the root of our site but the master page is a few folders higher up the directory tree. The problem is one of relative locations. Here are the relative paths required for each page to find the logo:

- Master Page— `../../images/nav/redlogo.191.jpg`
- Content Page— `images/nav/redlogo.191.jpg`

At the moment, the Master Page is telling our content page to use the first of these locations. So what to do?

One solution that would mean that the icon would display correctly in both pages would be to programmatically specify the file location using `Server.MapPath("images/nav/redlogo.191.jpg")`, but unfortunately code blocks are not permitted in Master Pages so we cannot use that method.

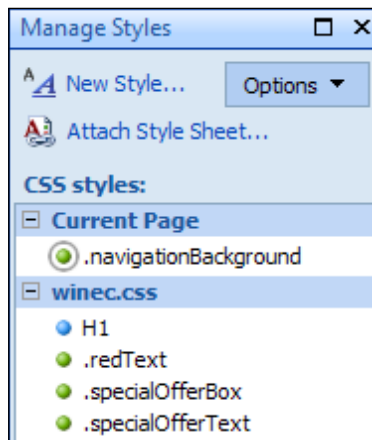
At the end of the day, it is the content page that is important because that is the one the visitor to the site will see. To solve the problem, go into the **Code** view of our Master Page and remove the two extra directory levels from `../../images/nav/redlogo.191.jpg` by removing the leading `../../`. Once we have saved `winec.master`, we will see that our logo appears correctly on our content page (and any other content pages we create at the same directory level).

Attaching Our StyleSheet to Our Master Page

The final thing that we will do is attach our style sheet to our Master Page, in order to maintain a uniform style across our site:

1. Open `winec.master` (it is inside the `_catalogs/masterpage/` folder) in **Design** view.
2. Click on **Attach Style Sheet...** in the **Manage Styles** task pane.
3. Click **Browse** on the **Attach Style Sheet** dialog.
4. Select `winec.css`.
5. Click **Open**.
6. Click **OK**.
7. Select **File | Save** to save the changes to `winec.master`.

Notice that our **Manage Styles** task pane has now been populated with the styles within **winec.css**:



Summary

We are now equipped to create new pages that follow a consistent theme. Even better than that, we can allow other users to contribute towards our site while still keeping our styles protected.

Read the following chapter to learn more about the tools SharePoint provides to allow contributors to publish their content.

5

Collaborating with Other Contributors

In this chapter, we are going to digress ever so slightly and learn about the great collaboration features that set SharePoint apart from other web development solutions.

First of all, we will examine Contributor mode, which allows us to define the changes individual solution creators can make to the site.

We will then take a look at the Workflow features that are built into SharePoint and that enable our site to automatically trigger specified actions when associated conditions occur.

Contributor Mode

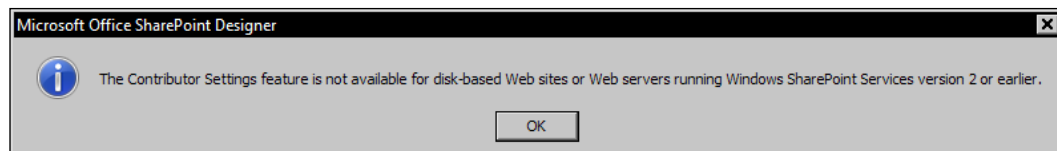
In developing the Wine Company website, Tony may wish to use Contributor mode to limit Gavin's level of participation. This would be a good idea because it would prevent the less technically savvy staff member from inadvertently breaking the site. By using SharePoint's Contributor mode, Gavin may be restricted in what he can do on the site. For example, he may be limited to only adding and editing pages, or adding and editing Web Parts. SharePoint Designer would ensure that he cannot alter the general layout of the site or perform other specified detrimental activities, such as adding huge images to the site.

It can often be difficult to decide which abilities to grant to users. When specifying the different abilities for the staff who will be maintaining our site, we should follow the 'principle of least privilege'. That is to say we give them as little access as possible but as much as they require in order to accomplish their tasks.

Server-Based Sites versus Disk-Based Sites

In our previous examples, we created our website locally and then published it to the SharePoint server so it could be viewed. This "disk-based" approach is a good way for an individual to develop a site (at least, in doing so, we are inadvertently creating a backup of our site) but is unsuitable if other users would like to contribute to our site.

If we were to try to enable collaboration on our Wine Company site, we would get the following message:



If collaboration is to be used by one of our sites, then the site must be based on the SharePoint server rather than on our personal computer.

Let's instruct our SharePoint server to set up a new server-based site for us:

1. Click **File | New | Web Site....**
2. Select **Empty Web Site** from the list of possible pages.
3. This time we are going to set the location on our SharePoint server. Type **http://<SERVERNAME>/share/** as the location of the new site (where **<SERVERNAME>** is the name of our SharePoint Server).
4. Click **OK**.

Now go ahead and create a new `default.aspx` page in the root of our new site.

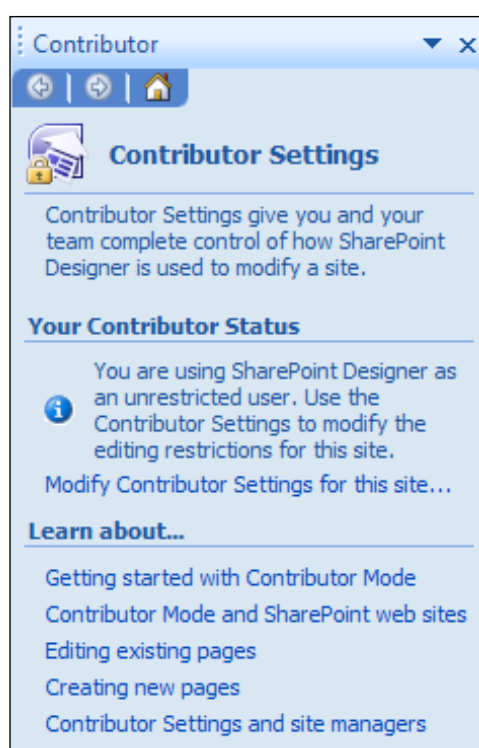
The first thing to notice is that when we browse to `http://<SERVERNAME>/share/` using our favorite web browser, the page appears (albeit without any content because we have not done that yet). This happened without the need for us to enter the **Remote Web site** view and publish our site. Easy, huh?

Even if we are not going to collaborate with other users on the development of our site, there are very few reasons why we should not use this server-based approach rather than the disk-based approach. These reasons for sticking with the disk-based approach may include:

- Uncertainty that we will always be able to access our files on the remote server.
- A desire to use the preview feature on our own computer to check we are happy with changes we have made before we publish them to the SharePoint server.

Enabling Contributor Settings

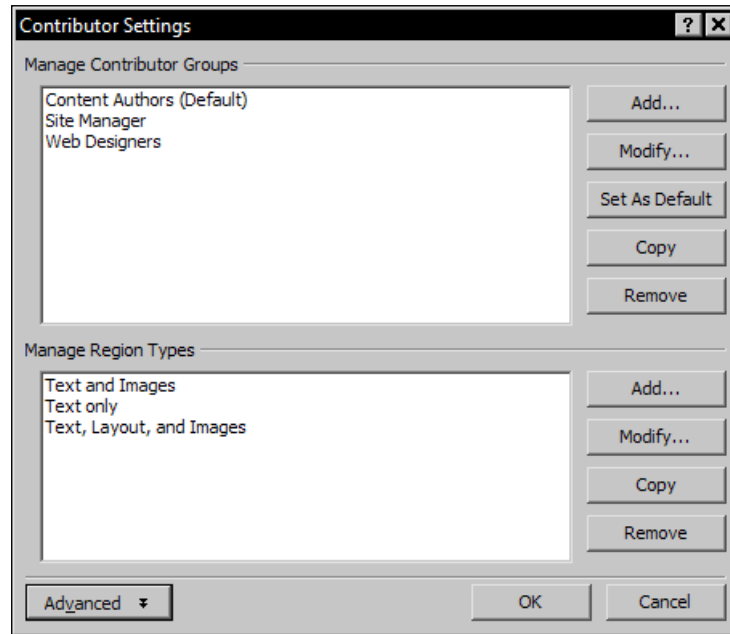
Now that we have our server-based site, we can go ahead and set the contributor settings. Start by opening the **Contributor** task pane.



You will notice that the task pane tells us that we have unrestricted contributor status. This status allows us to specify the contributor settings for our site, controlling which users can modify the different parts of our site.

Click on the **Modify Contributor Settings for this site...** link to bring up the **Contributor Settings** dialog. Here, we see that there are two categories we can manage:

- **Manage Contributor Groups**
- **Manage Region Types**



Let's look at both of these in more detail.

Contributor Groups

We can add the users in our organization to contributor groups, depending on the function that we would like those users to perform. We can set up as many contributor groups as we like. Each of the contributor groups can be set to use one of SharePoint's built-in permission levels.

By default, there are three contributor groups, each of which maps to a SharePoint permission level:

- **Content Authors** have the **Contribute** permission level that allows them to make changes in Design view. They can create new content pages and edit content in existing regions.
- **Web Designers** have the **Design** permission level, allowing them to make changes in both Design and Code view. They have unrestricted use of SharePoint Designer, including the ability to change the regions that are laid out on a page.
- The **Site Manager** has the **Full Control** permission level. This speaks for itself. They have the same unrestricted use as Designers.

If a user, who opens the site in SharePoint designer, is not assigned to one of the contributor groups, they will automatically be assigned to the group that is specified as the default group. For this reason, it is usual that the default contributor group is the one with the lowest set of permissions (e.g. the **Content Authors**).

Region Types

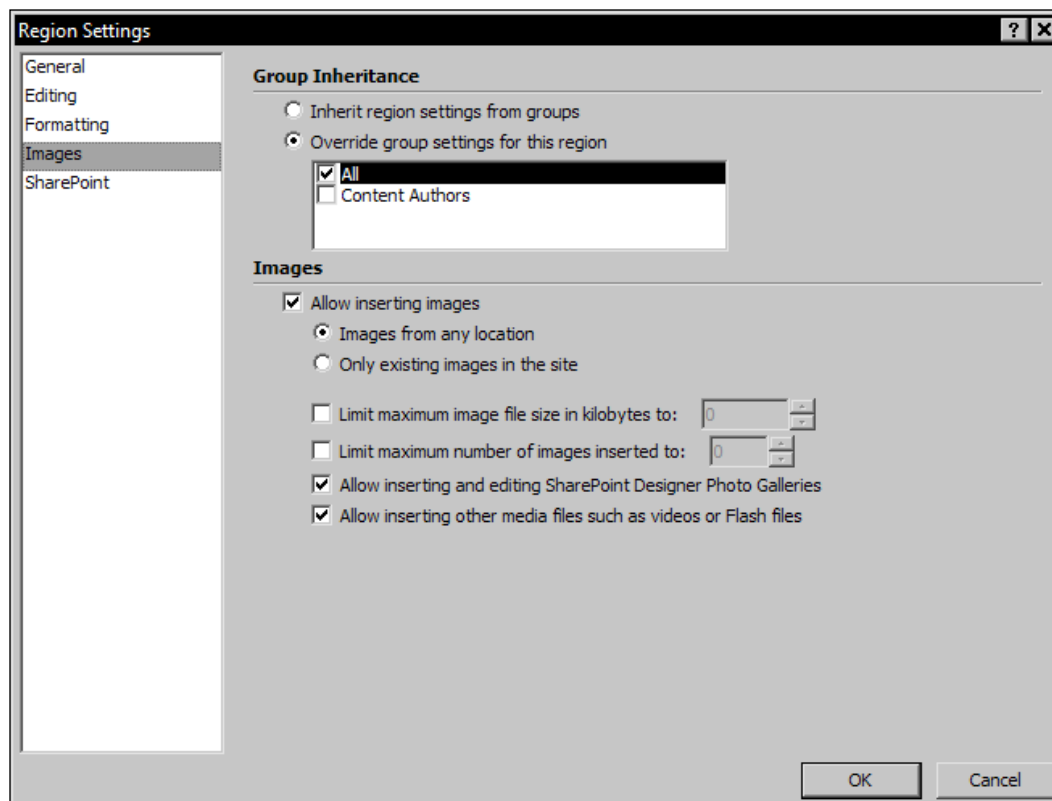
You will remember from the previous chapter that master pages contain content placeholders and that these placeholders in turn contain content regions.

Using the contribution feature, we can assign different region types to our content regions. In this way, we can specify the types of content that can be added to specific content regions. For example, we could assign the default **Text only** region type to a content region on our homepage that we have set up to display the name of the winner of our previous month's competition. The solution creator, who is just allowed to add text to the page, would be able to add the winner's name, but if they try to add a picture of the winner as well they would not be permitted to.

For each region type, we have a huge degree of control and can specify editing restrictions within four categories:

- Editing
- Formatting
- Images
- SharePoint

To see an example of one of these restrictions, click on the **Modify...** button within the **Manage Region Types** section in the **Contributor Settings** dialog and then click on the **Images** category on the left-hand side of the **Region Settings** dialog.



In this dialog, you will notice that it is not only possible to specify if this region type allows images to be added but it also allows much greater control. The example that I like the most is that you can specify the maximum file size of an image that can be added to a region. By using this feature, a site manager can prevent their staff from adding images that would take too long to download (as might be the case if Gavin were to be allowed free reign).

Before leaving the **Contributor Settings** dialog, we should also examine what is behind the **Advanced** button. The contact email address, which can be specified there, is an important feature. It provides users with a link in their **Contributor** task pane, which they can use to send an email, requesting changes to the **Contributor Settings** so they can perform tasks that they require.

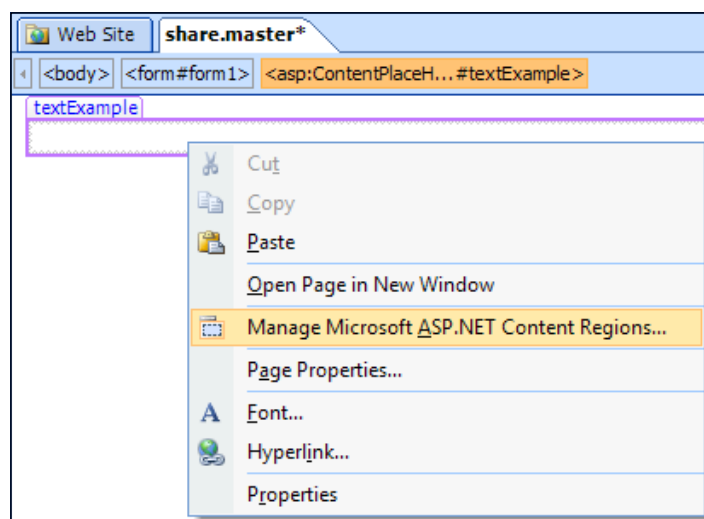
Setting Up Contribution on Our Master Page

Let's go ahead and put some of these features into practice. The first thing that we will do is create a new master page, to which we can add our content placeholders:

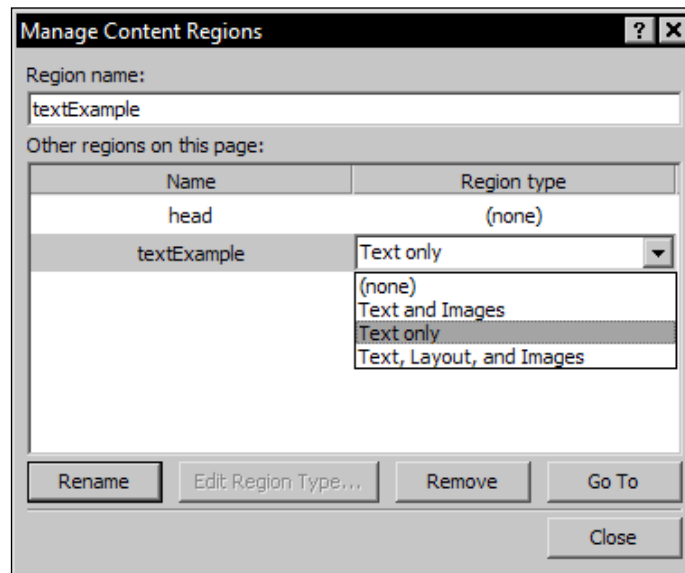
1. Go to **File | New | Page**.
2. Click on **Master Page**.
3. Click **OK**.
4. **File | Save As**.
5. Browse into the `_catalogs/masterpage/` folders.
6. Specify the **File name** as **share**.
7. **Save**.
8. When SharePoint prompts you to include a SharePoint Web Part Manager to the master page, click **Yes**.
9. When SharePoint prompts you to include a SharePoint Robots Meta Tag control to the master page, click **Yes**.

SharePoint has already created a content placeholder called `ContentPlaceHolder1` for us on our new master page. Go ahead and rename this placeholder to `textExample` (we can do this by opening the **Tag Properties** task pane and changing the ID or by clicking on **Manage Microsoft ASP.NET Content Regions**).

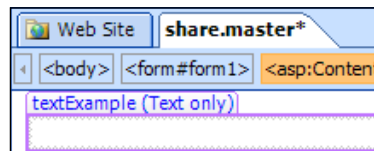
To assign a region type to our new content region, we right-click on the placeholder and select **Manage Microsoft ASP.NET Content Regions...** from the shortcut menu.



This brings up the following dialog. All we need to do is assign a region type (we will choose **Text Only**) to our region by selecting it from the **Region type** list and then click the **Close** button.



Notice how the region type that we chose appears in parentheses beside our placeholder name in the **Design** view:



Once we save our Master page, contributors will only be permitted to add text items into that placeholder. If they try to add images, they will be notified that they are not permitted to do so.

The Contributor's Experience

When working on a SharePoint site in Contributor mode, a restricted user will notice that only some of the features that would otherwise be available to them are functional. Users are able to view the permissions that are available to them by clicking on the **View your Contributor Settings...** link in their **Contributor** task pane.

One common way of using Contributor mode is to restrict most Contributor groups by only allowing them to create new pages from a specified master page. If the user tries to create a page in a different manner, they will be unable to and will be notified with the message "Your current Contributor Settings prevent you from creating new pages of the selected file type."

The **Contributor** task pane is very helpful when a user is in Contributor mode because it displays additional links such as **Create a new file in this site...**, which will allow the user to perform permitted tasks in an authorized manner.

Workflows

Whereas Contributor controls which users are allowed to perform actions, workflows define what happens when they do so.

Workflows allow rules to be created (without the need to write any code) that are triggered to perform actions, when associated conditions occur in SharePoint lists or libraries.

The Wine Company team is interested in workflows because they can use the functionality already built into SharePoint to allow them to focus on what they are good at (selling wine), while SharePoint takes care of the workflow for them.

Whenever a new order is placed, it triggers a rule in the workflow that sends an email to the accounts department notifying them of the new order. Depending on the payment status, the accounts department either approve or reject the order. If the order is approved, the workflow progresses and sends an email to the warehouse staff, notifying them to ship the order.

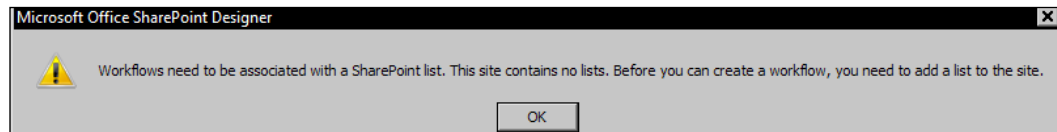
Workflow Designer

Windows Workflow Foundation is built into Windows SharePoint Services 3.0. SharePoint Designer allows us to plug into this functionality by using its Workflow Designer feature.

The Microsoft Windows Workflow Foundation component must be installed on the server and on your computer. If it is not, then you will be prompted to do this the first time you create a workflow.

Workflows and Lists

Because workflows monitor changes to lists or libraries, SharePoint Designer will not allow us to establish a new workflow unless we have a list or library in our site.



Lists and libraries are discussed in detail in the next few chapters, which cover collecting and displaying data. For the moment, we can think of lists as being data stores and libraries as being document stores.

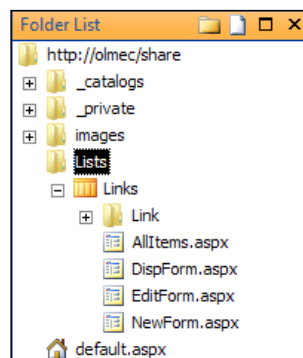
We will create a new list in the example share site, which we are using in this chapter. SharePoint Designer allows us to create custom lists, but we will be using one of the predefined lists called **Links**. We will use this list to hold a list of links to other websites. To use the predefined lists, follow these steps:

1. **File | New | SharePoint Content.**
2. The **Lists** option should already be specified in the left-hand pane of the **New** dialog.
3. Click on **Links** in the central pane.
4. Leave the name of our list as **Links**.
5. Click **OK**.

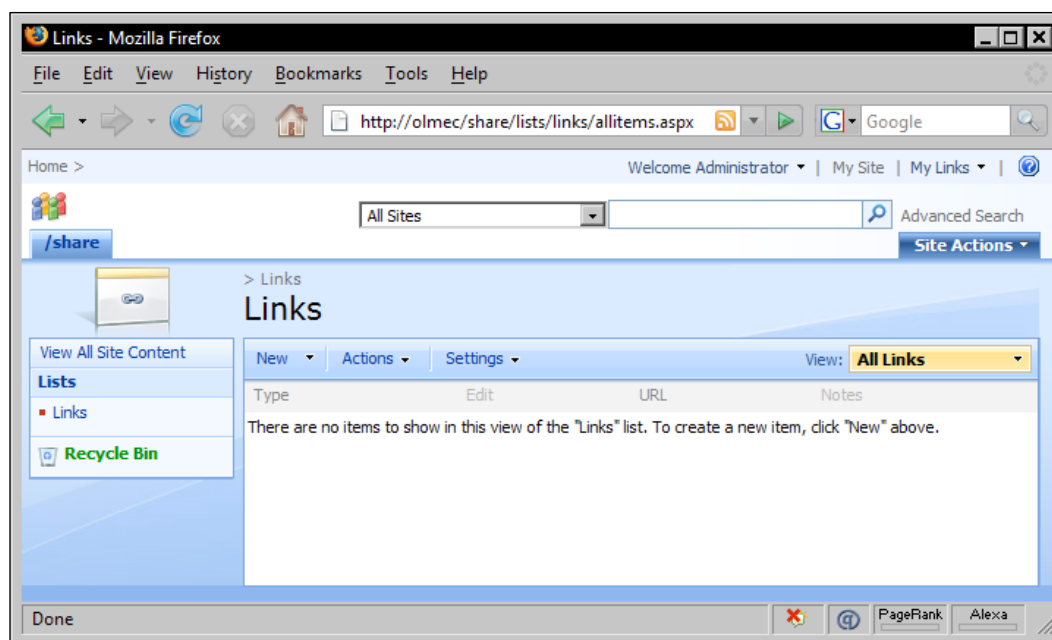


It is important to bear in mind that such use of SharePoint Content is only possible when editing a server-based site. SharePoint Content cannot be used on disk-based sites.

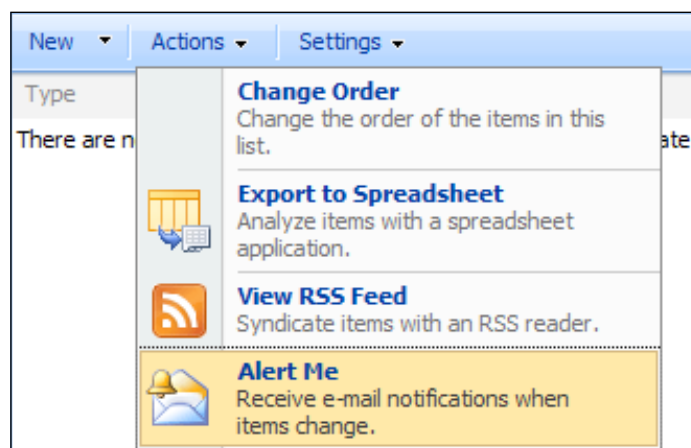
Our new **Links** list has been created within our share site, as can be seen in the **Folder List**:



Notice that four ASPX forms have also been created to allow us to maintain our Links list. If we browse to one of these new pages in our share team site (e.g. <http://olmec/share/lists/links/allitems.aspx>), we will see that a page has been created using the default master page to display our links.



Notice too how the new page allows new links (and folders of links) to be created and that within the **Actions** drop-down menu, we can choose to be alerted by email whenever the links change.

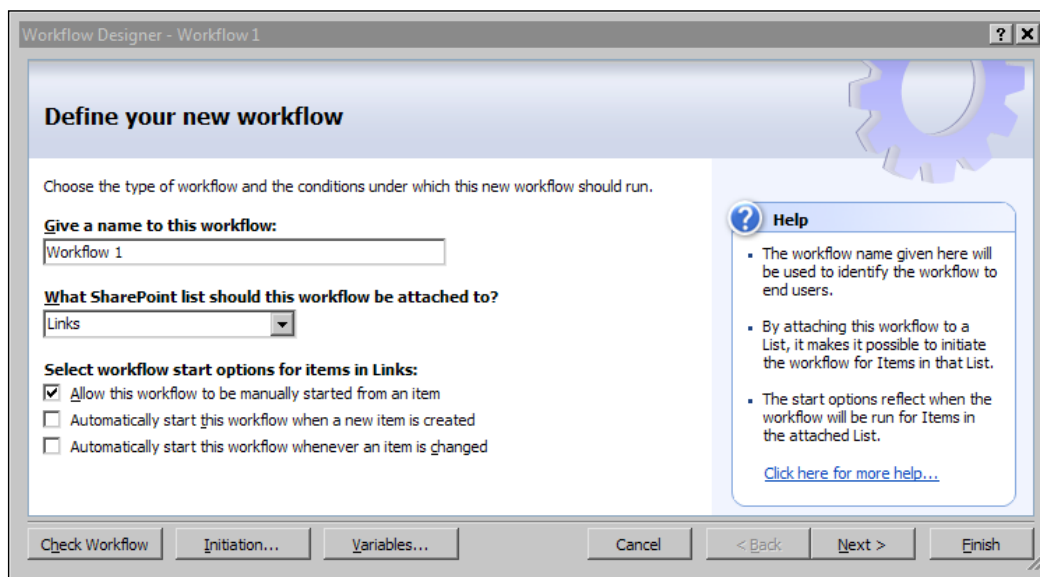


It will come as no surprise to learn that this **Alert Me** feature is an example of Windows Workflow Foundation functionality. Using **Alert Me**, new alerts can be created that alert one or more users to different types of changes to the list. In particular, I like the way you can specify to only be alerted when other users update the list (there is little point in alerting yourself to changes you have made, unless you are having a really bad day!). It is also possible to specify to receive daily or weekly summaries of changes rather than being inundated everytime a list is updated.

As comprehensive as these alerts are, we can create even more complex workflows using SharePoint Designer, so let's return there and learn more about creating our workflows using its Workflow Designer.

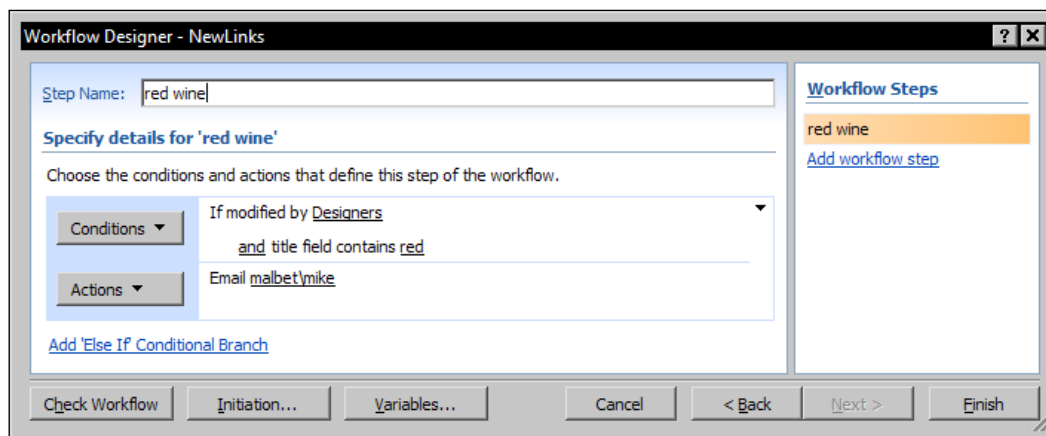
Defining New Workflows

We can create workflows from within SharePoint Designer by going to **File | New | Workflow**. The screen below is the first of the two wizard dialogs.



We should select a name for our workflow that will make sense in future (e.g. NewLinks). It is preferable that the name we choose does not contain any spaces because it will be used as a filename, as we will see shortly.

When we click on the **Next >** button, we are taken to the core screen of the wizard. Here, we are able to define the steps that our workflow progresses through.



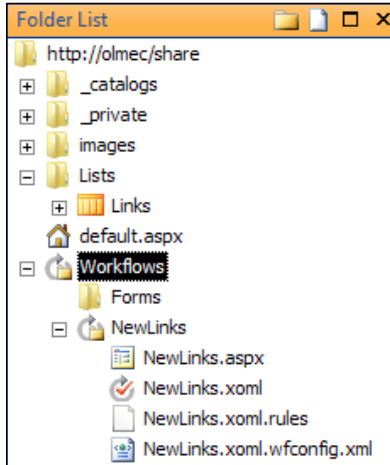
We can add as many steps to our workflow as we wish. For each step, we can add a number of conditions and actions:

- **Conditions** are the triggers for the rules, such as if the list has been updated by a specific person or user group (e.g. a new member of staff that we are supervising) or that contains a keyword (e.g. "red"). These conditions can be set to run together with the **and** text selected or independently with the **or** text selected.
- **Actions** are the events that happen when the conditions are met. The most common action is for an email to be sent.

Once we have set up our steps, we click on **Finish**. This will add a new **Workflows** folder to our **Folder List**. SharePoint creates an ASPX page within this folder that allows us to stop and start our workflow from within a browser. On my server, the address of this workflow would be `http://olmec/share/workflows/newlinks/newlinks.aspx`.

To edit our workflow within SharePoint Designer, we would double-click on the XOML file (e.g. `NewLinks.xoml`) in our **Folder List**.

XOML (pronounced zommel) files are standard XAML (pronounced zammel) files that specifically contain workflow serialization information.



Summary

In this chapter, we have seen that SharePoint Designer is more than just a tool to allow us to design pretty pages. When used properly, it becomes an integral part of a company's business processes.

Hopefully, you are as impressed by the contribution and workflow tools that we have at our disposal as I am. In the next chapter, we will learn how to bring new data into our site.

6

Collecting Data

It is difficult to specify which of SharePoint's features are the most powerful, but the ability to allow information to be easily submitted to a range of data stores must be very near to the top of any feature list.

Storing our data in data stores, such as a database or XML file, opens up a large number of opportunities to us. It allows our site to be easily searchable, our product list to be easily updated, and data to be shared between our site and external applications (such as our invoicing or warehousing systems).

In this chapter, we will take a look at the various types of data stores that SharePoint Designer allows us to connect to and how we can create forms that allow users to edit, add, and delete records in those stores.

Data Sources

SharePoint Designer allows us to connect to virtually every conceivable form of data store. Details of the data sources that SharePoint Designer can access are:

- **Business Data Catalog** is a new feature in Microsoft Office SharePoint Server that allows us to create views of business data contained in back-end server applications (e.g. SAP or Siebel). This feature does not exist in WSS.
- **Database Connections** allows us to connect to database servers on our network that use an SQL Server or OLE data provider.
- **Linked Sources** is not so much a data source in its own right, rather a wizard that allows us to link existing data sources together and access the resulting views.
- **Server-side Scripts** is a great way of easily updating RSS feeds.

- **SharePoint Libraries** are a less conventional type of data stores in that they store documents and the information about those documents rather than being used to store data. These documents can be any type of file (e.g. Word documents) but can also take the form of InfoPath forms, pictures, or wiki pages.
- **SharePoint Lists.** We already saw an example of a list in the previous chapter, when we used our links list to demonstrate workflow. Lists can be thought of as SharePoint-specific databases.
- **XML Files** can be used as a data source, as we will see in the example in a moment.
- **XML Web Services** can also be used as a data source. This allows us to connect to information that is exposed as a web service (e.g. weather forecasts or stock information).

When working in sites that use many data sources, the **Find Data Source** task pane can be used to help find the data source we would like to use.

Creating Our XML Data Source

We would like to display a list of products on a new page in our *SHARE* site. To allow us to do this, we will first create a new XML file containing our product data:

1. Select **File | New | Page**.
2. With the **General** option selected, click on **Text File** and then **OK**.

Now that we have our `Untitled_1.txt` file open, we can paste the following XML data into it:

```
<?xml version="1.0" encoding="utf-8"?>
<ProductsRoot>
<Products>
<ProductID>1</ProductID>
<ProductName>Chateau de l'Hospital</ProductName>
<Price>16.99</Price>
<InStock>509</InStock>
</Products>
<Products>
<ProductID>2</ProductID>
<ProductName>Chateau le Gay</ProductName>
<Price>94.99</Price>
<InStock>20</InStock>
</Products>
</Products>
```

```
<ProductID>3</ProductID>
<ProductName>Goats do Roam Goat Roti</ProductName>
<Price>14.99</Price>
<InStock>0</InStock>
</Products>
<Products>
<ProductID>4</ProductID>
<ProductName>Mad Housewife</ProductName>
<Price>10.68</Price>
<InStock>1206</InStock>
</Products>
<Products>
<ProductID>5</ProductID>
<ProductName>Thirsty Lizard</ProductName>
<Price>6.49</Price>
<InStock>42</InStock>
</Products>
</ProductsRoot>
```

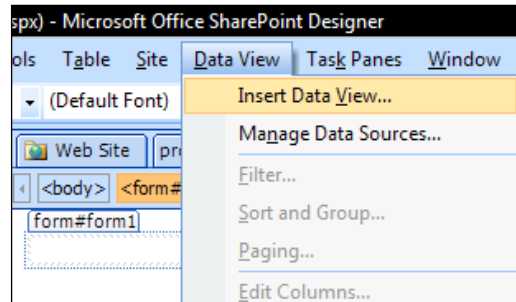
Once we have added the data to our text file, we can save it as an XML file:

1. Go to **File | Save As**.
2. Click on **New Folder**.
3. Name the new folder **data**.
4. Browse into the new folder.
5. Change the **File name** to **products.xml**.
6. Change the **Save as type** to **XML**.
7. Click **Save**.

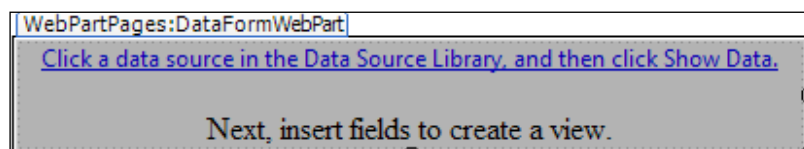
Creating a Data View

We will now create a new page in our site to allow us to display our products. Create this new ASPX page in the root of our site and save it as **products.aspx**. We are going to place a data view onto this page so that we can view and edit the data that is stored in our XML file.

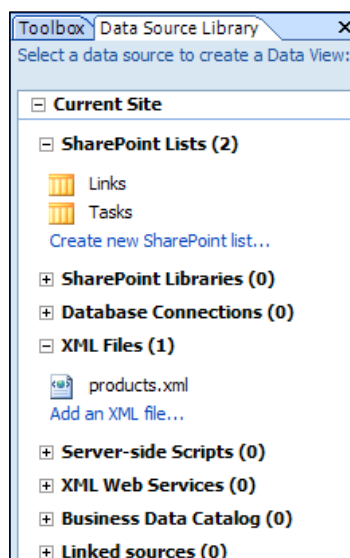
To add our data view into the form tag, go to **Data View | Insert Data View**. SharePoint Designer will take a few moments to process this command.



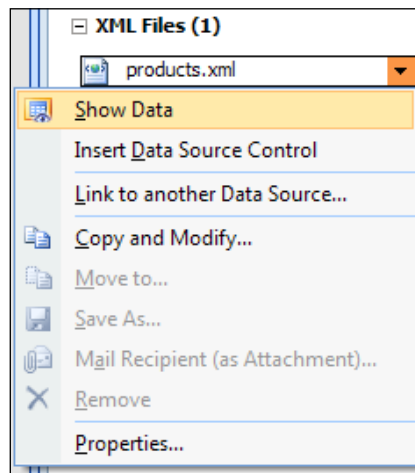
Once SharePoint Designer has finished doing its thing, you will see that a new Data View (called DataFormWebPart) has been added to our web page.



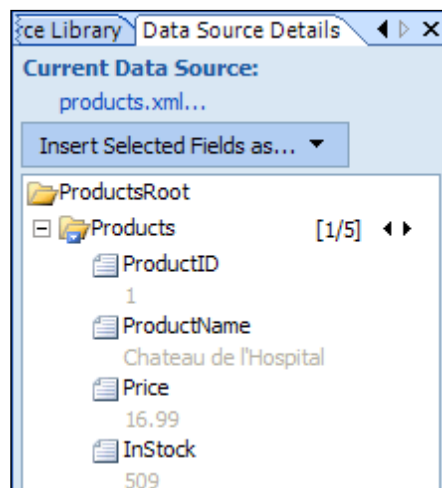
You will also notice that the **Data Source Library** task pane has opened automatically to give us easy access to our data stores. Clicking on the plus symbol to the left of **XML Files** will expand the list of XML files, showing our `products.xml` file.



If you have used older methods of web development, such as creating Classic ASP sites using DreamWeaver alongside separate database packages, then you are going to love how simple it is to add data to our SharePoint page. Gone are the days of flicking backwards and forwards between different applications to look up table and field names. All we need to do is click on our data source in the task pane and select **Show Data** from the drop-down list:



Doing this displays the contents of our data source in the **Data Source Details** task pane. We would like to take the fields from the data source and display them in the data view that we have on our web page. We select the fields that we would like to use in our data view by left-clicking on them (holding *Ctrl* while we do this allows us to select multiple fields).

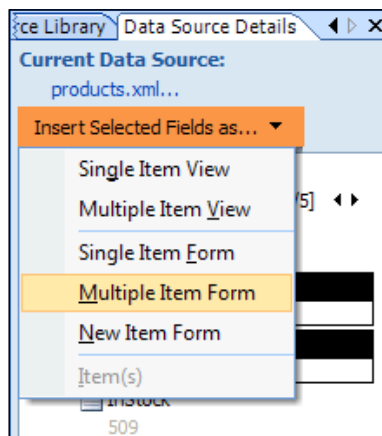


We will select the **ProductName** and **Price** fields.

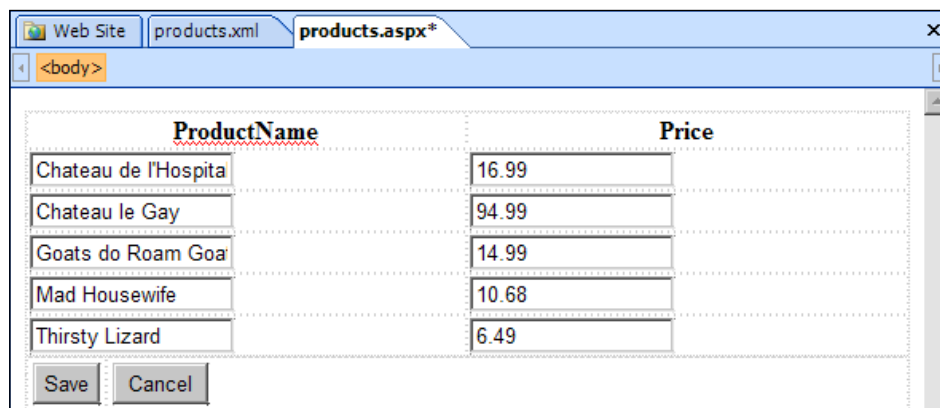
Once we have selected the fields we would like to use, clicking **Insert Selected Fields as...** will allow us to select the type of data view we would like to create.

As the name suggests, the View options will display the data in the data view but will not allow the data to be edited (probably a good idea for the likes of a public-access price list). The Form options will display the data in a way that allows items to be edited.

It is important to note that although the Form option may be selected, the data source may not be updatable because of its own permissions. You will find that for some data sources (such as custom SQL queries), the Form options won't be visible at all. In our XML example, there are no such problems.



We will select the **Multiple Item Form**, which will display all our records in one long editable data view:



ProductName	Price
Chateau de l'Hospita	16.99
Chateau le Gay	94.99
Goats do Roam Goa	14.99
Mad Housewife	10.68
Thirsty Lizard	6.49

Save Cancel

If we open `products.aspx` in our web browser (e.g. by visiting `http://olmec/share/products.aspx`), we see that we have a fully operational update form. You will see that if we change the price of the first wine to 20 and click **Save**, the `data/products.xml` file in our site has changed (you will need to re-open the file to see this change). We can also see this change in the **Data Source Details** task pane when we click the **Refresh data source** link at the bottom of the pane.

The first thing I like to do to my new data views is to change the column names. In our example, we can change `ProductName` to `Product Name`. In this example, it is probably also a good idea to constrain the width of the table (i.e. right-click on the table, select **Table Properties...**, and specify the width as 350 pixels) so that the table does not have as much white space inside it.

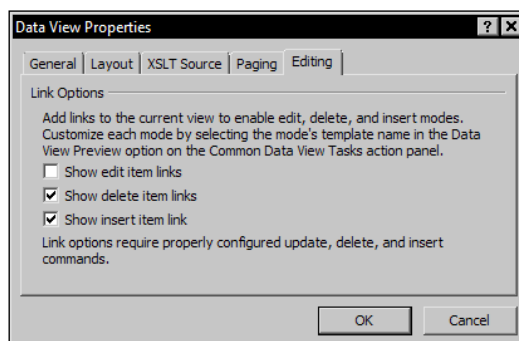
Adding and Deleting Records

Modifying records is all very well but we also usually like to give our users the ability to insert and delete records.

To change our data view so that users can insert or delete records, we do the following:

1. Click on our data view.
2. Click on the arrow in the top right-hand corner of the data view. This will show the **Common Data View Tasks** list. (We will cover the features in this list in more depth in the next chapter.)
3. Click on the **Data View Properties** link.
4. Select the **Editing** tab.
5. We will check the **Show delete item links** and the **Show insert item links** check boxes.
6. Click **OK**.

The following image displays the editing link options:



It should be noted that when working with a SQL data source, we are also given the option to auto-synchronize the update, delete, and insert commands with the edit, delete, and insert controls that appear in the data view.

If we now browse to our site, we see that **insert** and **delete** links have been added to our form, allowing users to perform those actions.

As we have not included the ability to edit the records in our example, we did not need to begin with a form, but it is useful that you know how to add such a form for the times that you will want to allow users to edit records.

	Product Name	Price
delete	Chateau de l'Hospital	20
delete	Chateau le Gay	94.99
delete	Goats do Roam Goat Roti	14.99
delete	Mad Housewife	10.68
delete	Thirsty Lizard	6.49
insert		

Clicking on the **insert** link creates a simple blank form that we can fill the with details for our new product (a relatively cheap number from Avalon called Vivacious Vicky):

	Product Name	Price
save	Vivacious Vicky	8.99
cancel		

Clicking **save** adds the record to our XML file and returns us to our list of products. If we click on the **delete** link beside the newly created record, it removes it from the database for us (note that it does not ask us to confirm that we would like the record to be deleted to avoid users accidentally deleting data).

InfoPath

It's not possible to spend very long reading about SharePoint before the word InfoPath comes along. InfoPath 2007 is a member of the Microsoft Office family that allows electronic forms to be created to collect and manage data.

InfoPath forms can be submitted via Outlook email messages, forms for mobile devices, or browser-based forms. When a form is submitted, the data is saved as XML. As a result, InfoPath forms can be integrated into most data workflows. InfoPath fits seamlessly into business processes that use Microsoft BizTalk Server or SharePoint.

We can integrate InfoPath forms into our SharePoint site through the InfoPath form library (part of the SharePoint Libraries). Creating a new form library is simple:

1. Expand the **SharePoint Libraries** node in the **Data Source Library** task pane.
2. Click on the **Create new Document Library...** link.
3. Ensure that **Document Libraries** is selected in the left-hand column.
4. Click on **Form Library** in the central column.
5. Specify a name for your document library.
6. Click **OK**.

Summary

Now that we are equipped to collect data in our SharePoint site, we will move on to the next chapter, where we will examine the many methods that SharePoint has for allowing us to display that data.

7

Displaying Data

Now that we have the data view firmly embedded in our page, it is a breeze to change the way the data looks and operates.

In this chapter, we will learn to apply nice formatting to data that is displayed on our page. We will learn to do this by making use of Cascading Style Sheets. We will also learn how to format our data automatically, depending on the data values (a technique known as conditional formatting). Then we will learn to filter and sort our data, use formulae to perform calculations, and how to split our data up into multiple pages.

Formatting the Data View

The default data view that we are presented with uses uninspiring black serif text on a white background. We can jazz up our data view using two different methods:

Direct Formatting

It is possible to apply formatting directly to our data view by highlighting the cells that we wish to format and then using the formatting tools that we learned about in Chapter 4. This can be a good option if we only want to format a single data view but is not the best approach if we would like to apply our formatting on a site-wide basis.

CSS Formatting

A more manageable way to apply formatting to our data views is to make the changes across the entire site by editing our style sheet.

When we click in the cell that has the **Price** heading in it, notice that a tag appears above it, telling us that this cell is referred to as `th.ms-vh`. That is to say that it is a table heading (`th`) element that is being rendered using the `ms-vh` class (which I assume stands for Microsoft View Heading). Similarly, if we click in any of the cells further down the data view, we see that they are referred to as `td.ms-vb` (standing for Microsoft View Body). This reference is used to specify the format of the cells in our table that display the actual data.

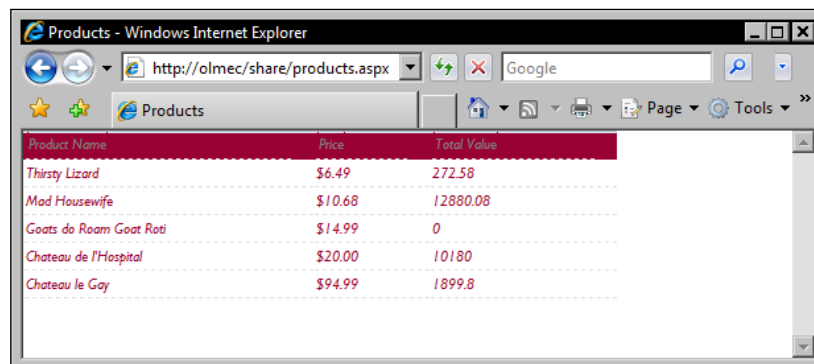
In addition, there is an `ms-alternating` class that renders every other row with a different background color.

To allow us to edit our styles, we must first create a new blank style sheet, which we will call `share.css`:

1. Select **File | New | CSS**.
2. Go to **File | Save**.
3. Give the file the name **share**.
4. Save the file as **CSS Files** file type.
5. Click **Save**.

The next step is to attach our style sheet to our site so that our pages can refer the styles that we will create within the style sheet:

1. Open the **Apply Styles** task pane.
2. Click **Attach Style Sheet**.
3. Click **Browse**.
4. Browse to `share.css` and click it.
5. Click **Open**.
6. Under **Attach to**, check **All HTML pages**.
7. Click **OK**.
8. Click **Close** on the information dialog.



The screenshot shows a Windows Internet Explorer browser window titled "Products - Windows Internet Explorer". The address bar shows the URL "http://olmec/share/products.aspx". The page displays a table with three columns: "Product Name", "Price", and "Total Value". The table contains five rows of data, with alternating background colors for the rows.

Product Name	Price	Total Value
Thirsty Lizard	\$6.49	272.58
Mad Housewife	\$10.68	12880.08
Goats do Roam Goat Roti	\$14.99	0
Chateau de l'Hospital	\$20.00	10180
Chateau le Gay	\$94.99	1899.8

We will then make our style sheet ready to use by defining some styles. Adding the following code to the style sheet will change the `ms-vh` and `ms-vb` classes so that they are formatted in a more inspiring manner:

```
th.ms-vh, td.ms-vb {
    font-family:gill sans, gill sans mt, arial, sans-serif;
}
th.ms-vh {
    border-width:0px;
    background-color:#903;
    color:#FFF;
}
td.ms-vb {
    color:#903;
    border-top-width:0px;
    border-left-width:0px;
    border-right-width:0px;
    border-bottom-width:1px;
    border-bottom-style:dashed;
    font-style:italic;
}
ourtable{
    border-width:0px;
}
```

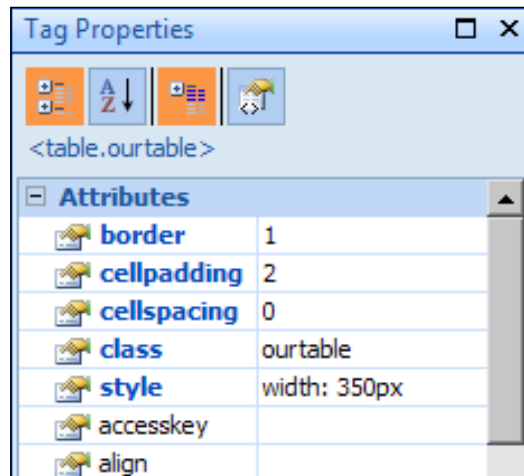
Notice how the first line of our style sheet refers both `ms-vh` and `ms-vb`, with a comma separating them. This allows us to specify the font face in one place rather than needing to enter it separately into each class. Grouping styles in this way not only saves time when creating our site but also makes the site more easily maintainable when we and other people make changes in the future.

If you are eagle-eyed, you will also notice that the color references (e.g. `#903`) only have three digits rather than six. This is possible when a color has repeating numbers (e.g. 99 or 00), allowing us to condense `#990033` to `#903`.

If you are not familiar with using CSS, do not be put off using it because it is easy to learn. We also get a helping hand, because when we type code into the style sheet, SharePoint Designer uses IntelliSense to suggest code we may like to use.

Once we have saved our style sheet, SharePoint Designer instantly reflects changes to the style sheet in the Design view of `products.aspx`.

When creating our `td.ms-vb` style, we specified that dashed lines should appear below each cell. By default, SharePoint Designer has a default value of 0 for our borders, meaning that they will not display. In order for the dotted lines to appear, we will need to make sure that our table has a border value of 1. We can do this by highlighting the whole table and typing 1 into the **border** attribute in our **Tag Properties** task pane. Selecting `ourtable` as the **class** for the table in this task pane will remove the solid border, allowing our dashed lines to be visible in all their glory.



We can take our formatting even further and use CSS to format the **delete** and **insert** text links so they look like buttons, by giving them a border and some padding. We can also give our "buttons" a different background color whenever the cursor is positioned over them.

To add this additional formatting, we will place the following code into our style sheet:

```
a:link, a:visited, a:active {
    color:#903;
    padding-top:1px;
    padding-bottom:1px;
    border-color:#903;
    border-width:1px;
    border-style:solid;
    text-decoration:none;
}
a:hover {
    background-color:#EEE;
}
```

A word of warning about specifying these styles for your links—they will apply to all links in the share site (and it is unlikely that you would want to do that). It would be better for us to create a new class (e.g A.funkyButton) that we can use whenever we want to render our links to look like buttons.

	Product Name	Price
delete	Chateau de l'Hospital	20
delete	Chateau le Gay	94.99
delete	Goats do Roam Goat Roti	14.99
delete	Mad Housewife	10.68
delete	Thirsty Lizard	6.49
insert		

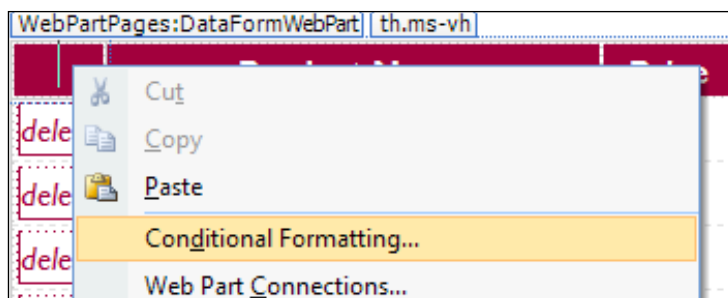
Conditional Formatting

We can make our data views more meaningful by setting up our data view to automatically use different styles in response to differing data values. This technique is known as conditional formatting.

Applying conditional formatting in SharePoint Designer requires us to use the XPath language. Don't worry though because SharePoint Designer makes this easy for us.

We would like to apply some conditional formatting to our data view so that it shows a differently formatted price for all wines that have a low number in stock.

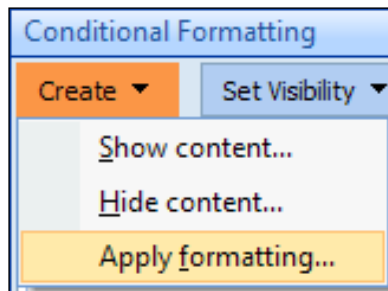
We start this process by right-clicking on our data view and selecting **Conditional Formatting** from the shortcut menu. This will open up the **Conditional Formatting** task pane.



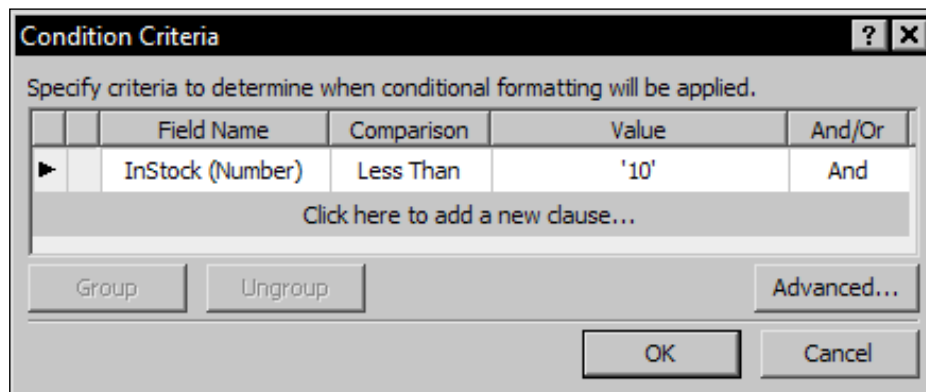
We then click on the data value that we would like to apply our conditional formatting to. We will select any of the prices in the data form. Below, I have selected the price of the most expensive wine, **Chateau le Gay**.

	Product Name	Price
delete	Chateau de l'Hospital	20
delete	Chateau le Gay	94.99
delete	Goats do Roam Goat Roti	14.99

We then click the **Create** button in our **Conditional Formatting** task pane and select **Apply Formatting**.

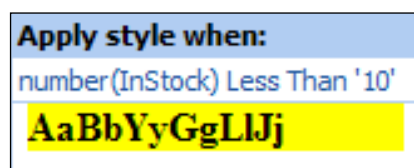


Doing so presents us with the **Condition Criteria** dialog box. This dialog is used to specify the conditions for our formatting. To begin using the dialog, we click on the line that says **Click here to add a new clause....** We would like our **Price** to be formatted differently if the number of products in stock is less than 10, and so select the options in the diagram below and then click **OK**:



Now that we have chosen our criteria, the next screen allows us to specify the style that will be automatically applied to our **Price** when the condition is met. In the **Font** category, we will select a **font-weight** of **bold**, and in the **Background** category, we will select a nice bright **background-color** of **yellow** (**#FFFF00**). This vivid style will ensure that the information stands out to the users of the site.

To modify the condition at a later date, we can click on the condition in our **Conditional Formatting** task pane and select **Edit condition...** from the drop-down list.



It is also worth noting that it is also possible to use conditional formatting to change the visibility of the data so that it can be displayed or hidden, depending on the value. This is useful to show or hide images. For example, if the type of wine is red, show a red bottle. If white, show a white bottle.

Formatting Numbers

It is worth pointing out that it is very easy to format numbers in our data view so that they display as we would like. We can display our numbers as percentages or as a currency and specify the number of decimal places that the number should have. We can also control the 1000 separator so that commas appear in the correct places to make large numbers more easily readable.

We will take the opportunity to display our prices with a dollar sign in front of them by doing the following:

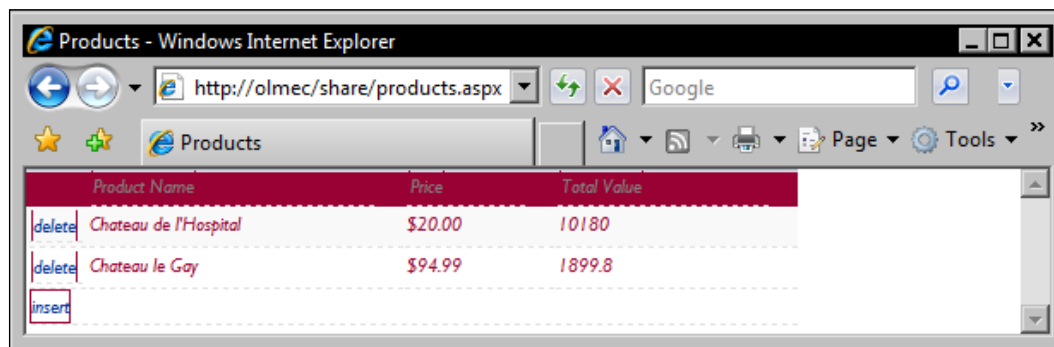
1. Right-click on one of the prices in our data view.
2. Select **Format Item as**.
3. Select **Currency**.
4. This will open the **Format Number** dialog. Ensure that the **Symbol** is set to \$.
5. Click **OK**.

Notice that we only needed to select one record and that all prices were formatted in the same manner.

Filtering Data

It is possible to filter our data view so that only certain records are displayed. We will use this functionality to display only products with a price of over \$15. This feature works in a similar manner to the conditional formatting (albeit there is no formatting to be applied). To filter our data view, we follow these steps:

1. Right-click on our data view.
2. Select **Show Common Control Tasks**.
3. Select **Filter:** from the list.
4. Click on the first row to add our new clause.
5. Specify the following values:
 - a. **Field Name** = Price (Number)
 - b. **Comparison** = Greater Than
 - c. **Value** = 15
6. Click **OK**.



	Product Name	Price	Total Value
delete	Chateau de l'Hospital	\$20.00	10180
delete	Chateau le Gay	\$94.99	1899.8
insert			

You will notice that our data view now only displays the products that are priced over \$15.

To remove the filtering, we follow the first three steps so that the **Filter Criteria** dialog appears again. We remove our criteria by right-clicking on the black arrow to the left of the criteria and selecting **Remove**. Finally, we click **OK**.

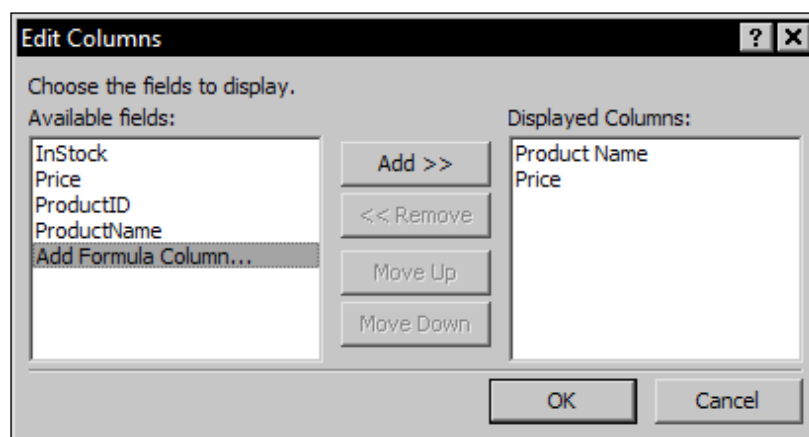
Using Formulae

When using SharePoint Designer, it should be remembered that it is a fully-fledged member of the Microsoft Office suite of products. As such, we can use it to perform powerful calculations that you would expect other products in the suite (such as Excel) to perform.

We will add a new column to our data view that will display the total value of the wine the Wine Company has in stock. We begin this process by clicking on our data view to select it and then going to **Data View | Edit Columns....**

The **Edit Columns** dialog allows us to select the fields in our data set that we would like to include in our data view. If we would like to add columns to or remove columns from our data view, this is the dialog that we would use.

The dialog also allows us to add a formula column. We will do this now by clicking on **Add Formula Column...** and then clicking on the **Add >>** button.



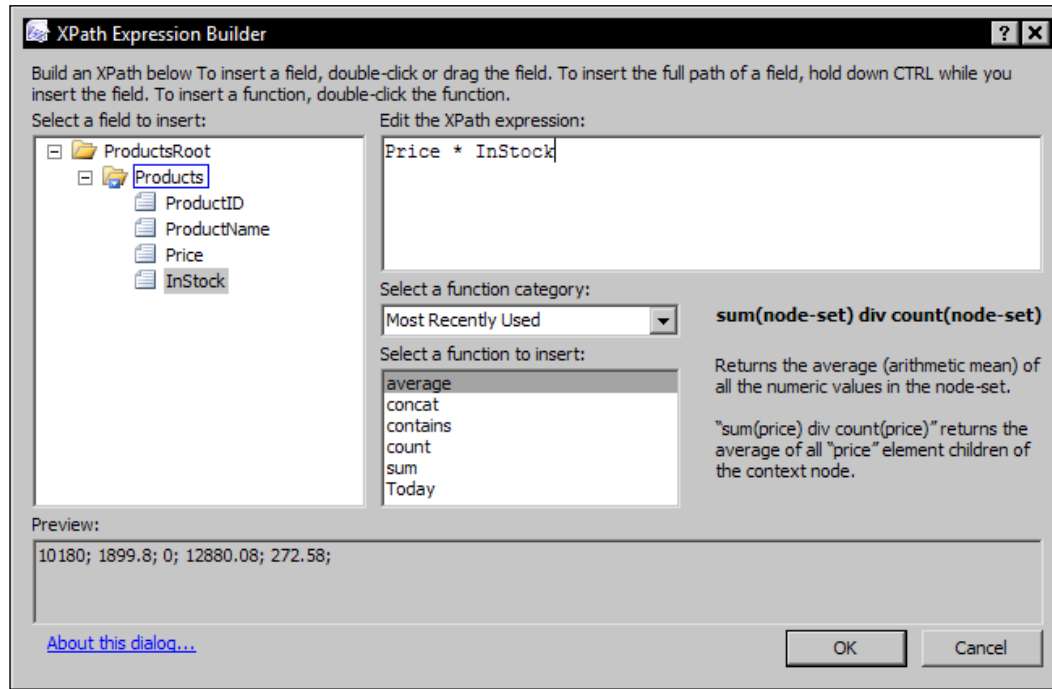
The **XPath Expression Builder** then appears. If you have previously created formulae in Microsoft Excel or Microsoft Access, then you will be familiar with the principle behind creating formulae using a builder like this.

We can either type our field names into the pane titled **Edit the XPath expression** or drag them across from the field list in the left pane. We can also use my personal favorite method and double-click them across. Whenever we type a space into the expression pane, IntelliSense suggests appropriate code items including field names and arithmetic operators (such as +, -, /, and *).

We will use the following simple expression:

```
Price * InStock
```

The following image shows the Expression Builder in use:



Once we click the **OK** button, SharePoint Designer adds our formula to our **Displayed Columns** list as **Formula 1** (important note: this has nothing to do with fast racing cars!). When we click the next **OK** button on the **Edit Columns** dialog, our formula is added as a new column on the right of our data view. You may wish to take a moment to resize the width of your data view to 450 pixels and to rename the last column **Total Value**.

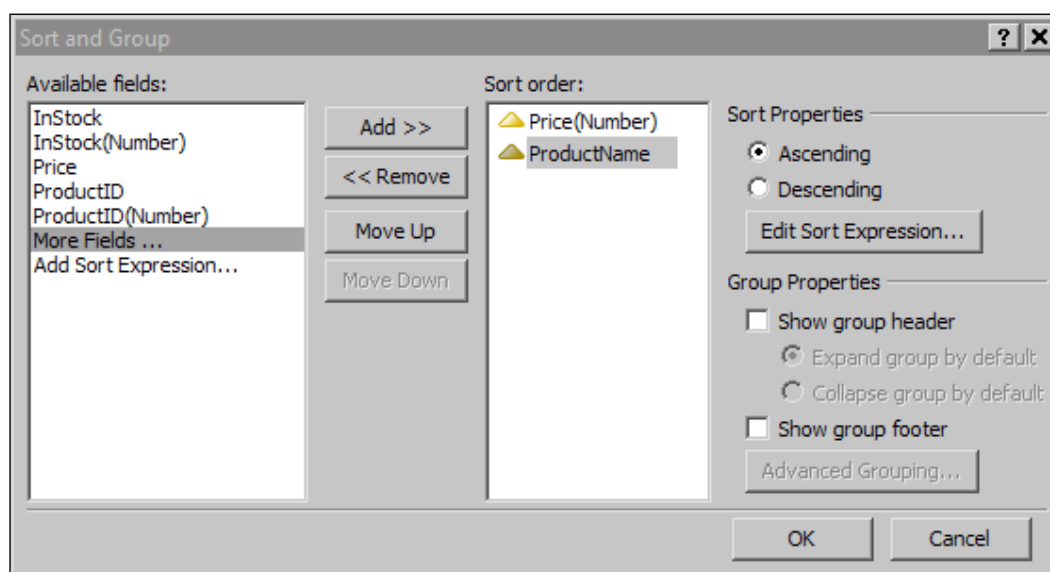
	Product Name	Price	Total Value
delete	Chateau de l'Hospital	20	10180
delete	Chateau le Gay	94.99	1899.8
delete	Goats do Roam Goat Roti	14.99	0
delete	Mad Housewife	10.68	12880.08
delete	Thirsty Lizard	6.49	272.58
insert			

Sorting Data

Hopefully, by now you are impressed with how easy it is to display data using SharePoint Designer. When it comes to ordering our data, you will continue to be impressed.

We would like to sort our wines so that the cheapest wines appear first and the more costly bottles appear last. To do this, we use the sort feature:

1. Right-click on our data view.
2. Select **Show Common Control Tasks**.
3. Select **Sort and Group** from the list.
4. Click on the `Price (Number)` field and click **Add>>**.
5. Click on the `ProductName` field and click **Add>>**.
6. Click **OK**.



It is worth noting that we can also group similar items together within the data view. For example, if our data specified whether the wine was red, white, or rosé, then we would be able to group the data by type.

Allowing Users to Sort the Data

It is easy for us to give users the ability to sort the records in the data view. We can enable this sorting like so:

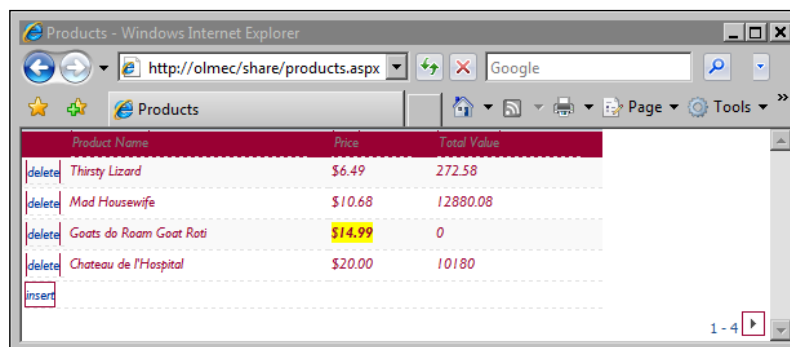
1. Click on our data view to select it.
2. Go to **Data View | Change Layout**.
3. Click on the **General** tab.
4. Check the **Enable sorting and filtering on column headers** checkbox.
5. Click **OK**.

Once we have saved the page, users will be able to click on the column headings to sort the records by alphabetical order (or numerical order, provided that the data in that column are numerical values).

Paging

Although our example data only contains five different products, there are a huge number of wine products in the market (there are about 6,000 wineries in the United States alone). Naturally, we would not want to display tens of thousands of products all on one page. By using paging, we can specify the maximum number of records that will be displayed in our data view at one time. Depending on the options we have specified, users may be able to click forwards and backwards through the data view, displaying the next records each time they do so.

1. Right-click on our data view.
2. Select **Show Common Control Tasks**.
3. Select **Paging** from the list.
4. Click on the **Display items in sets of this size** radio button.
5. Enter **4** into the field.
6. Click **OK**.



	Product Name	Price	Total Value
delete	Thirsty Lizard	\$6.49	272.58
delete	Mad Housewife	\$10.68	12880.08
delete	Goats do Roam Goat Roti	\$14.99	0
delete	Chateau de l'Hospital	\$20.00	10180
insert			

Summary

Over the course of the last two chapters, we have discovered how easy it is to interrogate a whole range of data sources and display information from them in our SharePoint site in an attractive and useful manner.

In the following chapters, we will be examining a range of different controls that we can add to our site to perform specific actions, with the need for virtually no coding on our part.

8

Adding Web Parts

In the final four chapters of this book, we will see how we can make our SharePoint site do more, adding ASP.NET controls and Web Parts to our pages.

In this chapter, we will focus on Web Parts. We will learn what they are used for and how to insert them into our pages. We will also take some time to examine the benefits and use of Web Part Zones.

What Are Web Parts?

Components have traditionally been used to add extra functionality to websites in a modular fashion. With the advent of Microsoft's .NET, came the ability for programmers to create ASP.NET Web Parts that could be embedded within web pages to give those pages additional functionality (e.g. a shopping basket). One of the main benefits of these Web Parts is that they only need to be programmed once and can be reused in different parts of our site (or indeed in many different sites).

SharePoint provides us with specialist Web Parts that can be placed in our page to allow the end user to perform a specific function (e.g. interact with a calendar) within their browser.

Web Parts are much more powerful than traditional components because they also allow us to specify which users can edit the Web Parts.

There are many Web Parts included with Microsoft Office SharePoint Server and others are available to download free of charge. Microsoft used to host an online SharePoint Products and Technologies Web Component Directory (which it renamed to the snappier-titled **SharePoint Utility Suite**), but this directory is no longer available.

Other Web Parts are full-blown commercial products and must be paid for. The prices of components are often high and usually priced per SharePoint server. Some components also require an ongoing annual license to be paid.

You can access Web Parts for SharePoint that will do the following (and more):

- Calendar and Schedule
- Charting, Graphing, and Gauging
- Data Entry
- Data Grids
- E-commerce
- Search
- Toolbars and Menus
- Tree Views and Lists
- Modification of the User Interface
- Wikis

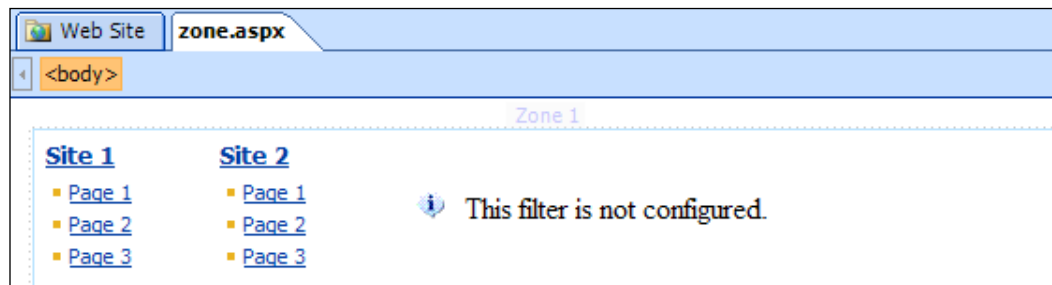
Web Part Zones

Web Parts can be placed directly onto our page, as we did with the Data Form Web Part in the last chapter. This is nice and simple to do.

Often we wish to have greater control over our Web Parts. This is where Web Part Zones come in. By adding our Web Parts into Web Part Zones, we can perform two different types of functions.

Firstly, the use of Web Part Zones allows us to specify the permissions for the Web Parts contained within them. In this way, we can dictate which users are able to modify the Web Parts in their browser.

The other purpose of Web Part Zones is to allow us to lay out our Web Parts in a pre-determined manner (without needing to resort to using a table to do so). We can choose to either lay out our Web Parts vertically or horizontally, as in this image (where the Category Web Part is placed beside the Choice Filter Web Part):

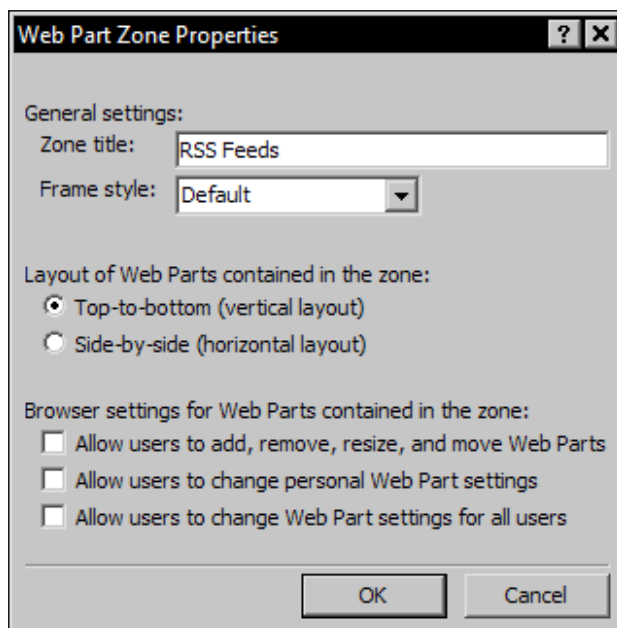


Inserting a Web Part Zone

Let's create a new page and add a Web Part Zone to that page:

1. Open up the share site in SharePoint Designer.
2. Go to **File | New | ASPX**.
3. Select **File | Save**.
4. Give your page the name **feeds**.
5. Set the **Save as type** to **ASPX Files**.
6. Change the title to **RSS Feeds**.
7. Click **Save**.
8. If you do not already have the **Web Parts** task pane open, then open it using **Task Panes | Web Parts**.
9. Click on the **New Web Part Zone** button, which is located at the bottom of the **Web Parts** task pane. This will add a new zone to our page called Zone 1.
10. In the **Web Part Zone Properties** dialog, give the zone the title **RSS Feeds** and uncheck the three permissions check boxes at the foot of the dialog.
11. Click **OK**.

The following image shows the **Web Part Zone Properties** dialog:

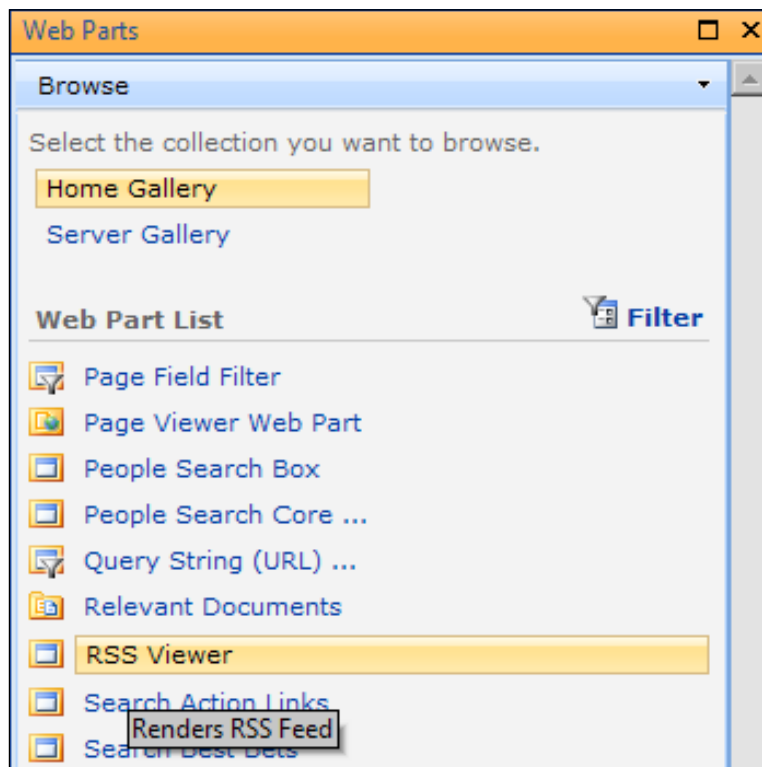


Inserting a Web Part

Now that we have created our Web Part Zone, we will go ahead and add a Web Part inside the new zone. To do this, we simply click and drag the **RSS Viewer** Web Part from the **Web Parts** task pane into the RSS Feeds Web Part zone that we just created.



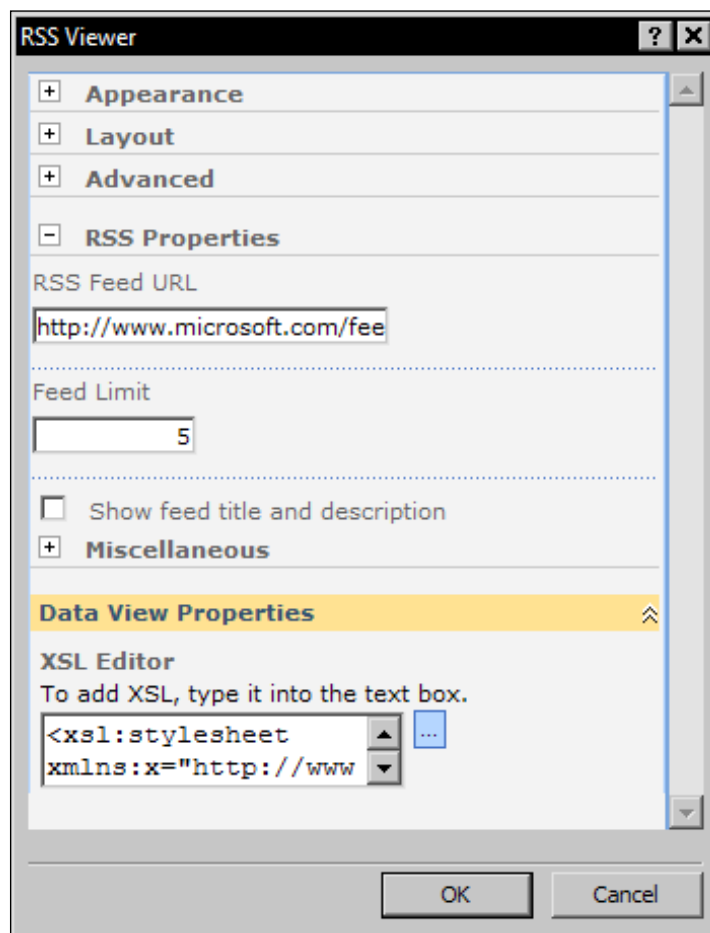
It should be noted that the RSS Viewer Web Part is only available on MOSS. It is not included with WSS.



Once we have done this, the Web Part will display a message that it is not bound to a feed. We can remedy this by specifying an RSS feed for it to display.

Right-click on the Web Part we have just placed in the designer and select **Web Part Properties** from the shortcut menu. We are going to use our Web Part to display the contents of an interesting feed that Microsoft's SharePoint team publishes. To do this, all we need to do is add the following URL into the **RSS Feed URL** field:

`http://www.microsoft.com/feeds/msdn/en-us/sharepoint/rss.xml`



Once we have clicked **OK**, our Web Part is ready to display the contents of the feed on our web page. Do not worry if SharePoint Designer gives you a vague message about a non-specific data source error; simply save the page and point your browser to the `feeds.aspx` page we created. You should then be able to view the feed contents and drill down into each of them for more detail.



We could of course use any RSS feed, but this one seemed like a suitable example. Feel free to add other RSS Viewer Web Parts to the Web Part Zone so that this page contains other useful SharePoint resources. For example, you may wish to add one displaying the contents of `http://www.2f3.com/xml/sharepoint.asp`.

In Chapter 10, we will see how to add other SharePoint Web Parts when we integrate our site with Exchange Server.

Adding Graphs

It is also possible to add components that are not part of the standard SharePoint installation. We will use a Web Part that will display order information graphically. To allow us to do this, we will first of all add a new list to our site that will contain the order data:

1. Open the **Data Source Library** task pane.
2. Maximize the **SharePoint Lists** category within the task pane, if it is not already open.

3. Click once on the `Lists` folder in the **Folder List** task pane. We do this to specify that our new list should be created inside this folder.
4. Click on the **Create New SharePoint List...** link in our **Data Source Library** task pane.
5. Click on **Custom List** in the central column of the dialog box.
6. Specify `OrderTotals` as the list name.
7. Click **OK**.

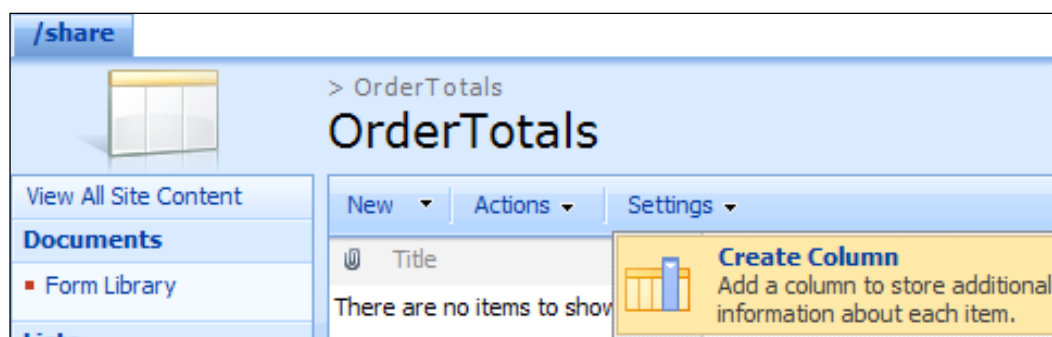
Now that we have created our new list, we can add a couple of fields to it:

1. Browse to our list using your favorite web browser (e.g. `http://olmec/share/Lists/OrderTotals/`).
2. Click on **Settings | Create Column**.
3. Give your column the name **sMonth** and ensure that the type of information for our column is set to **Single Line of Text**, then click **OK**.
4. Click on **Settings | Create Column** once more.
5. Give your column the name **iOrders** and ensure that the type of information for our column is set to **Number**, then click **OK**.



Notice that I use the prefix **s** in my variable names to denote that it is a string and the prefix **i** to denote the use of an integer.

The following image shows the **Create Column** button that we select during this process:



Now that we have set up our list, let's add some data to it. Clicking on the **New** button in our site allows us to add a new record. I have added examples for the first half of 2008. Note that `iOrders` does not mean anything specific in our example. In real life, it could be the number of distinct orders, the number of products ordered, or the total value of the orders.

> OrderTotals > New Item

OrderTotals: New Item

OK Cancel

Attach File | Spelling... * indicates a required field

Title *	January 2008 Orders
sMonth	Jan 08
iOrders	980

Please go ahead and add the following six records to your list (note that February was a particularly boozy month, perhaps because of Valentine's Day):

/share

> OrderTotals

OrderTotals

New Actions Settings

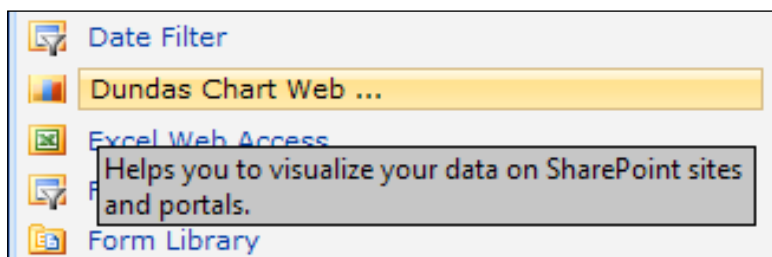
	Title	sMonth	iOrders
	January 2008 Orders ! NEW	Jan 08	980
	February 2008 Orders ! NEW	Feb 2008	1,600
	March 2008 Orders ! NEW	Mar 08	867
	April 2008 ! NEW	Apr 08	925
	May 2008 Orders ! NEW	May 08	845
	June 2008 Orders ! NEW	Jun 08	1,132

Now that we have exposed our data as a list, we can go ahead and display it in a graph. The graphing Web Part we are going to use is called **Dundas Chart for SharePoint**. This component is fairly pricey, but thankfully, we can download a free evaluation copy at www.dundas.com. The evaluation has no limit on how long you can use it for (it has a watermark on it to differentiate it from the full price version). If you would prefer to use a free product, then Ian Morrish's Chart Web Part is worth a look. It can be downloaded from <http://cid-62b84429c3a8a991.skydrive.live.com/browse.aspx/SharePoint>.

Once you have downloaded and installed the evaluation software on your SharePoint server, you will be able to access the component in your Web Part Gallery. When installing the Web Part, ensure that you select the MOSS option.

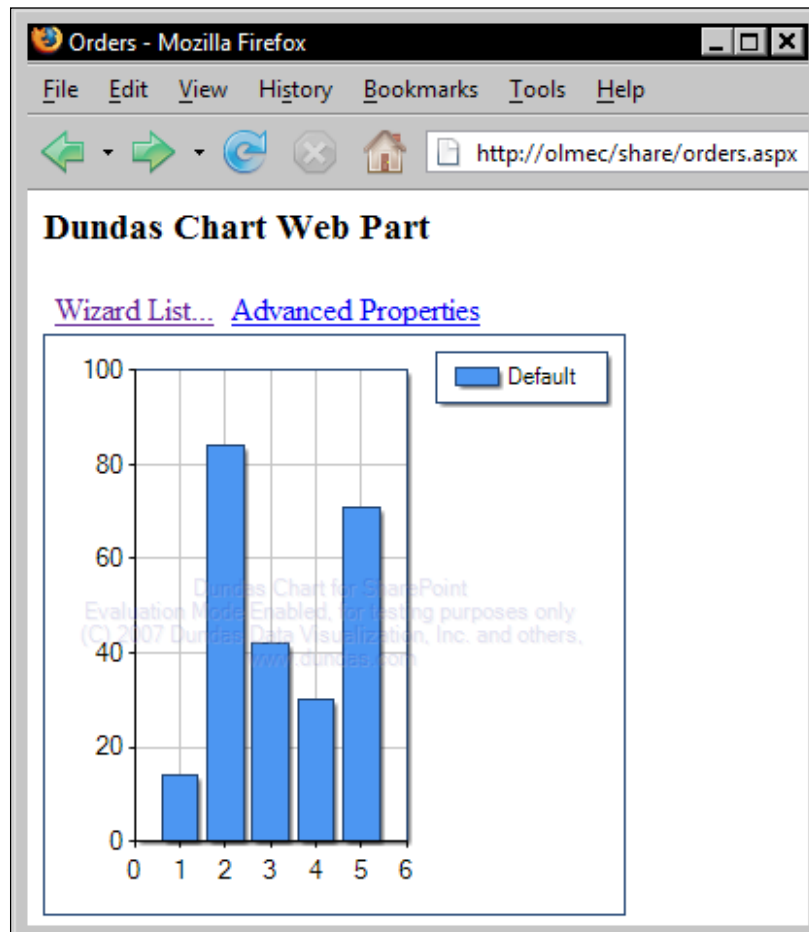
We will use these pretty graphs to display some order information on a new page in our share site. Go ahead and create a new page called `orders.aspx` and add a new Web Part Zone called Order Graphs to the page. The only difference with this new Web Part Zone will be that we should specify that the layout of the Web Parts is **Side-by-side** rather than **Top-to-bottom**.

Once we have created our Web Part Zone, click and drag the **Dundas Chart Web Part** from the **Web Parts** task pane into our Order Graphs Web Part Zone.



With this particular Web Part, we do not use the **Web Part Properties** dialog in SharePoint Designer to configure the chart. Instead, let's save our page and browse to it using our browser (e.g. <http://olmec/share/orders.aspx>).

We can customize the appearance of our graph and connect to our data by clicking on the **Wizard List...** link that appears in the browser window above the example graph.



The **Customize Your Chart** link in the Chart Wizard List allows us to customize the look of our chart. For this example, I have selected the **3D Column** chart type. Great fun can be had tweaking the chart appearance. I have added transparency of 50% and a Flash animation (**Growing One By One**) to my chart.

Once you are done making your chart look pretty, it is time to connect it to our data source. This can be done by clicking on the **Connect Chart To Data** link in the Chart Wizard List:

1. In step 1, we will choose to **Connect to A Site Definition List**.
2. In step 2, we select `OrderTotals` from the **List** drop-down.
3. We could filter our list in step 3, but we will not do that in our example.
4. In step 4, the wizard automatically suggests that we would like to use our order totals (`iOrders`) on the Y-axis and our long title (`Title`) on the X-axis. We will change our X-axis to use our shorter month titles (`sMonth`) so that they fit on the page better.

Once we click **Finish**, our graph is displayed for us.

Summary

In this chapter, we learned what Web Parts are and how they can be added to our site (with the help of examples, where we added an RSS Viewer and a chart to our site). We also learned about the benefits of grouping our Web Parts within Web Part Zones.

In the following chapter, we will examine how to use similar controls, specifically the ASP.NET user controls.

9

Using ASP.NET Controls

As SharePoint and the underlying .NET Framework developed over the years, more readymade controls have become available for us to use in our applications. Such controls allow us to drop common, and often complex, features into our site. For example, if we would like to add user registration and login features (often referred to as *personalization and membership*) to our site, it is a chore to build everything we require (registration form with validation, login box, password reminder, change password, etc.) each time we build a site. Microsoft has built these controls for us so that we can easily add them to our site. In this chapter, we will examine the use of these controls.

Very occasionally, we find that we are not able to do what we would like to do with the SharePoint configuration that comes "out of the box". A common example of this is using forms authentication in our site. Later on in the chapter, we will also make some configuration changes to SharePoint and associated products, such as Internet Information Services (IIS), SQL Server, and Visual Studio, to demonstrate how forms authentication can be enabled.

ASP.NET Controls

We can view the controls that are available by opening up the **Toolbox** task pane and maximizing the **ASP.NET Controls** option. Within this option, there are six categories of controls. Of these, we can ignore the **WebParts** option because that is simply a link to the Web Parts task pane.

It should be noted that it is not possible to use ASP.NET WebParts in SharePoint 2007 sites. It is, however, possible for developers to create their new user controls as SmartParts (see <http://www.codeplex.com/smartpart>) to allow them to be included within WSS and MOSS sites.

The five remaining categories are:

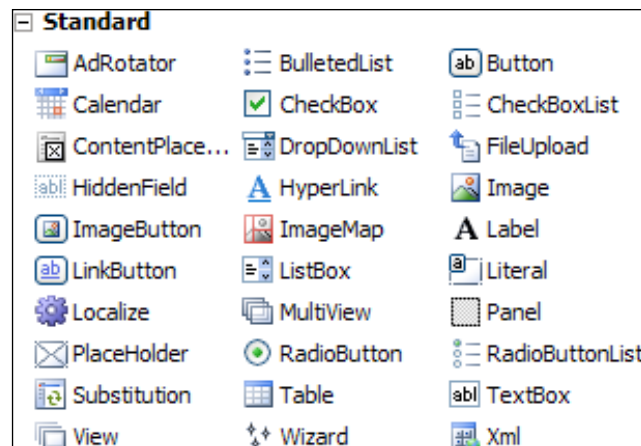
Standard Controls

This contains a large number of controls. Most of these are simple, for example:

- The **Image** control, which displays a picture.
- The **Label** control, which displays text.
- The **Hyperlink** control, which we will use to link from our homepage shortly.

In addition, we have standard controls that are somewhat more exciting:

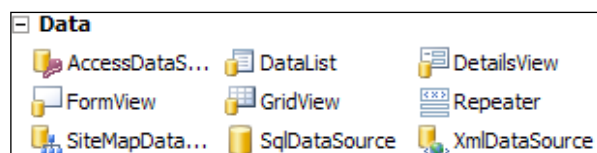
- The **FileUpload** control allows files to be uploaded to the server from the user's computer.
- The **Wizard** control allows a multi-step form to be added to our site. At each step of the wizard, the user is asked for different information.
- The **Xml** control allows data from an XML document to be displayed on our page (although you may find it easier to use a Data View to display the contents of your XML file).
- The **Calendar** control, which we will also add to our site later in this chapter.



Data Controls

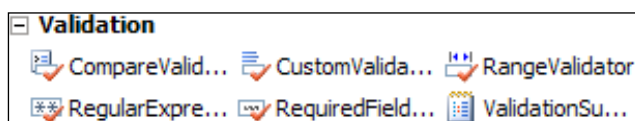
The data controls allow us to connect to various different data sources, display the results on our page, and update the data in the source.

Because SharePoint already provides us with access to data sources in the **Data Source Library** task pane, we can safely ignore these controls.



Validation Controls

The validation controls allow us to validate information input by the user. We will see this in practice towards the end of the chapter, when we add validation to a form.



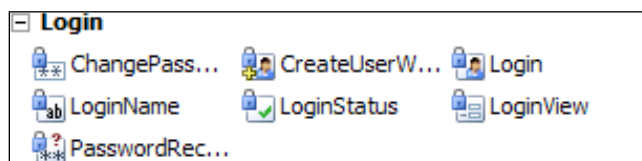
Navigation Controls

The navigation controls provide us with three different methods of helping users navigate our site. The **Menu** and **TreeView** controls would allow users to browse to a particular page, while the **SiteMapPath** control displays breadcrumbs that show their position within the page hierarchy. We will see an example of the **Menu** control later in this chapter.



Login Controls

The login controls allow us to take advantage of ASP.NET's extensive membership provider system. The benefit of this is that we no longer need to manually create all the elements required by a login system (i.e. registration, login, password reminder, etc.). We will also look at this in detail in this chapter.



Further information about virtually all of the ASP.NET server controls, including examples and code snippets, is available in the ASP.NET Control Reference in the .NET Framework SDK, which is available at <http://quickstarts.asp.net/>.

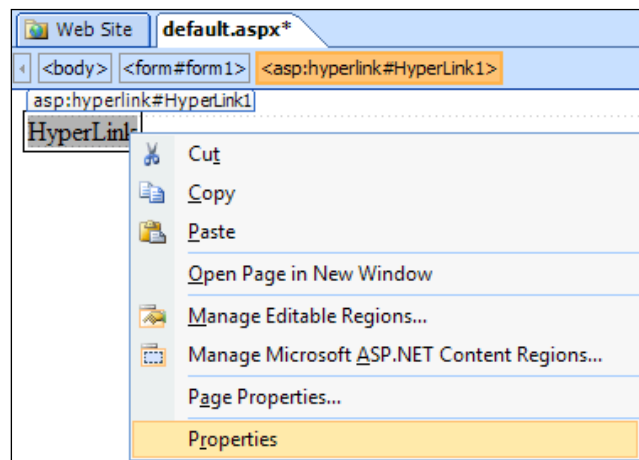
Adding a Simple Control

In this chapter, we will be using various server controls in our pages.

We will start by adding one of the simplest controls to our homepage:

1. Open the share site in SharePoint Designer.
2. Open the `default.aspx` page in Design view.
3. Make sure that the **Standard** controls are visible in the **Toolbox** task pane.
4. Click on the **HyperLink** control and drag it onto our page.
5. Right-click on the control that we have just placed and select **Properties** from the shortcut menu. This will open the **Tag Properties** task pane to the left of our design window.
6. Type View Orders Graph into the **Text** field in the **Tag Properties** task pane.
7. Click on the ellipsis to the right of the **NavigateUrl** field and select `orders.aspx` from the list of pages.
8. Finally, save the page.

The following image shows the **Properties** shortcut menu that we see as we work through this process:



Now, when we open <http://olmec/share/> in our browser, we see that the default page links to the Order Graphs.

The Menu Control

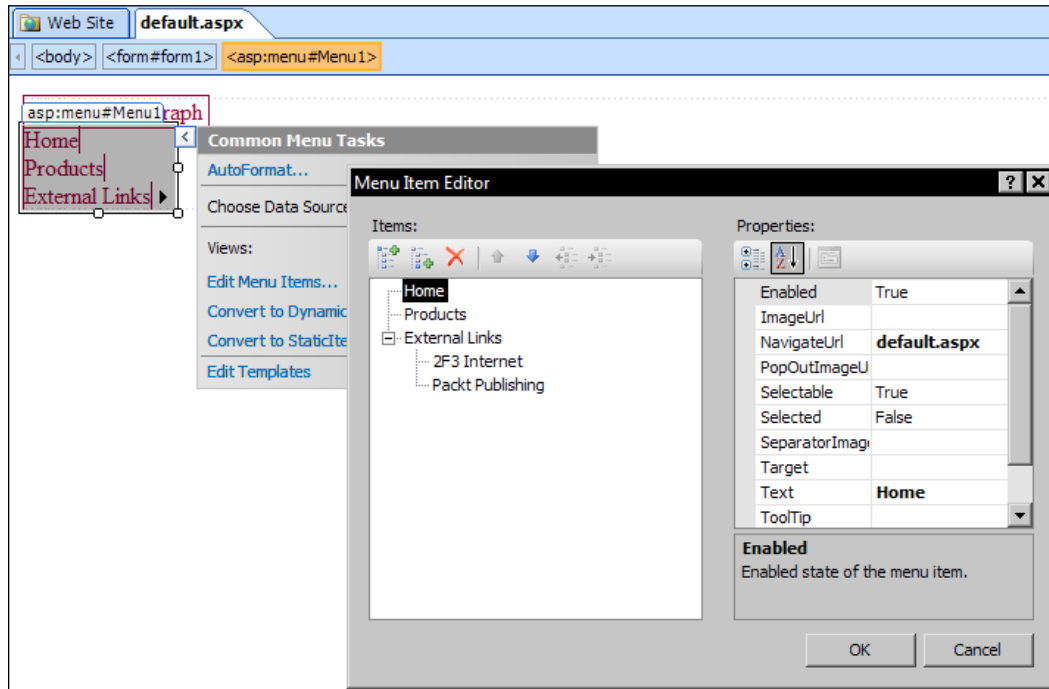
One of the most popular controls at our disposal is the Menu control. This allows us to add a navigation menu to the pages of our site to allow users to move from one page to another.

The Menu control is particularly useful because it can be populated by a data source. In this way, we could automatically display all the wines in our product range (we could even have a red wine menu button with all the red wines as a submenu and likewise with the white wines).

For our example, we will stay with the `default.aspx` page and add a menu control beneath our link to the graph page:

1. Drag a **Menu** control from the **Navigation** group in the **Standard** controls in the **Toolbox** onto the `default.aspx` page beneath the existing link.
2. The **Common Menu Tasks** options will automatically open for us. We will click on **Edit Menu Items**.
3. This brings up the **Menu Item Editor**, which allows us to add our links to the menu.
4. We will add our first link by clicking on the **Add a root item** button (which is the first one on the small menu bar).
5. We can then set the properties for the link. We will do this by specifying `default.aspx` as our **NavigateUrl** and `Home` as our **Text**.
6. We can add more root menu items by repeating steps 4 and 5 as often as we wish. We will add a further root item with `products.aspx` as our **NavigateUrl** and `Products` as our **Text**.
7. Before we close our **Menu Item Editor**, we will add a couple of external links as child items of another root item. We will begin this by adding a third root item and giving it the **Text** `External Links`. There is no need to add a **NavigateUrl** this time.
8. With the `External Links` node selected, we will then click on the **Add a child item** button (the second one on the small menu bar) and give this new item `http://www.2f3.com` as our **NavigateUrl** and `2F3 Internet` as our **Text**.
9. Taking care to select the `External Links` node again, we will add another child item with `http://www.packtpub.com` as our **NavigateUrl** and `Packt Publishing` as our **Text**.
10. Finally, we will click **OK** and then save our page.

This image shows the **Menu Item Editor** that we use to add our menu items:



When we view the homepage of our site, we will see the new menu items available to the user. Note that the `External Links` link has an arrow beside it to notify the user that it contains child links. It should also be noted that this control would usually be placed on every page of the site rather than just one page.

The Calendar Control

One of the nicest controls that SharePoint provides us with is the Calendar control. This control makes it easy for users to add a date into a form field. You will no doubt have seen these controls in use when booking a flight or a hotel room online. They are especially useful in helping to prevent confusion that can arise from the different way that North America and Europe treat dates (i.e. month/day/year as opposed to day/month/year).

We will add a Calendar control to a new page and use it to populate a text field:

1. Create a new ASPX page and call it `birthday.aspx`.
2. Type `My birthday is:` into the form on the page.

3. Drag a **TextBox** control from the **Standard** controls in the **Toolbox** onto the page to the right of the text we have just added.
4. Right-click on the **TextBox** control and select **Properties** from the shortcut menu.
5. Change the **ID** of the TextBox from `TextBox1` to `txtBirthday`.
6. Next, we will drag a **Calendar** control from the **Standard** controls in the **Toolbox** onto the page.
7. Right-click on the **Calendar** control and select **Properties** from the shortcut menu.
8. Change the **ID** of the Calendar from `Calendar1` to `calBirthday`.

Now that we have added the controls to our page, we must make the Calendar control interact with the TextBox. The following code will place the date into the text box when the calendar is used. We will put it at the top of our page in Code view:

```
<%@ Page Language="C#" %>
<script Language="C#" runat="server">
private void calBirthday_SelectionChanged (object sender, System.
EventArgs e) {
    txtBirthday.Text = calBirthday.SelectedDate.ToShortDateString();
}
</script>
```

We must also edit the Calendar control so that it fires that event:

```
<asp:Calendar runat="server" id="calBirthday" OnSelectionChanged="calB
irthday_SelectionChanged"></asp:Calendar>
```

We can now view the page in our browser (e.g. <http://olmec/share/birthday.aspx>). Notice that when we save our page, SharePoint will give us the following error message:

An error occurred during the processing of /share/birthday.aspx. Code blocks are not allowed in this file.

This is because SharePoint is protective about what code it allows us to run. By default, it disables server-side scripts. This is not a problem though, because we can override the prohibition. To do so, we require a small lesson in editing the `web.config` file.

Editing the web.config File

The `web.config` file specifies the settings not just for the share site but also for other sites sharing the same port (port 80) on our SharePoint server. It should be noted that any mistakes we make could stop SharePoint from working. Even if you are working using administrator permissions, it is recommended that you check with your server administrator that it is OK for you to edit this file. It is always a good idea to do this before you edit it, rather than after you have edited it and the sites have crashed!

By default, we will find the file in `C:\Inetpub\wwwroot\wss\VirtualDirectories\80` (where 80 is the port number of our SharePoint site) on our server. Before we begin tinkering with this file, it is a good idea to make a backup of it (e.g. `web.config.bak`).



You may find using Remote Desktop the easiest way to access your server to edit this file.

We will change the following lines of code in our `web.config` file:

```
<PageParserPaths>
</PageParserPaths>
```

to:

```
<PageParserPaths>
  <PageParserPath VirtualPath="/share/birthday.aspx"
    CompilationMode="Always" AllowServerSideScript="true" />
</PageParserPaths>
```

We could have left the `VirtualPath` blank so that server-side scripts are allowed in all SharePoint sites, but limiting it in this way ensures that only our birthday page has script access.

Once this change to the configuration file has been saved, the calendar on our birthday page will operate correctly. When we click on one of the dates, it will be pasted into the `txtBirthday` field.

While we are tinkering with the `web.config` file, you may find it useful to change the following line:

```
<customErrors mode="On" />
```

to:

```
<customErrors mode="Off" />
```

Making this change will provide us with additional information in the run-time error messages, which will help us troubleshoot our code. For security reasons, this should be returned to the On mode when our site is ready to go live.

Validating Our Forms

We can add validation controls to our form that will check the information input by the user. If they input an incorrect value, then an error message will be displayed.

There are several different types of validation controls. Each performs a specific type of validation when the form is updated.

We will add a **RangeValidator** to our form to check that the birthday we have selected is credible:

1. Click and drag a **RangeValidator** control onto our form.
2. Right-click on the validation control and select **Properties** from the shortcut menu.
3. Set the **ControlToValidate** to **txtBirthday**.
4. Change the **Type** from **String** to **Date**.
5. Set the **MinimumValue** to **1/1/1880**.
6. Set the **MaximumValue** to **1/1/2009**.
7. Set the **ErrorMessage** to **Please select a credible birthday**.

If we return to our birthday page in our web browser, we see that if we enter a spurious date such as 1/1/1850 and then press *Enter* to cause a postback, the validation kicks in and we are greeted with an error message.

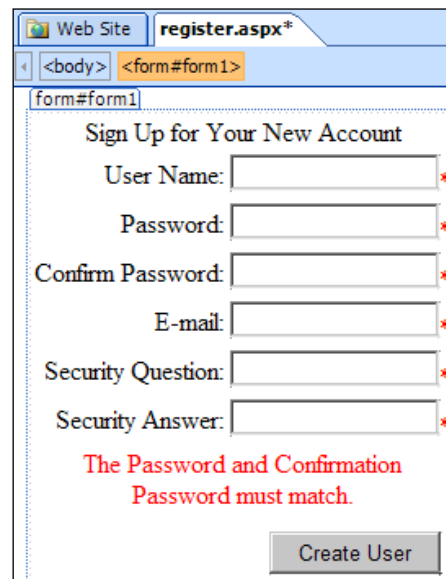
Creating a Login Feature

We know that SharePoint allows users to log in using Windows Authentication (i.e. their network login). What if we would like to provide a login to our site to allow users from outside the network to log in? This is a very common requirement, yet with SharePoint 2003, it proved highly problematic because there was no out-of-the-box alternative and the workarounds that programmers came up with had undesirable side effects.

The good news is that with SharePoint 2007, we can specify the Membership Provider that we would like our site to use. As you will see, implementing this functionality still requires a fair amount of configuring the various servers in our farm, but the solution we end up with is more elegant than it was with SharePoint 2003.

We are going to use the `AspNetSqlMembershipProvider` in conjunction with the ready-made ASP.NET Login controls to allow our users to create and use a username and password. This login will be stored in our SQL Server database as a new user account in the ASP.NET 2.0 membership system.

Let's go ahead and create a new ASPX page, which we will call `register.aspx`. We then click and drag the **CreateUserWizard** control from the **Login** category of our ASP.NET controls and drop it in the form on our new page. SharePoint Designer automatically creates a fairly ubiquitous registration form for us:



The screenshot shows a web browser window with the title 'Web Site' and the address bar displaying 'register.aspx*'. The page content is a registration form titled 'Sign Up for Your New Account'. The form contains six text input fields, each preceded by a label and followed by a red asterisk: 'User Name:', 'Password:', 'Confirm Password:', 'E-mail:', 'Security Question:', and 'Security Answer:'. Below these fields, a red error message reads 'The Password and Confirmation Password must match.' At the bottom right of the form is a button labeled 'Create User'.

As with our WebParts, we can open the list of Common Tasks by clicking on the form and then clicking on the right-hand arrow that appears in the top right-hand corner of the control. This allows us to control our **CreateUserWizard** control.

It should be noted that the registration process is highly configurable, allowing us to specify exactly which pieces of information are required from the registrant. It is also possible to add as many stages to the registration process as we wish.

If you agree that this form looks uninspiring, then go ahead and give it a pre-defined formatting scheme by clicking on the **AutoFormat...** link and taking your pick from the five different schemes. I have chosen the **Classic** scheme for my example.

If you save this page, then view it in your browser (e.g. `http://olmec/share/register.aspx`), you will get a SharePoint error when you try to submit the form. The error occurs because we have not specified where SharePoint should store the user data. Let's go ahead and set up our data store.

Configuring SQL Server

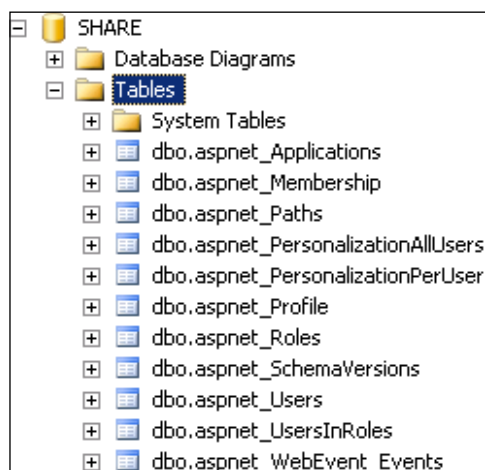
To provide a home for the user data, we will need to set up specific tables in our SQL Server. Thankfully, Microsoft has provided us with a tool that will do this for us. You will find this `aspnet_regsql.exe` program in the relevant .NET Framework folder on your server (e.g. `C:\WINDOWS\Microsoft.NET\Framework\v2.0.50727`). Running through the wizard steps in this tool is simple:

1. Click **Next >**.
2. Select **Configure SQL Server for application services**.
3. Click **Next >**.
4. Type in the name of your server into the **Server** box. Note that this may be more than just the name of the server that your database is stored on and should also include the instance name (e.g. `OLMEC/OFFICESERVERS` or `.\SQLEXPRESS`).
5. Select your database from the drop-down list. In my example, it is called `SHARE`.
6. Click **Next >**.
7. Click **Next >** on the summary screen and **Finish** on the confirmation screen.



Note that SQL Server must be configured to allow remote connections.

If we peek inside our database, we will see that the following tables were created by the wizard:



The `aspnet_Users` table holds information about users (e.g. `UserID` and `UserName`), while the `aspnet_Membership` table holds additional information about the user (such as `Password`, `Email`, `PasswordQuestion`).

The next thing that we need to do is to set up the authentication so we can connect to our SQL database:

1. Open SQL Server Management Studio.
2. Right-click on your chosen database server in Object Explorer.
3. Select **Properties** from the shortcut menu.
4. Select the **Security** option in the left-hand pane of the **Server Properties** dialog.
5. Change the **Server authentication** to **SQL Server and Windows Authentication mode**.
6. Click **OK**.
7. Open the **Security**, then the **Logins** folders in the Object Explorer.
8. Right-click on the `sa` user.
9. Select **Properties** from the shortcut menu.
10. Choose a new password that conforms to your password policy (e.g. `Bunnyrabb1t`) and type it into the **Password** and **Confirm Password** fields.
11. Click **OK**.
12. Right-click on the chosen database server in Object Explorer again.
13. Select **Restart** from the shortcut menu.
14. Click **Yes**.

Adding Our First User

In order to access our application, we will add a new user to our tables. Unfortunately, this is not as straightforward as one would hope.

The easiest way to do this would be to drop the **Login** control onto our `register.aspx` page and then select **Administer Website** from the **Common Login Tasks** list. This should in turn launch the Web Site Administration Tool (WAT), which would allow us to add our user. Unfortunately, the WAT is only available in Visual Studio 2005 (which also uses this control).

In order to use the WAT to add our new user, we must download Microsoft Visual Web Developer 2005 Express Edition from <http://www.microsoft.com/express/> and install it on our workstation.

Once we have installed Visual Studio, we come across another issue that we must overcome. We cannot simply add a new site to our SharePoint server because the default site is already being used for SharePoint sites. We must set up a new virtual site in Internet Information Server (IIS) and connect Visual Studio to that site using a different port number.

Adding a New Virtual Server

As we have mentioned, it will not be possible to use Visual Studio to set up a new site because SharePoint is already using port 80. We will need to manually set up a new site on our SharePoint server that uses a different port number:

1. Go to **Start | All Programs | Administrative Tools | Internet Information Services (IIS) Manager**.
2. Expand the server tag (e.g. Olmec).
3. Right-click on **Web Sites** and select **New | Web Site...** from the shortcut menu.
4. Click **Next >**.
5. Give our site a **description** (e.g. adduser).
6. Click **Next >**.
7. Set the **TCP port number** to 8080.
8. Click **Next >**.
9. Click **Browse** and select the location in which we would like to create a new site (e.g. c:\inetpub\wwwroot\), then create a new folder called adduser.
10. Click **OK**.
11. Click **Next >**.
12. Check **Run Scripts (such as ASP)**.
13. Click **Next >**.
14. Click **Finish**.

Adding a Host Header

Visual Studio will not recognize an address with a port number in it as a valid virtual path (see <http://connect.microsoft.com/VisualStudio/feedback/ViewFeedback.aspx?FeedbackID=106146> for more information), so before we can connect to our new site with Visual Studio, we must set up a host header using IIS:

1. Right-click on our adduser site and select **Properties**.
2. With the **Web Site** tab selected, click on the **Advanced** button.
3. Click **Add...**
4. Type **8080** into the **TCP port**.
5. Type in a **Host Header value** (e.g. `adduser.olmec.malbet.local`).

Adding an A Record

In order for traffic to find the correct server, we must add an (A) name to our DNS server. We will do the following on the server that looks after our network's DNS:

1. Go to **Start | All Programs | Administrative Tools | DNS**.
2. Expand the host name server.
3. Expand the **Forward Lookup Zones**.
4. Right-click on the zone you would like to administer (e.g. `malbet.local`).
5. Select **New Host (A)...**
6. Type `adduser.olmec` in the **Name** field.
7. Type the IP address of your SharePoint server into the **IP address** box.
8. Click **Add Host**.

It will take a short while (e.g. 10 minutes) for the DNS changes to propagate through the network.

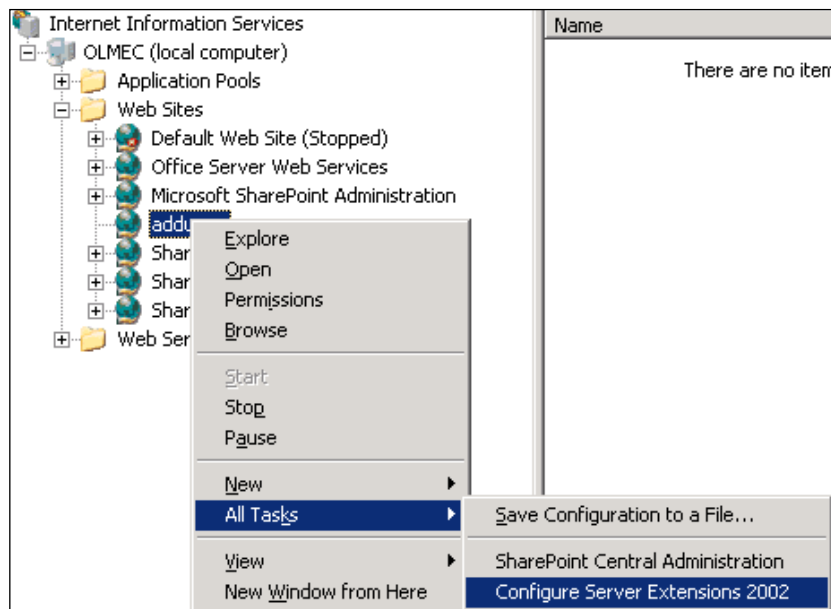
Extending the Virtual Server

We will be connecting to our new virtual server with Visual Studio over HTTP. For this to work, FrontPage Server extensions must be installed and enabled on our server.



If FrontPage Extensions are not installed on the server, this can be done by going into **Control Panel | Add or Remove Programs | Add/Remove Windows Components | Application Server | Details | Internet Information Services | Details | FrontPage 2002 Server Extensions**.

You may have thought that having done this would allow the FrontPage extensions to work, but there is one more step we must perform before they will work. We must extend our virtual server. To do this, right-click on our new virtual server in the IIS server tree and select **All Tasks | Configure Server Extensions 2002**:



This brings up a simple form that allows us to extend the server with the FrontPage extensions. Simply click **Submit** and watch the very funky cogwheel graphic while we wait.

Administration | Help

FrontPage Server Extensions 2002

Extend virtual server with FrontPage Server Extensions 2002

Use this page to extend a virtual server with FrontPage Server Extensions 2002.

Server to Extend The following virtual server will be extended.	Virtual server name: adduser
Administrator Specify an administrator account for this virtual server.	Administrator user name: MALBET\Administrator

Submit
Cancel

If we are not logged into our workstation with the username that we specified in the extend form, then we must add our username in the **Manage users** section of the administration site.

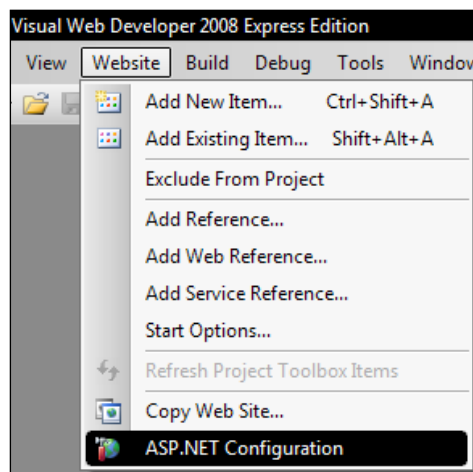
Using Visual Studio's Web Site Administration Tool

It is now possible to use Visual Studio to set up a new site on our new website:

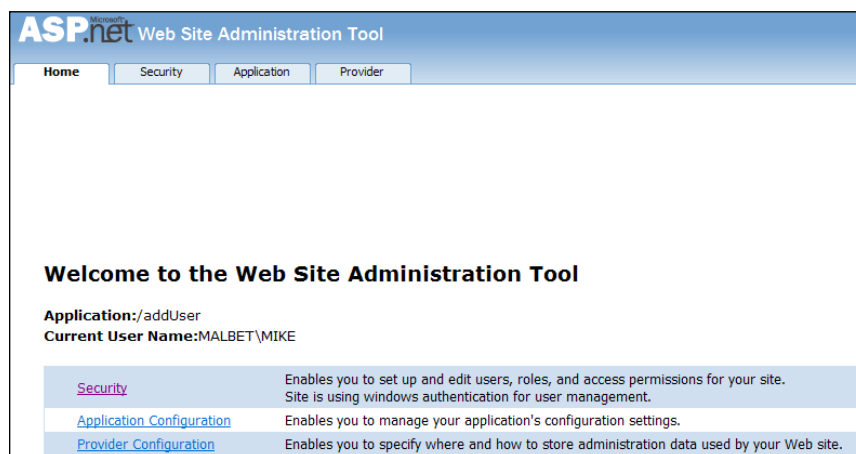
1. Go to **File | New Web Site**.
2. Select **ASP.NET Web Site**.
3. Set Location to **HTTP**.
4. Specify **http://adduser.olmec.malbet.local/** as the location.
5. Click **OK**.

Your new blank website will be set up on the server for you. Notice that you can also view the new blank page by pointing your web browser at <http://olmec:8080/>.

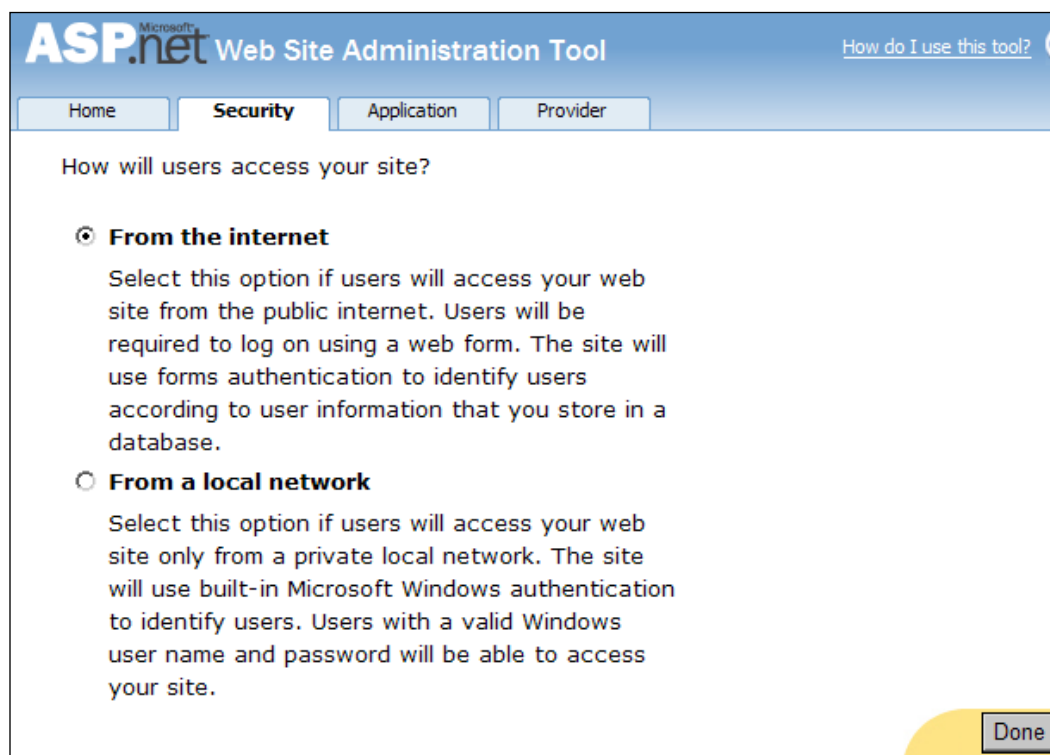
Now that we have our new website set up, we can access the WAT by selecting **Website | ASP.NET Configuration**.



Once you have the WAT open, you will want to click on the **Security** tab:



Once in the security page, you will be able to go to **Select authentication type** and change the authentication type from the **local network** (i.e. Windows authentication) to the **internet** (i.e. Forms authentication):



Now that you have set up the authentication type, you can click on the **Create User** link on the **Security** page and create your user:

The screenshot shows the 'ASP.NET Web Site Administration Tool' interface. The 'Security' tab is selected. The page contains instructions on managing security settings, a link to the security Setup Wizard, and a table with links to manage users, roles, and access rules.

Users	Roles	Access Rules
Existing users: 0 Create user Manage users Select authentication type	Roles are not enabled Enable roles Create or Manage roles	Create access rules Manage access rules

Thankfully, setting up a new user in the membership provider will be far easier in Windows Server 2008, which will allow the "Add User" feature in IIS7 to be used.

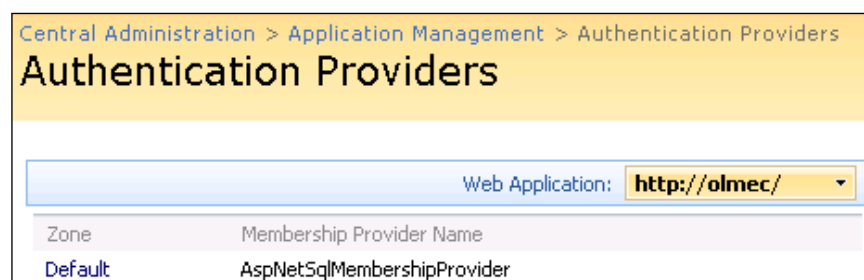
Changing the Authentication Provider

The next thing that we need to do is specify that our site should use forms authentication:

1. Open the SharePoint Central Administration Website (SCAW) in our browser (e.g. <http://olmec:44936/>).
2. Click on **Application Management**.
3. Under the **Application Security** heading, click the **Authentication Providers** link.
4. Click on the **Web Application** drop-down list and select **Change Web Application**.

5. Click on the site we would like to change (e.g. SharePoint - 80).
6. Click on our zone (e.g. Default).
7. Change the **Authentication Type** from **Windows** to **Forms**.
8. We will also check the **Enable anonymous access** box for now, so that we are not left with the Catch 22 situation of not being able to log in to register because we do not have a login yet.
9. Type **AspNetSqlMembershipProvider** into the **Membership provider name** field.
10. Click **Save**.

You should now see the sites on port 80 (the default port) correctly configured to use SQL Server as our membership provider:



It is important to note that changing the authentication type of our sites in this way means that SharePoint Designer will no longer be able to connect to them. When we need to edit the site in SharePoint Designer, we will need to switch the authentication type back to Windows authentication.

Return to the **Application Management** screen in your SCAW and select **Policy for Web Application**.

1. Click the **Add Users** link.
2. Select the zone our site is in (Default).
3. Click **Next >**.
4. We are then able to select the user(s) from our user database and the permissions we would like to grant them.
5. Once we have finished, click **Finish**.

Return to the web.config File

The next thing that we need to do is add the following code to the <configuration> section of the web.config file.



Note that the <connectionStrings> section cannot be placed before the <configSections> section.

This will allow our SharePoint site to connect to our user database:

```
<connectionStrings>
  <remove name="LocalSqlServer" />
  <add name="LocalSqlServer" connectionString="data
source=OLMEC/OFFICESERVERS; initial catalog=aspnetdb; user id=sa;
password=Bunnyrabb1t;"/>
</connectionStrings>
```

or if you prefer to use integrated security, you can use this:

```
<connectionStrings>
  <remove name="LocalSqlServer" />
  <add name="LocalSqlServer" connectionString="data source= OLMEC/
OFFICESERVERS;Integrated Security=SSPI; AttachDBFilename=|DataDire
ctory|aspnetdb.mdf; User Instance=true" providerName="System.Data.
SqlClient"/>
</connectionStrings>
```

Once these changes have been made, it is a good idea to restart the SharePoint Server (or stop and start the site in IIS).

Additional Configuration Tweaks

The following can also be added to the <system.web> section to specify the strength of the password policy used (by default, it is 7 characters, including one non-alphanumeric character, so we are allowing it to be a bit weaker):

```
<membership defaultProvider="SHAREProvider">
  <providers>
    <add name="SHAREProvider" type="System.Web.
Security.SqlMembershipProvider" connectionStringNam
e="ConnectionString" minRequiredPasswordLength="5"
minRequiredNonalphanumericCharacters="0"/>
  </providers>
</membership>
```

While we are on the topic of the Create User control, you might be interested to know that it is possible to automatically send a welcome email to the user by adding a MailDefinition tag between the existing asp:CreateUserWizard tags in the code view:

```
<asp:CreateUserWizard id="CreateUserWizard1" runat="server">
  <MailDefinition
    BodyFileName="NewUserEmail.txt"
    From="welcome@2f3.com"
    Subject="Welcome to the site!"/>
</asp:CreateUserWizard>
```

For the email to be relayed, you will need to specify the SMTP server in the web.config file (and have the SMTP service enabled on the server):

```
<configuration>
  <system.web>
    <authentication mode="Forms"/>
    <smtpMail serverName="Localhost"/>
  </system.web>
</configuration>
```

Summary

In this chapter, we have learned what ASP.NET controls are and which ones are available to us. We also saw how to implement simple controls, menu controls, calendar controls, validation controls, and login controls into our pages.

In the following chapter, we will take this one step further and make use of the controls and Web Parts that are unique to SharePoint.

10

Integrating with Exchange

One of the reasons that I like SharePoint is that it helps remove boundaries between Office and server products, allowing us to provide the user with more of the information they need. There is no better example of this interconnectivity than displaying mailbox items in the user's browser.

In this chapter, we will learn about the opportunities that exist to embed information from our Exchange Server within our SharePoint site. By way of example, we will be adding our tasks list to the site, but it is also possible to add other Exchange information such as calendars, contacts, and of course email.

When adding this functionality, there are a few configuration changes that may be required. We will examine these so that if you meet these hurdles when implementing your solution, they can be easily overcome.

Introduction to Outlook Web Access Web Parts

Over the years, as Microsoft's Exchange Server has matured, the number of access methods that users have to the information stored on the server has increased. The last five years in particular have seen great improvements in the ability to connect to your mailbox using nothing more than a web browser or a mobile phone.

Because the nature of SharePoint is to allow easy access to information and to improve team collaboration, it is only right that SharePoint should increase our options when it comes to being able to access Exchange information.

In this chapter, we will take a look at the Web Parts that are available to us and learn how to add them to our SharePoint site. The Web Parts that we are going to use can be integrated into any ASP.NET application (irrespective of whether it is running on a SharePoint server or not) and make use of Outlook Web Access (OWA), which exposes the Exchange data. You will hear the Web Parts referred to as either Outlook Web Access Web Parts or, less commonly, Exchange Web Parts. There is no difference between the two.

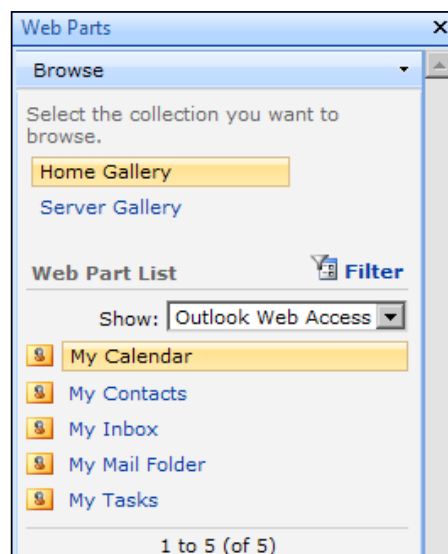
One of the beauties of using Exchange Web Parts is that we can mix and match the views that are displayed on our web page to suit our business needs. If we want to have our company's customer service, sales, and info inboxes all displayed on one page, then Exchange Web Parts allow us to do that.



It should be noted that Exchange integration is not supported by WSS. MOSS will be required if you would like to integrate Exchange information within your SharePoint site.

Viewing the Outlook Web Access Web Parts

We can access our OWA Web Parts by opening the **Web Parts** task pane in SharePoint Designer. We can then filter the Web Part list to only show Outlook Web Access Web Parts by clicking on the **Filter** link and selecting **Outlook Web Access** from the **Show** drop-down list.



You will note that there are five Outlook Web Access Web Parts, each of which allows us to add Exchange functionality to our web page:

- **My Calendar** allows us to display a calendar for an Exchange user. We can specify that the calendar will default to either a weekly or a daily view (although the user can switch views when they are using the calendar). I find the day view particularly useful because when we click on the appointment, it displays the details of the appointment to the right of the calendar, allowing us to change them. When using this Web Part, it should be noted that there is no design-time preview.
- **My Contacts** will display the user's entire contact list. We can again specify the default view as either a phone list or a two-line view. The user cannot change the type of view that they have displayed on the page.
- **My Inbox** does exactly as we would expect, by displaying the emails that are in the root folder of the user's inbox. There are lots of default views for this Web Part. We can select Two-line, messages, or order by sender, subject, or conversation. The user then has the option to arrange their mail by 11 different parameters.
- **My Mail Folder** allows us to display the contents of any private or public folder. There are no default views for this Web Part. When setting up the Web Part, we specify the name of the folder of which we would like to display the contents. If you would like to display the contents of a subfolder, you can specify `myFolder/mySubfolder`.
- **My Tasks** will display the user's task list. We will learn more about this Web Part shortly.

In order for our page to connect to the Exchange Server, we will need to correctly configure our Exchange Server by ensuring that OWA and Forms-Based Authentication are both enabled.

Enabling Outlook Web Access

OWA should be enabled by default. We can verify if OWA is set up on the server by attempting to view our mail in our web browser. If you are using Exchange Server 2007, then the address will be in the format `https://servername/owa/`.

If OWA is not enabled on our server, then we can enable it by using the Exchange Management Console. Managing Exchange Server is outside the scope of this book. For further information about managing OWA, see <http://technet.microsoft.com/en-us/library/aa996373.aspx>.

Enabling Forms-Based Authentication

In order for our OWA Web Parts to display correctly, we must configure our Exchange Server to use forms-based authentication.

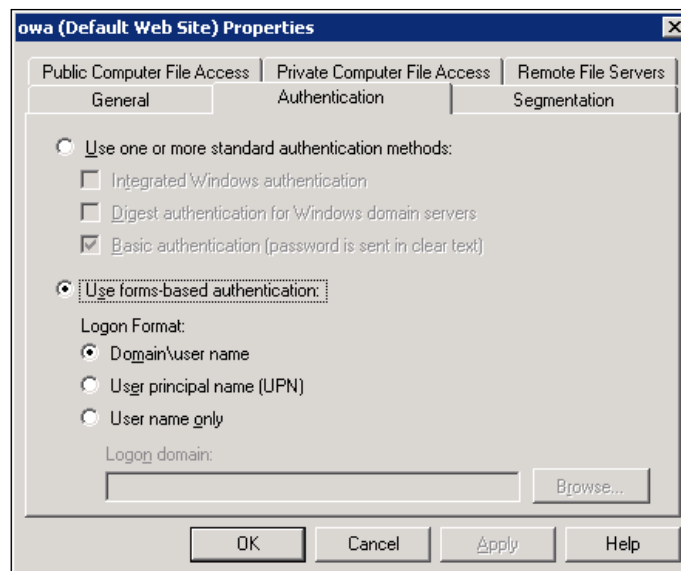


The following instructions are for Exchange Server 2007. The same thing can be achieved using Exchange System Manager in earlier versions of Exchange.

To enable forms-based authentication, do the following on our server:

1. Open the Exchange Management Console.
2. Click on **Server Configuration**.
3. Then click on **Client Access**.
4. Make sure the **Outlook Web Access** tab is selected in the lower middle pane.
5. Right-click on the virtual directory that we would like to edit (the default Exchange 2007 virtual directory is /owa).
6. Select **Properties** from the shortcut menu.
7. Click on the **Authentication** tab.
8. Ensure that **Use forms-based authentication** is selected.
9. Click **OK**.

As we work our way through this process, we will see the following **owa Properties** dialog:





Switching to forms-based authentication will cause integrated Windows authentication (e.g. `https://servername/owa/` or `https://servername/exchange/`) to stop functioning. If you still need to use integrated authentication for internal access, then an additional virtual server will need to be created using Exchange System Manager to allow this.

Integrating the My Tasks Web Part

Once our Exchange Server is correctly configured, it is really very easy to add the Exchange Web Parts to our site.

Let's add one of the Web Parts to a new page on our site. We will choose the My Tasks Web Part to allow us to keep an eye on our task list:

1. Open the share site.
2. Create a new ASPX page called `tasks.aspx`.
3. Click on the **My Tasks** Web Part and drag it onto our new page.

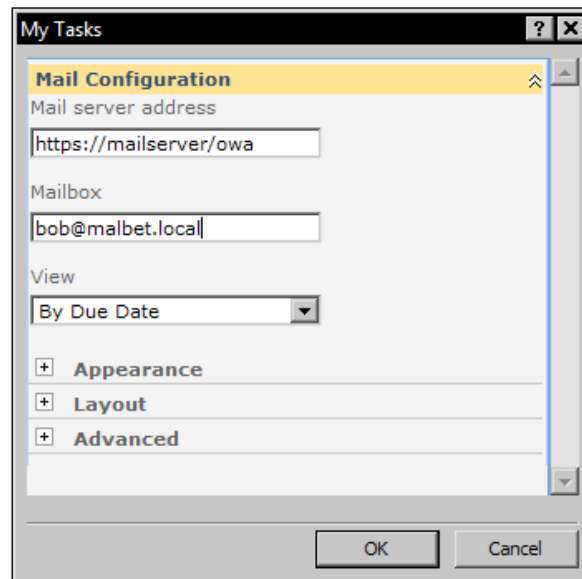
Once the Web Part has been added to the page, it will display the following text:

My Tasks: This Web Part does not have a preview mode when editing the page in design view.

Double-click on the Web Part and the **My Tasks** properties will be displayed so that we can fill in the following properties:

1. The **Mail server address** should be the OWA address. If your OWA access is over a secure connection, then `https` must be specified in the address (e.g. `https://mailserver/exchange`).

2. Add your mailbox address (e.g. bob@malbet.local_or bob@winecompany.co.uk) to the **Mailbox** field. This will, of course, only work if that user has a mailbox on the Exchange Server.



Once this is done, click **OK**. If you have specified reasonable-looking information, then the properties window will disappear. If you have not, then the **OK** button will gray out for a while and the window will remain open.

We can then go ahead and save the page, then open it in our favorite browser (provided that our favorite browser happens to be Internet Explorer 6 or later!) e.g. <http://olmec/share/tasks.aspx>.

Troubleshooting the Error Messages

If you are using a secure connection to display your OWA data, then the chances are that you will get the following error message when you try to view your web page:

Content was blocked because it was not signed by a valid security certificate.

If you do get the message, then select the option to allow blocked content. This message will appear every time you visit the page in a new browser session and will appear for every Exchange Web Part that you have on the page (i.e. if you have three inboxes displayed on the page, you will need to click to allow blocked content three times, each time you open the browser).

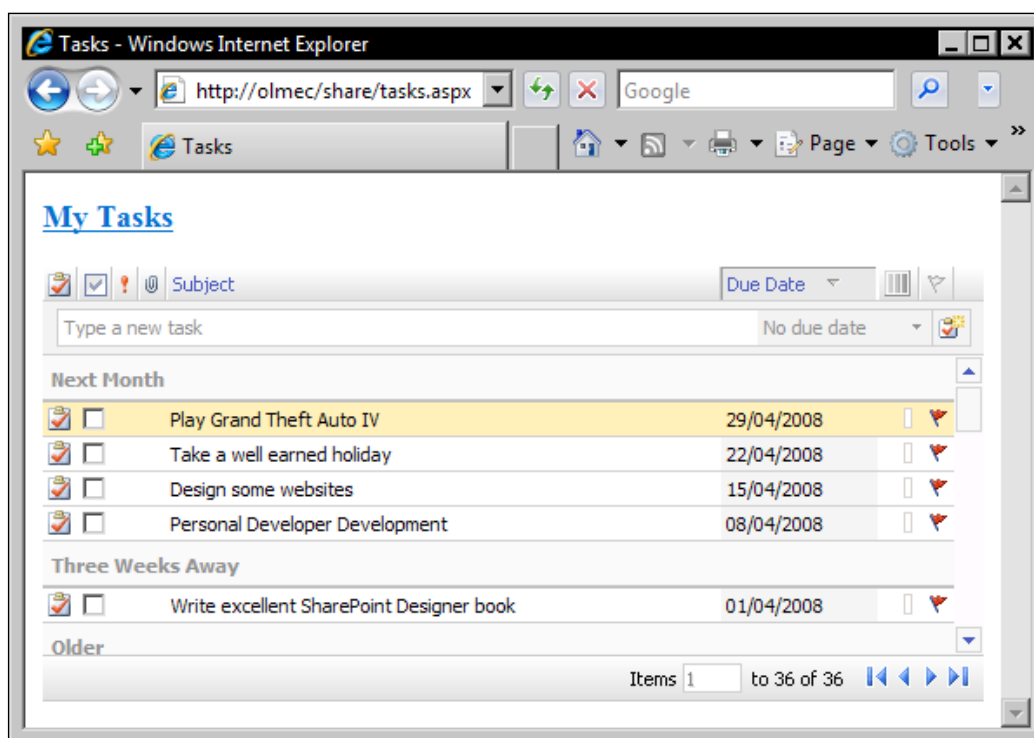
To permanently stop the content from being blocked, you will need to install an SSL certificate on your Exchange Server. An explanation of this is outside of the remit of this book. You will be able to find many helpful articles about how to do this by typing **create and install SSL certificate** into your favorite search engine.

You may also receive a message saying that cookies are disabled. If they are, then you will need to enable them by going into your Internet Options. This can be done in Internet Explorer 7 by going to **Tools | Internet Options | Privacy | Sites**, then typing in the address of your domain (e.g. `malbet.local`) and clicking the **Allow** button.

If you get the following error message, then it most likely means that the mailbox you have specified does exist but is not on the server that you have specified:

The specified Outlook Web Access Web Part URL contains an invalid value for the view parameter.

Once we have worked our way through these various error messages, then we should be able to see our task list successfully embedded in our web page.



Summary

In this chapter, we have witnessed the power of the ready-made tools that SharePoint makes available to us. We learned how to use OWA Web Parts to display our Exchange information in our SharePoint site and are now aware of the configuration changes that are required to do so successfully and how to troubleshoot any error messages that we may come across.

No doubt, you will be keen to embed your Exchange data into your applications, and pleasantly surprised about how easy it is to do so.

11

Search Tools

It is well known that the amount of data in the world doubles very rapidly. Estimates of the rate vary: once every three months, once a year, every two years, or every three years. There is even a law predicting the growth, which is known as "Reed's Law". One reason for the differing estimates is that they predict growth for different types of data (business data, social networks, etc.). Whichever rate of growth we choose, it is clear that the amount of data in the world is increasing at a phenomenal rate. According to a study, by the University of California, Berkley, there were five exabytes of new data created worldwide in 2002 alone. That is about 5,000,000,000 GB of information.

As the amount of data increases, the problem is not so much that we do not have the information in our organization, it is more that it is difficult to find answers amongst all the other information that we have.

A key requirement of any collaboration system is that it allows users to easily search for the information they need so it is at their fingertips.

"Information overload" and "knowledge transfer" have been the focus of numerous keynote speeches by Microsoft's top executives, over the past couple of years. So, it comes as no surprise that SharePoint is no slouch when it comes to providing powerful search tools. The improvements to the search capabilities in MOSS 2007 over its predecessor are numerous and include useful new functionality, such as "Did you mean..." and the ability to search for people who have certain knowledge or skills.

In this final chapter, we will examine the search tools we have at our disposal and use them to add search capabilities to our site.

Federated Searching

The power of SharePoint Enterprise Search is that it does not only query data stored in one location but can pull data from a variety of sources, a technique known as "federated searching".

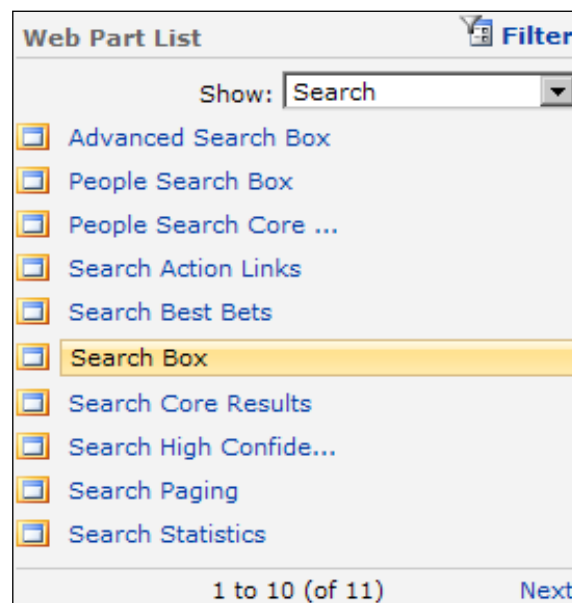
When a user performs a search, different search engines are queried using the "Open Search" standard. This allows various data sources including SharePoint sites, desktops, line-of-business applications (e.g. Lotus Notes), and even users' brains to be searched and the results displayed in one aggregated results list.



Don't worry about SharePoint searching our minds. This is not as scary as it sounds and requires no brain implants. SharePoint's people search merely identifies which brain the information is stored in and allows the data to be retrieved in a traditional manner, such as a telephone conversation or email exchange.

Search Web Parts

It will come as no surprise that the key components that we will use to add search functionality to our pages are available in the **Web Part List**. When we filter the list to display the Search components, we have the choice of no fewer than eleven different Search Web Parts that we can use in our applications.



The following Web Parts are available to us:

- **Advanced Search Box** will display an extremely comprehensive search form, which includes the ability for users to choose if they would like an exact phrase to be found, to specify words that should not appear in the search results, the type of documents they would like to search (e.g. Excel), and to select the languages they would like to search.
- **People Search Box** does exactly as you would expect, allowing the user to search for staff in our organization by name, department, title, and their responsibilities, skills, or memberships. The results of the search can be displayed in the **People Search Core Results** Web Part. A great feature of this tool is that it can also make use of an employee's "social distance" within the organization (i.e. a sales rep would not want to call up the CIO to ask him to set up his mailbox). It is important to note that these Web Parts (and the identical people search and employee lookup features in the default SharePoint site) will not work straight out of the box. We must first define the Active Directory from which we wish to import user profiles to allow people search to work.
- **Search Action Links** allows the user to perform additional actions on the search results. The main way that these action links are used is to allow the user to sort the results by either relevance or by modified date. It is also possible to use **Search Action Links** to create alerts based on the search results or to view an RSS feed of the results.
- **Search Best Bets** provides a list of the top search results.
- **Search Box** is a lovely little component and is the simplest of the search forms. It allows the user to specify the scope of their search (i.e. search our site, all sites, or people) and to enter a simple keyword search.
- **Search Core Results** is the main results Web Part. By placing this Web Part on our page, we can display the results of the search to the user.
- **Search High Confidence Results** will only display the results that SharePoint thinks are very good matches (as opposed to the **Search Best Bets** Web Part, which will display the top results no matter how good SharePoint thinks they are).
- **Search Paging** allows the user to skip backwards and forwards between the different pages of results.

- **Search Statistics** is nothing to get too excited about. When I first heard there was a Search Statistics Web Part, I immediately assumed that it would display a report of the top searches that users were performing on the site. Alas, it does not provide those types of statistics (there is a separate feature known as the query report that allows you to do that). It will display the number of results that are on the page, the total number of results, and the time taken to complete the search.
- **Search Summary** displays a summary of the search query and can also display a "Did You Mean..." feature, which second-guesses what the user really wanted to search for. The summary can either be displayed in a compact or an extended mode.

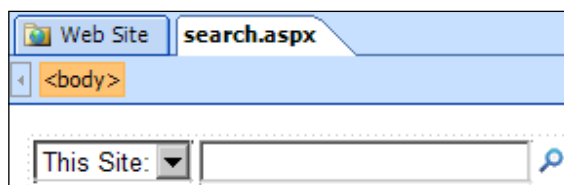
Using the Search Web Parts

We will waste no time and create an example search. As we do so, you will see that adding basic search capabilities to our site could not be simpler.

For our example, we will use the **Search Box** Web Part to allow users to search our site and the **Search Core Results** Web Part to display the results.

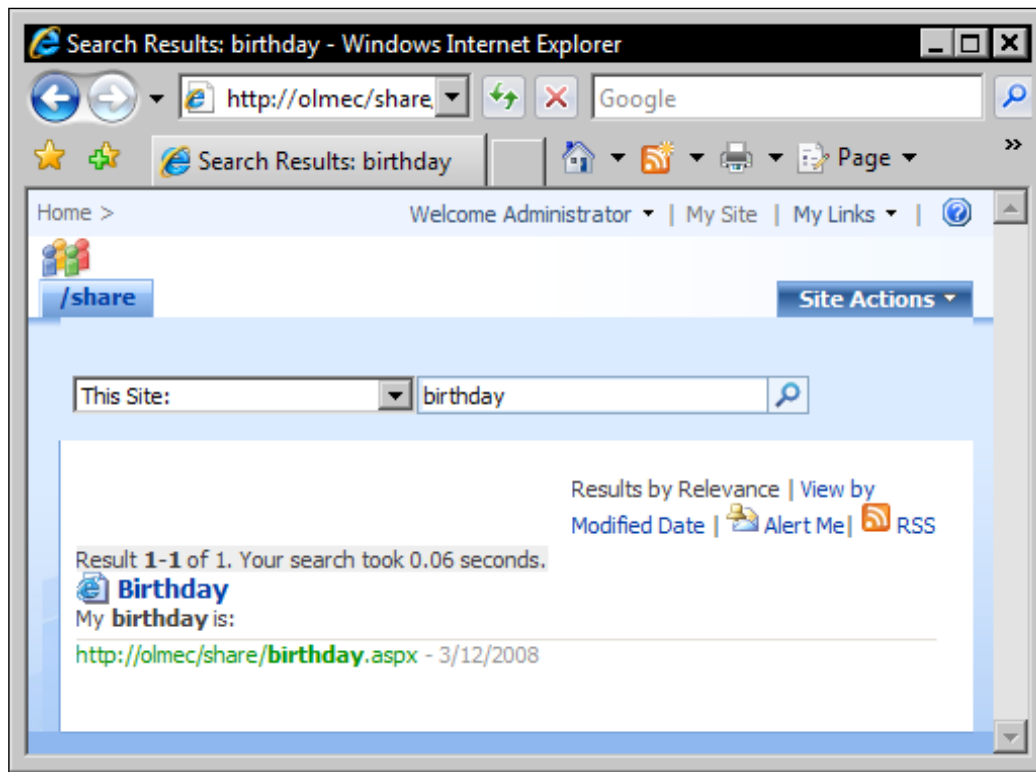
Let's start off by creating a new ASPX page on our site called `search.aspx`. This will be a simple page that will do nothing more than sending a query string to the results page.

Drag and drop the **Search Box** Web Part onto the page and then save the page. To begin with, we will not set any of the properties for this Web Part.



Let's test our search page by opening it (e.g. <http://olmec/share/search.aspx>) in our web browser and typing in a search term (e.g. "birthday"). When we click on the magnifying glass search button, the results are presented to us.

Notice that the search results have appeared in the default results page for our share site without us needing to build a results page.



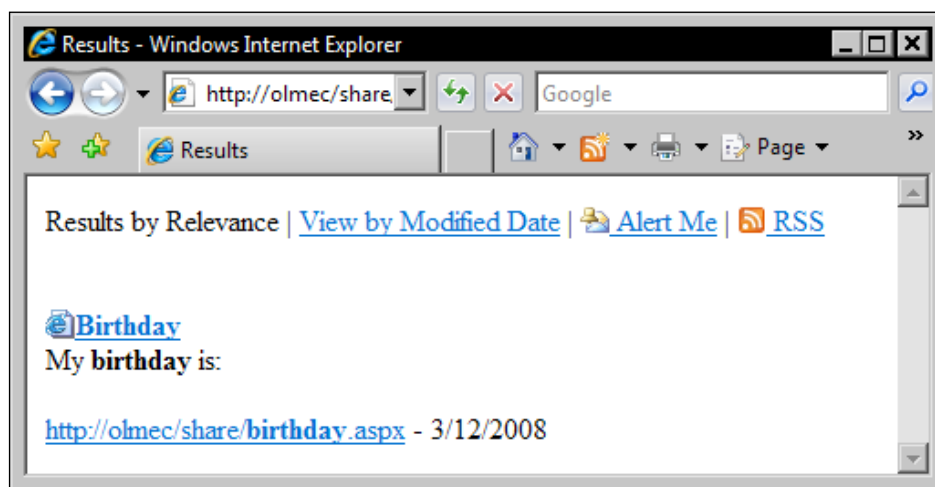
By default, search results are displayed in either the site you are working in (when using the Search Box Web Part, e.g. `http://olmec/share/_layouts/OSSSearchResults.aspx`) or in the Search Center (when using the Advanced Search Box Web Part, e.g. `http://olmec/searchcenter/Pages/results.aspx`).

It is possible to create our own results page. To do this, we must first of all tell our search form to point to the new results page that we will create:

1. Right-click on the **Search Box** Web Part that we placed on our page and select **Web Part Properties...** from the shortcut menu. This will open a dialog with the title **Search Box**.
2. Click on the plus sign to the left of the **Miscellaneous** title halfway down the dialog to display the **Miscellaneous** options.
3. Change the **Target search results page URL** to `results.aspx`.
4. Click **OK** and then save the page.

Next, we will create a second ASPX page called `results.aspx` onto which we will drag and drop the **Search Core Results** Web Part.

When we have saved both our pages and repeat the search (making sure, of course, that we have refreshed the search page in our browser so that it is using the latest version), we will see the results displayed in our customized results page.



Hard-Coding Results

When searching the site, notice how the keywords that we use are displayed in the address bar on the results page. For example, when we search for `birthday`, we see that the value `birthday` is displayed with the name `k` (for keyword) in the URL:

```
http://olmec/share/results.aspx?k=birthday
```

Why is this good to know? Well, it opens up the possibilities when it comes to providing users with pre-defined searches. If we would like to provide our users with a basic way of searching for `chardonnay`, we may hard-code the following URL into a link so that when the user clicks the link they are provided with `chardonnay` results:

```
http://olmec/share/results.aspx?k=chardonnay
```

This means that when we are displaying records in a grid view, we can program SharePoint to display the results of a field (e.g. wine type) as hyperlinks, which will use SharePoint Search to display the results.

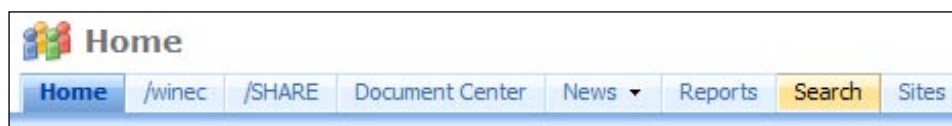
Other SharePoint Search Solutions

Apart from building our own search pages in SharePoint with the controls it provides, we can use solutions that Microsoft provides straight out of the box (or a couple of boxes, in fact).

Search Center

The most obvious of these search tools would be the Search Center that is built into the default SharePoint site.

By default, the Search Center is available at <http://servername/searchcenter/> (e.g. <http://olmec/searchcenter/>). When we open up our default SharePoint site (e.g. <http://olmec/>), we can see the **Search** button on the menu bar that links to the **Search Center**.



One of the beauties of the Search Center is that we can edit the appearance and content of the page either with SharePoint Designer or by clicking on the **Site Actions** button in the top right-hand corner of the Search Center and selecting **Edit Page**.

The other great thing about the Search Center is that the scope of searches includes all SharePoint sites on the server. Many companies have separate sites for different departments (HR, Finance, IT, etc.). The Search Center will provide results from all these sites in one federated list.

Search Server 2008

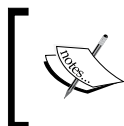
The SharePoint Server for Search product has been rebranded as Microsoft Search Server 2008 (and the free Search Server 2008 Express version) and is available to install separately from SharePoint Server so that you can benefit from the same search functionality that exists in SharePoint Server on sites that do not use SharePoint.

Further details are available at <http://www.microsoft.com/enterprisesearch/>.

Search Term Vocabulary

If you are serious about implementing searching in your site, it might be useful to take a moment to familiarize yourself with some terms used:

- **Noise** words can be disabled or enabled in SharePoint. These are words that add no meaning to the search (e.g. "the", "or", "a").
- **Stemming** is where matches are also made on similar words based on the stem of the word. This works particularly well with verbs (e.g. the stem "fly" will bring up matches with "flying" and "flew"). Stemming is not enabled by default in SharePoint but this can be configured.
- **Soundex** allows words with similar pronunciation to be grouped so that when you search for a word (e.g. "Mike"), then similar sounding results will be returned (e.g. "mic", "Michael", and "Michal"). Soundex is built into Microsoft SQL Server 2005 and works by ignoring the vowels and converting the remaining string into four-digit codes, which is then compared. Soundex is not built into SharePoint server, but by adding some simple SQL queries (e.g. `SELECT * FROM tblPeople WHERE SOUNDEX('Mike') = SOUNDEX(firstname)`), it is possible to add Soundex searches to our site.



While it is possible to create SharePoint searches using SQL queries, it is outside the remit of this book. If you would like to write some code to make use of SQL queries, then it is recommended that you investigate the *Microsoft.Sharepoint.Search.Query* namespace.

Summary

This brings us to the end of our chapter on using the search capabilities that SharePoint provides us with. We have not only seen how to add search forms and results lists to our pages but have also learned about the range of Search Web Parts that are available to us.

As ever, I hope that you have been as impressed as I am with how easy SharePoint makes it to create powerful applications. More than that though, I hope that you have enjoyed not just this chapter but also the rest of the book and that it helps you to understand and use SharePoint Designer.

Index

A

ASP.NET controls

- about 129
- calendar control, standard controls 130
- data controls 130
- fileupload control, standard controls 130
- hyperlink control, standard controls 130
- image control, standard controls 130
- label control, standard controls 130
- login controls 131
- navigation controls 131
- standard controls 130
- validation controls 131
- wizard control, standard controls 130
- Xml control, standard controls 130

B

borders, table properties 44

C

Calendar control

- about 134, 135
- web.config file, editing 136

conditional formatting 107

Contributor mode 79

contributor settings, website

- content authors, contributor groups 83
- contributor groups 82
- region types 83
- site manager, contributor groups 83
- web designers, contributor groups 83

D

data formatting

- conditional formatting 107

data sources

- about 93
- business data catalog 93
- database connections 93
- linked sources 93
- server-side scripts 93
- SharePoint libraries 94
- SharePoint lists 94
- XML files 94
- XML web services 94

data view

- conditional formatting, applying 107-109
- creating, on website 95
- CSS formatting 103-106
- data sorting by users 114
- direct formatting 103
- filtering 110
- formatting 103
- formula column, adding 110-112
- numbers, formatting 109
- paging 114
- sorting 113

E

error messages

- troubleshooting 156

existing website

- editing 53

F

federated searching 160

I

InfoPath 100

integrated development environment

- button bar 27
- code view 26, 27
- status bar 28
- task panes 19, 20

L

login feature, creating

- A record, adding 142
- authentication provider, changing 146, 147
- first user, adding 140, 141
- host reader, adding 142
- SQL server, configuring 139, 140
- virtual server, adding 141
- virtual server, extending 142-144
- Visual Studio web site administration tool, using 144, 145
- web.config file, returning to 148

M

Master Page

- about 70
- attaching, to existing page 74
- content region, adding 73
- creating 71
- editing 71, 72
- modifying 76, 77
- new page, creating 74, 75
- saving 74
- storing 71
- stylesheet, attaching 77

Menu control

- about 133
- adding, to graph page 133

Microsoft Office SharePoint Server 2007

- about 8
- functionality 8

Microsoft technologies

- Active Directory 8

ASP.NET 8

Internet Information Services 8

SQL Server 8

My Tasks Web Part

- integrating 155

O

Outlook Web Access Web Parts

- about 151
- forms-based authentication, enabling 154
- my calendar 153
- my contacts 153
- my inbox 153
- my mail folder 153
- my tasks 153
- OWA, enabling 153
- viewing 152

R

records

- adding 99
- deleting 99

S

search solutions

- search center 165
- search server 2008 165

Search Web Parts

- about 160
- example 162, 163

SharePoint

- about 7
- ASP.NET controls 129
- Calendar control 134
- contribution, setting up on Master page 85, 86
- Contributor mode 79
- data sources 93
- development tool 18
- error messages, troubleshooting 156
- existing website, editing 53
- federated searching 160
- hard coding results 164
- IDE 18
- information 15

- integrated development environment 18
- Menu control 133
- Microsoft technologies 8
- My Tasks Web Part, integrating 155
- Outlook Web Access Web Parts 151
- search center, search solutions 165
- search server 2008, search solutions 165
- search solutions 165
- search term vocabulary 166
- Search Web Parts 160
- server-based sites versus disk-based sites 80
- SharePoint designer 9, 10
- SharePoint designer, need for 9
- simple control, adding to website 132
- website, creating 33
- wine company website example 33
- workflows 87
- SharePoint designer**
 - about 10
 - conditional formatting 107
 - installing 10-13
- SharePoint site**
 - connecting to 13-15
 - login feature, creating 138
 - softwares used 18
- softwares, SharePoint site**
 - Expression Web 18
 - Notepad 18
 - Visual Studio 2005 18
- status bar, integrated development environment**
 - contextual message 28
 - CSS schema 30
 - document schema 30
 - download statistics 29
 - page size 30
 - standard rendering mode 29
 - style application 29
 - visual aids 29
- T**
- table properties, website**
 - alignment, layout 43
 - background 44
 - borders 44
 - cell padding, layout 43
 - cell spacing, layout 43
 - collapse table border, borders 44
 - color, background 44
 - color, borders 44
 - columns spanned 45
 - float, layout 43
 - header cell 46
 - horizontal alignment 45
 - layout 43
 - no wrap 46
 - rows spanned 45
 - set as default for new tables 45
 - size 43
 - size, borders 44
 - specify height, layout 43
 - specify width, layout 43
 - use background picture 44
 - vertical alignment 45
- task panes, integrated development environment**
 - accessibility 24
 - apply styles 22
 - behaviors 22
 - clip art 24
 - clipboard 24
 - compatibility 24
 - conditional formatting 23
 - contributor 24
 - CSS properties 22
 - CSS reports 24
 - data source details 23
 - data source library 23
 - find 1 24
 - find 2 24
 - find data source 23
 - folder list 20
 - hyperlinks 24
 - layers 22
 - layout tables 22
 - manage styles 22
 - navigation 21
 - organizing 25
 - tag properties 21
 - toolbox 22
 - web parts 23

V

validation controls

- about 137
- adding, to forms 137
- rangevalidator control 137

W

webpage

- viewing 51, 52

Web Part

- inserting, to website 120, 122

Web Part list

- advanced search box 161
- people search box 161
- search action links 161
- search best bets 161
- search box 161
- search core results 161
- search high confidence results 161
- search paging 161
- search statistics 162
- search summary 162

Web Parts

- about 117
- Dundas Chart Web Part 125

Web Part Zones

- about 118
- inserting, to website 119

website

- authorization 53
- contributor settings 81
- creating 33
- data view, creating 95-99
- graphs, adding 122
- homepage, formatting 58
- images, publishing 65
- layers, adding to web page 61
- Master Pages 70
- publishing 49, 51
- records, adding 99, 100
- records, deleting 99, 100
- styles, editing 69, 70
- styles, renaming 67
- styles sheets, cascading 67, 69
- XML data source, creating 94, 95

website, creating

- div versus tables 48
- first page, creating 35, 36
- hyperlinks, creating 39, 40
- images, adding 41
- layout tables 47
- table properties 43
- tables, creating 42
- text, adding 37, 38
- text, formatting 37, 38
- web page, previewing 38, 39

Windows SharePoint services 3.0

- about 8
- basics 8

wine company 33

wine company website

- about 55
- creating 56
- homepage, creating 57
- homepage, formatting 58-61
- images, adding to layer 64, 65
- images, publishing 65, 66
- layers, adding to web pages 61-64
- SharePoint site, creating 56
- site, publishing 58
- styles, renaming 67
- style sheets, cascading 67

workflows

- about 87
- library 88-90
- lists 88-90
- new workflows, defining 90, 91
- workflow designer 87

X

XML data source

- creating, on website 94, 95



**Thank you for buying
SharePoint Designer Tutorial**

About Packt Publishing

Packt, pronounced 'packed', published its first book "*Mastering phpMyAdmin for Effective MySQL Management*" in April 2004 and subsequently continued to specialize in publishing highly focused books on specific technologies and solutions.

Our books and publications share the experiences of your fellow IT professionals in adapting and customizing today's systems, applications, and frameworks. Our solution based books give you the knowledge and power to customize the software and technologies you're using to get the job done. Packt books are more specific and less general than the IT books you have seen in the past. Our unique business model allows us to bring you more focused information, giving you more of what you need to know, and less of what you don't.

Packt is a modern, yet unique publishing company, which focuses on producing quality, cutting-edge books for communities of developers, administrators, and newbies alike. For more information, please visit our website: www.packtpub.com.

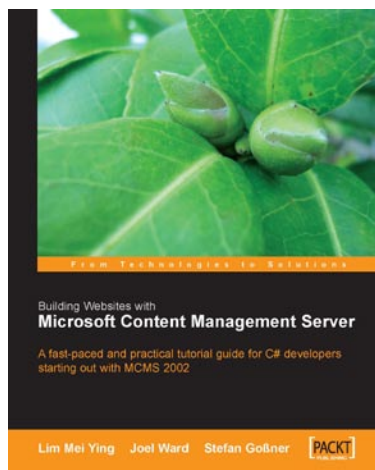
Writing for Packt

We welcome all inquiries from people who are interested in authoring. Book proposals should be sent to authors@packtpub.com. If your book idea is still at an early stage and you would like to discuss it first before writing a formal book proposal, contact us; one of our commissioning editors will get in touch with you.

We're not just looking for published authors; if you have strong technical skills but no writing experience, our experienced editors can help you develop a writing career, or simply get some additional reward for your expertise.



This material is copyright and is licensed for the sole use by George McCAVITT on 7th March 2009
Corndon, Manor Road, Stourpaine, BLANDFORD FORUM, Dorset, DT11 8TQ

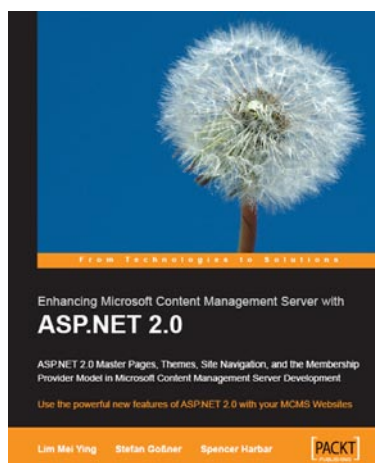


Building Websites with Microsoft Content Management Server

ISBN: 1-904811-16-7 Paperback: 638 pages

A fast-paced and practical tutorial guide for C# developers starting out with MCMS 2002

1. Learn directly from recognized community experts
2. Rapid developer level tutorials build logically through out the book
3. Develops a feature rich custom site incrementally
4. Tips and Tricks from developer newsgroups and online communities



Enhancing Microsoft Content Management Server with ASP.NET 2.0

ISBN: 978-1-904811-52-7 Paperback: 180 pages

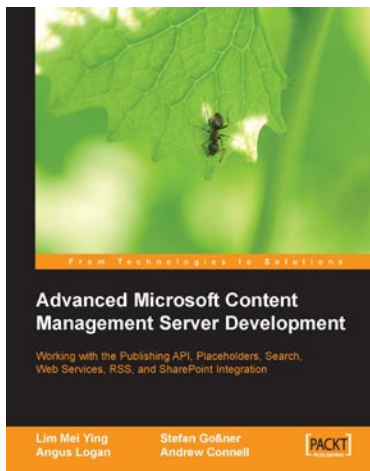
Use the powerful new features of ASP.NET 2.0 with your MCMS Websites

1. Get Microsoft Content Management Server Service Pack 2 up and running
2. Use the most exciting features of ASP.NET 2.0 in your MCMS development

Please check www.PacktPub.com for information on our titles



This material is copyright and is licensed for the sole use by George McCAVITT on 7th March 2009
Corndon, Manor Road, Stourpaine, BLANDFORD FORUM, Dorset, DT11 8TQ

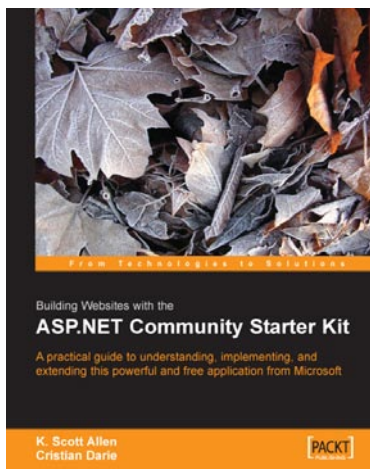


Advanced Microsoft Content Management Server Development

ISBN: 1-904811-53-1 Paperback: 512 pages

This book has the most in depth-coverage of important MCMS development topics found anywhere. Each author of the book is a renowned expert in the area

1. Learn directly from recognized community experts
2. Extensive coverage of the Publishing API (PAPI)
3. Get Sharepoint and MCMS working together
4. InfoPath, RSS and hot topics covered



Building Websites with the ASP.NET Community Starter Kit

ISBN: 1-904811-00-0 Paperback: 268 pages

A comprehensive guide to understanding, implementing, and extending the powerful and freely available application from Microsoft

1. Learn .NET architecture through building real-world examples
2. Understand, implement, and extend the Community Starter Kit
3. Learn to create and customize your own website
4. For ASP.NET developers with a sound grasp of C#

Please check www.PacktPub.com for information on our titles



This material is copyright and is licensed for the sole use by George McCAVITT on 7th March 2009
Corndon, Manor Road, Stourpaine, BLANDFORD FORUM, Dorset, DT11 8TQ