Teun Duynstee

The
Web Part Infrastructure
Uncovered

Acknowledgements

Writing a book is not easy. I have done it a few times now and it still isn't. For me, the hardest part is to keep going. I have a strong tendency to encounter many other problems that seem a lot more interesting than finishing a book (I already know the ending anyway). With my first writing experiences, I used to have an editor that would remind me of the schedule and who would keep asking 'how I was doing'. I never realized how much I needed someone to do that for me.

Anyway, after 2 years of writing on and off, I think that the result is publishable. Web parts are not as new anymore as they were when I started out, but on the other hand, some real world experience could slip in as well, which made the booklet better. Better still: it looks as if web parts will be around for a long time, with SharePoint becoming more and more the lynchpin of all other Microsoft server products.

And again, even without a publisher and without co-authors, there is a number of people that deserve a big thank you from the author. First, there are the proof-readers. You guys had to muddle through a crude version, you gave me some really valuable feedback. I recognize that proof-reading is harder than it seems: some of you even never recuperated enough to send me back your comments. ☺ Special thanks to Remco Kapinga, Eric Willeke and Leon Zandman.

I'm not quite an artist, so I'm glad that people like Linnell Esler post their photo material free for use on sites like sxc.hu. I really like the uncovered box of chocolates on the cover (http://www.sxc.hu/photo/386007).

Then there are my colleagues at Macaw. The reason that Macaw is a great place to work and the reason that I still go to work every morning is you: the people I can talk with about the latest and greatest technology thingies, sharing my enthusiasm on the 'interesting subject of the day' (thanks for listening).

And then of course there is Marjolein, the love of my life. She was the one to keep me on track and writing ("Is that your book you're working on?"). Now let's get out your thesis too. I love you.

Table of Contents

1 Introduction

The Web Part Infrastructure is one of the most conspicuous new features in Microsoft's ASP.NET 2.0. Spectacular demonstrations can be given, where only some declarative code and drag-and-drop operations whip up a configurable website. In these websites, end-users can easily change the appearance of the pages to their liking. By dragging blocks over the page, they can place the parts of most interest to them at the top, remove parts they don't need or even add new parts to the page. These modifications will only be visible for this user. Other users can make their own modifications and all of the personalization information is stored with their user name. The great thing is that most of this functionality, from the storage of the info up to the scripts that enable drag-and-drop in the web page, comes out of the box with no code written at all.
This allows developers to achieve personalized portal functionality at a fraction of the usual cost. Many ASP.NET developers are enthusiastic about these new opportunities. This book will show you all of the possibilities of using web parts in your projects. We'll also show you how to use web parts in a real site (and not just a demo); what is the appropriate way to use web parts in an intranet, a web portal, a content publishing site. We'll get into the many options for setting privileges, how to make your own configurable web parts, how to create web part catalogs to pick parts from and how to let parts on a page communicate to make them work together.

The vision of web parts

The main idea of web parts is to let business users create and configure web pages by dragging in functional blocks (called web parts) and setting properties on these blocks. This configuration happens right inside the web page, using nothing but the web browser. To change a page in the site, you browse to this page, switch to edit mode and change it. Non-technical users can change the layout of their page by dragging and dropping the web parts to different zones on the page. Each of the parts exposes a configuration user interface to change its behavior and appearance.

Combining the web part paradigm with personalization opens up a whole new world of possibilities. With the personalization framework inside ASP.NET 2.0, the settings applied on the parts on a page can be different per user. You can allow any user to make changes to the page (like remove a function from the page) and these changes will be visible to that user alone. And the changes will be automatically persisted and still be there for the user in a future session.
Without code! You as a developer can concentrate on programming the parts and decide which properties the user will want to configure.

Who this book is for

This book is aimed at ASP.NET developers, who already have some experience building websites on this platform. You should know what an ASPX page is and what a User Control is. You don't need knowledge of any of the new features in version 2.0 of ASP.NET; we will touch some of these new features, but where needed, we will shortly introduce them so you know enough to get on with the main subject of the chapter.
It would help if you had some experience in working with databases such as MS SQLServer or MS Access. Basic knowledge of HTML is required and to really understand some of the examples, experience in client-side JavaScript would be nice.
But most of all: you want to learn how to create websites with web parts and how to create cool web parts yourself.

This book builds on the idea that is it more important to understand the technology than to show cookbook recipes for performing tricks. If you understand the concepts, the tricks are trivial and the recipes never match the requirements of real projects anyway. Still, not everyone will be interested in the same depth of information. The first half of this book will give you a good understanding of web parts and you will have seen examples and some of the working under the hood. The second half delves into some special techniques and more advanced subjects. You can cherry-pick from these chapters according to you needs and liking: there is no required order of reading.

This book uses C# as its programming language. Of course, all of the examples in this book can be created in any .NET language. Converting the code to your favorite language should be trivial.

About SharePoint, WSS, the Digital Dashboard and web parts: some history

Release 2.0 of ASP.NET is not the first Microsoft product featuring something called Web Parts. The term appeared first (I think) when Microsoft shipped the Digital Dashboard (DD). DD was basically a VBScript that rendered web pages based on documents in folder structure that could be accessed by HTTP-DAV. The basic idea was that a folder on the file system would contain small pieces of script code. At runtime, the engine would execute the pieces of code and combine them into a web page. The small pieces of code were called web parts and where the most componentized form of server code available on the Microsoft web platform at the time (of course, the .NET framework and ASP.NET were still to be conceived). A page like this was called a Dashboard and a dashboard manager would be able to set properties on the web parts and thus change the page to his liking.

Soon after this, Microsoft unveiled their first real product based upon the DD: SharePoint Portal Server (SPS). SPS offered collaboration features, a strong search engine and a versioned document store. All user interaction was based upon the DD and when building applications upon the SPS, one would write and deploy web parts. SPS was a large commercial success, but the predicted market for commercial web parts has never really materialized.

Then the .NET framework came along: the functionality of web parts seemed a natural match for the control based architecture of ASP.NET and so SharePoint was rebuilt from the ground up. The DD core was removed and replaced by SQLServer based storage. Web parts were still an important concept in Windows SharePoint Services (as it is now called), but now they are a .NET class that you can subclass from. The marriage of web parts with the .NET framework was natural and made the writing of non-trivial web parts a lot easier. WSS is freely available on Windows Server 2003. WSS is also a rather complex system that includes much more

than the web part framework. The learning curve for developers is rather steep.

In ASP.NET 2.0, the ASP.NET team has brought the concept of web parts to the masses. It's now freely available on all platforms that run ASP.NET 2.0. It is also made easy to use it without the overhead of the WSS. Version 3 of WSS is built upon this new core, while remaining backward compatible with the WSS2 style of web parts. This means that you can apply the skills learned through this book on several platforms:

- You can build personalizable web sites using web parts in ASP.NET 2
- You can build applications within WSS3 leveraging web parts as UI components and using the tons of features that also come with WSS (Page and Site provisioning, Lists, Content Types, Workflows, Versioning, etc…)
- Many enterprise level server applications leverage WSS as their web platform (like MS Office Sharepoint Server 2007, Team Foundation Server, PerformancePoint). Using you web part skills you can modify and extend these applications

The platform for web parts has thus stretched both the low budget end and to the high end. This means that more developers will be able to use the concept of web parts in their solutions and that the market for third party web parts may finally become commercially feasible.

Compatibility

Web parts built for SPS1 should be considered legacy. They do not work on WSS or inside the Web Part Infrastructure in ASP.NET. Your current .NET web parts for WSS will work in the next version of SharePoint and can be used inside the Web Part Infrastructure. Keep in mind though that many of the current WSS web parts rely on other parts of WSS being available. These web parts will obviously not work outside the WSS environment.
The WSS 3 is built upon the Web Part Infrastructure, so parts built for ASP.NET 2.0 will work in WSS 3. In the further future, the ASP.NET version of web parts will probably be the only path.

So: if you have invested in WSS web parts, keep them. If you have invested in SPS1 web parts, investigate the cost of rewriting them. If you have no current investments and you do not need to support the current version of WSS, go with ASP.NET web parts. The future is yours!
This book will teach you how to use the ASP.NET web parts to build great web sites and how to create good customizable web parts for them. For a bit more detail on specific WSS 3 features and concerns, check out chapter 15.

The sample site we will build

In the course of this book, we will see a number of techniques to achieve specific effects. In the code download belonging to this book, you will find a sample website that makes use of many of these techniques. Appendix II contains a walkthrough of the sample site, but here we show a short feature list of the site.

- On the homepage, we will have several zones containing web parts. One of these zones will display web parts that must always be visible. The contents of this zone should only be modified by administrators and their changes should be reflected on the page by all other users. The users can modify the other zones as they like by dragging and dropping the parts to other locations on the page and editing the settings of web parts. We will make a list of web parts available that they can add to their personalized homepage.
- Besides the home page, users have access to a personal page that they can fill with web parts to their liking. Users must be logged in to use their personal page.
- The site also contains a small application that should only be accessible to administrators. This application can be used for user management.