



Robert L. Bogue
MS MVP, MCSE, MCSA:Security
317-844-5310
Rob.Bogue@ThorProjects.com

Understanding SharePoint Branding Options

When we're talking about branding in Microsoft SharePoint there are several options. You've probably heard about site definitions, site templates, themes, master pages, page layouts, CSS, and so on. In this article we'll tear apart each of these options and discuss what you'll want to use to modify the appearance of your pages.

SharePoint Designer

The starting point for most discussions about branding is SharePoint Designer. SharePoint Designer is the preeminent editor for SharePoint and can handle editing the master page, page layouts, and pages themselves. The master pages, ASP.NET Master Pages to be more specific are the master layout for the page. The content from a page layout which is used in web content management, or a page drops into placeholders in the master page. Page layouts further contain content from the items they're associated with but otherwise are edited the same way that regular pages are.

Certainly that's not a bad capabilities list. There are some pages that SharePoint Designer can't modify. It can't modify any of the "application" pages. Those are the pages that exist in the _layouts directory. This can be an issue because some of the pages that users see -- such as upload file, and view all site content, come from this directory.

SharePoint Designer is also a great editor for Cascading Style Sheets (CSS) both from the context of editing your own CSS sheets in a site as well as for editing the CSS sheets that are essential to themes.

The challenge with SharePoint Designer is that the changes apply only to the site collection, site, and page that SharePoint Designer is editing. That's great if you just need to customize the look of one site -- but what if you need the look and feel to apply to multiple sites?

(If you want to learn more about how and when to use SharePoint Designer see: [When You Need SharePoint Designer](#))

Site Template

The first option for getting the SharePoint Designer changes beyond the current site is to take the site you've modified with SharePoint designer and save it as a template. Then when you need a new site you create it from the site template you've created. This approach works but it means that you either have to create a new site template for each of the built in site definitions -- or you have to live with just the one site template that you've created. It also means that you can't go back and apply the changes after they've been created.

Finally, to add insult to injury, if you ever change the work that you did on the original site those changes won't be reflected across the other SharePoint sites that were created from the template because each page exists separately in the database. So site templates solve the problem of getting SharePoint Designer changes into multiple sites but without the ability to adapt to changes in the future.

Site Definition

Site Definitions solve the problem of site templates in that changes can be affected to them after they've been created -- but at the cost of additional work. Site definitions exist on the file system of each of the front end web server unlike site templates which exist in the content database. Each site which is created from the site definition doesn't make a copy of the page in the database, instead it stores a pointer to the site definition file. The good news is that when the page on the file system changes it changes all of the references in the database -- thus solving the problem of making changes to existing branding.

In order to do a site definition, you'll first have to get the changes to the files that you want into files. You can do this from SharePoint Designer by selecting Save As and pointing to a directory. From there you have to create the support files for the site definition including the ONET.XML file which drives how the site definition is used, and the WEBTEMP*.XML file which makes the site definition show up as an option. In most cases you'll copy the STS site definition that comes with SharePoint and make your changes there rather than starting from scratch.

So with site definitions you can get the ability to modify the look and feel of the site after the site has been created, but at the cost of additional effort and with the same limitation of needing to create a separate site definition (or configuration) for each of the out of box site definitions, and they still don't solve the problem of branding the application pages. Before we get into a solution for that, we need to talk briefly about one of the features of SharePoint that works with Site Definitions in order to facilitate changes.

(You can learn more about Site Definitions and how to create them in the article "[SharePoint Site Definitions: Why You Need Them and How to Use Them](#)." The article was written for 2003 but the process is largely the same between SharePoint 2003 and SharePoint 2007. You can find additional information in the SDK article "[Creating a Site Definition from an Existing Site Definition](#)".)

Feature

The SharePoint Feature (notice the capital F) allows you to enable and disable functionality in a site or sites. Technically speaking you're not supposed to change a site definition after it's been created, however, it's perfectly acceptable to make changes to SharePoint Features. Site definitions can automatically activate features when instances are created of them. Because of this the best way to manage changes in a site definition is to have SharePoint Features which are activated when a site is created and then allow those SharePoint Features to be changed.

Features can deploy pages into a site definition -- and can also define content types, delegate controls, and a variety of other types of functionality for a site. So features are an integral part of building site definitions and a great way to manage changes over time.

(You can find out more in the SDK article "[Working with Features](#)".)

Solutions

When talking about site definitions, I made the point that they are files on the file system -- they're not in the content database. This means that they must be deployed to each of the front end web servers. However, that can be painful -- particularly if you have a number of SharePoint servers. However, SharePoint has a solution -- SharePoint Solutions (notice the Capital S). SharePoint Solutions, WSP files, allow you to deploy site definitions, features, web parts, and files.

They're a simple CAB file with a WSP extension that includes a manifest.xml file that tells SharePoint how to use the files that are in the file. The beauty of the SharePoint Solution is that SharePoint includes an infrastructure to deploy these WSP files to all of the servers on the farm. This means the hassle of deploying files manually to multiple servers can be eliminated.

(You can find out more about building solutions in the article: "[Using SharePoint Solutions to Deliver Server-Side Solution](#)".)

Theme

With all of the other options we've seen we've been able to do a lot of customization. SharePoint Designer or directly editing the master page, page layouts, or pages themselves allows you to control not only the colors and images -- it allows you to control every aspect of how the page looks. As a result you can change the number of web part zones on the page, their location, or even how they operate. However, most of the time, all you are really seeking is an opportunity to get the site's colors and images -- particularly the logo - to match the company standards -- that's something that's well within the reach of a Theme. The limitation of a Theme is that it can't change the layout of the page. (Technically CSS can influence layout but in the hardcoded HTML that some of SharePoint's controls emit layout isn't as easy to change with CSS as it should be.) However, what it can do, that the other techniques can't do is that it can change the application pages. A theme is applied across all of the pages in the site - including the application pages. Thus a Theme is the only way to customize a site such that all of the pages reflect the new colors and logo.

The one remaining hitch to Themes is that deployment isn't perfect. Although they can be modified within a site with SharePoint Designer, to make them accessible to every site they should be saved to disk and deployed to all of the front end web servers. Of course, SharePoint Solution packages (WSP files) should be able to handle that task -- and they do except for the XML file you have to modify to tell SharePoint to use the Theme files you've created. The problem is solvable -- you can write a feature receiver to handle this. However, no one has publically released a feature receiver to help with the installation of a Theme -- yet.

(You can find out more about themes in the SDK article "[How to: Customize Themes.](#)")

Conclusion

If you want to really have a customized site that covers all of the pages in a site including the application pages, you'll need a theme. In most cases you'll also want a custom site definition to handle additional layouts. That site definition will use custom features. If it seems like a tall order to get it all together, it can feel that way. But when you're all done, clients won't be able to realize that they're even on a SharePoint site -- unless they see all of the functionality they have.

About The Author

Robert Bogue, MCSE (NT4/W2K), MCSA:Security, A+, Network+, Server+, INet+, IT Project+, E-Biz+, CDIA+, is president of Thor Projects LLC, which provides SharePoint Consulting services to clients around the country. He has contributed to more than 100 book projects and numerous other publishing projects. His latest book is The SharePoint Shepherd's Guide for End Users. (You can find out more about the book at www.SharePointShepherd.com.)

Bogue has been part of the Microsoft Most Valuable Professional (MVP) program for the past 5 years. He was most recently awarded for Microsoft Office SharePoint Server. Before that, Bogue was a Microsoft Commerce Server MVP and Microsoft Windows Servers-Networking MVP.

Bogue runs the SharePoint Users Group of Indiana (SPIN, www.spindiana.com), and he is also a member of the steering committees for the Indiana Windows Users Group and Indianapolis .NET Developer Association. In addition to speaking at local and regional events, Bogue speaks at national conferences. He blogs at www.thorprojects.com/blog , and you can reach him at Rob.Bogue@thorprojects.com .