STEVEN MANN

PowerShell™ for SharePoint® 2010 HOW-TO

SAMS | 800 East 96th Street, Indianapolis, Indiana 46240 USA

PowerShell™ for SharePoint® 2010 How-To

ISBN-10: 0-672-33559-X

ISBN-13: 978-0-672-33559-4

Library of Congress Cataloging-in-Publication Data

Mann, Steve, 1950-
PowerShell for SharePoint 2010 : how to / Steven Mann.
p. cm.
ISBN 978-0-672-33559-4
1. Intranets (Computer networks)–Management–Miscellanea. 2. Microsoft SharePoint (Electronic resource)–Miscellanea. 3. Groupware (Computer software)–Miscellanea. 4. Windows PowerShell (Computer program language)–Miscellanea. 5. Microsoft Windows (Computer file)–Miscellanea. I. Title.
TK5105.875.I6M365 2011
006.7'8–dc23
2011020669

Printed in the United States of America

First Printing: June 2011

Editor-in-Chief
Greg Wiegand

Executive Editor
Neil Rowe

Development Editor
Mark Renfrow

Managing Editor
Kristy Hart

Project Editor
Andy Beaster

Copy Editor
Bart Reed

Indexer
Tim Wright

Proofreader
Debbie Williams

Technical Editor
J. Boyd Nolan

Team Coordinator
Cindy Teeters

Book Designer
Gary Adair

Composition
Gloria Schurick

Contents at a Glance

Table of Contents

PART I

Getting Started

CHAPTER 1

SharePoint 2010 Management Shell

IN THIS CHAPTER

- How Do I Run PowerShell?
- How Do I Make PowerShell Aware of SharePoint?
- What Is the SharePoint 2010 Management Shell?
- What Is the SharePoint 2010 Management Shell Loading?

This chapter is an overview of the PowerShell and SharePoint 2010 Management Shell console applications. It provides background information on the logistics behind running scripts for SharePoint.

How Do I Run PowerShell?

On every Windows Server running a variation of Windows Server 2008 or higher, PowerShell is installed and can be used for administration purposes. The PowerShell application, by default, appears in the Windows taskbar. Right-clicking the application button displays the available menu options, as shown in Figure 1.1.

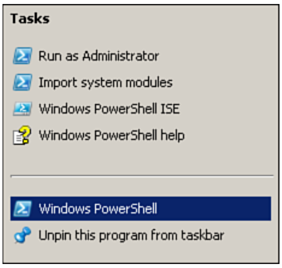


FIGURE 1.1 The PowerShell application button provides a menu of options.

NOTE The OS used to demonstrate the PowerShell application pinned to the taskbar is Windows Server 2008 R2. Not all OS installations and/or configurations may reproduce the displayed behavior.

Selecting the Import System Modules menu item launches PowerShell and loads up all modules from all services and server applications running on the current server. This is useful when you're scripting out code for more than one set of services (for example, SharePoint and SQL Server).

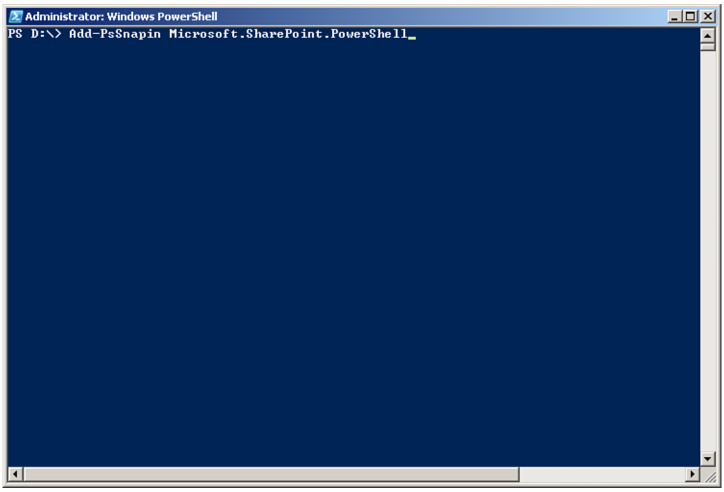
Selecting the Windows PowerShell ISE menu item launches a pseudo-development environment where you may create, execute, test, and debug PowerShell scripts. It is a glorified text editor but is helpful in creating scripts and running them.

Selecting the Windows PowerShell menu item launches PowerShell in plain-vanilla form (no additional modules are supported for the server applications that are loaded, including SharePoint). If you are not logged on as an administrator, you may select the Run As Administrator menu item to launch PowerShell with administrative privileges.

How Do I Make PowerShell Aware of SharePoint?

Launching PowerShell on the server opens a blue console application window. Without any modules imported, this instance does not know how to execute any SharePoint commands. To make this PowerShell instance aware of SharePoint, you must add the SharePoint PowerShell snap-in using the following command (as shown in Figure 1.2):

```
Add-PsSnapIn Microsoft.SharePoint.PowerShell
```



```
PS D:\> Add-PsSnapin Microsoft.SharePoint.PowerShell_
```

FIGURE 1.2 Adding the SharePoint PowerShell snap-in

What Is the SharePoint 2010 Management Shell?

The SharePoint 2010 Management Shell is a variation of PowerShell geared toward the administration and management of SharePoint. Once SharePoint is installed on the server, the management console is available from the Start menu, as shown in Figure 1.3.

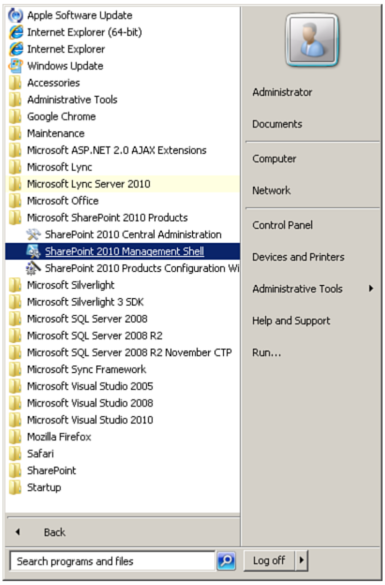
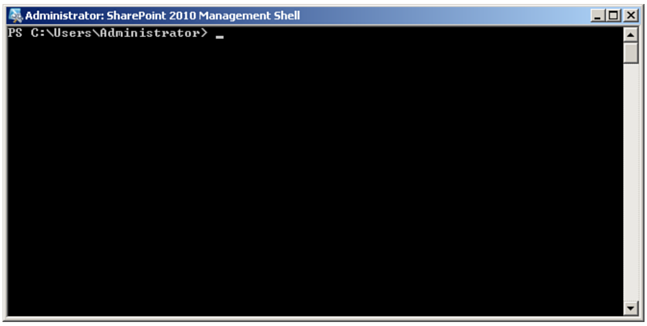

Apple Software Update
Internet Explorer (64-bit)
Internet Explorer
Windows Update
Accessories
Administrative Tools
Google Chrome
Maintenance
Microsoft ASP.NET 2.0 AJAX Extensions
Microsoft Lync
Microsoft Lync Server 2010
Microsoft Office
Microsoft SharePoint 2010 Products
SharePoint 2010 Central Administration
SharePoint 2010 Management Shell
SharePoint 2010 Products Configuration Wi
Microsoft Silverlight
Microsoft Silverlight 3 SDK
Microsoft SQL Server 2008
Microsoft SQL Server 2008 R2
Microsoft SQL Server 2008 R2 November CTP
Microsoft Sync Framework
Microsoft Visual Studio 2005
Microsoft Visual Studio 2008
Microsoft Visual Studio 2010
Mozilla Firefox
Safari
SharePoint
Startup
Back
Search programs and files
Administrator
Documents
Computer
Network
Control Panel
Devices and Printers
Administrative Tools
Help and Support
Run...
Log off

FIGURE 1.3 SharePoint 2010 Management Shell on the Start menu

Selecting the SharePoint 2010 Management Shell menu item launches the management console, as shown in Figure 1.4.



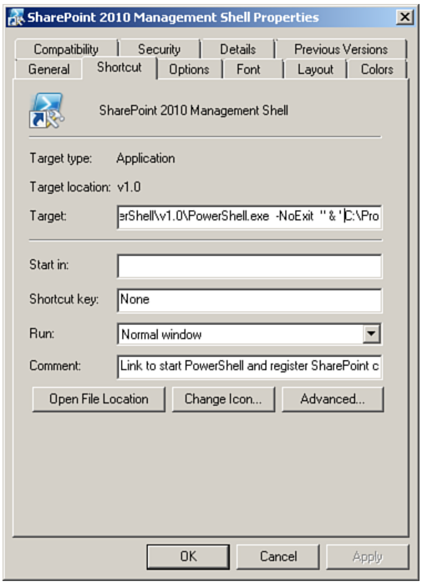
```
PS C:\Users\Administrator> _
```

FIGURE 1.4 SharePoint 2010 Management Shell console window

The management console window knows about SharePoint commands as well as general PowerShell commands, and it may be used to execute command lines or scripts. Some PowerShell basics are outlined in Chapter 2, "PowerShell Basics."

What Is the SharePoint 2010 Management Shell Loading?

If you dig a little deeper and look at the properties of the SharePoint 2010 Management Shell menu item, you will notice that the SharePoint 2010 Management Shell does nothing more than run PowerShell with a SharePoint script, as shown in Figure 1.5.



Target type: Application

Target location: v1.0

Target: erShell\v1.0\PowerShell.exe -NoExit " & ' C:\Pro

Start in:

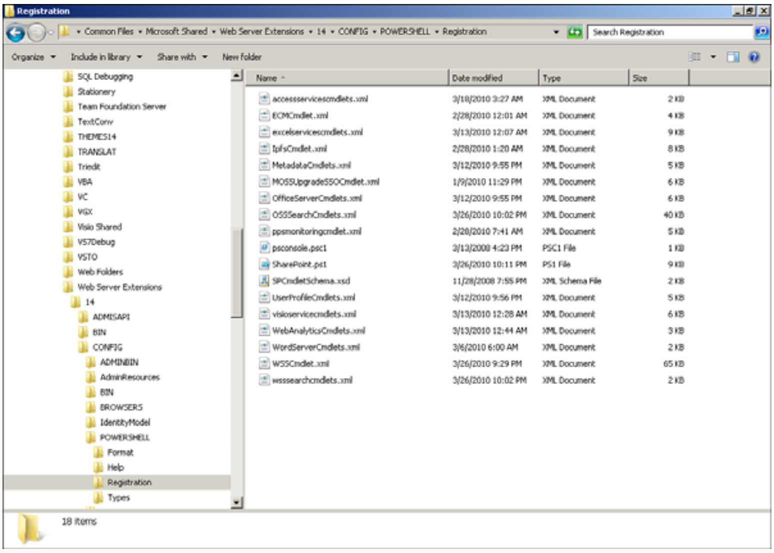
Shortcut key: None

Run: Normal window

Comment: Link to start PowerShell and register SharePoint c

FIGURE 1.5 SharePoint 2010 Management Shell properties

The SharePoint script that the menu item is loading is located within the SharePoint "14 hive" under the PowerShell Registration folder, as shown in Figure 1.6.



Name	Date modified	Type	Size
accessservicescmdlets.xml	3/18/2010 3:27 AM	XML Document	2 KB
ECMCmdlet.xml	2/28/2010 12:01 AM	XML Document	4 KB
excelservicescmdlets.xml	3/13/2010 12:07 AM	XML Document	9 KB
IpfsCmdlet.xml	2/28/2010 1:20 AM	XML Document	8 KB
MetadataCmdlets.xml	3/12/2010 9:55 PM	XML Document	5 KB
MOSSUpgradeSSOCmdlet.xml	1/9/2010 11:29 PM	XML Document	6 KB
OfficeServerCmdlets.xml	3/12/2010 9:55 PM	XML Document	6 KB
OSSSearchCmdlets.xml	3/26/2010 10:02 PM	XML Document	40 KB
ppsmonitoringcmdlet.xml	2/20/2010 7:41 AM	XML Document	5 KB
psconsole.psc1	3/13/2008 4:23 PM	PSC1 File	1 KB
SharePoint.ps1	3/26/2010 10:11 PM	PS1 File	9 KB
SPCmdletSchema.xsd	11/28/2008 7:55 PM	XML Schema File	2 KB
UserProfileCmdlets.xml	3/12/2010 9:56 PM	XML Document	5 KB
visioservicecmdlets.xml	3/13/2010 12:28 AM	XML Document	6 KB
WebAnalyticsCmdlets.xml	3/13/2010 12:44 AM	XML Document	3 KB
WordServerCmdlets.xml	3/6/2010 6:00 AM	XML Document	2 KB
WSSCmdlet.xml	3/26/2010 9:29 PM	XML Document	65 KB
wsssearchcmdlets.xml	3/26/2010 10:02 PM	XML Document	2 KB

FIGURE 1.6 SharePoint 2010 PowerShell Registration folder contents

TIP Crack open some of the files (with Notepad or a similar text editor) located in the registration folder to gain some insights on the SharePoint commands and operations.

CHAPTER 2

PowerShell Basics

IN THIS CHAPTER

This chapter explains some of the PowerShell basics such that a general understanding may be acquired. The sections and scenarios in this chapter help form a foundation for the tasks and solutions within this book.

What Is a Cmdlet?

PowerShell commands are called *cmdlets*. These are structured by a verb and noun concatenation in the form of verb-noun. So in the cmdlet `Get-Help`, `Get` is the verb and `Help` is the noun.

TIP Use the `Get-Help` cmdlet to view helpful information in working with cmdlets.

How Can I See the Possible Verbs for a Noun Command?

If you know the noun but are unsure of the possible verbs available, you can enter the noun with a `"-?"` to display the available verb-noun combinations. This may also provide additional nouns that are similar to the one you provided. For example, `SPSite -?` not only displays the available verbs for the `SPSite` noun but also for `SPSiteAdministration`, `SPSiteSubscription`, and other cmdlets available beginning with `SPSite`.

What Is a Parameter?

Most cmdlets need values to perform the desired actions. These values are provided to the cmdlet using parameters in the following form:

```
-parametername <value>
```

What Is a Switch Parameter?

A *switch parameter* is a parameter used with a cmdlet that does not take a value (or is Boolean in nature and can be set to a “false” setting). The fact that the switch parameter is present “switches” that option on (“true” setting) when executing the cmdlet.

Most of the time switch parameters are optional, but in some cases you may need to provide one switch or another. This is dependent on the cmdlet being executed.

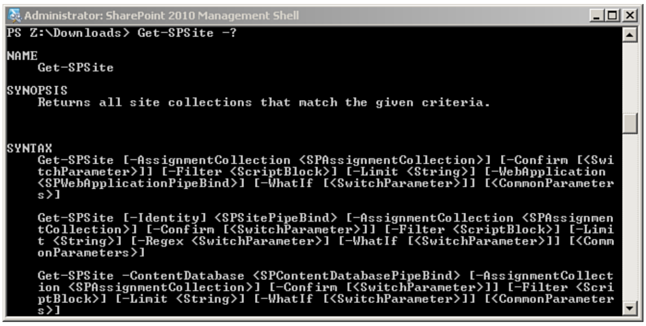
How Can I See the Possible Parameters for a Cmdlet?

Entering the cmdlet in the console with `"-?"` as a parameter displays information about that cmdlet along with the available parameters. An example is shown in Listing 2.1.

LISTING 2.1 **Getting Information about a Cmdlet**

```
Get-SPSite -?
```

Some cmdlets, such as `Get-SPSite`, have different variations of parameters. These variations will also be displayed within the information provided by the `-?` parameter, as shown in Figure 2.1.


Administrator: SharePoint 2010 Management Shell

```
PS Z:\Downloads> Get-SPSite -?

NAME
    Get-SPSite

SYNOPSIS
    Returns all site collections that match the given criteria.


SYNTAX
    Get-SPSite [-AssignmentCollection <SPAssignmentCollection>] [-Confirm [<Swi
    tchParameter>]] [-Filter <ScriptBlock>] [-Limit <String>] [-WebApplication
    <SPWebApplicationPipeBind>] [-WhatIf [<SwitchParameter>]] [<CommonParameter
    s>]

    Get-SPSite [-Identity] <SPSitePipeBind> [-AssignmentCollection <SPAssignmen
    tCollection>] [-Confirm [<SwitchParameter>]] [-Filter <ScriptBlock>] [-Limi
    t <String>] [-Regex <SwitchParameter>] [-WhatIf [<SwitchParameter>]] [<Comm
    onParameters>]

    Get-SPSite -ContentDatabase <SPContentDatabasePipeBind> [-AssignmentCollect
    ion <SPAssignmentCollection>] [-Confirm [<SwitchParameter>]] [-Filter <Scri
    ptBlock>] [-Limit <String>] [-WhatIf [<SwitchParameter>]] [<CommonParameter
    s>]
```

FIGURE 2.1 Using `-?` with a cmdlet provides various syntax information.

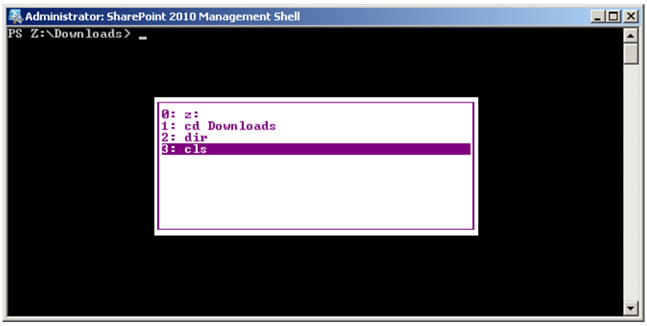
What Does F3 Do?

Pressing the F3 function key in the console window displays the last executed statement on the current prompt line. Using the up and down arrows will page through the other previous statements. This is useful for repeating or correcting previous command entries. Use the left and right arrow keys to place the cursor within the command text for corrections.

TIP Pressing the Insert key on the keyboard allows you to overwrite text within the command text without pushing the text out.

What Does F7 Do?

This is probably the best-kept secret in command consoles. Every time I show this to someone, that person had no idea it was available. If you have been entering several commands in a console screen, pressing the F7 function key will display a menu of those previously executed commands, as shown in Figure 2.2.



```
PS Z:\Downloads> _
```

```
0: z:
1: cd Downloads
2: dir
3: cls
```

FIGURE 2.2 Pressing the F7 function key presents a command history menu.

Use the arrow keys to change the selection in the menu. Pressing Enter will execute the command selected. Also, pressing F9 on the menu allows you to enter the command number you wish to reexecute but only places it on the line (that is, it does not execute the command automatically).

What Are Console Commands?

There are a handful of (what I like to call) console commands that stem from the good-ole DOS days. They have been available in command prompts ever since, with the current PowerShell consoles included. These commands provide standard directory navigation, file handling, and screen handling.

Here are several common console commands:

- `cd`—Change directory.
- `cls`—Clear the console screen.
- `dir`—Display a listing of the current directory (folder).
- `type`—Used with a text-based filename. This will display the contents of a text file.

TIP These commands have been grouped within PowerShell as aliases. To see all of the available aliases, use `Get-Alias`.

Path Environment Variable

Scenario/Problem: You have scripts within a local folder that you want PowerShell to recognize, such that you do not need to use the full pathname.

Solution: Add the local directory path to the `PSModulePath` variable.

You may add the local folder's path to the `PSModule` environment variable by entering the following command in the PowerShell command prompt (substituting <folder path> with the local directory path), as shown in Listing 2.2.

LISTING 2.2 **Adding a Folder Path to the PSModulePath**

```
$env:PSModulePath = $env:PSModulePath + ";c:\<folder path>"
```

TIP Add the same exact command line into the PowerShell profile configuration file to always have your local folder added to the `PSModulePath`. See http://technet.microsoft.com/en-us/library/dd315342.aspx for more information about PowerShell profiles.

Run Unsigned Scripts

Scenario/Problem: You need to be able to run unsigned scripts within PowerShell.

Solution: Use the `Set-ExecutionPolicy` command.

In order to be allowed to run unsigned scripts, you must change the execution policy:

```
Set-ExecutionPolicy remotesigned
```

You will be prompted with a confirmation. Enter **Y** and press Enter (or just press Enter since Y is the default).

In some cases you may need to set the policy to `Unrestricted`:

```
Set-ExecutionPolicy unrestricted
```

TIP For more information on signing scripts, see http://technet.microsoft.com/en-us/library/dd347649.aspx.

Disable the Confirmation Prompt

Scenario/Problem: You need to automatically confirm operations such that a script runs unattended.

Solution: Include the `–Confirm` parameter with a `$false` setting.

When running cmdlets discussed in this book, you may receive a confirmation message to confirm the action being processed. If you use these cmdlets in scripts, the script will not run straight through without prompting for the confirmation.

The cmdlets that have a `-Confirm` optional parameter that can be called in unattended mode by passing in `-Confirm:$false` to the cmdlet, as shown in Listing 2.3.

LISTING 2.3 **Suppressing the Confirmation Prompt**

```
Remove-item *.png -Confirm:$false
```

NOTE Not all cmdlets have a `-Confirm` parameter. Use the `help (-?)` parameter to verify each cmdlet in question.

Generate Inline Credentials

Scenario/Problem: You need to provide cmdlets with credentials without prompting for them.

Solution: Create a new `PSCredential` object.

Various cmdlets discussed in this book require SharePoint and/or SQL Server account credentials for proper authentication when you are performing the desired operations. The examples will include using `(Get-Credential)` as the parameter value, which prompts the user for credentials.

When you are running these cmdlets in scripts, the prompting for credentials pauses the execution. If you need the script to run straight through, instead of using `Get-Credential`, you may generate a new `PSCredential` object inline, as shown in Listing 2.4.

LISTING 2.4 **Generating Inline Credentials**

```
(New-Object System.Management.Automation.PSCredential
"domain\user", (ConvertTo-SecureString "password" -AsPlainText
-Force))
```

Therefore, simply substitute `(Get-Credential)` for the text shown in Listing 2.4 with the proper username and password when you are using a cmdlet with authentication requirements.

Reference an Assembly

Scenario/Problem: You want to be able to call methods and/or use objects within a trusted assembly (DLL).

Solution: Use a reflection declaration and then use a variable to instantiate an object.

You may reference an assembly in your PowerShell script and then use any objects available by assigning them to a variable. Any methods or properties can then be used within your script. There is no IntelliSense, so you need to know the object model beforehand. Listing 2.5 demonstrates a sample assembly reference and usage.

LISTING 2.5 **Referencing and Using a SharePoint Assembly**

```
[Void][System.Reflection.Assembly]::LoadWithPartialName(
"Microsoft.SharePoint")

[Void][System.Reflection.Assembly]::LoadWithPartialName(
"Microsoft.SharePoint.Administration")

$spFarm = [Microsoft.SharePoint.Administration.SPFarm]::Local
$spFarmSolutions = $spFarm.Solutions
```

TIP Instead of coding and compiling a console application, use a PowerShell script that references the SharePoint assemblies.

PART II

Server Management

CHAPTER 3

SharePoint Installation and Configuration

IN THIS CHAPTER

- Prepare the Microsoft SharePoint Installation Module
- Install SharePoint Unattended
- Configure a New SharePoint Farm
- Join a Server to the SharePoint Farm
- Create a New Web Application
- Create a New Site Collection
- Create a New Subsite

This chapter explores how you may use PowerShell scripts to perform the installation and configuration of SharePoint 2010. Microsoft provides a set of tools that aids in the installation. Use these procedures to script out and run unattended installations.

Prepare the Microsoft SharePoint Installation Module

Scenario/Problem: You need to have access to extended SharePoint installation commands within PowerShell.

Solution: Download and install the SPModule from Microsoft.

Before you can install SharePoint from within PowerShell, you need to download and install the SPModule code from Microsoft. The downloadable file can be found at http://www.microsoft.com/downloads/en/details.aspx?FamilyID=c57556ff-8df0-44fd-aba6-3df01b9f80ce (or search for SPModule Powershell).

Unzip the file and place it into a local folder. Add this folder to the PSModulePath environment variable. The code is unsigned, so you must allow unsigned scripts to run. See Chapter 2, “PowerShell Basics,” for steps on adding an environment variable and allowing the execution of unsigned scripts.

Launch Windows PowerShell by right-clicking the icon and selecting Run As Administrator. Import the modules extracted from the zipped SPModule by using the commands in Listing 3.1.

LISTING 3.1 **Importing the SPModule Components**

```
Import-Module SPModule.misc
Import-Module SPModule.setup
```

TIP You will need to set your execution policy to unrestricted in order to successfully import the SPModule.misc and SPModule.setup files. You will therefore be prompted on the import of some of the scripts. Entering an `R` for Run Once will install the scripts properly.

The cmdlets to install and configure SharePoint are now installed and ready for use.

NOTE Importing the SPModule.misc file performs a check for updates. If updates are available, you will be notified. Perform any updates before proceeding.

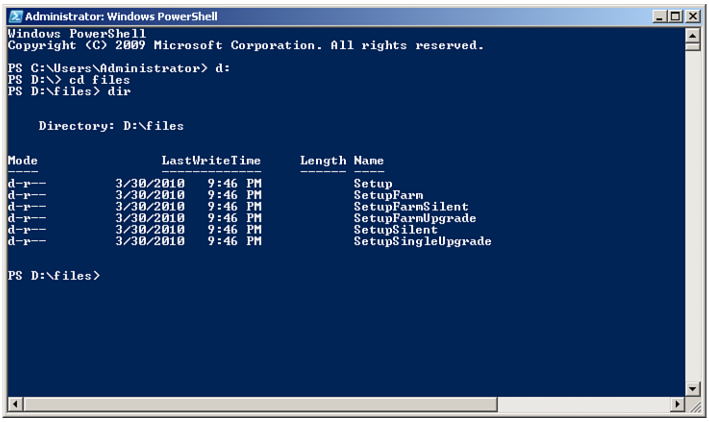
Install SharePoint Unattended

Scenario/Problem: You need to install SharePoint from PowerShell.

Solution: Use the `Install-SharePoint` cmdlet from the SPModule.

The first step in unattended installation and configuration is to actually install SharePoint onto the server. The `Install-SharePoint` cmdlet is provided by the SPModule to perform the installation. This cmdlet is used in conjunction with a setup configuration file that contains the installation parameters.

Sample setup configuration files are included on the SharePoint installation disc. They are contained within their own folder within Files, as shown in Figure 3.1. An example of the configuration file contents is shown in Listing 3.2.



```
Windows PowerShell
Copyright (C) 2009 Microsoft Corporation. All rights reserved.

PS C:\Users\Administrator> d:
PS D:\> cd files
PS D:\files> dir


    Directory: D:\files


Mode                LastWriteTime     Length Name
----                -------------     ------ ----
d-r--         3/30/2010   9:46 PM            Setup
d-r--         3/30/2010   9:46 PM            SetupFarm
d-r--         3/30/2010   9:46 PM            SetupFarmSilent
d-r--         3/30/2010   9:46 PM            SetupFarmUpgrade
d-r--         3/30/2010   9:46 PM            SetupSilent
d-r--         3/30/2010   9:46 PM            SetupSingleUpgrade


PS D:\files>
```

FIGURE 3.1 Sample configuration files are contained within the folders shown.

LISTING 3.2 **Sample Farm Setup Configuration File**

```
<Configuration>
    <Package Id="sts">
        <Setting Id="LAUNCHEDFROMSETUPSTS" Value="Yes"/>
    </Package>

    <Package Id="spswfe">
        <Setting Id="SETUPCALLED" Value="1"/>
    </Package>

    <Logging Type="verbose" Path="%temp%"
Template="SharePoint Server Setup(*).log"/>
    <!--<PIDKEY Value="Enter Product Key Here" />-->
    <Setting Id="SERVERROLE" Value="APPLICATION"/>
    <Setting Id="USINGUIINSTALLMODE" Value="1"/>
    <Setting Id="SETUPTYPE" Value="CLEAN_INSTALL"/>
    <Setting Id="SETUP_REBOOT" Value="Never"/>
</Configuration>
```

Copy the config.xml file that matches your intended installation to a local drive location. Open the file in Notepad and uncomment the `PIDKEY` line by removing the `"<!--"` from the front and `"-->"` from the back. Enter your SharePoint product key in place of "Enter Product Key Here." Save your changes.

You may now use this configuration file with the Import-SharePoint cmdlet, which has the following syntax:

```
Install-SharePoint –setupexepath <path to setup.exe on disc>
-configxmlpath <path to the setup config.xml>
```

A sample command-line entry is shown in Listing 3.3.

LISTING 3.3 **Installing SharePoint from the Command Line**

```
Install-SharePoint –setupexepath D:\setup.exe –configxmlpath
c:\config.xml
```

SharePoint will be installed on the server accordingly.

TIP Install SharePoint on all servers that will participate in the farm (except for the database server) before configuring a new farm or joining to an existing farm.

Installing Without a Product Key in the Configuration File

For security purposes, you may not want to store the product key within the configuration file. Instead you can use the `–PIDKey` parameter followed by your product key in quotes on the command line with the `Install-SharePoint` cmdlet, as shown in Listing 3.4.

LISTING 3.4 **Installing SharePoint from the Command Line with a Product Key**

```
Install-SharePoint –setupexepath D:\setup.exe –configxmlpath
c:\config.xml -PIDKey "ABDCE-FGHIJ-KLMNO-PQRST-UVWXY"
```

Configure a New SharePoint Farm

Scenario/Problem: You need to configure a new SharePoint farm from the command line.

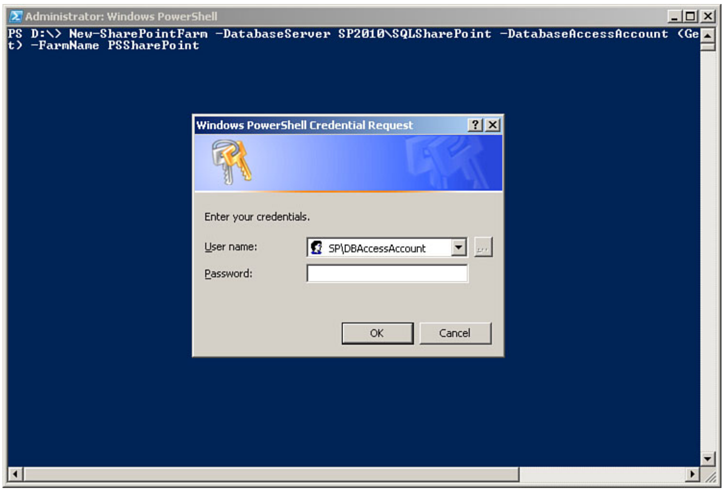
Solution: Use the `New-SharePointFarm` cmdlet from the SPModule.

The `New-SharePointFarm` cmdlet allows you to configure a new SharePoint farm on the server. The syntax is as follows:

```
New-SharePointFarm -databaseserver <server name>
-databaseaccessaccount (Get-Credential <domain\account>)
-farmname <name of new farm>
-passphrase (ConvertTo-SecureString -asplaintext -force -string "Enter a
farm pass phrase here")
```

This command requires credentials for the database server. Therefore, the `Get-Credential` PowerShell command is used in conjunction with `New-SharePointFarm`. After you enter the command, PowerShell will prompt for the password of the database access account entered, as shown in Figure 3.2.

TIP Use inline credentials instead of `(Get-Credential)`. See Chapter 2 for more details.



```
PS D:\> New-SharePointFarm -DatabaseServer SP2010\SQLSharePoint -DatabaseAccessAccount (Ge
t) -FarmName PSSharePoint
```

Windows PowerShell Credential Request

Enter your credentials.

User name: SP\DBAccessAccount

Password:

FIGURE 3.2 Using `Get-Credential` prompts for the corresponding password.

Once the credentials have been entered, the new farm will be configured using the database server entered. This essentially creates the SharePoint_Config database and prepares the farm for use.

NOTE The farm passphrase is a password that is used when you are adding new servers to the farm or making farm configuration changes. It needs to be entered into the cmdlet as a secure string; therefore, the `ConvertTo-SecureString` cmdlet is also used in conjunction with `New_SharePointFarm`.

Join a Server to the SharePoint Farm

Scenario/Problem: You need to join the servers to the new SharePoint farm.

Solution: Use the `Join-SharePointFarm` cmdlet from the SPModule.

The previous cmdlet (`New-SharePointFarm`) created the database and configured the SharePoint farm. You cannot run this again on the other servers. Instead, you need to join the servers to the farm using the `Join-SharePointFarm` cmdlet:

```
Join-SharePointFarm –databaseserver <database server name>
-configurationdatabasename <config database>
-passphrase (ConvertTo-SecureString -asplaintext -force -string
"<farm pass phrase>")
```

The farm passphrase used with the `Join-SharePointFarm` cmdlet should be the same passphrase used when you are creating the new SharePoint farm.

Create a New Web Application

Scenario/Problem: You need to create a new web application for the SharePoint farm.

Solution: Use the `New-SPWebApplication` cmdlet from the SharePoint 2010 Management Shell.

You may now use the SharePoint 2010 Management Shell because SharePoint has been installed and the SPModule commands have been utilized. If you are running a full script and/or still wish to use the PowerShell console, be sure to add the SharePoint snap-in (`Add-PsSnapin Microsoft.SharePoint.PowerShell`) such that the SharePoint cmdlets are available.

To add a new web application, use the `New-SPWebApplication` cmdlet as follows:

```
New-SPWebApplication –applicationpool <name of new application pool>
-name <name of new web app>
-applicationpoolaccount <service account to be used for the
application pool> -port <desired port number>
```

A sample command-line entry for creating the main SharePoint web application is shown in Listing 3.5.

LISTING 3.5 **Creating a New Web Application**

```
New-SPWebApplication –applicationpool SharePoint-80
-name SharePoint-80 -applicationpoolaccount SP\AppPoolAccount -port 80
```

This process may take a few minutes. When it's completed, the display name and the URL of the new web application are presented in the console.

TIP If you do not include the port number, a random port number will be assigned. The first web application created will use port 80.

Create a New Site Collection

Scenario/Problem: You need to create a new top-level site collection.

Solution: Use the `New-SPSite` cmdlet from the SharePoint 2010 Management Shell.

To add a new site collection, use the `New-SPSite` cmdlet, as follows:

```
New-SPSite –url <full url of new site collection>
-name <name of new site collection>
-owneralias <site collection administrator>
-template <site collection template to use>
```

A sample command-line entry for creating the root site collection using the Team Site template is shown in Listing 3.6.

LISTING 3.6 **Creating the Root Top-level Site Collection**

```
New-SPSite –url http://sp2010/ -name Home -owneralias SP\SiteAdmin
-template STS#0
```

The site template names can be found using `Get-SPWebTemplates`. If no template is provided, the site collection will still be created. In this case, once the site is accessed via the browser, SharePoint will prompt for a site collection template to be selected.

TIP You can only create one site collection at the root of the web application. Subsequent site collections must be created using a managed path. See Chapter 4, "Farm Management," for more information on creating a managed path.

Create a New Subsite

Scenario/Problem: You want to create sites under your site collection.

Solution: Use the `New-SPWeb` cmdlet from the SharePoint 2010 Management Shell.

To add a new subsite, use the `New-SPWeb` cmdlet as follows:

```
New-SPWeb –url <full url of new site> -name <name of new site>
-template <site template to use>
```

A few switch parameters are available for the `New-SPWeb` cmdlet:

- `AddToTopNav`
- `UniquePermissions`
- `UseParentTopNav`

The `AddToTopNav` switch parameter places the new site within the top navigation of the site collection. Using the `UseParentTopNav` switch parameter replicates the top nav of the site collection onto the new subsite. Creating a site with `UniquePermissions` forces the site to not inherit permissions from the site collection and only grants the System Account Full Control access.

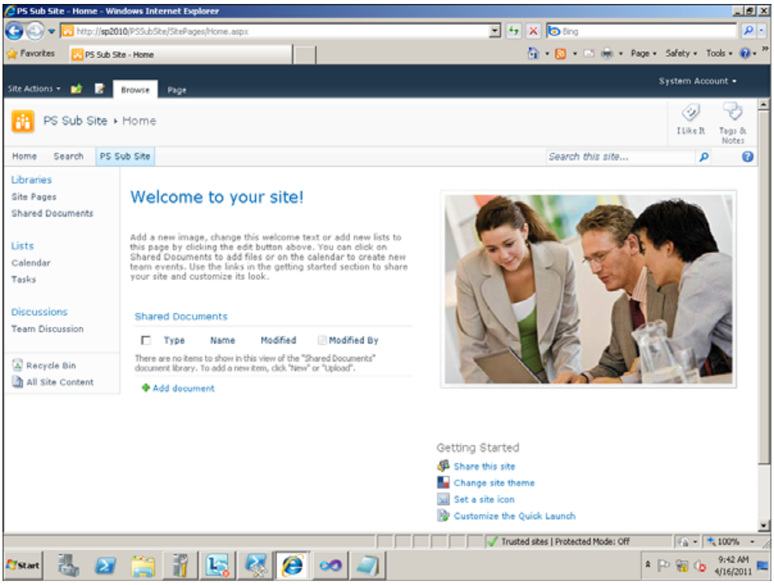
A sample command-line entry for creating a subsite using the Team Site template is shown in Listing 3.7.

LISTING 3.7 **Creating a Subsite in the Root Site Collection**

```
New-SPWeb –url http://sp2010/PSSubSite -name "PS Sub Site"
-template STS#0 –AddToTopNav –UniquePermissions -UseParentTopNav
```

The site template names can be found using `Get-SPWebTemplates`. If no template is provided, the site will still be created. In this case, once the site is accessed via the browser, SharePoint will prompt for a site template to be selected.

Using both `AddToTopNav` and `UseParentTopNav` ensures the new site always appears in the top navigation, as shown in Figure 3.3.



Add a new image, change this welcome text or add new lists to this page by clicking the edit button above. You can click on Shared Documents to add files or on the calendar to create new team events. Use the links in the getting started section to share your site and customize its look.

There are no items to show in this view of the "Shared Documents" document library. To add a new item, click "New" or "Upload".

FIGURE 3.3 Using the switch parameters places the subsite in the top nav.

CHAPTER 4

Farm Management

IN THIS CHAPTER

- Review Farm Configuration Values
- Set the Farm Configuration Values
- Refresh Installed Products
- Change the Port of Central Admin
- Change the Farm Passphrase
- Retrieve the System Accounts
- Retrieve Managed Accounts
- Rename a Site Template
- Rename a Server on the Farm
- Display the Configured Managed Paths
- Create a New Managed Path
- Remove a Managed Path
- Merge Log Files
- End the Current Log File
- Review SharePoint Designer Settings
- Configure SharePoint Designer Settings
- Review Workflow Configuration Settings
- Modify Workflow Configuration Settings
- Display Available Timer Jobs on the Farm
- Get a Specific Timer Job
- Enable a Timer Job
- Disable a Timer Job
- Start a Timer Job
- Set the Schedule for a Timer Job

This chapter outlines the SharePoint PowerShell commands that are used to manage general operations within your SharePoint farm. Whereas some operations are specific to the farm level, others are overall farm administration cmdlets that could not be categorized into a separate chapter.

Review Farm Configuration Values

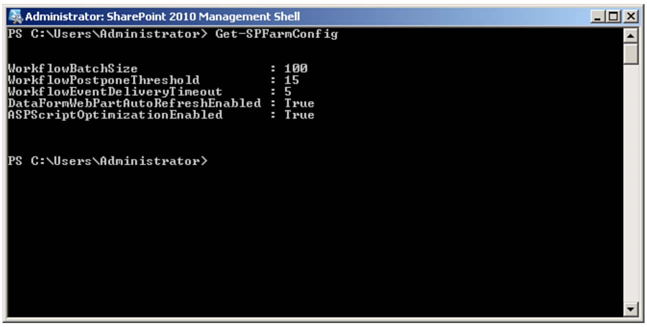
Scenario/Problem: You want to see the farm configuration values.

Solution: Use the `Get-SPFarmConfig` cmdlet without any parameters.

Several configuration settings are available using `SPFarmConfig`. Execute the `Get-SPFarmConfig` without any parameters, as shown in Listing 4.1, to display the current values. Some sample output is shown in Figure 4.1.

LISTING 4.1 **Displaying All Farm Configuration Values**

```
Get-SPFarmConfig
```



```
PS C:\Users\Administrator> Get-SPFarmConfig

WorkflowBatchSize                   : 100
WorkflowPostponeThreshold           : 15
WorkflowEventDeliveryTimeout        : 5
DataFormWebPartAutoRefreshEnabled   : True
ASPScriptOptimizationEnabled        : True


PS C:\Users\Administrator>
```

FIGURE 4.1 `Get-SPFarmConfig` displays the current farm values.

This is not the most exciting list of configuration values; however, they could impact the behavior of your SharePoint farm. The next section explains how to change these values.

Set the Farm Configuration Values

Scenario/Problem: You want to modify the farm configuration values displayed in the previous section.

Solution: Use the `Set-SPFarmConfig` cmdlet with the configuration parameters.

The values shown in the previous section may be modified using the `Set-SPFarmConfig` cmdlet with the associated parameter of each value. The parameters for these configuration values are as follows:

- `ASPScriptOptimizationEnabled`
- `DataFormWebPartAutoRefreshEnabled`
- `WorkflowBatchSize`
- `WorkflowEventDeliveryTimeout`
- `WorkflowPostponeThreshold`

The `ASPScriptOptimizationEnabled` parameter is a Boolean parameter that determines whether ASP scripting is allowed to be optimized by IIS.

The `DataFormWebPartAutoRefreshEnabled` parameter is a Boolean parameter that determines whether the data form web parts on the SharePoint web pages are allowed to automatically refresh using the asynchronous Ajax settings. If this value is set to `false`, the data form web parts will not automatically refresh regardless of the web part Ajax settings.

The `WorkflowBatchSize` value defaults to 100 and determines the amount of events that may be processed for a single workflow instance. The value must be greater than 0 but can be as large as the maximum integer value.

The `WorkflowEventDeliveryTimeout` parameter specifies the number of minutes that a workflow can run before it times out. The default value is 5 minutes but can be any positive integer. If the workflow times out, it is placed back into the queue to attempt execution again.

The `WorkflowPostponeThreshold` parameter determines how many workflows may operate at the same time. The default is 15 workflows but can be any integer value. If the number of workflows that need to run exceed this amount, the excess workflows are placed in the queue and must wait for the running workflows to finish.

If you haven't noticed, the parameter names are exactly the same as the output from `Get-SPFarmConfig`. A sample command line for setting these values is shown in Listing 4.2.

LISTING 4.2 **Setting the Farm Configuration Values**

```
Set-SPFarmConfig –ASPScriptOptimizationEnabled:$true
–DataFormWebPartAutoRefreshEnabled:$false –WorkflowBatchSize 50
–WorkflowEventDeliveryTimeout 20 –WorkflowPostponeThreshold 10
```

Run the `Get-SPFarmConfig` cmdlet to verify your changes.

NOTE The `ASPScriptOptimizationEnabled` parameter is not documented as part of the cmdlet help but is indeed an accepted Boolean parameter for `Set–SPFarmConfig`.

Refresh Installed Products

Scenario/Problem: You need to ensure that the SharePoint products installed on a farm server are properly registered.

Solution: Use the `Set-SPFarmConfig` cmdlet with the `InstalledProductsRefresh` switch parameter.

Using the `InstalledProductsRefresh` switch parameter with the `Set-SPFarmConfig` cmdlet refreshes the current server's license state with the products installed on the SharePoint farm. A sample command line is shown in Listing 4.3.

LISTING 4.3 **Refreshing the Installed Products**

```
Set-SPFarmConfig -InstalledProductsRefresh
```

Change the Port of Central Admin

Scenario/Problem: You want to change the HTTP port value for Central Administration.

Solution: Use the `Set-SPCentralAdministration` cmdlet with the `Port` parameter.

The `Set-SPCentralAdministration` cmdlet is solely used to change the port number configured for the Central Administration web application. Provide a valid port number, as shown in Listing 4.4.

LISTING 4.4 **Changing the Port of Central Admin**

```
Set-SPCentralAdministration –Port 20222
```

Change the Farm Passphrase

Scenario/Problem: You want to change the farm passphrase.

Solution: Use the `Set-SPPassPhrase` cmdlet with the `PassPhrase` parameter.

The `Set-SPPassPhrase` cmdlet is solely used to change the SharePoint farm passphrase. This passphrase was initially entered during the installation of the SharePoint farm. Provide a secure string value with the `PassPhrase` parameter, as shown in Listing 4.5.

LISTING 4.5 **Changing the Farm Passphrase**

```
Set-SPPassPhrase –PassPhrase (ConvertTo-SecureString -asplaintext
-force -string "SharePoint1")
```

For scripting purposes, if you do not wish to be prompted to confirm the passphrase entry, include the `ConfirmPassPhrase` parameter and switch the general confirmation off, as shown in Listing 4.6.

LISTING 4.6 **Changing the Farm Passphrase Without Prompts**

```
Set-SPPassPhrase –PassPhrase (ConvertTo-SecureString -asplaintext
-force -string "SharePoint1") –ConfirmPassPhrase
(ConvertTo-SecureString -asplaintext -force -string "SharePoint1")
–Confirm:$false
```

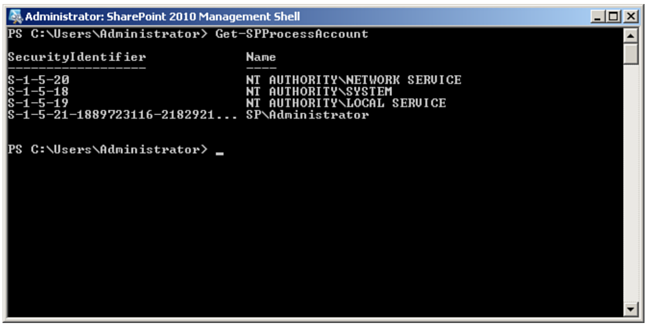
TIP The default operation performs the passphrase modification for the entire farm. If you run into issues, use the `LocalServerOnly` switch parameter to only perform the modification on the current server. Continue on each server in the farm to pinpoint the problem.

Retrieve the System Accounts

Scenario/Problem: You need to retrieve system account objects.

Solution: Use the `Get-SPProcessAccount` cmdlet.

Issuing the `Get-SPProcessAccount` cmdlet without any parameters displays all system and managed accounts on the SharePoint farm, as shown in Figure 4.2.



```
PS C:\Users\Administrator> Get-SPProcessAccount

SecurityIdentifier                   Name
------------------                   ----
S-1-5-20                             NT AUTHORITY\NETWORK SERVICE
S-1-5-18                             NT AUTHORITY\SYSTEM
S-1-5-19                             NT AUTHORITY\LOCAL SERVICE
S-1-5-21-1889723116-2182921...       SP\Administrator


PS C:\Users\Administrator> _
```

FIGURE 4.2 `Get-SPProcessAccount` displays all system and managed accounts.

Several mutually exclusive switch parameters may be used to retrieve a specific system account. The switch parameters are `LocalSystem`, `LocalService`, and `NetworkService`. Use these separately, as shown in Listing 4.7.

LISTING 4.7 **Retrieving Individual System Accounts**

```
Get-SPProcessAccount –LocalSystem
Get-SPProcessAccount -LocalService
Get-SPProcessAccount –NetworkService
```

TIP Pipe the result to other cmdlets that accept process accounts or assign the result to a variable for use within a cmdlet script.

Retrieve Managed Accounts

Scenario/Problem: You need to retrieve managed accounts on the farm.

Solution: Use the `Get-SPManagedAccount` cmdlet.

Issuing the `Get-SPManagedAccount` cmdlet without any parameters displays all managed accounts on the SharePoint farm. To retrieve a specific managed account, you may use the account name, as shown in Listing 4.8.

LISTING 4.8 **Retrieving a Specific Managed Account**

```
Get-SPManagedAccount "SP\Administrator"
```

TIP Pipe the result to other cmdlets that accept managed accounts or assign the result to a variable for use within a cmdlet script.

Rename a Site Template

Scenario/Problem: You want to rename an installed custom site template.

Solution: Use the `Set-SPWebTemplate` cmdlet.

The `Set-SPWebTemplate` cmdlet is used to update the name and/or description of an installed site template within your SharePoint farm. You need the identity of the site template to update. A sample command line is shown in Listing 4.9.

TIP Use `Get-SPWebTemplate` to list out the installed templates on the SharePoint farm.

LISTING 4.9 **Updating the Name and Description of a Site Template**

```
Set-SPWebTemplate –Identity "CUSTOMSITE#1" –Name "Custom Site"
–Description "Custom Site Template"
```

NOTE As the scenario suggests, you can only modify custom site templates; attempting to modify a system-based site template results in an error.

Rename a Server on the Farm

Scenario/Problem: You want to rename a server that participates in the SharePoint farm.

Solution: Use the `Rename-SPServer` cmdlet.

The `Rename-SPServer` cmdlet may be used to rename a server in the farm. This only renames the server from a SharePoint standpoint and does not actually rename the physical server. Use the current name as the `Identity` parameter and the new name for the `Name` parameter, as shown in Listing 4.10.

LISTING 4.10 **Renaming a Server on the Farm**

```
Rename-SPServer –Identity "OldServerName" –Name "NewServerName"
```

WARNING If the rename procedure does not complete successfully, the server being renamed may no longer function as a SharePoint server. Running the SharePoint 2010 Products Configuration Wizard may resolve any issues; however, there is the potential of configuration database corruption forcing you to rebuild the SharePoint farm.

Display the Configured Managed Paths

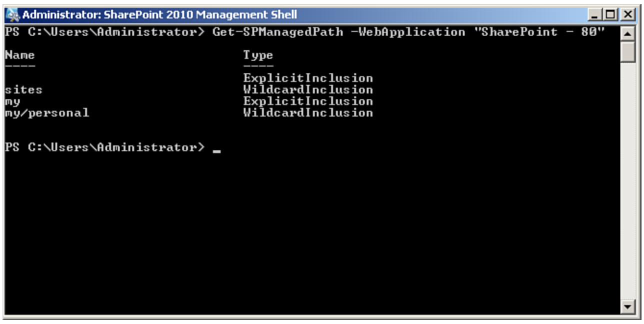
Scenario/Problem: You need to display the current managed paths configured on a web application in the farm.

Solution: Use the `Get-SPManagedPath` cmdlet with an associated web application.

Issuing the `Get-SPManagedPath` cmdlet with the `WebApplication` parameter (as shown in Listing 4.11) displays all configured managed paths on the SharePoint farm, as shown in Figure 4.3.

LISTING 4.11 **Retrieving Configured Managed Paths**

```
Get-SPManagedPath –WebApplication "SharePoint – 80"
```



```
PS C:\Users\Administrator> Get-SPManagedPath -WebApplication "SharePoint - 80"

Name                          Type
----                          ----
                              ExplicitInclusion
sites                         WildcardInclusion
my                            ExplicitInclusion
my/personal                   WildcardInclusion


PS C:\Users\Administrator> _
```

FIGURE 4.3 `Get-SPManagedPath` displays all configured managed paths.

Create a New Managed Path

Scenario/Problem: You want to create a new managed path in a web application on the SharePoint farm.

Solution: Use the `New-SPManagedPath` cmdlet.

Using the `New-SPManagedPath` cmdlet allows you to create a new managed path on the specified web application, as shown in Listing 4.12.

LISTING 4.12 **Creating a New Managed Path**

```
New-SPManagedPath –WebApplication "SharePoint – 80"
–RelativeURL "/teamsites/"
```

The forward slashes (/) in the name of the managed path are optional but do provide a visual queue (similar to manually creating a managed path through Central Administration).

By default, the managed path is created as a wildcard inclusion type. You may use the `Explicit` switch parameter to generate the managed path as an explicit inclusion type, as shown in Listing 4.13.

LISTING 4.13 **Creating a New Managed Path with Explicit Inclusion**

```
New-SPManagedPath –WebApplication "SharePoint – 80" –RelativeURL
"/departments/" -Explicit
```

TIP You may also create a managed path using the `HostHeader` parameter instead of the `WebApplication` parameter. `New-SPManagedPath` has two parameter sets for this purpose.

Remove a Managed Path

Scenario/Problem: You need to delete a managed path from a specific web application on your SharePoint farm.

Solution: Use the `Remove-SPManagedPath` cmdlet.

Using the `Remove-SPManagedPath` cmdlet allows you to remove a managed path from the specified web application, as shown in Listing 4.14.

LISTING 4.14 **Removing a Managed Path**

```
Remove-SPManagedPath –Identity "departments"
–WebApplication "SharePoint – 80"
```

The `Identity` parameter is simply the name of the relative URL that was used to generate the managed path, which becomes the name of the managed path. Use the `Get-SPManagedPath` cmdlet to review the names.

TIP You may also remove a managed path using the `HostHeader` parameter instead of the `WebApplication` parameter. `Remove-SPManagedPath` has two parameter sets for this purpose.

Merge Log Files

Scenario/Problem: You need to trace an issue but don’t want to have to review the logs on each server in the farm.

Solution: Use the `Merge-SPLogFile` cmdlet.

The `Merge-SPLogFile` cmdlet is a wonderful process that will consolidate the logs from all servers in the farm into one log file. This makes it easier to trace and find issues. A sample command line is shown in Listing 4.15.

LISTING 4.15 **Merging Log Files**

```
Merge-SPLogFile –Path E:\Logs\MergedLog.log –StartTime "1/19/2010"
–Overwrite
```

The `Path` parameter is the path and filename of the log file you wish to create. If it already exists, the cmdlet will fail unless you include the `Overwrite` switch parameter. You should use the `StartTime` parameter (and/or the `EndTime` parameter) to specify the range of the log entries that should be gathered and merged.

Several other parameters can help you narrow down the log search:

- `Area`
- `Category`
- `ContextFilter`
- `Correlation`
- `Level`
- `EventID`
- `Message`
- `Process`
- `ThreadID`

End the Current Log File

Scenario/Problem: You want SharePoint to be finished with the current log file and start a new one.

Solution: Use the `New-SPLogFile` cmdlet.

The `New-SPLogFile` cmdlet will stop the current log file and start up a new one. This is helpful if you are tracing an issue and do not need more entries in the log file or do not want any issues with the log file being in use. Simply issue the `New-SPLogFile` cmdlet without any parameters on the server you want to start a new log file, as shown in Listing 4.16.

LISTING 4.16 **Start a New Log File**

```
New-SPLogFile
```

Review SharePoint Designer Settings

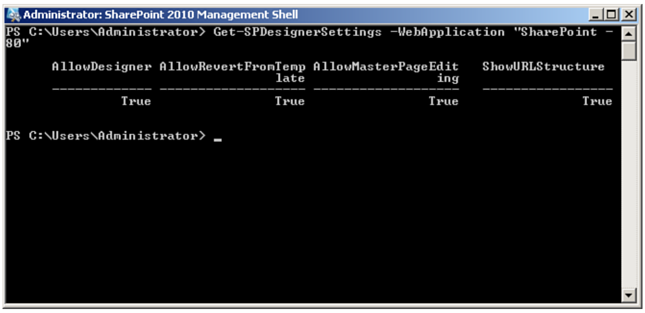
Scenario/Problem: You need to review the overall SharePoint Designer settings for a particular web application on the SharePoint farm.

Solution: Use the `Get-SPDesignerSettings` cmdlet.

The `Get-SPDesignerSettings` cmdlet allows you to review the current SharePoint Designer settings for a specified web application. Running this cmdlet with the `WebApplication` parameter, as shown in Listing 4.17, displays the current settings, as shown in Figure 4.4.

LISTING 4.17 **Reviewing the SharePoint Designer Settings**

```
Get-SPDesignerSettings –WebApplication "SharePoint – 80"
```



```
PS C:\Users\Administrator> Get-SPDesignerSettings -WebApplication "SharePoint -
80"

      AllowDesigner AllowRevertFromTemp AllowMasterPageEdit   ShowURLStructure
                                   late                 ing
      ------------- ------------------- -------------------   ----------------
               True                True                True               True


PS C:\Users\Administrator> _
```

FIGURE 4.4 `Get-SPDesignerSettings` displays the SharePoint Designer settings.

Configure SharePoint Designer Settings

Scenario/Problem: You need to modify the overall SharePoint Designer settings for a particular web application on the SharePoint farm.

Solution: Use the `Set-SPDesignerSettings` cmdlet.

The `Set-SPDesignerSettings` cmdlet allows you to modify the SharePoint Designer settings for a specific web application. Each setting, listed here, is a Boolean parameter:

- `AllowDesigner`
- `AllowMasterPageEditing`

- `AllowRevertFromTemplate`
- `ShowURLStructure`

A sample modification command line is shown in Listing 4.18.

LISTING 4.18 **Modifying the SharePoint Designer Settings**

```
Set-SPDesignerSettings –WebApplication "SharePoint – 80"
–AllowDesigner:$true –AllowMasterPageEditing:$false
–AllowRevertFromTemplate:$false –ShowURLStructure:$true
```

The `AllowDesigner` parameter determines whether the sites in the web application can be modified with SharePoint Designer. The user performing the action must have the proper permissions to modify the site. The default value is `True`.

The `AllowMasterPageEditing` parameter determines whether the master pages and page layouts in the sites of the web application can be modified with SharePoint Designer by site administrators or users with the appropriate permissions. The default value is `True`.

The `AllowRevertFromTemplate` parameter specifies whether site pages can be detached from the site definition using SharePoint Designer. The default value is `True`.

The `ShowURLStructure` parameter specifies whether the URL structure of the web application may be modified in SharePoint Designer. The default value is `True`.

TIP Use `Get-SPDesignerSettings` to validate the modifications.

Review Workflow Configuration Settings

Scenario/Problem: You need to review the overall workflow configuration settings for a particular web application on the SharePoint farm or a site collection.

Solution: Use the `Get-SPWorkflowConfig` cmdlet.

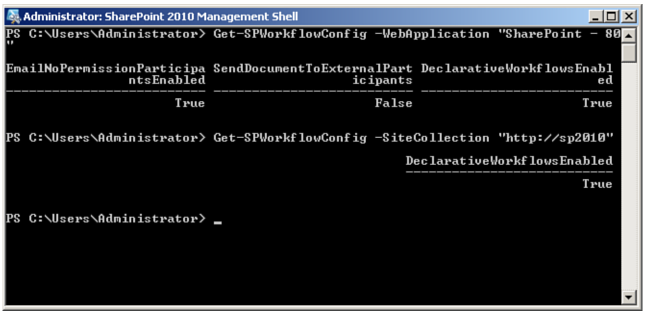
The `Get-SPWorkflowConfig` cmdlet allows you to review the current workflow configuration settings for a specified web application or site collection. Running this cmdlet with the `WebApplication` parameter is shown in Listing 4.19. The site collection command line is shown in Listing 4.20. Sample results of both command lines are shown in Figure 4.5.

LISTING 4.19 **Reviewing the Workflow Configuration for a Web Application**

```
Get-SPWorkflowConfig –WebApplication "SharePoint – 80"
```

LISTING 4.20 **Reviewing the Workflow Configuration for a Site Collection**

```
Get-SPWorkflowConfig –SiteCollection "http://sp2010"
```



```
PS C:\Users\Administrator> Get-SPWorkflowConfig -WebApplication "SharePoint - 80
"

EmailNoPermissionParticipa SendDocumentToExternalPart DeclarativeWorkflowsEnabl
             ntsEnabled                   icipants                        ed
-------------------------- -------------------------- --------------------------
                      True                      False                       True

PS C:\Users\Administrator> Get-SPWorkflowConfig -SiteCollection "http://sp2010"

                                                     DeclarativeWorkflowsEnabled
                                                     ---------------------------
                                                                            True

PS C:\Users\Administrator> _
```

FIGURE 4.5 `Get-SPWorkflowConfig` displays the workflow configuration settings.

Modify Workflow Configuration Settings

Scenario/Problem: You need to modify the overall SharePoint Designer settings for a particular web application on the SharePoint farm.

Solution: Use the `Set-SPWorkflowConfig` cmdlet.

The `Set-SPWorkflowConfig` cmdlet allows you to modify the workflow configuration settings for a specific web application or site collection. Each setting, listed here, is a Boolean parameter:

- `DeclarativeWorkflowsEnabled`
- `EmailNoPermissionParticipantsEnabled`
- `SendDocumentToExternalParticipants`

A sample modification command line for a web application is shown in Listing 4.21. Modification of the site collection workflow configuration is shown in Listing 4.22.

LISTING 4.21 **Modifying the Workflow Settings for a Web Application**

```
Set-SPWorkflowConfig –WebApplication "SharePoint – 80"
–DeclarativeWorkflowsEnabled:$true
–EmailNoPermissionParticipantsEnabled:$false
–SendDocumentToExternalParticipants:$false
```

LISTING 4.22 **Modifying the Workflow Settings for a Site Collection**

```
Set-SPWorkflowConfig –SiteCollection "http://sp2010"
–DeclarativeWorkflowsEnabled:$true
–EmailNoPermissionParticipantsEnabled:$false
–SendDocumentToExternalParticipants:$false
```

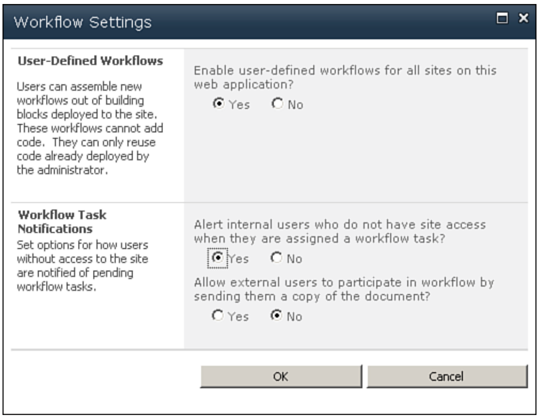
TIP Use `Get-SPWorkflowConfig` to validate the modifications. The site collection output only displays the `DeclarativeWorkflowEnabled` setting.

The `DeclarativeWorkflowEnabled` parameter determines whether users may define workflows from deployed managed code. The default value is `True`.

The `EmailNoPermissionParticipantsEnabled` parameter determines whether a user who does not have permissions to the site should receive an email if a task is assigned to him or her. The default value is `True`.

The `SendDocumentToExternalParticipants` parameter specifies whether external users should receive a copy of the workflow document. The default value is `False`.

These parameters correspond, respectively, to the workflow settings available in Central Administration, as shown in Figure 4.6.


Workflow Settings

User-Defined Workflows
Users can assemble new workflows out of building blocks deployed to the site. These workflows cannot add code. They can only reuse code already deployed by the administrator.

Enable user-defined workflows for all sites on this web application?
[✓ Yes] No

Workflow Task Notifications
Set options for how users without access to the site are notified of pending workflow tasks.

Alert internal users who do not have site access when they are assigned a workflow task?
[✓ Yes] No

Allow external users to participate in workflow by sending them a copy of the document?
Yes [✓ No]

FIGURE 4.6 Workflow settings for a web application in Central Administration.

Display Available Timer Jobs on the Farm

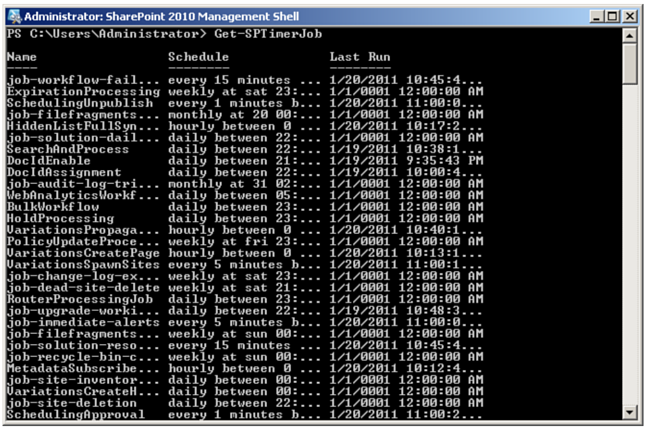
Scenario/Problem: You need to obtain a timer jobs reference and therefore need a listing of available timer jobs.

Solution: Use the `Get-SPTimerJob` cmdlet without any parameters to retrieve all available timer jobs.

In order to obtain a timer job reference object, you need to find the identity of the timer job. Using `Get-SPTimerJob` by itself (see Listing 4.23) displays all available timer jobs, as shown in Figure 4.7.

LISTING 4.23 **Displaying All Available Timer Jobs**

```
Get-SPTimerJob
```



Administrator: SharePoint 2010 Management Shell

```
PS C:\Users\Administrator> Get-SPTimerJob

Name                     Schedule                 Last Run
----                     --------                 --------
job-workflow-fail...     every 15 minutes ...     1/20/2011 10:45:4...
ExpirationProcessing     weekly at sat 23:...     1/1/0001 12:00:00 AM
SchedulingUnpublish      every 1 minutes b...     1/20/2011 11:00:0...
job-filefragments...     monthly at 20 00:...     1/1/0001 12:00:00 AM
HiddenListFullSyn...     hourly between 0 ...     1/20/2011 10:17:2...
job-solution-dail...     daily between 22:...     1/1/0001 12:00:00 AM
SearchAndProcess         daily between 22:...     1/19/2011 10:38:1...
DocIdEnable              daily between 21:...     1/19/2011 9:35:43 PM
DocIdAssignment          daily between 22:...     1/19/2011 10:00:4...
job-audit-log-tri...     monthly at 31 02:...     1/1/0001 12:00:00 AM
WebAnalyticsWorkf...     daily between 05:...     1/1/0001 12:00:00 AM
BulkWorkflow             daily between 23:...     1/1/0001 12:00:00 AM
HoldProcessing           daily between 23:...     1/1/0001 12:00:00 AM
VariationsPropaga...     hourly between 0 ...     1/20/2011 10:40:1...
PolicyUpdateProce...     weekly at fri 23:...     1/1/0001 12:00:00 AM
VariationsCreatePage     hourly between 0 ...     1/20/2011 10:13:1...
VariationsSpawnSites     every 5 minutes b...     1/20/2011 11:00:1...
job-change-log-ex...     weekly at sat 23:...     1/1/0001 12:00:00 AM
job-dead-site-delete     weekly at sat 21:...     1/1/0001 12:00:00 AM
RouterProcessingJob      daily between 23:...     1/1/0001 12:00:00 AM
job-upgrade-worki...     daily between 22:...     1/19/2011 10:48:3...
job-immediate-alerts     every 5 minutes b...     1/20/2011 11:00:0...
job-filefragments...     weekly at sun 00:...     1/1/0001 12:00:00 AM
job-solution-reso...     every 15 minutes ...     1/20/2011 10:45:4...
job-recycle-bin-c...     weekly at sun 00:...     1/1/0001 12:00:00 AM
MetadataSubscribe...     hourly between 0 ...     1/20/2011 10:12:4...
job-site-inventor...     daily between 00:...     1/1/0001 12:00:00 AM
VariationsCreateH...     daily between 00:...     1/1/0001 12:00:00 AM
job-site-deletion        daily between 22:...     1/1/0001 12:00:00 AM
SchedulingApproval       every 1 minutes b...     1/20/2011 11:00:2...
```

FIGURE 4.7 `Get-SPTimerJob` displays all available timer jobs.

Get a Specific Timer Job

Scenario/Problem: You need to obtain a specific timer job reference.

Solution: Use the `Get-SPTimerJob` cmdlet to assign a variable the results using the `–Identity` parameter.

Providing the identity of the timer job with the `Get-SPTimerJob` cmdlet retrieves a specific timer job. Use this to assign the results to an object for use in other cmdlets, as shown in Listing 4.24.

LISTING 4.24 **Assigning a Variable to a Specific Timer Job**

```
$timerJob = Get-SPTimerJob -Identity "DocIDEnable"
```

The `Identity` parameter in this case is the name shown when retrieving all timer jobs (`Get-SPTimerJob`). You may also use the GUID of the timer job.

Enable a Timer Job

Scenario/Problem: You want to enable a specific timer job.

Solution: Use the `Enable-SPTimerJob` cmdlet.

To enable a timer job, execute the `Enable-SPTimerJob` cmdlet with the `Identity` of the timer job or a timer job variable. A sample command line is shown in Listing 4.25.

LISTING 4.25 **Enabling a Timer Job**

```
$timerJob = Get-SPTimerJob -Identity "DocIDEnable"
Enable-SPTimerJob $timerJob
```

Disable a Timer Job

Scenario/Problem: You want to disable a specific timer job.

Solution: Use the `Disable-SPTimerJob` cmdlet.

To disable a timer job, execute the `Disable-SPTimerJob` cmdlet with the `Identity` of the timer job or a timer job variable. A sample command line is shown in Listing 4.26.

LISTING 4.26 **Disabling a Timer Job**

```
$timerJob = Get-SPTimerJob -Identity "DocIDEnable"
Disable-SPTimerJob $timerJob
```

Start a Timer Job

Scenario/Problem: You want to start a specific timer job.

Solution: Use the `Start-SPTimerJob` cmdlet.

To start a timer job, execute the `Start-SPTimerJob` cmdlet with the `Identity` of the timer job or a timer job variable. A sample command line is shown in Listing 4.27.

LISTING 4.27 **Starting a Timer Job**

```
$timerJob = Get-SPTimerJob -Identity "DocIDEnable"
Start-SPTimerJob $timerJob
```

The timer job will kick off once when you start it manually in this scenario. This is synonymous with selecting the Run Now button in Central Administration.

TIP You cannot start a timer job that is disabled. Use `Enable-SPTimerJob` first to ensure that the timer job is enabled before attempting to start the job.

Set the Schedule for a Timer Job

Scenario/Problem: You need to set or modify the schedule of a specific timer job.

Solution: Use the `Set-SPTimerJob` cmdlet.

Setting the schedule for a timer job can be facilitated by using the `Set-SPTimerJob` cmdlet. Execute the `Start-SPTimerJob` cmdlet with the Identity of the timer job or a timer job variable along with the `Schedule` parameter. A sample command line is shown in Listing 4.28.

LISTING 4.28 **Setting the Schedule for a Timer Job**

```
$timerJob = Get-SPTimerJob -Identity "DocIDEnable"
Set-SPTimerJob –Identity $timerJob –Schedule "Daily at 18:00:00"
```

The `Schedule` parameter is a string value but must conform to one of the following schedule string formats:

- Every X minutes between 0 and 59.
- Hourly between 0 and 59.
- Daily at hh:mm:ss.
- Weekly between <day> hh:mm:ss and <day> hh:mm:ss. (<day> is the three letter day of the week: Sun, Mon, Tue, Wed, Thu, Fri, or Sat.)
- Monthly at dd hh:mm:ss.
- Yearly at <month> dd hh:mm:ss. (<month> is the three letter month: Jan, Feb, Mar, and so on.)

CHAPTER 5

Web Application Management

IN THIS CHAPTER

- Display Available Web Applications on the Farm
- Get a Specific Web Application
- Remove a Web Application
- Create a New Web Application
- Configure Web Application Settings
- Extend a Web Application
- Create an Alternate Access Mapping URL
- Display All Alternate Access Mappings
- Get a Specific Alternate URL Entry
- Change the Zone of an Alternate Access Mapping
- Remove an Alternate Access Mapping

This chapter outlines the SharePoint PowerShell commands that are used to manage the web applications within your SharePoint farm.

Display Available Web Applications on the Farm

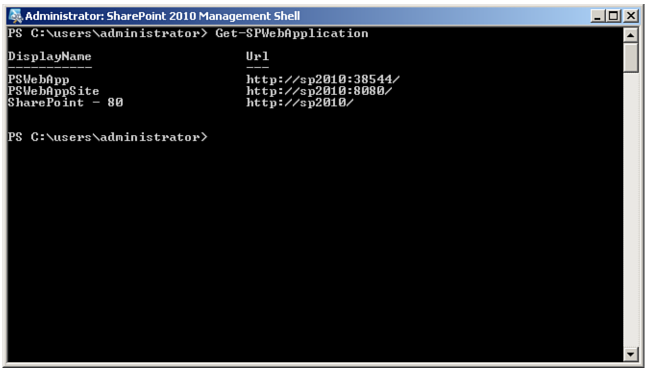
Scenario/Problem: You need to obtain a web application reference and therefore need a listing of available web applications.

Solution: Use the `Get-SPWebApplication` cmdlet without any parameters to retrieve all available web applications.

In order to obtain a web application reference object, you need to find the identity of the web application. Using the `Get-SPWebApplication` by itself (see Listing 5.1) displays all available web applications, as shown in Figure 5.1.

LISTING 5.1 **Displaying All Available Web Applications**

```
Get-SPWebApplication
```



```
PS C:\users\administrator> Get-SPWebApplication

DisplayName                     Url
-----------                     ---
PSWebApp                        http://sp2010:38544/
PSWebAppSite                    http://sp2010:8080/
SharePoint - 80                 http://sp2010/


PS C:\users\administrator>
```

FIGURE 5.1 The `Get-SPWebApplication` cmdlet displays all available web applications.

Get a Specific Web Application

Scenario/Problem: You need to obtain a specific web application reference.

Solution: Use the `Get-SPWebApplication` cmdlet to assign a variable the results using the `–Identity` parameter.

Providing the identity of the web application with the `Get-SPWebApplication` cmdlet retrieves a specific web application. Use this to assign the results to an object for use in other cmdlets, as shown in Listing 5.2.

LISTING 5.2 **Assigning a Variable to a Specific Web Application**

```
$webApp = Get-SPWebApplication -Identity "PSWebApp"
```

The `Identity` in this case can be the display name shown when retrieving all web applications (`Get-SPWebApplication`). You may also use the URL instead.

Remove a Web Application

Scenario/Problem: You need to remove a web application from the current server configuration.

Solution: Use the `Remove-SPWebApplication` cmdlet with a web application variable.

The `Remove-SPWebApplication` cmdlet allows you to remove a specific web application from the current server. Use this in conjunction with the `Get-SPWebApplication` cmdlet (explained in a previous section), as shown in Listing 5.3, or just use the web application identity directly in line, as shown in Listing 5.4.

LISTING 5.3 **Removing a Web Application Using a Variable**

```
$webApp = Get-SPWebApplication -Identity "PSWebApp"
Remove-SPWebApplication $webApp
```

LISTING 5.4 **Removing a Web Application Directly**

```
Remove-SPWebApplication –Identity "PSWebApp"
```

TIP You may also remove the underlying database(s) and data by using the –`RemoveContentDatabases` switch parameter. Only use this if you are no longer using the web application at all on the farm.

Create a New Web Application

Scenario/Problem: You need to create a new web application.

Solution: Use the `New-SPWebApplication` cmdlet.

To add a new web application, use the `New-SPWebApplication` cmdlet as follows:

```
New-SPWebApplication –applicationpool <name of new application pool>
-name <name of new web app> -applicationpoolaccount <service account
to be used for the application pool> -port <desired port number>
```

A sample command-line entry for creating the main SharePoint web application is shown in Listing 5.5.

LISTING 5.5 **Creating a New Web Application**

```
New-SPWebApplication –applicationpool NewWebApp -name NewWebApp
-applicationpoolaccount SP\AppPoolAccount -port 80
```

This process may take a few minutes. When it is completed, the display name and the URL of the new web application are presented in the console.

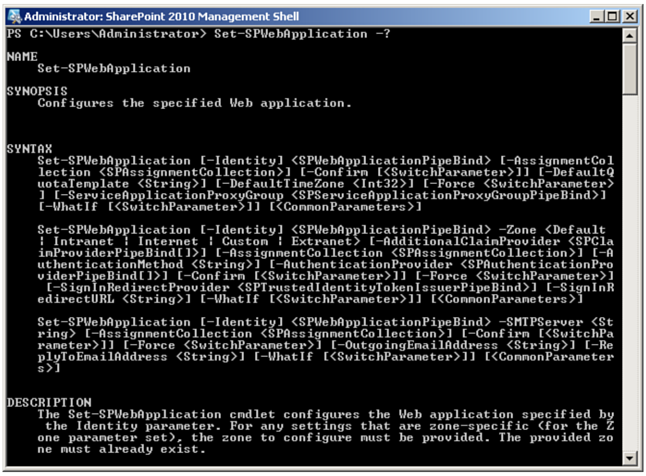
TIP If you do not include the port number, a random port number will be assigned. The first web application created will use port 80.

Configure Web Application Settings

Scenario/Problem: You need to modify the settings of a web application.

Solution: Use the `Set-SPWebApplication` cmdlet with a web application variable.

The `Set-SPWebApplication` cmdlet has several parameter sets for configuring various settings on the web application. Running `Set-SPWebApplication` with the `-?` parameter displays each parameter set, as shown in Figure 5.2.



```
PS C:\Users\Administrator> Set-SPWebApplication -?

NAME
    Set-SPWebApplication

SYNOPSIS
    Configures the specified Web application.


SYNTAX
    Set-SPWebApplication [-Identity] <SPWebApplicationPipeBind> [-AssignmentCol
    lection <SPAssignmentCollection>] [-Confirm [<SwitchParameter>]] [-DefaultQ
    uotaTemplate <String>] [-DefaultTimeZone <Int32>] [-Force <SwitchParameter>
    ] [-ServiceApplicationProxyGroup <SPServiceApplicationProxyGroupPipeBind>]
    [-WhatIf [<SwitchParameter>]] [<CommonParameters>]

    Set-SPWebApplication [-Identity] <SPWebApplicationPipeBind> -Zone <Default
    | Intranet | Internet | Custom | Extranet> [-AdditionalClaimProvider <SPCla
    imProviderPipeBind[]>] [-AssignmentCollection <SPAssignmentCollection>] [-A
    uthenticationMethod <String>] [-AuthenticationProvider <SPAuthenticationPro
    viderPipeBind[]>] [-Confirm [<SwitchParameter>]] [-Force <SwitchParameter>]
     [-SignInRedirectProvider <SPTrustedIdentityTokenIssuerPipeBind>] [-SignInR
    edirectURL <String>] [-WhatIf [<SwitchParameter>]] [<CommonParameters>]

    Set-SPWebApplication [-Identity] <SPWebApplicationPipeBind> -SMTPServer <St
    ring> [-AssignmentCollection <SPAssignmentCollection>] [-Confirm [<SwitchPa
    rameter>]] [-Force <SwitchParameter>] [-OutgoingEmailAddress <String>] [-Re
    plyToEmailAddress <String>] [-WhatIf [<SwitchParameter>]] [<CommonParameter
    s>]


DESCRIPTION
    The Set-SPWebApplication cmdlet configures the Web application specified by
     the Identity parameter. For any settings that are zone-specific (for the Z
    one parameter set), the zone to configure must be provided. The provided zo
    ne must already exist.
```

FIGURE 5.2 The `Set-SPWebApplication` cmdlet has three different parameter sets.

Use the first parameter set to configure the Default Quota Template, the Default Time Zone, and/or the Service Application Proxy Group. Use the second parameter set to configure the Zone of the web application, along with the associated authentication settings. Finally, use the third parameter set to configure various email settings for the web application.

Extend a Web Application

Scenario/Problem: You need to extend a web application to a new zone.

Solution: Use the `New-SPWebApplicationExtension` cmdlet.

If you need to extend a web application to a different zone, that zone must not already be in use. The `New-SPWebApplicationExtension` cmdlet only requires the web application identity, the name of the new website that will extend the web application, and the zone. However, it is a good idea to include the new URL and host header that the web application extension will use for the zone. A sample command-line entry for extending the main SharePoint web application is shown in Listing 5.6.

LISTING 5.6 **Extending a Web Application Example**

```
New-SPWebApplicationExtension –Identity "SharePoint – 80"
–Name "SP Internet Site" –Zone Internet –Url "http://internet"
–HostHeader "http://internet.sp2010.com"
```

The URL is the public-facing HTTP address, whereas the host header entry is the internal alternate access mapping. The `Zone` parameter may be one of the following values:

- `Default`
- `Intranet`
- `Internet`
- `Extranet`
- `Custom`

The zone used in the command must not already be used. That is, there can't already be a web application extended to that zone.

TIP There are a handful of optional parameters that can be used to configure security and authentication.

Create an Alternate Access Mapping URL

Scenario/Problem: You need to create a new alternate access mapping for a web application.

Solution: Use the `New-SPAlternateURL` cmdlet.

You may programmatically generate alternate access mappings using the `New-SPAlternateURL` cmdlet. The main parameter set requires the URL and the web application name, as shown in Listing 5.7.

LISTING 5.7 **Creating an Alternate Access Mapping**

```
New-SPAlternateURL –URL "http://intranet.sp2010.com"
–WebApplication "SharePoint - 80" –Internal –Zone Intranet
```

Using the `–Internal` switch parameter creates the mapping for internal use. Omitting this parameter generates a public URL entry. Use the `–Zone` parameter to specify the zone. Here are the valid values:

- `Default`
- `Intranet`
- `Internet`
- `Extranet`
- `Custom`

TIP For external resources, use the second parameter set, which requires the external resource name (`-ResourceName`) instead of the web application.

Display All Alternate Access Mappings

Scenario/Problem: You need to obtain an alternate access mapping reference and therefore need a listing of the available alternate URLs.

Solution: Use the `Get-SPAlternateURL` cmdlet without any parameters.

In order to obtain a reference variable, you need to find the identity of the URL. Using the `Get-SPAlternateURL` cmdlet without any parameters, as shown in Listing 5.8, displays the current URL mappings.

LISTING 5.8 **Sample Command to Display All Alternate URLs**

```
Get-SPAlternateURL
```

Get a Specific Alternate URL Entry

Scenario/Problem: You need to obtain a specific alternate URL reference.

Solution: Use the `Get-SPAlternateURL` cmdlet using the `–Identity` parameter and assign a variable the results.

Providing the identity of the alternate URL with the `Get-SPDatabase` cmdlet retrieves a specific alternate access mapping instance. Use this to assign the results to a variable for use in other cmdlets, as shown in Listing 5.9.

LISTING 5.9 **Assigning a Variable to a Specific Alternate URL Entry**

```
$altURL = Get-SPAlternateURL -Identity "http://intranet.sp2010.com"
```

The `Identity` parameter is the incoming URL address for the specific alternate access mapping.

Change the Zone of an Alternate Access Mapping

Scenario/Problem: You need to change the zone of an existing alternate access mapping.

Solution: Use the `Set-SPAlternateURL` cmdlet.

The `Set-SPAlternateURL` cmdlet allows you to modify the zone. This is the only setting that may be modified using the cmdlet. The `–Zone` parameter may be one of the following values:

- `Default`
- `Intranet`
- `Internet`
- `Extranet`
- `Custom`

A sample command line is shown in Listing 5.10.

LISTING 5.10 **Changing the Zone of an Alternate Access Mapping**

```
Set-SPAlternateURL –Identity "http://intranet.sp2010.com"
–Zone Intranet
```

The `–Identity` parameter can be the incoming URL or a variable identifying the alternate access mapping (using the `Get-SPAlternateURL` cmdlet).

NOTE If the alternate access mapping entry is public and the zone is `Default`, you may not change the zone until another public entry is created for the default zone.

Remove an Alternate Access Mapping

Scenario/Problem: You need to remove an existing alternate access mapping entry.

Solution: Use the `Remove-SPAlternateURL` cmdlet.

The `Remove-SPAlternateURL` cmdlet allows you to remove the specified alternate URL entry. The only required parameter is the identity of the URL mapping, as shown in Listing 5.11.

LISTING 5.11 **Removing an Existing Alternate Access Mapping**

```
Remove-SPAlternateURL –Identity "http://intranet.sp2010.com"
```

The `–Identity` parameter can be the incoming URL or a variable identifying the alternate access mapping (using the `Get-SPAlternateURL` cmdlet).

NOTE If the alternate access mapping entry is public and the zone is `Default`, you may not remove the alternate URL entry.

CHAPTER 6

Service Application Management

IN THIS CHAPTER

- Install Service Applications
- Display Available Service Applications on the Farm
- Get a Specific Service Application
- Configure IIS Settings for a Service Application
- Share a Service Application
- Remove a Service Application
- Display Service Instances on a Server
- Get a Specific Service Instance
- Start a Service Instance
- Stop a Service Instance
- Ancillary Cmdlets for Service Applications

This chapter outlines the SharePoint PowerShell commands that are used to manage the service applications within your SharePoint farm. Service applications are extensions of functionality within SharePoint and may be scaled out across servers (depending on the architecture topology). Some service applications allow spanning across farms (cross-farm services). The cmdlets in this chapter are generic. Specific service application cmdlets and configuration options are described in later chapters.

Install Service Applications

Scenario/Problem: You need to install all service applications to the farm using a script.

Solution: Use the `Install-SPService` cmdlet with the `-Provision` switch parameter.

Using the `Install-SPService` cmdlet installs all service applications on a server based on the services, service proxies, and service instances stored in the local registry. The `–Provision` switch parameter allows the service applications to be installed using the default settings. A sample command line is shown in Listing 6.1.

LISTING 6.1 **Installing All Available Service Applications**

```
Install-SPService -Provision
```

Display Available Service Applications on the Farm

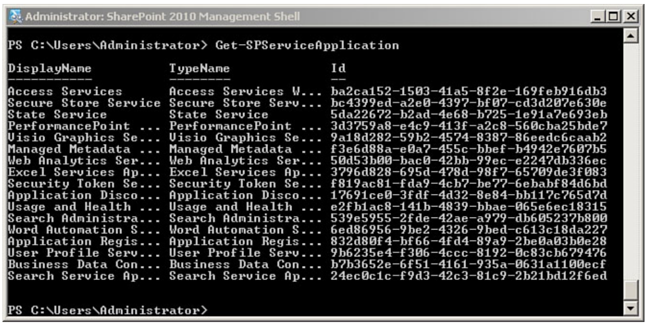
Scenario/Problem: You need to obtain a service application reference and therefore need a listing of the available service applications.

Solution: Use the `Get-SPServiceApplication` cmdlet without any parameters to retrieve all available service applications.

In order to obtain a service application reference object, you need to find the identity of the service application. Using the `Get-SPServiceApplication` by itself (see Listing 6.2) displays all available service applications, as shown in Figure 6.1.

LISTING 6.2 **Displaying All Available Service Applications**

```
Get-SPServiceApplication
```



```
PS C:\Users\Administrator> Get-SPServiceApplication

DisplayName          TypeName             Id
-----------          --------             --
Access Services      Access Services W... ba2ca152-1503-41a5-8f2e-169feb916db3
Secure Store Service Secure Store Serv... bc4399ed-a2e0-4397-bf07-cd3d207e630e
State Service        State Service        5da22672-b2ad-4e68-b725-1e91a7e693eb
PerformancePoint ... PerformancePoint ... 3d3759a8-e4c9-413f-a2c8-560cba25bde7
Visio Graphics Se... Visio Graphics Se... 9a18d282-59b2-4574-8387-86eedc6caab2
Managed Metadata ... Managed Metadata ... f3e6d88a-e0a7-455c-bbef-b4942e7607b5
Web Analytics Ser... Web Analytics Ser... 50d53b00-bac0-42bb-99ec-e2247db336ec
Excel Services Ap... Excel Services Ap... 3796d828-695d-478d-98f7-65709de3f083
Security Token Se... Security Token Se... f819ac81-fda9-4cb7-be77-6ebabf84d6bd
Application Disco... Application Disco... 17691ce0-3fdf-4d32-8e84-bb117c765d7d
Usage and Health ... Usage and Health ... e2fb1ac8-141b-4839-bbae-065e6ec18315
Search Administra... Search Administra... 539e5955-2fde-42ae-a979-db605237b800
Word Automation S... Word Automation S... 6ed86956-9be2-4326-9bed-c613c18da227
Application Regis... Application Regis... 832d80f4-bf66-4fd4-89a9-2be0a03b0e28
User Profile Serv... User Profile Serv... 9b6235e4-f306-4ccc-8192-0c83cb679476
Business Data Con... Business Data Con... b7b3652e-6f51-4161-935a-0631a1100ecf
Search Service Ap... Search Service Ap... 24ec0c1c-f9d3-42c3-81c9-2b21bd12f6ed


PS C:\Users\Administrator>
```

FIGURE 6.1 The `Get-SPServiceApplication` cmdlet displays all available service applications.

Get a Specific Service Application

Scenario/Problem: You need to obtain a specific service application reference.

Solution: Use the `Get-SPServiceApplication` cmdlet to assign a variable the results using the `–Identity` parameter.

Providing the identity of the service application with the `Get-SPServiceApplication` cmdlet retrieves a specific service application. Use this to assign the results to an object for use in other cmdlets, as shown in Listing 6.3.

LISTING 6.3 **Assigning a Variable to a Specific Service Application**

```
$sApp = Get-SPServiceApplication -Identity bc4399ed-a
2e0-4397-bf07-cd3d207e630e
```

TIP Instead of using `–Identity`, you may use `–Name` instead and provide the name of the service application in quotes.

Configure IIS Settings for a Service Application

Scenario/Problem: You need to configure IIS settings for a service application.

Solution: Use the `Set-SPServiceApplication` cmdlet to set IIS settings using a service application variable.

The `Set-SPServiceApplication` cmdlet allows you to set the default end point, the application pool, and/or the service proxy for a specific service application. Use this in conjunction with the `Get-SPServiceApplication` cmdlet (explained in the previous section), as shown in Listing 6.4.

LISTING 6.4 **Sample Configuration Script for a Service Application**

```
$sApp = Get-SPServiceApplication -Identity bc4399ed-a
2e0-4397-bf07-cd3d207e630e
Set-SPServiceApplication $sApp
-IISWebServiceApplicationPool "ServiceAppPool"
```

The main configuration parameters are as follows:

- `-DefaultEndPoint`
- `-IISWebServiceApplicationPool`
- `-ServiceApplicationProxyGroup`

TIP Use the `Get-SPServiceApplicationProxyGroup` cmdlet to create an object to pass into the `–ServiceApplicationProxyGroup` parameter.

TIP Use the `Get-SPServiceApplicationEndPoint` cmdlet to create an object to pass into the `–DefaultEndPoint` parameter. The example in the help page that uses "https" does not work.

Share a Service Application

Scenario/Problem: You need to make a service application available outside of the local farm.

Solution: Use the `Publish-SPServiceApplication` cmdlet with a service application variable.

The `Publish-SPServiceApplication` cmdlet allows you to make a specific service application available outside of the current farm. Use this in conjunction with the `Get-SPServiceApplication` cmdlet (explained in a previous section), as shown in Listing 6.5.

LISTING 6.5 **Sample Configuration Script for Sharing a Service Application**

```
$sApp = Get-SPServiceApplication -Identity bc4399ed-a
2e0-4397-bf07-cd3d207e630e
Publish-SPServiceApplication $sApp
```

TIP You may optionally provide a help page link (URL) using the `–Infolink` parameter with the `Publish-SPServiceApplication` cmdlet.

TIP To stop sharing a service application, use `UnPublish-SPServiceApplication`.

Remove a Service Application

Scenario/Problem: You need to remove a service application from the current server configuration.

Solution: Use the `Remove-SPServiceApplication` cmdlet with a service application variable.

The `Remove-SPServiceApplication` cmdlet allows you to remove a specific service application from the current server. Use this in conjunction with the `Get-SPServiceApplication` cmdlet (explained in a previous section), as shown in Listing 6.6.

LISTING 6.6 **Sample Configuration Script for Removing a Service Application**

```
$sApp = Get-SPServiceApplication -Identity bc4399ed-a
2e0-4397-bf07-cd3d207e630e
Remove-SPServiceApplication $sApp
```

TIP You may also remove the underlying database and data by using the `–RemoveData` switch parameter. Only use this if you are no longer using the service application at all on the farm.

Display Service Instances on a Server

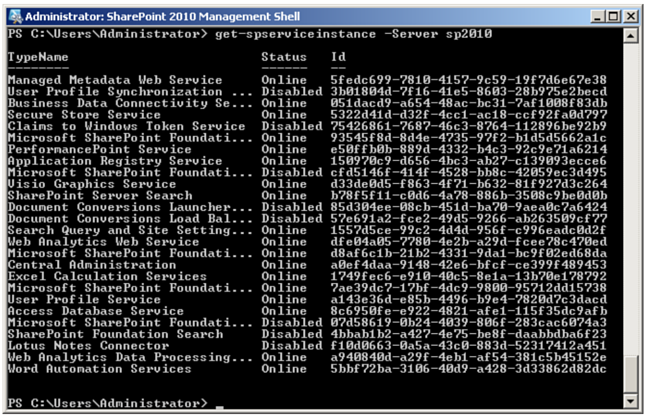
Scenario/Problem: You need to obtain a service instance reference and therefore need a listing of available service instances.

Solution: Use the `Get-SPServiceInstance` cmdlet with the `Server` parameter to display all of the service instances on that server.

In order to obtain a service instance reference object, you need to find the identity of the service instance. Using the `Get-SPServiceInstance` cmdlet with the `Server` parameter (see Listing 6.7) displays all available service applications, as shown in Figure 6.2.

LISTING 6.7 **Displaying All Available Service Instances Example**

```
Get-SPServiceInstance –Server sp2010svr01
```



```
PS C:\Users\Administrator> get-spserviceinstance -Server sp2010

TypeName                          Status   Id
--------                          ------   --
Managed Metadata Web Service      Online   5fedc699-7810-4157-9c59-19f7d6e67e38
User Profile Synchronization ... Disabled 3b01804d-7f16-41e5-8603-28b975e2becd
Business Data Connectivity Se... Online   051dacd9-a654-48ac-bc31-7af1008f83db
Secure Store Service              Online   5322d41d-d32f-4cc1-ac18-ccf92fa0d797
Claims to Windows Token Service   Disabled 75426861-7687-46c3-8764-112896be92b9
Microsoft SharePoint Foundati... Online   93545f8d-8d4e-4735-97f2-b1d5d5662a1c
PerformancePoint Service          Online   e50ffb0b-889d-4332-b4c3-92c9e71a6214
Application Registry Service      Online   150970c9-d656-4bc3-ab27-c139093ecce6
Microsoft SharePoint Foundati... Disabled cfd5146f-414f-4528-bb8c-42059ec3d495
Visio Graphics Service            Online   d33de0d5-f863-4f71-b632-81f927d3c264
SharePoint Server Search          Online   b78f5f11-c0d6-4a78-886b-3508c9be0d0b
Document Conversions Launcher... Disabled 85d304ee-08cb-451d-ba70-9aea0c7a6424
Document Conversions Load Bal... Disabled 57e691a2-fce2-49d5-9266-ab263509cf77
Search Query and Site Setting... Online   1557d5ce-99c2-4d4d-956f-c996eadc0d2f
Web Analytics Web Service         Online   dfe04a05-7780-4e2b-a29d-fcee78c470ed
Microsoft SharePoint Foundati... Online   d8af6c1b-21b2-4331-9da1-bc9f02ed68da
Central Administration            Online   a0ef4daa-9148-42e6-bfcf-ce399f489453
Excel Calculation Services        Online   1749fec6-e910-40c5-8e1a-13b70e178792
Microsoft SharePoint Foundati... Online   7ae39dc7-17bf-4dc9-9800-95712dd15738
User Profile Service              Online   a143e36d-e85b-4496-b9e4-7820d7c3dacd
Access Database Service           Online   8c6950fe-e922-4821-afe1-115f35dc9afb
Microsoft SharePoint Foundati... Disabled 07d58619-0b24-4039-806f-283cac6074a3
SharePoint Foundation Search      Disabled 4bbab1b2-a427-4e75-be8f-daabbdba6f23
Lotus Notes Connector             Disabled f10d0663-0a5a-43c0-883d-52317412a451
Web Analytics Data Processing... Online   a940840d-a29f-4eb1-af54-381c5b45152e
Word Automation Services          Online   5bbf72ba-3106-40d9-a428-3d33862d82dc


PS C:\Users\Administrator> _
```

FIGURE 6.2 The `Get-SPServiceInstance` cmdlet displays all service instances on the specified server.

TIP Running the `Get-SPServiceInstance` cmdlet without the `Server` parameter returns all service instances on the entire farm.

Get a Specific Service Instance

Scenario/Problem: You need to obtain a specific service instance reference.

Solution: Use the `Get-SPServiceInstance` cmdlet to assign a variable the results using the `–Identity` parameter.

Providing the identity of the service instance with the `Get-SPServiceInstance` cmdlet retrieves a specific service instance. Use this to assign the results to an object for use in other cmdlets, as shown in Listing 6.8.

LISTING 6.8 **Assigning a Variable to a Specific Service Instance**

```
$serviceInst = Get-SPServiceInstance
-Identity 5fedc699-7810-4157-9c59-19f7d6e67e38
```

Start a Service Instance

Scenario/Problem: You need to start a specific service instance.

Solution: Use the `Get-SPServiceInstance` cmdlet to assign a variable the results using the `–Identity` parameter and then use `Start-SPServiceInstance` using that variable.

The `Start-SPServiceInstance` cmdlet starts the specified service on the current server. You may provide the identity directly to the cmdlet or use a variable, as shown in Listing 6.9.

LISTING 6.9 **Starting a Service Instance**

```
$serviceInst = Get-SPServiceInstance
-Identity 5fedc699-7810-4157-9c59-19f7d6e67e38
Start-SPServiceInstance $serviceInst
```

Stop a Service Instance

Scenario/Problem: You need to stop a specific service instance.

Solution: Use the `Get-SPServiceInstance` cmdlet to assign a variable the results using the `–Identity` parameter and then use `Stop-SPServiceInstance` using that variable.

The `Stop-SPServiceInstance` cmdlet stops the specified service on the current server. You may provide the identity directly to the cmdlet or use a variable, as shown in Listing 6.10.

LISTING 6.10 **Stopping a Service Instance**

```
$serviceInst = Get-SPServiceInstance
-Identity 5fedc699-7810-4157-9c59-19f7d6e67e38
Stop-SPServiceInstance $serviceInst
```

Ancillary Cmdlets for Service Applications

Scenario/Problem: You need to create instances of service application items for use within other cmdlets covered in this chapter.

Solution: Use the cmdlets listed in this section to obtain various instances.

This chapter covers the basic or more common cmdlets. For more advance scripting with service applications, the following cmdlet nouns are available:

- `SPServiceApplicationProxy`
- `SPServiceApplicationProxyGroup`
- `SPServiceApplicationProxyGroupMember`
- `SPServiceApplicationSecurity`
- `SPServiceContext`
- `SPServiceApplicationEndPoint`

CHAPTER 7

SharePoint Database Management

IN THIS CHAPTER

This chapter outlines the SharePoint PowerShell commands that are used to manage the underlying SharePoint databases. Cmdlets to create and remove databases are explained as well as several backup and restore operations.

Display All SharePoint Databases

Scenario/Problem: You need to obtain a service application reference and therefore need a listing of available service applications.

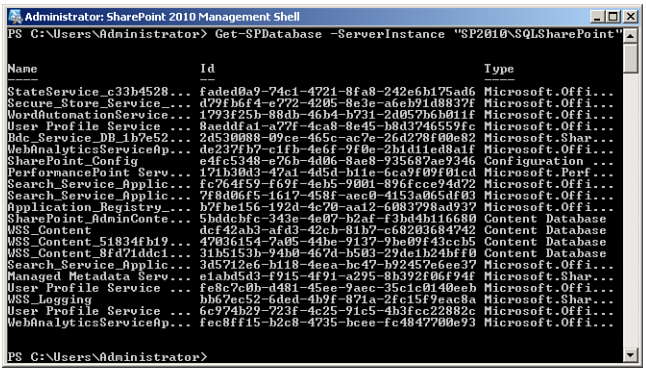
Solution: Use the `Get-SPDatabase` cmdlet with the `ServerInstance` parameter to retrieve all available databases.

In order to obtain a database reference object, you need to find the identity of the database. Using the `Get-SPDatabase` cmdlet with the `ServerInstance` parameter (see Listing 7.1) displays all available databases, as shown in Figure 7.1.

You need to pass in the SQL Server instance to the `ServerInstance` parameter. This is not any of the SharePoint servers.

LISTING 7.1 **Sample Command to Display All Databases**

```
Get-SPDatabase –ServerInstance "SP2010\SQLSharePoint"
```



```
PS C:\Users\Administrator> Get-SPDatabase -ServerInstance "SP2010\SQLSharePoint"


Name                         Id                                     Type
----                         --                                     ----
StateService_c33b4528...     faded0a9-74c1-4721-8fa8-242e6b175ad6   Microsoft.Offi...
Secure_Store_Service_...     d79fb6f4-e772-4205-8e3e-a6eb91d8837f   Microsoft.Offi...
WordAutomationService...     1793f25b-88db-46b4-b731-2d057b6b011f   Microsoft.Offi...
User Profile Service ...     8aeddfa1-a77f-4ca8-8e45-b8d3746559fc   Microsoft.Offi...
Bdc_Service_DB_1b7e52...     2d530088-09ce-465c-ac7e-26d278f00e82   Microsoft.Shar...
WebAnalyticsServiceAp...     de237fb7-c1fb-4e6f-9f0e-2b1d11ed8a1f   Microsoft.Offi...
SharePoint_Config            e4fc5348-e76b-4d06-8ae8-935687ae9346   Configuration ...
PerformancePoint Serv...     171b30d3-47a1-4d5d-b11e-6ca9f09f01cd   Microsoft.Perf...
Search_Service_Applic...     fc764f59-f69f-4eb5-9001-896fcce94d72   Microsoft.Offi...
Search_Service_Applic...     7f8d06f5-1617-458f-aec0-4153a065df03   Microsoft.Offi...
Application_Registry_...     b7fbe156-192d-4c70-aa12-6083798ad937   Microsoft.Offi...
SharePoint_AdminConte...     5bddcbfc-343e-4e07-b2af-f3bd4b116680   Content Database
WSS_Content                  dcf42ab3-afd3-42cb-81b7-c68203684742   Content Database
WSS_Content_51834fb19...     47036154-7a05-44be-9137-9be09f43ccb5   Content Database
WSS_Content_8fd71ddc1...     31b5153b-94b0-467d-b503-29de1b24bff0   Content Database
Search_Service_Applic...     3d5712e6-b118-4eea-bc47-b92457e6ee37   Microsoft.Offi...
Managed Metadata Serv...     e1abd5d3-f915-4f91-a295-8b392f06f94f   Microsoft.Shar...
User Profile Service ...     fe8c7c0b-d481-45ee-9aec-35c1c0140eeb   Microsoft.Offi...
WSS_Logging                  bb67ec52-6ded-4b9f-871a-2fc15f9eac8a   Microsoft.Shar...
User Profile Service ...     6c974b29-723f-4c25-91c5-4b3fcc22882c   Microsoft.Offi...
WebAnalyticsServiceAp...     fec8ff15-b2c8-4735-bcee-fc4847700e93   Microsoft.Offi...


PS C:\Users\Administrator>
```

FIGURE 7.1 The `Get-SPDatabase –ServerInstance` cmdlet displays all databases on the SQL Server.

Get a Specific Database

Scenario/Problem: You need to obtain a specific database reference.

Solution: Use the `Get-SPDatabase` cmdlet using the `–Identity` parameter to assign a variable the results.

Providing the identity of the database with the `Get-SPDatabase` cmdlet retrieves a specific database. Use this to assign the results to an object for use in other cmdlets, as shown in Listing 7.2.

LISTING 7.2 **Assigning a Variable to a Specific Database**

```
$database = Get-SPDatabase
-Identity 47036154-7a05-44be-9137-9be09f43ccb5
```

Create a New Content Database

Scenario/Problem: You want to create a new content database and associate it to an existing web application.

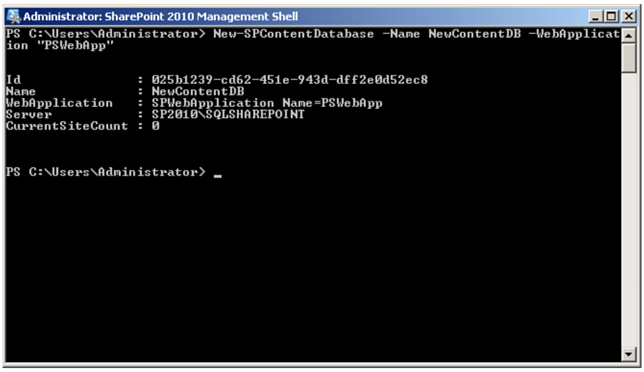
Solution: Use the `New-SPContentDatabase` cmdlet to create a new content database and associate it to the specified web application.

The `New-SPContentDatabase` cmdlet allows you to create a new content database using the `Name` parameter for the name and the `WebApplication` parameter to associate it to the specified web application. You may use the name of the web application and therefore do not need to obtain the actual identity. A sample command-line entry is shown in Listing 7.3

LISTING 7.3 **Creating a New Content Database Example**

```
New-SPContentDatabase –Name NewContentDB –WebApplication "PSWebApp"
```

It will take a moment to create the new content database. The creation is confirmed by an output of the content database information, as shown in Figure 7.2.



```
Administrator: SharePoint 2010 Management Shell
PS C:\Users\Administrator> New-SPContentDatabase -Name NewContentDB -WebApplicat
ion "PSWebApp"


Id                : 025b1239-cd62-451e-943d-dff2e0d52ec8
Name              : NewContentDB
WebApplication    : SPWebApplication Name=PSWebApp
Server            : SP2010\SQLSHAREPOINT
CurrentSiteCount  : 0



PS C:\Users\Administrator> _
```

FIGURE 7.2 Content database information is displayed after creation.

TIP Use the `Get-SPWebApplication` cmdlet to list out the names of the web applications.

Display All Content Databases for a Web Application

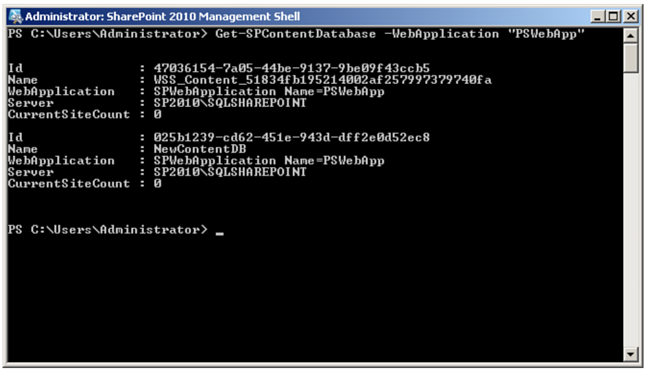
Scenario/Problem: You want to see all content databases associated to a specific web application.

Solution: Use the `Get-SPContentDatabase` cmdlet with the `WebApplication` parameter.

The `Get-SPContentDatabase` cmdlet has several parameter sets. One of the sets allows for a `WebApplication` parameter to be used. Using the `WebApplication` parameter with the `Get-SPContentDatabase` cmdlet (see Listing 7.4) will display information about all content databases associated to the specified web application, as shown in Figure 7.3.

LISTING 7.4 **Displaying All Content Databases**

```
Get-SPContentDatabase –WebApplication "PSWebApp"
```



```
PS C:\Users\Administrator> Get-SPContentDatabase -WebApplication "PSWebApp"

Id               : 47036154-7a05-44be-9137-9be09f43ccb5
Name             : WSS_Content_51834fb195214002af257997379740fa
WebApplication   : SPWebApplication Name=PSWebApp
Server           : SP2010\SQLSHAREPOINT
CurrentSiteCount : 0

Id               : 025b1239-cd62-451e-943d-dff2e0d52ec8
Name             : NewContentDB
WebApplication   : SPWebApplication Name=PSWebApp
Server           : SP2010\SQLSHAREPOINT
CurrentSiteCount : 0


PS C:\Users\Administrator> _
```

FIGURE 7.3 Results of using the `Get-SPContentDatabase` cmdlet with the `WebApplication` parameter

TIP Use the `Get-SPContentDatabase` cmdlet with the `Identity` parameter to assign the specific content database reference to a variable for use in other cmdlets.

Detach a Content Database from a Web Application

Scenario/Problem: You need to dissociate a content database from a web application without removing the database or deleting the data.

Solution: Use the `Dismount-SPContentDatabase` cmdlet with a content database variable.

The `Dismount-SPContentDatabase` cmdlet allows you to dissociate the specified content database with the web application without removing the underlying database itself. Use this in conjunction with the `Get-SPContentDatabase` cmdlet, as shown in Listing 7.5.

TIP Use the `Get-SPContentDatabase` cmdlet with the `WebApplication` parameter to confirm that the content database was dismounted from the web application.

LISTING 7.5 **Sample Script for Dismounting a Content Database**

```
$database = Get-SPContentDatabase
-Identity 025b1239-cd62-451e-943d-dff2e0d52ec8
Dismount-SPContentDatabase $database
```

Attach a Content Database to a Web Application

Scenario/Problem: You need to associate a content database to a web application.

Solution: Use the `Mount-SPContentDatabase` cmdlet with the `WebApplication` parameter.

The `Mount-SPContentDatabase` cmdlet allows you to associate the specified content database with a web application. The command-line entry is similar to creating a new content database because the parameters used are `Name` and `WebApplication`, as shown in Listing 7.6.

LISTING 7.6 **Sample Script for Mounting a Content Database**

```
Mount-SPContentDatabase –Name "NewContentDB"
–WebApplication "PSWebApp"
```

A confirmation listing of the content database information is displayed, just as in creating a new content database.

Delete a Content Database

Scenario/Problem: You want to remove a content database from its web application and delete the actual SQL Server database and data.

Solution: Use the `Remove-SPContentDatabase` cmdlet with a database variable.

The `Remove-SPContentDatabase` cmdlet allows you to delete the specified content database. Use this in conjunction with the `Get-SPContentDatabase` cmdlet, as shown in Listing 7.7.

LISTING 7.7 **Sample Script for Removing a Content Database**

```
$database = Get-SPContentDatabase
-Identity 025b1239-cd62-451e-943d-dff2e0d52ec8
Remove-SPContentDatabase $database
```

TIP Instead of using the database variable, you may also execute `Remove-SPContentDatabase` by just providing the name of the content database without any parameter entries (for example, `Remove-SPContentDatabase "NewContentDB"`).

Create a New Configuration Database

Scenario/Problem: You want to create a new configuration database.

Solution: Use the `New-SPConfigurationDatabase` cmdlet to create a new configuration database and associate it to the specified web application.

The `New-SPConfigurationDatabase` cmdlet allows you to create a new configuration database using the `DatabaseName` and `DatabaseServer` parameters. A sample command-line entry is shown in Listing 7.8.

LISTING 7.8 **Creating a New Configuration Database**

```
New-SPConfigurationDatabase –DatabaseName NewConfigurationDB
–DatabaseServer "SP2010\SQLSharePoint"
```

The farm credentials and a passphrase are needed to execute this cmdlet. If you do not provide these in line with the cmdlet, you will be prompted to provide this information.

NOTE If the current server is already joined to an existing farm, you will not be able to create a new configuration database. Use the `Disconnect-SPConfigurationDatabase` cmdlet first to disconnect the current server from the farm.

Delete a Configuration Database

Scenario/Problem: You want to remove a configuration database from the farm and delete the actual SQL Server database and data.

Solution: Use the `Remove-SPConfigurationDatabase` cmdlet.

The `Remove-SPConfigurationDatabase` cmdlet allows you to delete the current configuration database. You do not need to pass any parameters into this cmdlet, as shown in Listing 7.9.

LISTING 7.9 **Sample Script for Removing a Configuration Database**

```
Remove-SPConfigurationDatabase
```

This cmdlet already knows which configuration database is being used by the current server and SharePoint farm.

TIP Run the `Remove-SPConfigurationDatabase` cmdlet with the `–WhatIf` switch parameter to display the actual configuration database that will be removed.

Back Up a Configuration Database

Scenario/Problem: You want to back up a configuration database within your SQL Server.

Solution: Use the `Backup-SPConfigurationDatabase` cmdlet with the Directory parameter.

The `Backup-SPConfigurationDatabase` cmdlet allows you to perform a backup on a configuration database, placing the backup files in the directory specified. An example is shown in Listing 7.10.

LISTING 7.10 **Sample Script for Backing Up a Configuration Database**

```
Backup-SPConfigurationDatabase –Directory F:\Backups
```

This cmdlet already knows which configuration database is being used by the current server and SharePoint farm. Use the `–DatabaseServer` and `–DatabaseName` parameters to specify the actual configuration database name and location.

If you are backing up the current configuration database being used by the farm, you may also use the `Backup-SPFarm` cmdlet with the `ConfigurationOnly` switch parameter, as shown in Listing 7.11.

LISTING 7.11 **Using `Backup-SPFarm` to Back Up the Configuration Database**

```
Backup-SPFarm –BackupMethod Full –Directory F:\Backups
-ConfigurationOnly
```

TIP For backup and restore operations of the current farm, it is recommended to use the `Backup-SPFarm` cmdlet with the `ConfigurationOnly` switch parameter to back up the configuration database.

The `BackupMethod` parameter can be either `Full` or `Differential`. `Full` will back up the entire configuration, whereas `Differential` only backs up changes that have occurred since the last full backup was performed.

Restore a Configuration Database

Scenario/Problem: You want to restore a configuration database from a backup.

Solution: Use the `Restore-SPFarm` cmdlet with the Directory parameter and `ConfigurationOnly` switch parameter.

The `Restore-SPFarm` cmdlet allows you to restore a configuration database based on the backup files in the directory specified. Use the `–ConfigurationOnly` switch parameter to only restore the configuration database, as shown in Listing 7.12.

LISTING 7.12 **Sample Script for Restoring a Configuration Database**

```
Restore-SPFarm –Directory F:\Backups –ConfigurationOnly
–RestoreMethod Overwrite
```

The `RestoreMethod` parameter can be `New` or `Overwrite`. Using `New` creates a new location for the backup to be restored. `Overwrite` will restore the backup in place and overwrite the current configuration.

Back Up the SharePoint Farm

Scenario/Problem: You want to back up the entire farm.

Solution: Use the `Backup-SPFarm` cmdlet with the `Directory` parameter.

The `Backup-SPFarm` cmdlet allows you to perform a backup of the entire farm (or item in the farm), placing the backup files in the directory specified. An example is shown in Listing 7.13.

LISTING 7.13 **Sample Script for Backing Up the SharePoint Farm**

```
Backup-SPFarm –BackupMethod Full –Directory F:\Backups
```

The `BackupMethod` parameter can be either `Full` or `Differential`. `Full` will back up the entire configuration, whereas `Differential` only backs up changes that have occurred since the last full backup was performed.

TIP Run `Backup-SPFarm` with the `–ShowTree` parameter to display all items in the farm that may be backed up. Use the `–Item` parameter with `Backup-SPFarm` to back up a specific item in the farm.

Restore the SharePoint Farm

Scenario/Problem: You want to restore the farm from a backup.

Solution: Use the `Restore-SPFarm` cmdlet with the `Directory` parameter.

The `Restore-SPFarm` cmdlet allows you to restore the farm based on the backup files in the directory specified, as shown in Listing 7.14.

LISTING 7.14 **Sample Script for Restoring a Farm**

```
Restore-SPFarm –Directory F:\Backups –RestoreMethod Overwrite
```

The `RestoreMethod` parameter can be `New` or `Overwrite`. Using `New` creates a new location for the backup to be restored. `Overwrite` will restore the backup in place and overwrite the current farm content.

TIP Use the `–Item` parameter with `Restore-SPFarm` cmdlet to restore a specific item in the farm.

Back Up a Site Collection

Scenario/Problem: You want to back up a specific site collection.

Solution: Use the `Backup-SPSite` cmdlet.

The `Backup-SPSite` cmdlet allows you to perform a backup of a specific site collection, creating a backup file based on the path specified. An example is shown in Listing 7.15.

LISTING 7.15 **Sample Script for Backing Up a Site Collection**

```
Backup-SPSite –Identity "http://sp2010/sites/BICenter"
–Path F:\Backups\BICenter.bak
```

The `Identity` parameter can be the URL or the GUID of the site collection. Using the URL is usually easier. The `Path` parameter defines where the backup will be stored, along with the name of the backup file to create.

TIP Run `Get-SPSite` without any parameters to list out the available site collections.

Restore a Site Collection

Scenario/Problem: You want to restore a site collection from a backup.

Solution: Use the `Restore-SPSite` cmdlet.

The `Restore-SPSite` cmdlet allows you to restore a site collection based on the backup file in the path specified, as shown in Listing 7.16.

LISTING 7.16 **Sample Script for Restoring a Site Collection**

```
Restore-SPSite –Identity "http://sp2010/sites/BICenter"
–Path F:\Backups\BICenter.bak –Force
```

The `Force` switch parameter is needed to overwrite an existing site collection. You may also specify the `DatabaseName` and `DatabaseServer` parameters to specify where the content of the site collection should be stored.

PART III

Development and Administration

CHAPTER 8

Site Management

IN THIS CHAPTER

This chapter outlines the SharePoint PowerShell commands that are used to manage the site collections and sites within your SharePoint farm.

Display All Available Site Collections on the Farm

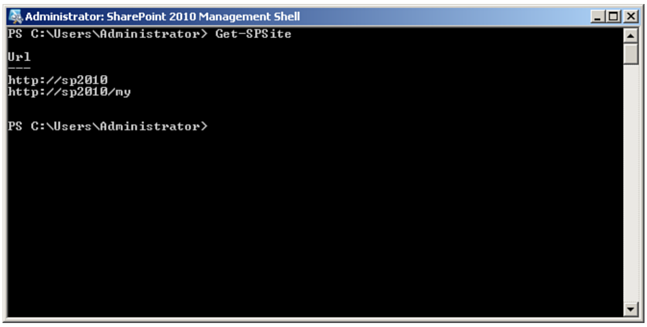
Scenario/Problem: You need to obtain a site collection reference and therefore need a listing of available site collection URLs.

Solution: Use the `Get-SPSite` cmdlet without any parameters to retrieve all available site collections.

In order to obtain a site collection reference object, you need to find the URL (`Identity`) of the site collection. Using the `Get-SPSite` cmdlet by itself (see Listing 8.1) displays all available site collections, as shown in Figure 8.1.

LISTING 8.1 **Displaying All Available Site Collections**

```
Get-SPSite
```


Administrator: SharePoint 2010 Management Shell

```
PS C:\Users\Administrator> Get-SPSite

Url
---
http://sp2010
http://sp2010/my

PS C:\Users\Administrator>
```

FIGURE 8.1 The `Get-SPSite` cmdlet displays all available sites.

Display Available Site Collections in a Web Application

Scenario/Problem: You need to obtain a site collection reference and therefore need a listing of available site collection URLs within a specific web application.

Solution: Use the `Get-SPSite` cmdlet with the `WebApplication` parameter.

In order to obtain a site collection reference object, you need to find the URL (`Identity`) of the site collection. Using the `Get-SPSite` cmdlet with the `WebApplication` parameter, as shown in Listing 8.2, displays all available site collections in the specified web application.

LISTING 8.2 **Displaying Available Site Collections in a Web Application**

```
Get-SPSite –WebApplication "SharePoint – 80"
```

Display Available Site Collections in a Content Database

Scenario/Problem: You need to obtain a site collection reference and therefore need a listing of available site collection URLs within a specific content database.

Solution: Use the `Get-SPSite` cmdlet with the `ContentDatabase` parameter.

In order to obtain a site collection reference object, you need to find the URL (`Identity`) of the site collection. Using the `Get-SPSite` cmdlet with the `ContentDatabase` parameter, as shown in Listing 8.3, displays all available site collections in the specified content database.

LISTING 8.3 **Displaying Available Site Collections in a Content Database**

```
Get-SPSite –ContentDatabase "WSS_CONTENT1"
```

TIP Use the `SP-GetContentDatabase` cmdlet to display the available content databases used by the SharePoint farm.

Get a Specific Site Collection

Scenario/Problem: You need to obtain a specific site collection reference.

Solution: Use the `Get-SPSite` cmdlet to assign a variable the results using the `–Identity` parameter.

Providing the identity of the site collection with the `Get-SPSite` cmdlet retrieves a specific site collection. Use this to assign the results to an object for use in other cmdlets, as shown in Listing 8.4.

LISTING 8.4 **Assigning a Variable to a Specific Web Application**

```
$webApp = Get-SPSite -Identity "http://sp2010"
```

The `Identity` parameter in this case is the URL shown when retrieving all site collections (`Get-SPSite`).

Remove a Site Collection

Scenario/Problem: You need to remove a site collection from your SharePoint farm.

Solution: Use the `Remove-SPSite` cmdlet with a site collection variable.

The `Remove-SPSite` cmdlet allows you to remove a specific site collection from the current server. Use this in conjunction with the `Get-SPSite` cmdlet (explained in a previous section), as shown in Listing 8.5, or just use the site collection identity directly in line, as shown in Listing 8.6.

LISTING 8.5 **Removing a Site Collection Using a Variable**

```
$siteCol = Get-SPSite -Identity "http://sp2010/sites/sitecol"
Remove-SPSite $siteCol
```

LISTING 8.6 **Removing a Site Collection Directly**

```
Remove-SPSite –Identity "http://sp2010/sites/sitecol"
```

The `Remove-SPSite` cmdlet has an optional switch parameter named `GradualDelete`. Using this parameter will remove the site collection and any subsites underneath gradually. This helps keeps the SharePoint load down and therefore is highly recommended for large site collections.

NOTE There is also a switch parameter for `Remove-SPSite` that removes Active Directory accounts. The parameter is named `DeleteADAccounts`. This can be dangerous and therefore it is recommended to not use this switch parameter.

Create a Site Collection

Scenario/Problem: You need to create a new site collection.

Solution: Use the `New-SPSite` cmdlet.

To add a new site collection, use the `New-SPSite` cmdlet as follows:

```
New-SPSite –url <full url of new site collection>
-name <name of new site collection>
-owneralias <site collection administrator>
-template <site collection template to use>
```

A sample command-line entry for creating the root site collection using the Team Site template is shown in Listing 8.7.

LISTING 8.7 **Creating a New Site Collection**

```
New-SPSite –url http://sp2010/sites/sitecol -name Home
-owneralias SP\SiteAdmin -template STS#0
```

The site template names can be found using `Get-SPWebTemplates`. If no template is provided, the site collection will still be created. In this case, once the site is accessed via the browser, SharePoint will prompt for a site collection template to be selected.

TIP You can only create one site collection at the root of the web application. Subsequent site collections must be created using a managed path. See Chapter 4, “Farm Management,” for more information on creating a managed path.

It may take a few minutes to create a new site collection using the `New-SPSite` cmdlet. Once this is completed, the new URL will be presented.

Set the Lock State of a Site Collection

Scenario/Problem: You want to allow or prohibit access to the site collection.

Solution: Use the `Set-SPSite` cmdlet with the `LockState` parameter.

The `Set-SPSite` cmdlet has several parameter sets for configuring various settings on the site collection. Running `Set-SPSite` with the `-?` parameter displays each parameter set. The `LockState` parameter allows you to lock or unlock the site collection. The following are valid values for the `LockState` parameter:

- `Unlock`
- `NoAdditions`
- `ReadOnly`
- `NoAccess`

The `Unlock` state allows full availability to users (based on the normal permissions). `NoAdditions` allows you to lock the site collection such that no new content may be added. This does not lock down updates or deletes. To fully lock down the content, use the `ReadOnly` state. Placing the site collection into the `ReadOnly` state prevents users from adding, updating, or deleting content. `NoAccess` disables all access to the site collection. Users will not even be able to navigate to the site collection location (or locations underneath).

A sample command line for changing the lock state is shown in Listing 8.8.

LISTING 8.8 **Changing the Lock State of a Site Collection**

```
Set-SPSite –Identity http://sp2010/sites/sitecol -LockState NoAdditions
```

Set the Storage Limits on a Site Collection

Scenario/Problem: You want to modify the storage limitations for a site collection.

Solution: Use the `Set-SPSite` cmdlet with the `MaxSize` and `WarningSize` parameters.

Two parameters work together to set the storage limits on a site collection: `MaxSize` and `WarningSize`. A sample command line for setting the storage limits is shown in Listing 8.9.

LISTING 8.9 **Setting the Storage Limits on a Site Collection**

```
Set-SPSite –Identity http://sp2010/sites/sitecol -MaxSize 1024
-WarningSize 768
```

Both parameter values are in megabytes. The `MaxSize` parameter can be any integer value but must be greater than or equal to the `WarningSize` value. Therefore, the `WarningSize` parameter can be any value between zero and the `MaxSize` value.

NOTE You may not be able to set the storage limits if a quota template is being used for the site collection.

Create a Site Under a Site Collection

Scenario/Problem: You want to create sites under your site collection.

Solution: Use the `New-SPWeb` cmdlet.

To add a new subsite, use the `New-SPWeb` cmdlet as follows:

```
New-SPWeb –url <full url of new site> -name <name of new site>
-template <site template to use>
```

The following switch parameters are available when you are using the `New-SPWeb` cmdlet:

- `AddToTopNav`
- `UniquePermissions`
- `UseParentTopNav`

The `AddToTopNav` switch parameter places the new site within the top navigation of the site collection. Using the `UseParentTopNav` switch parameter replicates the top nav of the site collection onto the new subsite. Creating a site with `UniquePermissions` forces the site to not inherit permissions from the site collection and only grants the System Account Full Control access.

A sample command-line entry for creating a subsite using the Team Site template is shown in Listing 8.10.

LISTING 8.10 **Creating a Site Under a Site Collection**

```
New-SPWeb –url http://sp2010/sites/sitecol/newsubsite
-name "New Sub Site" -template STS#0 –AddToTopNav –UniquePermissions
-UseParentTopNav
```

The site template names can be found using `Get-SPWebTemplates`. If no template is provided, the site will still be created. In this case, once the site is accessed via the browser, SharePoint will prompt for a site template to be selected.

Display All Subsites within a Site Collection

Scenario/Problem: You need to obtain a site reference and therefore need a listing of available subsites within a specific site collection.

Solution: Use the `Get-SPWeb` cmdlet with a wildcard `Identity` path.

To display all subsites within a site collection, you need to use the `Get-SPWeb` cmdlet with the `Identity` parameter. If you only provide the site collection URL, `Get-SPWeb` will only return the site collection web object. Therefore, you need to provide a wildcard path, as shown in Listing 8.11.

LISTING 8.11 **Displaying All Subsites Within a Site Collection**

```
Get-SPWeb –Identity "http://sp2010/sites/sitecol/*"
```

You may also use the `Site` parameter instead of the wildcard `Identity` parameter to retrieve all webs, including the site collection web. A sample command line is shown in Listing 8.12.

LISTING 8.12 **Displaying All Webs Within a Site Collection**

```
Get-SPWeb –Site "http://sp2010/sites/sitecol"
```

Get a Specific Subsite (Web)

Scenario/Problem: You need to obtain a specific subsite reference.

Solution: Use the `Get-SPWeb` cmdlet to assign a variable the results using the `–Identity` parameter.

Providing the identity of the subsite with the `Get-SPWeb` cmdlet retrieves a specific subsite. Use this to assign the results to an object for use in other cmdlets, as shown in Listing 8.13.

LISTING 8.13 **Assigning a Variable to a Specific Subsite**

```
$web = Get-SPWeb -Identity "http://sp2010/sites/sitecol/subsite"
```

Remove a Subsite (Web) from a Site Collection

Scenario/Problem: You need to remove a subsite from a site collection.

Solution: Use the `Remove-SPWeb` cmdlet.

The `Remove-SPWeb` cmdlet allows you to remove a specific subsite. Use this in conjunction with the `Get-SPWeb` cmdlet (explained in the previous section), as shown in Listing 8.14, or just use the path identity directly in line, as shown in Listing 8.15.

LISTING 8.14 **Removing a Subsite Application Using a Variable**

```
$web = Get-SPWeb -Identity "http://sp2010/sites/sitecol/subsite"
Remove-SPWeb $web
```

LISTING 8.15 **Removing a Subsite Directly**

```
Remove-SPWeb –Identity "http://sp2010/sites/sitecol/subsite"
```

Modify the Subsite (Web) URL

Scenario/Problem: You want to rename the relative URL name for a subsite.

Solution: Use the `Set-SPWeb` cmdlet with the `RelativeURL` parameter.

You may easily change the name of the relative URL in which a subsite is accessed by using the `Set-SPWeb` cmdlet. Provide the URL of the site as the `Identity` parameter and use the `RelativeURL` parameter to provide the new URL, as shown in Listing 8.16.

LISTING 8.16 **Renaming a Subsite**

```
Set-SPWeb –Identity "http://sp2010/sites/sitecol/newsubsite"

–RelativeURL teamsites
```

Notice the `RelativeURL` value is not a full URL. The example is essentially renaming the subsite URL from “newsubsite” to “teamsites.” Therefore, after execution the subsite is accessed by http://sp2010/sites/sitecol/teamsites/.

Move a Site Collection to a Different Content Database

Scenario/Problem: You want to move a particular site collection to a different content database.

Solution: Use the `Move-SPSite` cmdlet.

The `Move-SPSite` cmdlet allows you to easily move a site collection from one content database to another. All that is needed to execute this cmdlet is the site collection identity and the destination database name. A sample command line is shown in Listing 8.17.

LISTING 8.17 **Moving a Site Collection to a Different Content Database**

```
Move-SPSite –Identity "http://sp2010/sites/sitecol"
–DestinationDatabase "WSS_CONTENT_TEAMSITES"
```

You do not need to provide the originating database because there is no parameter for that anyway. The destination database must already exist.

TIP Use the `New-SPContentDatabase` cmdlet to quickly generate a new content database to move your site collection.

Move All Site Collections from One Content Database to Another

Scenario/Problem: You want to move all site collections stored in one content database to a different content database.

Solution: Use the `Move-SPSite` cmdlet with the `Get-SPSite` cmdlet.

You may pipe the results of `Get-SPSite` to the `Move-SPSite` cmdlet. Therefore, by using the `ContentDatabase` parameter with `Get-SPSite`, you can easily move all site collections from that content database to a new content database. A sample command line is shown in Listing 8.18.

LISTING 8.18 **Moving All Site Collections From a Content Database**

```
Get-SPSite –ContentDatabase "WSS_CONTENT_MAIN" | Move-SPSite
–DestinationDatabase "WSS_CONTENT_TEAMSITES"
```

Similar to moving a single site collection, the destination database must already exist.

TIP Use the `New-SPContentDatabase` cmdlet to quickly generate a new content database to move your site collections.

CHAPTER 9

Solutions and Features

IN THIS CHAPTER

- Add a Solution to the SharePoint Farm
- Display Available Solutions on the Farm
- Get a Specific Solution
- Deploy a Solution to a Web Application
- Retract a Solution
- Upgrade a Deployed Solution
- Remove a Solution from the SharePoint Farm
- Add a Sandboxed Solution to a Site Collection
- Display Available Sandboxed Solutions in a Site Collection
- Get a Specific Sandboxed Solution
- Activate a Sandboxed Solution
- Deactivate a Sandboxed Solution
- Upgrade a Deployed Sandboxed Solution
- Remove a Sandboxed Solution from a Site Collection
- Display Available Features
- Get a Specific Feature
- Activate a Feature
- Deactivate a Feature
- Install a Feature in SharePoint
- Uninstall a Feature from SharePoint
- Export Installed Farm Solutions

This chapter outlines the SharePoint PowerShell commands that are used to manage the solutions and features throughout the SharePoint farm and sites. Solutions are installed on the farm and deployed to specific site collections or globally. Solutions may contain features that become available at the site or site collection level. Using the cmdlets in this chapter assists in the automation of managing these solutions and features.

Add a Solution to the SharePoint Farm

Scenario/Problem: You need to add a solution to the farm using a script.

Solution: Use the `Install-SPSolution` cmdlet.

Using the `Add-SPSolution` cmdlet adds a solution package (.wsp) to the farm. All you need to provide the cmdlet is the actual location of the solution package, as shown in Listing 9.1.

LISTING 9.1 **Adding a Solution to the SharePoint Farm**

```
Add-SPSolution "c:\newsolution.wsp"
```

The solution location is the `–LiteralPath` parameter, which is required. The path may be a local drive location or network location using a UNC string (\\servername\folder).

Display Available Solutions on the Farm

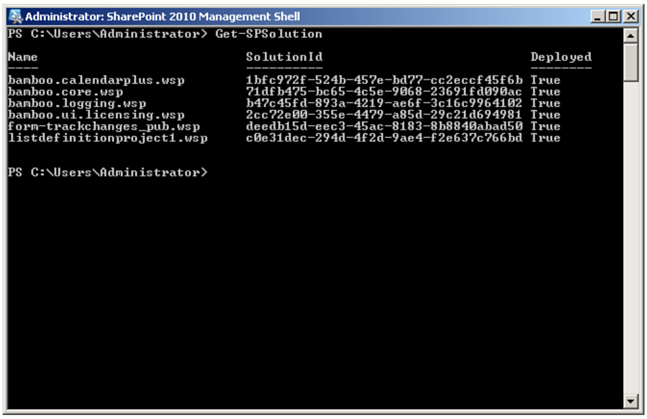
Scenario/Problem: You need to obtain a solution reference and therefore need a listing of available solutions.

Solution: Use the `Get-SPSolution` cmdlet without any parameters to retrieve all available solutions.

In order to obtain a solution reference object, you need to find the identity of the solution. Using the `Get-SPSolution` cmdlet by itself (see Listing 9.2) displays all available solutions, as shown in Figure 9.1.

LISTING 9.2 **Displaying All Available Solutions**

```
Get-SPSolution
```



```
PS C:\Users\Administrator> Get-SPSolution

Name                          SolutionId                           Deployed
----                          ----------                           --------
bamboo.calendarplus.wsp       1bfc972f-524b-457e-bd77-cc2eccf45f6b True
bamboo.core.wsp               71dfb475-bc65-4c5e-9068-23691fd090ac True
bamboo.logging.wsp            b47c45fd-893a-4219-ae6f-3c16c9964102 True
bamboo.ui.licensing.wsp       2cc72e00-355e-4479-a85d-29c21d694981 True
form-trackchanges_pub.wsp     deedb15d-eec3-45ac-8183-8b8840abad50 True
listdefinitionproject1.wsp    c0e31dec-294d-4f2d-9ae4-f2e637c766bd True


PS C:\Users\Administrator>
```

FIGURE 9.1 `Get-SPSolution` displays all available solutions.

Get a Specific Solution

Scenario/Problem: You need to obtain a specific solution reference.

Solution: Use the `Get-SPSolution` cmdlet to assign a variable the results using the `–Identity` parameter.

Providing the identity of the solution with the `Get-SPSolution` cmdlet retrieves a specific solution. Use this to assign the results to an object for use in other cmdlets, as shown in Listing 9.3.

LISTING 9.3 **Assigning a Variable to a Specific Solution**

```
$solution = Get-SPSolution
-Identity c0e31dec-294d-4f2d-9ae4-f2e637c766bd
```

TIP Instead of using the solution GUID, you may instead use the name of the solution as listed in the output when issuing `Get-SPSolution` without any parameters.

Deploy a Solution to a Web Application

Scenario/Problem: You need to deploy a solution that was added to the farm to a web application.

Solution: Use the `Install-SPSolution` cmdlet with a solution variable.

The `Install-SPSolution` cmdlet allows you to deploy a specific solution installed on the farm to a specified web application. Use this in conjunction with the `Get-SPSolution` cmdlet (explained in a previous section), as shown in Listing 9.4.

LISTING 9.4 **Sample Configuration Script for Deploying a Solution**

```
$solution = Get-SPSolution
-Identity c0e31dec-294d-4f2d-9ae4-f2e637c766bd
Install-SPSolution $solution –WebApplication "SharePoint – 80"
–Force -GACDeployment
```

NOTE You may only deploy a solution to a web application if the solution contains components that are scoped at the web application level. To install globally, do not include the `–WebApplication` parameter.

The `–Force` switch parameter forces the solution to be deployed to the specified web application. As noted, if there are no components scoped for the web application level, you cannot even force the solution to be deployed to the web application. However, if you are deploying the solution globally, the `–Force` parameter will redeploy the solution if it has already been previously deployed.

The `–GACDeployment` switch parameter needs to be included if there are assemblies in the solution that will be deployed to the Global Assembly Cache (GAC). Most of the time this is the case—especially with third-party products.

If code access security (CAS) policies are included in the solution, you need to provide the `–CASPolicies` switch parameter to the `Install-SPSolution` command. This is less common but does occur.

TIP To deploy to all web applications, instead of providing a specific web application (using the `–WebApplication` parameter), use the `–AllWebApplications` switch parameter.

Retract a Solution

Scenario/Problem: You need to retract a deployed solution from a web application or global deployment.

Solution: Use the `UnInstall-SPSolution` cmdlet with a solution variable.

The `Uninstall-SPSolution` cmdlet allows you to retract a specific solution that was previously deployed. Use this in conjunction with the `Get-SPSolution` cmdlet (explained in a previous section), as shown in Listing 9.5.

LISTING 9.5 **Retracting a Solution from a Web Application**

```
$solution = Get-SPSolution
-Identity c0e31dec-294d-4f2d-9ae4-f2e637c766bd
Uninstall-SPSolution $solution –WebApplication "SharePoint – 80"
```

Providing the `–WebApplication` parameter retracts the solution from the specified web application (assuming the solution was deployed to that web application and has resources scoped at the web application level). You may also retract the solution from all web applications, as shown in Listing 9.6.

LISTING 9.6 **Retracting a Solution from All Web Applications**

```
$solution = Get-SPSolution
-Identity c0e31dec-294d-4f2d-9ae4-f2e637c766bd
Uninstall-SPSolution $solution –AllWebApplications
```

If the solution is installed globally, all you need to provide to the `Uninstall-SPSolution` cmdlet is the solution identity, as shown in Listing 9.7.

LISTING 9.7 **Retracting a Solution That Was Globally Deployed**

```
$solution = Get-SPSolution
-Identity c0e31dec-294d-4f2d-9ae4-f2e637c766bd
Uninstall-SPSolution $solution
```

Upgrade a Deployed Solution

Scenario/Problem: You have an updated solution package that needs to be deployed to SharePoint.

Solution: Use the `Update-SPSolution` cmdlet with a solution variable.

The `Update-SPSolution` cmdlet allows you to upgrade a previously deployed solution with a newer version of the solution package (and underlying files). Use this in conjunction with the `Get-SPSolution` cmdlet (explained in a previous section), as shown in Listing 9.8.

LISTING 9.8 **Sample Configuration Script for Upgrading a Solution**

```
$solution = Get-SPSolution
-Identity c0e31dec-294d-4f2d-9ae4-f2e637c766bd
Update-SPSolution $solution –LiteralPath "c:\newsolution.wsp" –Force
-GACDeployment
```

The `–LiteralPath` determines the location of the updated solution package. The path may be a local drive location or network location using a UNC string (\\servername\folder).

The `–Force` switch parameter forces the solution to be upgraded.

The `–GACDeployment` switch parameter needs to be included if there are assemblies in the solution that will be redeployed to the Global Assembly Cache (GAC).

If code access security (CAS) policies are included in the solution, you need to provide the `–CASPolicies` switch parameter to the `Update-SPSolution` command.

NOTE If there are new files or changes to the underlying structure, you may not be able to upgrade the existing solution. In this case, you need to retract and remove the solution from the farm and then add the updated package as a new solution.

Remove a Solution from the SharePoint Farm

Scenario/Problem: You need to remove a solution from the SharePoint farm.

Solution: Use the `Remove-SPSolution` cmdlet with a solution variable.

The `Remove-SPSolution` cmdlet allows you to remove a specific solution from the current farm. Use this in conjunction with the `Get-SPSolution` cmdlet (explained in a previous section), as shown in Listing 9.9.

LISTING 9.9 **Sample Configuration Script for Removing a Solution**

```
$solution = Get-SPSolution
 -Identity c0e31dec-294d-4f2d-9ae4-f2e637c766bd
Remove-SPSolution $solution
```

NOTE You may not remove a solution that has been deployed. You must first retract the solution and then perform the removal operation.

Add a Sandboxed Solution to a Site Collection

Scenario/Problem: You need to add a sandboxed solution to a specific site collection.

Solution: Use the `Install-SPUserSolution` cmdlet.

Using the `Add-SPUserSolution` cmdlet adds a solution package (.wsp) to a site collection. All you need to provide the cmdlet is the actual location of the sandboxed solution package and the site collection, as shown in Listing 9.10.

LISTING 9.10 **Adding a Sandboxed Solution to a Site Collection**

```
Add-SPUserSolution –LiteralPath "c:\SBSolution.wsp" –Site
"http://intranet.sp2010.com/sites/UserSC"
```

The sandboxed solution package location is the `–LiteralPath` parameter, which is required. The path may be a local drive location or network location using a UNC string (\\servername\folder).

The `–Site` parameter is the site collection path to which the sandboxed solution should be deployed.

Display Available Sandboxed Solutions in a Site Collection

Scenario/Problem: You need to obtain a sandboxed solution reference and therefore need a listing of available solutions.

Solution: Use the `Get-SPUserSolution` cmdlet with a specified site collection.

In order to obtain a sandboxed solution reference object, you need to find the identity of the sandboxed solution. Using `Get-SPUserSolution` with the `–Site` parameter displays all available sandboxed solutions, as shown in Listing 9.11.

LISTING 9.11 **Displaying All Available Sandboxed Solutions**

```
Get-SPUserSolution –site "http://intranet.sp2010.com/sites/UserSC"
```

Get a Specific Sandboxed Solution

Scenario/Problem: You need to obtain a specific solution reference.

Solution: Use the `Get-SPUserSolution` cmdlet to assign a variable the results using the `–Identity` parameter.

Providing the identity of the sandboxed solution with the `Get-SPUserSolution` cmdlet retrieves a specific sandboxed solution. Use this to assign the results to an object for use in other cmdlets, as shown in Listing 9.12.

LISTING 9.12 **Assigning a Variable to a Specific Sandboxed Solution**

```
$userSolution = Get-SPUserSolution
-Identity c0e31dec-294d-4f2d-9ae4-f2e637c766bd
–site "http://intranet.sp2010.com/sites/UserSC"
```

TIP Instead of using the solution GUID, you may use the name of the solution as listed in the output when issuing `Get-SPUserSolution`.

Activate a Sandboxed Solution

Scenario/Problem: You need to activate a sandboxed solution that was added to the site collection.

Solution: Use the `Install-SPUserSolution` cmdlet with a sandboxed solution variable and site collection reference.

The `Install-SPUserSolution` cmdlet allows you to deploy a specific sandboxed solution installed on the farm to a specified web application. Use this in conjunction with the `Get-SPUserSolution` cmdlet (explained in a previous section), as shown in Listing 9.13.

LISTING 9.13 **Sample Configuration Script for Deploying a Sandboxed Solution**

```
$userSolution = Get-SPUserSolution
-Identity c0e31dec-294d-4f2d-9ae4-f2e637c766bd
Install-SPUserSolution $userSolution –Site
"http://intranet.sp2010.com/sites/UserSC"
```

Deactivate a Sandboxed Solution

Scenario/Problem: You need to deactivate an active sandboxed solution from a site collection.

Solution: Use the `UnInstall-SPUserSolution` cmdlet with a sandboxed solution variable and site collection reference.

The `Uninstall-SPUserSolution` cmdlet allows you to retract a specific sandboxed solution that was previously deployed. Use this in conjunction with the `Get-SPUserSolution` cmdlet (explained in a previous section), as shown in Listing 9.14.

LISTING 9.14 **Retracting a Sandboxed Solution from a Site Collection**

```
$userSolution = Get-SPUserSolution
-Identity c0e31dec-294d-4f2d-9ae4-f2e637c766bd
Uninstall-SPUserSolution $userSolution –Site
"http://intranet.sp2010.com/sites/UserSC"
```

Upgrade a Deployed Sandboxed Solution

Scenario/Problem: You have an updated sandboxed solution package that needs to be deployed to a site collection.

Solution: Use the `Update-SPUserSolution` cmdlet with a sandboxed solution variable.

The `Update-SPUserSolution` cmdlet allows you to upgrade a previously deployed sandboxed solution with a newer version of the sandboxed solution package (and underlying files). You first need to add the new package to the site collection, as shown in Listing 9.15.

LISTING 9.15 **Sample Configuration Script for Upgrading a Sandboxed Solution**

```
Add-SPUserSolution –LiteralPath "c:\SBSolution2.wsp"
–Site "http://intranet.sp2010.com/sites/UserSC"
$userSolution = Get-SPUserSolution
-Identity c0e31dec-294d-4f2d-9ae4-f2e637c766bd
Update-SPUserSolution $userSolution
–site "http://intranet.sp2010.com/sites/UserSC"
–ToSolution SBSolution2
```

Remove a Sandboxed Solution from a Site Collection

Scenario/Problem: You need to remove a sandboxed solution from a site collection.

Solution: Use the `Remove-SPUserSolution` cmdlet with a sandboxed solution variable.

The `Remove-SPUserSolution` cmdlet allows you to remove a specific sandboxed solution from a specified site collection. Use this in conjunction with the `Get-SPUserSolution` cmdlet (explained in a previous section), as shown in Listing 9.16.

LISTING 9.16 **Sample Configuration Script for Removing a Sandboxed Solution**

```
$userSolution = Get-SPUserSolution
-Identity c0e31dec-294d-4f2d-9ae4-f2e637c766bd
Remove-SPUserSolution $userSolution
–site "http://intranet.sp2010.com/sites/UserSC"
```

NOTE You may not remove a solution that has been deployed. You must first retract the solution and then perform the removal operation.

Display Available Features

Scenario/Problem: You need to obtain a feature reference and therefore need a listing of available features.

Solution: Use the `Get-SPFeature` cmdlet using the desired scope.

The `Get-SPFeature` cmdlet is handy for listing out the installed features at various levels within your SharePoint environment. Issuing the command without any parameters displays all features at every level. Because there are about 12 dozen features at all levels, it is usually not helpful to list them all out.

Luckily, the `Get-SPFeature` cmdlet has several parameter sets you may use to narrow your list of features. To display the features at the farm level, simply use the `–Farm` switch parameter, as shown in Listing 9.17.

LISTING 9.17 **Displaying All Installed Farm Features**

```
Get-SPFeature –Farm
```

To display the features at the web application level, use the `–WebApplication` parameter and provide the web application name or identity, as shown in Listing 9.18.

LISTING 9.18 **Displaying All Installed Web Application Features**

```
Get-SPFeature –WebApplication "SharePoint – 80"
```

This works for site collection and site features as well. For site collections, use the `–Site` parameter (see Listing 9.19) and for site features use the `–Web` parameter (see Listing 9.20).

LISTING 9.19 **Displaying All Installed Site Collection Features**

```
Get-SPFeature –Site "http://sp2010"
```

LISTING 9.20 **Displaying All Installed Site Features**

```
Get-SPFeature –Web "http://sp2010"
```

Get a Specific Feature

Scenario/Problem: You need to obtain a specific feature reference.

Solution: Use the `Get-SPFeature` cmdlet to assign a variable the results using the `–Identity` parameter.

Providing the identity of the solution with the `Get-SPSolution` cmdlet retrieves a specific solution. Use this to assign the results to an object for use in other cmdlets, as shown in Listing 9.21.

LISTING 9.21 **Assigning a Variable to a Specific Feature**

```
$feature = Get-SPFeature
-Identity e8389ec7-70fd-4179-a1c4-6fcb4342d7a0
```

TIP Although you may also retrieve the feature using its display name, it is recommended to use the feature GUID such that the exact feature and scope can be determined. The feature GUID uniquely identifies the feature, whereas there could be a site collection and site -level feature using the same display name.

Activate a Feature

Scenario/Problem: You need to activate a feature.

Solution: Use the `Enable-SPFeature` cmdlet with a feature variable.

The `Enable-SPFeature` cmdlet allows you to activate a feature at the determined level. Use this in conjunction with the `Get-SPFeature` cmdlet (explained in the previous section), as shown in Listing 9.22.

LISTING 9.22 **Sample Script for Activating a Feature**

```
$feature = Get-SPFeature
-Identity e8389ec7-70fd-4179-a1c4-6fcb4342d7a0
Enable-SPFeature $feature –Url "http://sp2010" -Force
```

The `–Url` parameter is either the web application, site collection, or site in which the feature should be activated. At the root level of the main site, the address is the same. Features can only be activated at the level in which they reside.

The `–Force` switch parameter forces the activation and reruns any customized code. This occurs if the feature is already activated and you are forcing a "reactivation."

NOTE To enable farm features, do not provide a `–Url` parameter.

Deactivate a Feature

Scenario/Problem: You need to deactivate a feature.

Solution: Use the `Disable-SPFeature` cmdlet with a feature variable.

The `Disable-SPFeature` cmdlet allows you to deactivate a feature at the determined level. Use this in conjunction with the `Get-SPFeature` cmdlet (explained in a previous section), as shown in Listing 9.23.

LISTING 9.23 **Sample Script for Deactivating a Feature**

```
$feature = Get-SPFeature
-Identity e8389ec7-70fd-4179-a1c4-6fcb4342d7a0
Disable-SPFeature $feature –Url "http://sp2010" -Force
```

The `–Url` parameter is either the web application, site collection, or site in which the feature should be activated. At the root level of the main site, the address is the same. Features can only be deactivated at the level in which they reside and are activated.

The `–Force` switch parameter forces the deactivation and does not produce any errors if the feature is already deactivated.

NOTE To disable farm features, do not provide a `–Url` parameter.

Install a Feature in SharePoint

Scenario/Problem: You need to install a feature to SharePoint.

Solution: Use the `Install-SPFeature` cmdlet with the `–Path` parameter.

The `Install-SPFeature` cmdlet allows you to install a feature that already has been copied or deployed to the SharePoint environment. The feature files should reside within a folder under the 14\TEMPLATE\FEATURES location of the SharePoint servers. The folder that contains the feature files is used as the `–Path` parameter with the `Install-SPFeature` cmdlet, as shown in Listing 9.24.

LISTING 9.24 **Sample Script for Installing a Feature**

```
Install-SPFeature –Path "CustomFeature" -Force
```

The `–Force` switch parameter forces the installation and does not produce any errors if the feature has already been installed.

NOTE The feature scope is contained within the feature.xml file and therefore no other parameters are required to perform the installation. The feature is available at the level of its scope.

Uninstall a Feature from SharePoint

Scenario/Problem: You need to uninstall a feature from SharePoint.

Solution: Use the `Install-SPFeature` cmdlet with a feature variable.

The `Uninstall-SPFeature` cmdlet allows you to uninstall a feature from SharePoint. Use this in conjunction with the `Get-SPFeature` cmdlet (explained in a previous section), as shown in Listing 9.25.

LISTING 9.25 **Sample Script for Uninstalling a Feature**

```
$feature = Get-SPFeature
-Identity e8389ec7-70fd-4179-a1c4-6fcb4342d7a0
Uninstall-SPFeature $feature  -Force
```

The `–Force` switch parameter forces the feature to be uninstalled and does not produce any errors if the feature has already been uninstalled.

NOTE The feature scope is contained within the feature.xml file and therefore no other parameters are required to perform the uninstallation. The feature is uninstalled at the level of its scope. This does not remove the files in the feature folder, however.

Export Installed Farm Solutions

Scenario/Problem: You want to export the installed farm solutions to local files.

Solution: Create a script that loops through the solutions and saves them locally.

Using the SharePoint assemblies, you may easily generate a script that loops through the installed farm solutions and actually extracts them into their .wsp files on a local drive. This becomes extremely valuable when you are moving solutions from one environment to another (for example, from development to staging). A sample script is shown in Listing 9.26.

LISTING 9.26 **Sample Script for Exporting All Farm Solutions**

```
[Void][System.Reflection.Assembly]::LoadWithPartialName(
"Microsoft.SharePoint")
[Void][System.Reflection.Assembly]::LoadWithPartialName(
"Microsoft.SharePoint.Administration")

    $spFarm = [Microsoft.SharePoint.Administration.SPFarm]::Local
    $spFarmSolutions = $spFarm.Solutions
    $localPath = "C:\Solutions\"

    foreach($spFarmSolution in $spFarmSolutions)
    {
         [string]$outputFileName = $localPath + $spFarmSolution.Name
        $spFarmSolution.SolutionFile.SaveAs($outputFileName);
    }
```

Set the `$localPath` variable to a location where you want the solution files to be extracted and saved.

CHAPTER 10

Content Deployment

IN THIS CHAPTER

- ▶ Create a New Deployment Path
- ▶ Display Deployment Paths Configured on the Farm
- ▶ Get a Specific Content Deployment Path
- ▶ Remove a Content Deployment Path
- ▶ Create a New Deployment Job
- ▶ Display Deployment Jobs Configured on the Farm
- ▶ Get a Specific Content Deployment Job
- ▶ Remove a Content Deployment Job
- ▶ Start a Content Deployment Job
- ▶ Modify Content Deployment Configurations

This chapter explains how to use the SharePoint cmdlets to manage and administer content deployment within your SharePoint farm. Content deployment involves the publishing of internal content to another site (usually public facing). Your farm topology must be properly configured to implement content publishing and deployment.

Create a New Deployment Path

Scenario/Problem: You need to set up and configure a new deployment path.

Solution: Use the `New-SPContentDeploymentPath` cmdlet.

Setting up a new deployment path involves mapping the source web application and site to a target web application and site. You may use the `New-SPContentDeploymentPath` cmdlet to facilitate this mapping, as shown in Listing 10.1

LISTING 10.1 **Creating a New Content Deployment Path**

```
New-SPContentDeploymentPath –Name "Marketing Internet Content"
–SourceSPWebApplication "SharePoint – 80"
–SourceSPSite "http://intranet.sp2010.com/Marketing"
–DestinationCentralAdministrationURL "http://internet.sp2010.com:12345"
–DestinationSPWebApplication "SharePoint – Internet"
–DestinationSPSite "http://internet.sp2010.com/Marketing"
–PathAccount (Get-Credential)
```

TIP Use inline credentials instead of `(Get-Credential)`. See Chapter 2, “PowerShell Basics,” for more details.

The `–PathAccount` parameter is used to authenticate to the destination server and site. Therefore, this account must have the appropriate permissions and should be an administrator of the destination Central Administration server.

NOTE The destination Central Administration server must be configured to accept content deployment jobs before the path can be created.

Display Deployment Paths Configured on the Farm

Scenario/Problem: You need to identify deployment paths for proper job creation.

Solution: Use the `Get-SPContentDeploymentPath` cmdlet without any parameters to retrieve all available content deployment paths.

In order to obtain a Content Deployment Path reference object, you need to find the identity of the path entry. Use the `Get-SPContentDeploymentPath` cmdlet by itself, as shown in Listing 10.2, to display all available content deployment paths.

LISTING 10.2 **Displaying All Available Content Deployment Paths**

```
Get-SPContentDeploymentPath
```

Get a Specific Content Deployment Path

Scenario/Problem: You need to obtain a specific content deployment path reference.

Solution: Use the `Get-SPContentDeploymentPath` cmdlet to assign a variable the results using the `–Identity` parameter.

Providing the identity of the content deployment path with the `Get-SPContent DeploymentPath` cmdlet retrieves a specific content deployment path. Use this to assign the results to an object for use in other cmdlets, as shown in Listing 10.3.

LISTING 10.3 **Assigning a Variable to a Specific Content Deployment Path**

```
$path = Get-SPContentDeploymentPath
-Identity "Marketing Internet Content"
```

Remove a Content Deployment Path

Scenario/Problem: You need to delete a content deployment path configuration from the SharePoint farm.

Solution: Use the `Remove-SPContentDeploymentPath` cmdlet.

The `Remove-SPContentDeploymentPath` cmdlet allows you to remove a specific content deployment path configured on the current farm. Simply provide the name of the content deployment path, as shown in Listing 10.4.

LISTING 10.4 **Removing a Specific Content Deployment Path**

```
Remove-SPContentDeploymentPath "Marketing Internet Content"
```

NOTE You may not remove a path that has a content deployment job configured. Remove the deployment job first before removing the deployment path.

Create a New Deployment Job

Scenario/Problem: You need to set up and configure a new deployment job.

Solution: Use the `New-SPContentDeploymentJob` cmdlet.

Setting up a new deployment job involves mapping the source web application and site to a target web application and site. You may use the `New-SPContentDeploymentJob` cmdlet to facilitate this mapping, as shown in Listing 10.5.

LISTING 10.5 **Creating a New Content Deployment Job**

```
New-SPContentDeploymentJob –Name "Marketing Deployment Job"
–SPContentDeploymentPath "Marketing Internet Content"
```

The `–SPContentDeploymentPath` parameter is the name of the deployment path that will be used for this deployment job.

NOTE A content deployment path must be configured first before a content deployment job can be created.

Display Deployment Jobs Configured on the Farm

Scenario/Problem: You need to obtain identify deployment paths for proper job creation.

Solution: Use the `Get-SPContentDeploymentJob` cmdlet without any parameters to retrieve all available deployment jobs.

In order to obtain a Content Deployment Job reference object, you need to find the identity of the job entry. Using the `Get-SPContentDeploymentJob` cmdlet by itself, as shown in Listing 10.6, displays all of the deployment jobs.

LISTING 10.6 **Displaying All Available Content Deployment Jobs**

```
Get-SPContentDeploymentJob
```

Get a Specific Content Deployment Job

Scenario/Problem: You need to obtain a specific content deployment path reference.

Solution: Use the `Get-SPContentDeploymentJob` cmdlet to assign a variable the results using the `–Identity` parameter.

Providing the identity of the content deployment job with the `Get-SPContentDeploymentJob` cmdlet retrieves a specific content deployment job. Use this to assign the results to an object for use in other cmdlets, as shown in Listing 10.7.

LISTING 10.7 **Assigning a Variable to a Specific Content Deployment Job**

```
$job = Get-SPContentDeploymentJob -Identity "Marketing Deployment Job"
```

Remove a Content Deployment Job

Scenario/Problem: You need to delete a content deployment path configuration from the SharePoint farm.

Solution: Use the `Remove-SPContentDeploymentJob` cmdlet.

The `Remove-SPContentDeploymentJob` cmdlet allows you to remove a specific deployment job from the current farm. Simply provide the name of the content deployment job, as shown in Listing 10.8.

LISTING 10.8 **Removing a Specific Content Deployment Job**

```
Remove-SPContentDeploymentJob "Marketing Deployment Job"
```

Start a Content Deployment Job

Scenario/Problem: You need to kick off a deployment job such that content is deployed from the source site to the target site.

Solution: Use the `Start-SPContentDeploymentJob` cmdlet.

The `Start-SPContentDeploymentJob` cmdlet allows you to manually start a content deployment job. Simply provide the name of the content deployment job, as shown in Listing 10.9.

LISTING 10.9 **Manually Starting a Specific Content Deployment Job**

```
Start-SPContentDeploymentJob "Marketing Deployment Job"
```

Modify Content Deployment Configurations

Scenario/Problem: You need to modify properties of content deployment paths or jobs.

Solution: Use the cmdlets listed in this section to change the configuration of existing deployment paths and jobs.

This chapter covers the basic or more common cmdlets for managing content deployments. For more advance scripting and modifications of content deployment jobs and paths, the following cmdlets are available:

- `Set-SPContentDeploymentPath`
- `Set-SPContentDeploymentJob`

PART IV

Service Application Management

CHAPTER 11

InfoPath Forms Services

IN THIS CHAPTER

This chapter explores the administrative aspects of InfoPath Form Services through the use of PowerShell.

Browser-Enable Form Templates

Scenario/Problem: You need to allow custom SharePoint forms to be rendered within SharePoint.

Solution: Use the `Set-SPInfoPathFormsService` cmdletcmdlet.

From the PowerShell cmdlet prompt, enter the following:

```
Set-SPInfoPathFormsService -AllowUserFormBrowserEnabling $true
-AllowUserFormBrowserRendering $true
```

TIP To disable the browser-enabled form templates, replace `$true` with `$false`.

Throttle Data Connection Timeouts

Scenario/Problem: You need to increase or decrease the data connection timeout limit.

Solution: Use the `Set-SPInfoPathFormsService` cmdlet with the data connection timeout parameters.

Your form will attempt to connect and access data sources based on your configured data connections. If the data source is inaccessible, busy, or down, InfoPath Forms Services needs a way to time out the connection such that the form or the browser doesn't stop responding.

Use the `–DefaultDataConnectionTimeout` and `–MaxDataConnectionTimeout` parameters with the `Set-SPInfoPathFormsService` cmdlet to set the timeout values, as shown in Listing 11.1.

LISTING 11.1 **Data Connection Timeout Cmdlet Line**

```
Set-SPInfoPathFormsService –DefaultDataConnectionTimeout 15
–MaxDataConnectionTimeout 25
```

The timeout values are in milliseconds. The default value for `DefaultDataConnectionTimeout` is 10 (10, milliseconds, or 10 seconds), whereas the default value for `MaxDataConnectionTimeout is 20` (20, milliseconds, 20 seconds).

TIP The default values are usually acceptable entries and do not need to be modified. However, if you have some long-running data processes or connectivity issues, you may want to increase these numbers slightly.

Throttle the Data Connection Response Size

Scenario/Problem: You need to increase or decrease the data connection response size.

Solution: Use the `MaxDataConnectionResponseSize` parameter with the `Set-SPInfoPathFormsService` cmdlet.

When connecting to data sources, you do not want to overload other systems. Limiting the response size determines how many kilobytes the data connection can process. You can modify the setting by using the `MaxDataConnectionReponseSize` parameter with the number of kilobytes, as shown in Listing 11.2.

LISTING 11.2 **Data Connection Response Size Cmdlet Line**

```
Set-SPInfoPathFormsService –MaxDataConnectionResponseSize 3
```

TIP The default value of 1500 (1,500KB) is usually a good setting. You do not want to strain other systems or databases. This also helps limit traffic on the network. Only increase this value if you have a large form that submits a great deal of data.

Modify Authentication Settings

Scenario/Problem: You need to modify authentication settings.

Solution: Use the `Set-SPInfoPathFormsService` cmdlet with the appropriate parameters.

Several data authentication options are available that either allow or prohibit certain types of data connectivity, as follows:

- `RequireSslForDataConnections`—Set this option to `true` to require SSL encryption when your form uses HTTP authentication (such as when accessing a web service).
- `AllowEmbeddedSqlForDataConnection`—Set this option to `true` if you have data connection files that contain SQL database connection information including the username and password.

- `AllowUdcAuthenticationForDataConnections`—Set this option to `true` to allow custom forms to access data sources through data connection files.
- `AllowUserFormCrossDomainDataConnections`—Select this option if your form needs to access data sources on a different domain than SharePoint.

To configure these settings, use the `Set-SPInfoPathFormsService` cmdlet with the desired parameters and settings, as shown in Listing 11.3.

LISTING 11.3 **Authentication Settings Cmdlet Line**

```
Set-SPInfoPathFormsService –RequireSslForDataConnections $true –
AllowEmbeddedSqlForDataConnection $true –
AllowUdcAuthenticationForDataConnections $true –
AllowUserFormCrossDomainDataConnections $true
```

Configure Session State

Scenario/Problem: You want to configure session state settings.

Solution: Use the `Set-SPInfoPathFormsService` cmdlet with the appropriate parameters.

When a user is filling out a SharePoint form, InfoPath Form Services uses the SharePoint Server State service to manage user sessions. (You can switch this to View State using PowerShell; see the next section.) Transient data is stored within the session state.

Several settings are involved in the session state configuration:

- `MaxPostbacksPerSession`—Typically, there shouldn't be too much chatter between the form and Forms Services. This setting prevents unintended post-backs or "out-of-control" communications.
- `MaxUserActionsPerPostback`—There should be only a handful of actions per postback. Once again, this is to prevent a form process from pegging server resources.
- `ActiveSessionTimeout`—By default, active sessions are terminated after 1,440 minutes, which essentially is 24 hours. This is more of a cleanup process than anything. Some forms may take a long time to fill out, depending on the information required, but one day should be plenty of time.
- `MaxSizeOfFormSessionState`—This determines how much data can be stored for the active user session. The default is 4,096KB, which equates to 4MB. This provides plenty of room to store user session data. Typical forms and user information should only take up several kilobytes, if not bytes.

If the user session lasts too long or there are too many postbacks to the server, the session is ended based on the configured settings. All form data will be lost, and the user will need to start over. This is done to prevent unintended communication, limit network traffic, and limit system resource utilization.

To configure these settings, use the `Set-SPInfoPathFormsService` cmdlet with the desired parameters and settings, as shown in Listing 11.4.

LISTING 11.4 **Session State Settings Cmdlet Line**

```
Set-SPInfoPathFormsService –MaxPostbacksPerSession 110 –
ActiveSessionTimeout 720
```

TIP If forms are constantly being filled out within your organization and are fairly lightweight, you might actually want to reduce some of these values to keep recycling the session state as well as system resources. Another option is to use View State instead, as described in the next section.

Enable View State

Scenario/Problem: Instead of using Session State, you would like to use View State.

Solution: Use the `AllowViewState` parameter with the `Set-SPInfoPathFormsService` cmdlet.

View State sessions (or Form View) store session data within the client browser, thus reducing the load on the database server. Although this assists in maximizing database performance, there is more bandwidth used in View State than in Session State. The maximum session size allowed in the View State is 40KB. If there is an instance where this limit is passed, the session automatically switches back to Session State.

To configure these settings, use the `Set-SPInfoPathFormsService` cmdlet with the `–AllowViewState` and `-ViewStateThreshold` parameters, as shown in Listing 11.5.

LISTING 11.5 **View State Settings Cmdlet Line**

```
Set-SPInfoPathFormsService –AllowViewState $true –ViewStateThreshold
40961
```

TIP You can only modify this setting using the PowerShell cmdlet line. There are no equivalent options in Central Administration.

Verify and Upload a Form Template

Scenario/Problem: You need to verify and upload a form template to InfoPath Form Services.

Solution: Use the `Test-SPInfoPathFormTemplate` and `Install-SPInfoPathFormTemplate` PowerShell cmdlets to verify and upload the form template.

Forms that have code-behind or require Full Trust permissions cannot be published directly to SharePoint. These forms must be published to a network location and then uploaded to Form Services. Before uploading, you should verify that the form is correct and can be installed.

To verify a form template, use the following cmdlet:

```
Test-SPInfoPathFormTemplate
-Path "<<full path to InfoPath form template>>"
```

To upload a form to InfoPath Form Services, use the following cmdlet:

```
Install-SPInfoPathFormTemplate
-Path "<<full path to InfoPath form template>>"
```

TIP Use the switch parameter `–EnableGradualUpgrade` with `Install-SPInfoPathFormTemplate` to ensure that existing forms are upgraded.

Upload Multiple Form Templates at Once

Scenario/Problem: You have many form templates to upload to InfoPath Form Services but find it difficult performing the same process for each one individually.

Solution: Pipe the paths of all form templates you want to upload to the `Install-SPInfoPathFormTemplate` cmdlet.

You may upload multiple form templates by listing out each path, separated by a comma, and then using the pipe (|) to pass that list into the `Install-SPInfoPathFormTemplate` cmdlet, as shown in Listing 11.6.

LISTING 11.6 **Multiple-Form Upload Cmdlet Line**

```
"<<path of form 1>>" , "<<path of form 2>" , "<<path of form 3>>" |
Install-SPInfoPathFormTemplate
```

Activate or Deactivate a Form to or from a Site Collection

Scenario/Problem: You need to activate or deactivate a form to or from a site collection.

Solution: Use the `Enable-SPInfoPathFormTemplate` or `Disable-SPInfoPathFormTemplate` cmdlet.

When you upload a form, it generates and deploys a feature for your form to each site collection in your farm. Using the `Enable-SPInfoPathFormTemplate` cmdlet activates that feature on the site collection you choose. Conversely, using the `Disable-SPInfoPathFormTemplate` cmdlet deactivates the form feature of the selected site collection.

To activate a form to a site collection, use the following cmdlet:

```
Enable-SPInfoPathFormTemplate
-Identity "<<name of form template>>"
-Site "<<URL of Site Collection>>"
```

To deactivate a form from a site collection, use the following cmdlet:

```
Disable-SPInfoPathFormTemplate
-Identity "<<name of form template>>"
-Site "<<URL of Site Collection>>"
```

Remove a Form from InfoPath Form Services

Scenario/Problem: You need to remove a form from InfoPath Form Services.

Solution: Use the `Uninstall-SPInfoPathFormTemplate` PowerShell cmdlet.

Removing a form deletes it from InfoPath Forms Services. It is no longer available on any site collection. To remove a form from InfoPath Form Services, use the following cmdlet:

```
Uninstall-SPInfoPathFormTemplate -Identity "<<name of form template>>"
```

TIP Removing a form does not always remove the content type it generates. Make sure the form is not being used anywhere in SharePoint. Otherwise, errors will occur when SharePoint tries to access the form template via the content type. (Hint: Remove the content type first because SharePoint will notify if it is being used.)

Quiesce a Form from InfoPath Form Services

Scenario/Problem: You need to quiesce a form from InfoPath Form Services.

Solution: Use the `Stop-SPInfoPathFormTemplate` PowerShell cmdlet.

Quiescing a form will halt the initiation of new form instances and allow existing user sessions to complete. To quiesce a form from InfoPath Form Services, use the following cmdlet:

```
Stop-SPInfoPathFormTemplate -Identity "<<name of form template>>"
```

Enable the Web Service Proxy

Scenario/Problem: You need to enable the use of the InfoPath Form Services web service proxy.

Solution: Use the `Set-SPInfoPathWebServiceProxy` cmdlet.

The InfoPath Form Services web services proxy can be used to access web services without passing the credentials of the form user. A set of credentials for the web service needs to be stored within SharePoint's Secure Store Service (SSS), which is the replacement for Single Sign-On (SSO).

A data connection file that your InfoPath form is using as a web service connection simply needs to be modified by adding a `UseFormsServiceProxy` attribute with a setting of `true` (see Listing 11.7), along with an `Authentication` element that references the Secure Store application ID (see Listing 11.8).

LISTING 11.7 **UseFormsServiceProxy Attribute in the Data Connection File**

```
<udc:ServiceUrl UseFormsServiceProxy="true"/>
```

LISTING 11.8 **Authentication Element in the Data Connection File**

```
<udc:Authentication>
    <udc:SSO AppId="<<AppID>>" CredentialType="<<credential type>>"/>
</udc:Authentication>
```

The possible credential types are as follows:

- Basic
- CD
- Digest
- Kerberos
- NTLM
- SQL

To enable the use of the web proxy, use the following cmdlet:

```
Set-SPInfoPathWebServiceProxy -Identity "<<URL of web application>>"
-AllowWebServiceProxy $true
```

To enable SharePoint forms to use the web proxy, use the following cmdlet:

```
Set-SPInfoPathWebServiceProxy -Identity "<<URL of web application>>"
-AllowForUserForms $true
```

TIP The `AllowWebServiceProxy` setting needs to be set to `true` first before `AllowForUserForms` can be set to `true`. This is why the cmdlets need to be separated.

CHAPTER 12

Enterprise Search

IN THIS CHAPTER

- Display the Enterprise Search Service Information
- Display Enterprise Search Service Instances
- Get a Specific Enterprise Search Service Instance
- Configure the Crawl Account for Enterprise Search
- Configure the Enterprise Search Service Performance Level
- Display Enterprise Search Service Applications
- Get a Specific Enterprise Search Service Application
- Create a Custom Metadata Category
- Display Available Metadata Categories
- Get a Specific Metadata Category
- Create a Custom Metadata Crawled Property
- Display Available Crawled Properties
- Get a Specific Metadata Crawled Property
- Create a Custom Metadata Managed Property
- Display Available Managed Properties
- Get a Specific Metadata Managed Property
- Create a Metadata Mapping

This chapter explores the administrative aspects of Enterprise Search through the use of SharePoint PowerShell cmdlets.

Display the Enterprise Search Service Information

Scenario/Problem: You want to view the information on the Enterprise Search Service.

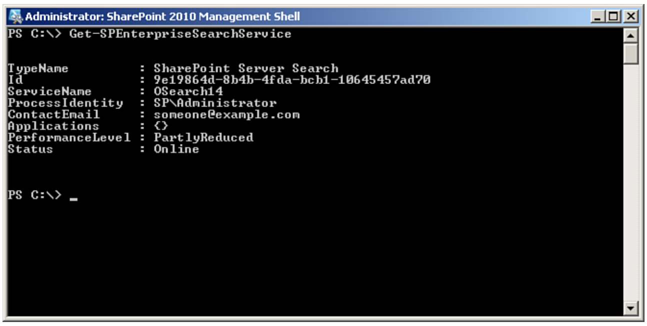
Solution: Use the `Get-SPEnterpriseSearchService` cmdlet.

The `Get-SPEnterpriseSearchService` cmdlet provides information on the Enterprise Search Service that is running on the SharePoint farm. No parameters are necessary, as shown in Listing 12.1.

LISTING 12.1 **Displaying the Enterprise Search Service Information**

```
Get-SPEnterpriseSear1chService
```

The search service information is displayed, as shown in Figure 12.1.



```
PS C:\> Get-SPEnterpriseSearchService


TypeName           : SharePoint Server Search
Id                 : 9e19864d-8b4b-4fda-bcb1-10645457ad70
ServiceName        : OSearch14
ProcessIdentity    : SP\Administrator
ContactEmail       : someone@example.com
Applications       : {}
PerformanceLevel   : PartlyReduced
Status             : Online



PS C:\> _
```

FIGURE 12.1 Displaying the Enterprise Search Service information

Display Enterprise Search Service Instances

Scenario/Problem: You want to view the information on the search service instances.

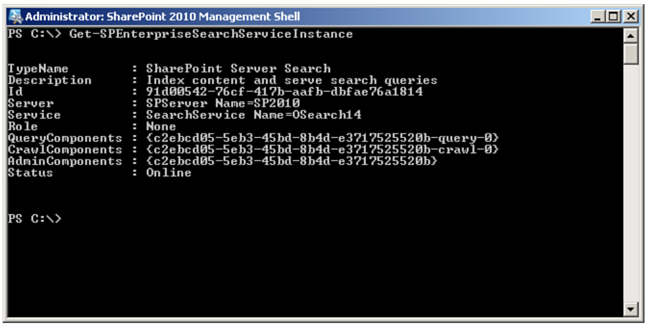
Solution: Use the `Get-SPEnterpriseSearchServiceInstance` cmdlet.

The `Get-SPEnterpriseSearchServiceInstance` cmdlet provides information on the Enterprise Search Service instances that are mapped to the search service. No parameters are necessary, as shown in Listing 12.2.

LISTING 12.2 **Displaying the Enterprise Search Service Instances**

```
Get-SPEnterpriseSearchServiceInstance
```

The Enterprise Search Service instance information is displayed, as shown in Figure 12.2.



```
PS C:\> Get-SPEnterpriseSearchServiceInstance


TypeName          : SharePoint Server Search
Description       : Index content and serve search queries
Id                : 91d00542-76cf-417b-aafb-dbfae76a1814
Server            : SPServer Name=SP2010
Service           : SearchService Name=OSearch14
Role              : None
QueryComponents   : {c2ebcd05-5eb3-45bd-8b4d-e371752520b-query-0}
CrawlComponents   : {c2ebcd05-5eb3-45bd-8b4d-e371752520b-crawl-0}
AdminComponents   : {c2ebcd05-5eb3-45bd-8b4d-e371752520b}
Status            : Online



PS C:\>
```

FIGURE 12.2 Displaying the Enterprise Search Service instance information

Get a Specific Enterprise Search Service Instance

Scenario/Problem: You need to obtain an Enterprise Search Service instance reference.

Solution: Use the `Get-SPEnterpriseSearchServiceInstance` cmdlet to assign a variable the results using the `–Identity` parameter.

Providing the identity of the Enterprise Search Service instance with the `Get-SPEnterpriseSearchServiceInstance` cmdlet retrieves a specific search service instance. Use this to assign the results to an object for use in other cmdlets, as shown in Listing 12.3.

LISTING 12.3 **Assigning a Variable to a Specific Search Service Application**

```
$ssInstance = Get-SPEnterpriseSearchServiceInstance
-Identity 91d00542-76cf-417b-aafb-dbfae76a1814
```

Configure the Crawl Account for Enterprise Search

Scenario/Problem: You need to configure the crawl account for the Enterprise Search Service.

Solution: Use the `Set-SPEnterpriseSearchService` cmdlet with the crawl account parameters.

You may set the crawl account information for the search service by using the `CrawlAccount` and `CrawlPassword` parameters with the `Set-SPEnterpriseSearchService` cmdlet, as shown in Listing 12.4.

LISTING 12.4 **Configuring the Search Service Crawl Account**

```
Set-SPEnterpriseSearchService
–Identity 9e19864d-8b4b-4fda-bcb1-10645457ad70
-CrawlAccount "SP\SharePointCrawl"
-CrawlPassword (ConvertTo-SecureString -asplaintext -force
 -string "SharePoint1")
```

The password needs to be a secure string; therefore, the `ConvertTo-SecureString` cmdlet is used to assist. The Crawl Account setting is the Content Access Account when you are managing the Enterprise Search Service on the server.

Configure the Enterprise Search Service Performance Level

Scenario/Problem: You need to change the performance level for the search service.

Solution: Use the `Set-SPEnterpriseSearchService` cmdlet with the `PerformanceLevel` parameter.

You may set the performance level using the `PerformanceLevel` parameter with the `Set-SPEnterpriseSearchService` cmdlet, as shown in Listing 12.5.

LISTING 12.5 **Configuring the Enterprise Search Service Performance Level**

```
Set-SPEnterpriseSearchService
–Identity 9e19864d-8b4b-4fda-bcb1-10645457ad70
–PerformanceLevel Reduced
```

The `PerformanceLevel` parameter determines how many threads can be utilized by the search service. The valid values are as follows:

- `Reduced`
- `PartlyReduced`
- `Maximum`

The default setting is `PartlyReduced`.

Display Enterprise Search Service Applications

Scenario/Problem: You want to view the information on the search service instances.

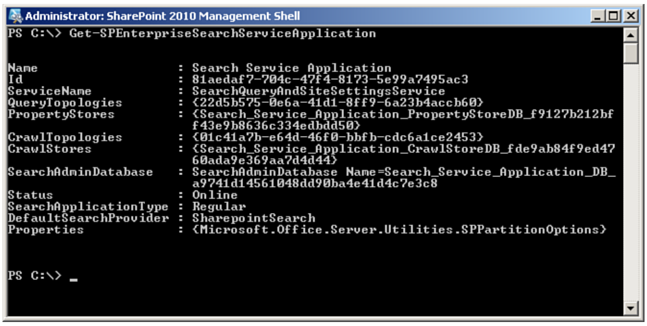
Solution: Use the `Get-SPEnterpriseSearchServiceApplication` cmdlet.

The `Get-SPEnterpriseSearchServiceApplication` cmdlet provides information on the Enterprise Search Service applications configured on the SharePoint farm. No parameters are necessary, as shown in Listing 12.6.

LISTING 12.6 **Displaying the Enterprise Search Service Applications**

```
Get-SPEnterpriseSearchServiceApplication
```

The Enterprise Search Service application information is displayed, as shown in Figure 12.3.



```
PS C:\> Get-SPEnterpriseSearchServiceApplication


Name                    : Search Service Application
Id                      : 81aedaf7-704c-47f4-8173-5e99a7495ac3
ServiceName             : SearchQueryAndSiteSettingsService
QueryTopologies         : {22d5b575-0e6a-41d1-8ff9-6a23b4accb60}
PropertyStores          : {Search_Service_Application_PropertyStoreDB_f9127b212bf
                          f43e9b8636c334edbdd50}
CrawlTopologies         : {01c41a7b-e64d-46f0-bbfb-cdc6a1ce2453}
CrawlStores             : {Search_Service_Application_CrawlStoreDB_fde9ab84f9ed47
                          60ada9e369aa7d4d44}
SearchAdminDatabase     : SearchAdminDatabase Name=Search_Service_Application_DB_
                          a9741d14561048dd90ba4e41d4c7e3c8
Status                  : Online
SearchApplicationType   : Regular
DefaultSearchProvider   : SharepointSearch
Properties              : {Microsoft.Office.Server.Utilities.SPPartitionOptions}



PS C:\> _
```

FIGURE 12.3 Displaying the Enterprise Search Service application information

Get a Specific Enterprise Search Service Application

Scenario/Problem: You need to obtain an Enterprise Search Service application reference.

Solution: Use the `Get-SPEnterpriseSearchServiceApplication` cmdlet to assign a variable the results using the `–Identity` parameter.

Providing the identity of the Enterprise Search Service application with the `Get-SPEnterpriseSearchServiceApplication` cmdlet retrieves a specific Enterprise Search Service application. Use this to assign the results to an object for use in other cmdlets, as shown in Listing 12.7.

LISTING 12.7 **Assigning a Variable to a Specific Search Service Application**

```
$ssApp = Get-SPEnterpriseSearchServiceApplication
-Identity 81aedaf7-704c-47f4-8173-5e99a7495ac3
```

Create a Custom Metadata Category

Scenario/Problem: You want to create a metadata category for custom crawled properties.

Solution: Use the `New-SPEnterpriseSearchMetadataCategory` cmdlet.

Creating a metadata category allows you to organize your custom crawled properties. This makes it easier to manage custom search configurations. Creating a new category first allows you to use it when generating your crawled properties. You need to provide an Enterprise Search Service application reference, as shown in Listing 12.8.

LISTING 12.8 **Creating a New Metadata Category**

```
$ssApp = Get-SPEnterpriseSearchServiceApplication
-Identity 81aedaf7-704c-47f4-8173-5e99a7495ac3
New-SPEnterpriseSearchMetaDataCategory -Name "Custom Properties"
–SearchApplication $ssApp
```

Several switch parameter options may be used when generating a new metadata category:

- `AutoCreateNewManagedProperties`
- `DiscoverNewProperties`
- `MapToContents`

Display Available Metadata Categories

Scenario/Problem: You want to view the information on the available search metadata categories.

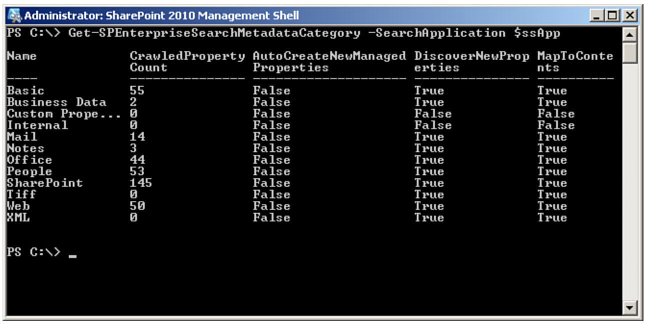
Solution: Use the `Get-SPEnterpriseSearchMetadataCategory` cmdlet.

The `Get-SPEnterpriseSearchMetadataCategory` cmdlet provides information on the Enterprise Search metadata categories configured for a particular search service application. Therefore, you need to provide a search service application with this cmdlet, as shown in Listing 12.9.

LISTING 12.9 **Displaying the Search Metadata Categories**

```
$ssApp = Get-SPEnterpriseSearchServiceApplication
-Identity 81aedaf7-704c-47f4-8173-5e99a7495ac3
Get-SPEnterpriseSearchMetadataCategory –SearchApplication $ssApp
```

The Enterprise Search Service metadata categories are displayed, as shown in Figure 12.4.



```
PS C:\> Get-SPEnterpriseSearchMetadataCategory -SearchApplication $ssApp

Name             CrawledProperty AutoCreateNewManaged DiscoverNewProp MapToConte
                 Count           Properties           erties          nts
----             --------------- -------------------- --------------- ----------
Basic            55              False                True            True
Business Data    2               False                True            True
Custom Prope...  0               False                False           False
Internal         0               False                False           False
Mail             14              False                True            True
Notes            3               False                True            True
Office           44              False                True            True
People           53              False                True            True
SharePoint       145             False                True            True
Tiff             0               False                True            True
Web              50              False                True            True
XML              0               False                True            True


PS C:\> _
```

FIGURE 12.4 Displaying the Enterprise Search Service application metadata categories

Get a Specific Metadata Category

Scenario/Problem: You need to obtain a metadata category reference for use in other search cmdlets.

Solution: Use the `Get-SPEnterpriseSearchMetadataCategory` cmdlet to assign a variable the results using the `–Identity` and `–SearchApplication` parameters.

Providing the identity of the metadata category with the `Get-SPEnterpriseSearchMetadataCategory` cmdlet retrieves a specific metadata category for the specified search service application. Use this to assign the results to an object for use in other cmdlets, as shown in Listing 12.10.

LISTING 12.10 **Assigning a Variable to a Specific Metadata Category**

```
$ssApp = Get-SPEnterpriseSearchServiceApplication
-Identity 81aedaf7-704c-47f4-8173-5e99a7495ac3
$category = Get-SPEnterpriseSearchMetadataCategory
–SearchApplication $ssApp –Identity "Custom Properties"
```

Create a Custom Metadata Crawled Property

Scenario/Problem: You want to create a metadata crawled property.

Solution: Use the `New-SPEnterpriseSearchMetadataCrawledProperty` cmdlet.

Instead of crawling content to generate a crawled property, you may use this search cmdlet to create a crawled property entry within a search service application. This makes it easier when you are replicating search configurations within various environments (development, staging, production, and so on). Sample command lines are shown in Listing 12.11.

LISTING 12.11 **Creating a New Metadata Crawled Property**

```
$ssApp = Get-SPEnterpriseSearchServiceApplication
-Identity 81aedaf7-704c-47f4-8173-5e99a7495ac3
$category = Get-SPEnterpriseSearchMetadataCategory
–SearchApplication $ssApp –Identity "SharePoint"
New-SPEnterpriseSearchMetadataCrawledProperty -Name "ows_NewField"
–SearchApplication $ssApp –Category $category -VariantType 31
-PropSet 00130329-0-0130-c-131346  -IsNameEnum $false
```

The `VariantType` parameter determines the type of data that is stored within the crawled property. The valid types and associated values are as follows:

- `Text`—31
- `Integer`—20
- `Date and Time`—64
- `Yes/No`—11
- `Binary Data`—12

`PropSet` is the property set associated with the category and crawled properties. The best way to find the correct property set is from investigating the crawled properties within Central Administration (from Manage Service Applications, select Search Service Application, select Managed Properties, and then click on the Crawled Properties link at the top of the screen) or using the `Get-SPEnterpriseSearchMetaDataCrawledProperty` cmdlet (explained in the next section).

The `IsNameEnum` switch parameter is required and determines whether the underlying data is actually an integer that is being converted to a string.

NOTE The execution of the commands will state that the `SearchApplication` parameter is ignored; however, if you do not provide `SearchApplication`, the cmdlet prompts for the entry.

Display Available Crawled Properties

Scenario/Problem: You want to view the information on the available search metadata crawled properties.

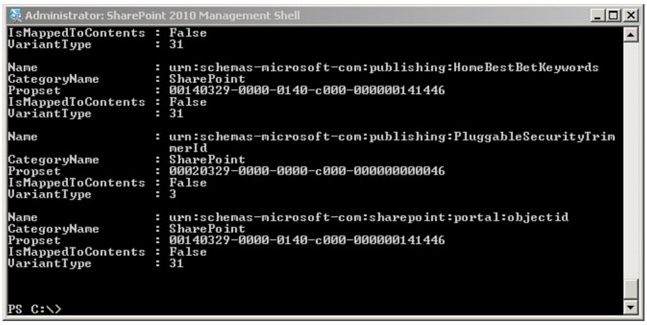
Solution: Use the `Get-SPEnterpriseSearchMetadataCrawledProperty` cmdlet.

The `Get-SPEnterpriseSearchMetaDataCrawledProperty` cmdlet provides information on the Enterprise Search metadata categories configured for a particular search service application. Therefore, you need to provide a search service application with this cmdlet, as shown in Listing 12.12.

LISTING 12.12 **Displaying the Search Metadata Categories**

```
$ssApp = Get-SPEnterpriseSearchServiceApplication
-Identity 81aedaf7-704c-47f4-8173-5e99a7495ac3
Get-SPEnterpriseSearchMetadataCrawledProperty
–SearchApplication $ssApp
```

The Enterprise Search Service metadata crawled properties are displayed, as shown in Figure 12.5.



```
IsMappedToContents : False
VariantType        : 31

Name               : urn:schemas-microsoft-com:publishing:HomeBestBetKeywords
CategoryName       : SharePoint
Propset            : 00140329-0000-0140-c000-000000141446
IsMappedToContents : False
VariantType        : 31

Name               : urn:schemas-microsoft-com:publishing:PluggableSecurityTrim
                     merId
CategoryName       : SharePoint
Propset            : 00020329-0000-0000-c000-000000000046
IsMappedToContents : False
VariantType        : 3

Name               : urn:schemas-microsoft-com:sharepoint:portal:objectid
CategoryName       : SharePoint
Propset            : 00140329-0000-0140-c000-000000141446
IsMappedToContents : False
VariantType        : 31


PS C:\>
```

FIGURE 12.5 Displaying the Enterprise Search Service application metadata crawled properties

TIP Use the `Where-Object` clause to limit the amount of crawled properties displayed. Here is an example:`Get-SPEnterpriseSearchMetadataCrawledProp-erty –SearchApplication $ssApp | Where-Object {$_.CategoryName -eq “SharePoint”}`

Get a Specific Metadata Crawled Property

Scenario/Problem: You need to obtain a metadata crawled property reference for use in other search cmdlets.

Solution: Use the `Get-SPEnterpriseSearchMetadataCrawledProperty` cmdlet to assign a variable the results using the `–Identity` and `–SearchApplication` parameters.

Providing the identity of the metadata crawled property with the `Get-SPEnterpriseSearchMetadataCrawledProperty` cmdlet retrieves a specific metadata crawled property for the specified search service application. Use this to assign the results to an object for use in other cmdlets, as shown in Listing 12.13.

LISTING 12.13 **Assigning a Variable to a Specific Metadata Crawled Property**

```
$ssApp = Get-SPEnterpriseSearchServiceApplication
-Identity 81aedaf7-704c-47f4-8173-5e99a7495ac3
$crawledproperty = Get-SPEnterpriseSearchMetadataCrawledProperty
–SearchApplication $ssApp –Name "ows_NewField"
```

Create a Custom Metadata Managed Property

Scenario/Problem: You want to create a metadata managed property.

Solution: Use the `New-SPEnterpriseSearchMetadataManagedProperty` cmdlet.

Instead of crawling content to generate a managed property, you may use this search cmdlet to create a managed property entry within a search service application. This makes it easier when you are replicating search configurations within various environments (development, staging, production, and so on). Sample command lines are shown in Listing 12.14.

LISTING 12.14 **Creating a New Metadata Managed Property**

```
$ssApp = Get-SPEnterpriseSearchServiceApplication
-Identity 81aedaf7-704c-47f4-8173-5e99a7495ac3
$category = Get-SPEnterpriseSearchMetadataCategory
–SearchApplication $ssApp –Identity "SharePoint"
New-SPEnterpriseSearchMetadataManagedProperty -Name "NewField"
–SearchApplication $ssApp -Type 1
```

The `Type` parameter determines the type of data that is stored within the managed property. The valid types and values are as follows:

- `Text`—1
- `Integer`—2
- `Decimal`—3
- `DateTime`—4
- `YesNo`—5
- `Binary`—6

Display Available Managed Properties

Scenario/Problem: You want to view the information on the available search metadata managed properties.

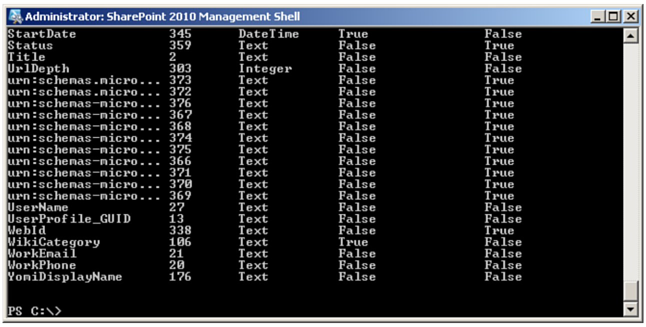
Solution: Use the `Get-SPEnterpriseSearchMetadataManagedProperty` cmdlet.

The `Get-SPEnterpriseSearchMetadataManagedProperty` cmdlet provides information on the Enterprise Search metadata categories configured for a particular search service application. Therefore, you need to provide a search service application with this cmdlet, as shown in Listing 12.15.

LISTING 12.15 **Displaying the Search Metadata Managed Properties**

```
$ssApp = Get-SPEnterpriseSearchServiceApplication
-Identity 81aedaf7-704c-47f4-8173-5e99a7495ac3
Get-SPEnterpriseSearchMetadataManagedProperty
–SearchApplication $ssApp
```

The Enterprise Search Service metadata managed properties are displayed, as shown in Figure 12.6.



Administrator: SharePoint 2010 Management Shell

```
StartDate              345      DateTime    True              False
Status                 359      Text        False             True
Title                  2        Text        False             False
UrlDepth               303      Integer     False             False
urn:schemas.micro...   373      Text        False             True
urn:schemas.micro...   372      Text        False             True
urn:schemas-micro...   376      Text        False             True
urn:schemas-micro...   367      Text        False             True
urn:schemas-micro...   368      Text        False             True
urn:schemas-micro...   374      Text        False             True
urn:schemas-micro...   375      Text        False             True
urn:schemas-micro...   366      Text        False             True
urn:schemas-micro...   371      Text        False             True
urn:schemas-micro...   370      Text        False             True
urn:schemas-micro...   369      Text        False             True
UserName               27       Text        False             False
UserProfile_GUID       13       Text        False             False
WebId                  338      Text        False             True
WikiCategory           106      Text        True              False
WorkEmail              21       Text        False             False
WorkPhone              20       Text        False             False
YomiDisplayName        176      Text        False             False

PS C:\>
```

FIGURE 12.6 Displaying the Enterprise Search Service application metadata managed properties

Get a Specific Metadata Managed Property

Scenario/Problem: You need to obtain a metadata managed property reference for use in other search cmdlets.

Solution: Use the `Get-SPEnterpriseSearchMetadataManagedProperty` cmdlet to assign a variable the results using the `–Identity` and `–SearchApplication` parameters.

Providing the identity of the metadata managed property with the `Get-SPEnterpriseSearchMetadataManagedProperty` cmdlet retrieves a specific metadata managed property for the specified search service application. Use this to assign the results to an object for use in other cmdlets, as shown in Listing 12.16.

LISTING 12.16 **Assigning a Variable to a Specific Metadata Managed Property**

```
$ssApp = Get-SPEnterpriseSearchServiceApplication
-Identity 81aedaf7-704c-47f4-8173-5e99a7495ac3
$managedproperty = Get-SPEnterpriseSearchMetadataManagedProperty
–SearchApplication $ssApp –Identity NewField
```

Create a Metadata Mapping

Scenario/Problem: You need to map a crawled property to a managed property.

Solution: Use the `New-SPEnterpriseSearchMetadataMapping` cmdlet.

In order to be able to use new search fields within SharePoint, you must map the crawled properties to managed properties. You may use the variables generated in the previous sections to map the two together using the `New-SPEnterpriseSearchMetadataMapping` cmdlet. Sample command lines are shown in Listing 12.17.

LISTING 12.17 **Creating a New Metadata Mapping**

```
$ssApp = Get-SPEnterpriseSearchServiceApplication
-Identity 81aedaf7-704c-47f4-8173-5e99a7495ac3
$crawledproperty = Get-SPEnterpriseSearchMetadataCrawledProperty
–SearchApplication $ssApp –Name "ows_NewField"
$managedproperty = Get-SPEnterpriseSearchMetadataManagedProperty
–SearchApplication $ssApp –Identity NewField
New-SPEnterpriseSearchMetadataMapping -SearchApplication $ssApp
-ManagedProperty $managedproperty -CrawledProperty $crawledproperty
```

TIP Change the `Get` statements to `New` statements such that you may generate the crawled and managed properties on the fly while performing the mapping.

CHAPTER 13

SharePoint Foundation Search

IN THIS CHAPTER

- Display the Search Service Information
- Display Search Service Instances
- Get a Specific Search Service Instance
- Configure the Crawl Account
- Configure the Maximum Backup Duration
- Configure the Search Service Performance Level

This chapter explores the administrative aspects of SharePoint Foundation Search through the use of SharePoint PowerShell cmdlets.

Display the Search Service Information

Scenario/Problem: You want to view the information on the search service.

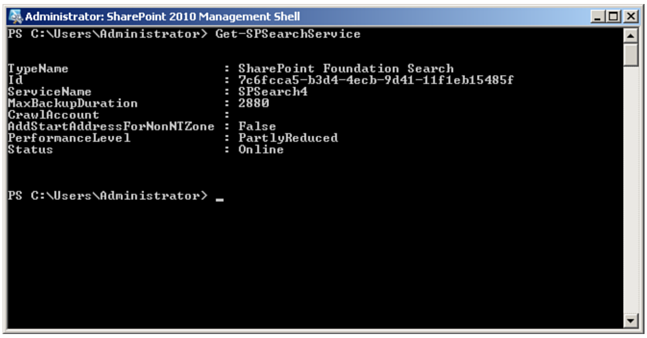
Solution: Use the `Get-SPSearchService` cmdlet.

The `Get-SPSearchService` cmdlet provides information on the SharePoint Foundation Search service that is running on the SharePoint farm. No parameters are necessary, as shown in Listing 13.1

LISTING 13.1 **Displaying the Search Service Information**

```
Get-SPSearchService
```

The search service information is displayed, as shown in Figure 13.1.



```
PS C:\Users\Administrator> Get-SPSearchService

TypeName                      : SharePoint Foundation Search
Id                            : 7c6fcca5-b3d4-4ecb-9d41-11f1eb15485f
ServiceName                   : SPSearch4
MaxBackupDuration             : 2880
CrawlAccount                  :
AddStartAddressForNonNTZone : False
PerformanceLevel              : PartlyReduced
Status                        : Online


PS C:\Users\Administrator> _
```

FIGURE 13.1 Displaying the search service information

Display Search Service Instances

Scenario/Problem: You want to view the information on the search service instances.

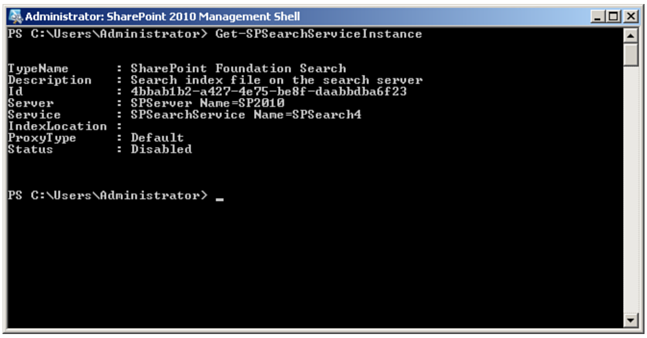
Solution: Use the `Get-SPSearchServiceInstance` cmdlet.

The `Get-SPSearchServiceInstance` cmdlet provides information on the SharePoint Foundation Search service instances that are mapped to the search service. No parameters are necessary, as shown in Listing 13.2.

LISTING 13.2 **Displaying the Search Service Instances**

```
Get-SPSearchServiceInstance
```

The search service instance information is displayed, as shown in Figure 13.2.



Administrator: SharePoint 2010 Management Shell

```
PS C:\Users\Administrator> Get-SPSearchServiceInstance


TypeName      : SharePoint Foundation Search
Description   : Search index file on the search server
Id            : 4bbab1b2-a427-4e75-be8f-daabbdba6f23
Server        : SPServer Name=SP2010
Service       : SPSearchService Name=SPSearch4
IndexLocation :
ProxyType     : Default
Status        : Disabled


PS C:\Users\Administrator> _
```

FIGURE 13.2 Displaying the search service instance information

Get a Specific Search Service Instance

Scenario/Problem: You need to obtain a search service instance reference.

Solution: Use the `Get-SPSearchServiceInstance` cmdlet to assign a variable to the results using the `–Identity` parameter.

Providing the identity of the search service instance with the `Get-SPSearchServiceInstance` cmdlet retrieves a specific search service instance. Use this to assign the results to an object for use in other cmdlets, as shown in Listing 13.3.

LISTING 13.3 **Assigning a Variable to a Specific Search Service Instance**

```
$ssInstance = Get-SPSearchServiceInstance
-Identity 4bbab1b2-a427-4e75-be8f-daabbdba6f23
```

Configure the Crawl Account

Scenario/Problem: You need to configure the crawl account for the search service.

Solution: Use the `Set-SPSearchService` cmdlet with the crawl account parameters.

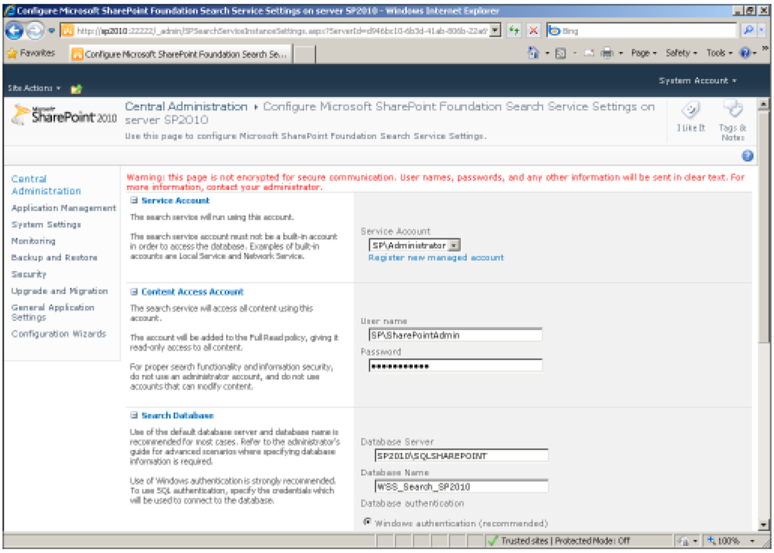
You may set the crawl account information for the search service by using the `CrawlAccount` and `CrawlPassword` parameters with the `Set-SPSearchService` cmdlet, as shown in Listing 13.4.

LISTING 13.4 **Configuring the Search Service Crawl Account**

```
Set-SPSearchService –Identity 7c6fcca5-b3d4-4ecb-9d41-11f1eb15485f
-CrawlAccount "SP\SharePointCrawl"
-CrawlPassword (ConvertTo-SecureString -asplaintext -force
 -string "SharePoint1")
```

The password needs to be a secure string; therefore, the `ConvertTo-SecureString` cmdlet is used to assist. The Crawl Account setting appears as “Content Access Account” when you’re managing the SharePoint Foundation Search service on the server, as shown in Figure 13.3.

TIP To get to these properties, from Central Admin, click on the Manage Service Applications link under Application Management. Locate the SharePoint Foundation Search Service entry and highlight the connection (the indented instance). Click on the Properties button from the top ribbon bar.


Configure Microsoft SharePoint Foundation Search Service Settings on server SP2010 - Windows Internet Explorer
Favorites
Configure Microsoft SharePoint Foundation Search Se...
Page
Safety
Tools
Site Actions
System Account
I Like It
Tags & Notes
Central Administration
Application Management
System Settings
Monitoring
Backup and Restore
Security
Upgrade and Migration
General Application Settings
Configuration Wizards
Trusted sites | Protected Mode: Off
100%

Central Administration ▸ Configure Microsoft SharePoint Foundation Search Service Settings on server SP2010

Use this page to configure Microsoft SharePoint Foundation Search Service Settings.

Warning: this page is not encrypted for secure communication. User names, passwords, and any other information will be sent in clear text. For more information, contact your administrator.

Service Account

The search service will run using this account.

The search service account must not be a built-in account in order to access the database. Examples of built-in accounts are Local Service and Network Service.

Service Account: SP\Administrator

Register new managed account

Content Access Account

The search service will access all content using this account.

The account will be added to the Full Read policy, giving it read-only access to all content.

For proper search functionality and information security, do not use an administrator account, and do not use accounts that can modify content.

User name: SP\SharePointAdmin

Password: ●●●●●●●●●●●●

Search Database

Use of the default database server and database name is recommended for most cases. Refer to the administrator's guide for advanced scenarios where specifying database information is required.

Use of Windows authentication is strongly recommended. To use SQL authentication, specify the credentials which will be used to connect to the database.

Database Server: SP2010\SQLSHAREPOINT

Database Name: WSS_Search_SP2010

Database authentication

● Windows authentication (recommended)

FIGURE 13.3 Managing the SharePoint Foundation Search service via Central Admin

Configure the Maximum Backup Duration

Scenario/Problem: You need to change the maximum backup duration for the search service.

Solution: Use the `Set-SPSearchService` cmdlet with the `MaxBackupDuration` parameter.

You may set the maximum backup duration using the `MaxBackupDuration` parameter with the `Set-SPSearchService` cmdlet, as shown in Listing 13.5.

LISTING 13.5 **Configuring the Maximum Backup Duration**

```
Set-SPSearchService
–Identity 7c6fcca5-b3d4-4ecb-9d41-11f1eb15485f
–MaxBackupDuration 1440
```

The `MaxBackupDuration` is the amount of minutes a backup of the search service can take before the search service resumes normal operation. The default is 2,880 minutes, which equates to 48 hours.

Configure the Search Service Performance Level

Scenario/Problem: You need to change the performance level for the search service.

Solution: Use the `Set-SPSearchService` cmdlet with the `PerformanceLevel` parameter.

You may specify the performance level setting using the `PerformanceLevel` parameter with the `Set-SPSearchService` cmdlet, as shown in Listing 13.6.

LISTING 13.6 **Configuring the Search Service Performance Level**

```
Set-SPSearchService
–Identity 7c6fcca5-b3d4-4ecb-9d41-11f1eb15485f
–PerformanceLevel Reduced
```

The `PerformanceLevel` parameter determines how many threads can be utilized by the search service. The valid values are as follows:

- `Reduced`
- `PartlyReduced`
- `Maximum`

The default setting is `PartlyReduced`.

CHAPTER 14

Profile Service

IN THIS CHAPTER

- Get the Identity of the Profile Service Application
- Get a Specific Profile Service Application Instance
- Configure Settings on the Profile Service Application
- Remove Old Comments
- Remove Old Ratings
- Remove Old Tags
- Update the Profile Photo Store
- Use Ancillary Cmdlets for the Profile Service

This chapter explores the administrative aspects of the User Profile Service through the use of SharePoint PowerShell cmdlets.

Get the Identity of the Profile Service Application

Scenario/Problem: You need to find the identity of the User Profile Service application.

Solution: Use the `Get-SPServiceApplication` cmdlet.

In order to get a specific instance of a metadata service application, you need the identity of the service application. The `Set-SPProfileServiceApplication` cmdlet requires an `Identity` parameter; however, there is no `Get-SPProfileServiceApplication` cmdlet. Therefore, you need to use the `Get-SPServiceApplication` cmdlet to display all service applications and find the User Profile Service. The command line without any parameters is shown in Listing 14.1.

LISTING 14.1 **Displaying the List of Service Applications**

```
Get-SPServiceApplication
```

Get a Specific Profile Service Application Instance

Scenario/Problem: You need to obtain a User Profile Service application instance reference.

Solution: Use the `Get- SPServiceApplication` cmdlet to assign a variable the results using the `–Identity` parameter.

Providing the identity of the User Profile Service application instance with the `Get-SPServiceApplication` cmdlet retrieves a specific secure store service instance. Use this to assign the results to an object for use in other cmdlets, as shown in Listing 14.2.

LISTING 14.2 **Assigning a Variable to a Specific Metadata Service Application**

```
$profileSA = Get-SPServiceApplication
-Identity 9b6235e4-f306-4ccc-8192-0c83cb679476
```

There is no cmdlet to get an object type related to a User Profile Service application. Any cmdlet using the service application identity uses the `SPServiceApplicationPipeBind` type, so a generic service application object needs to be utilized.

Configure Settings on the User Profile Service Application

Scenario/Problem: You want to configure settings on the User Profile Service.

Solution: Use the `Set-SPProfileServiceApplication` cmdlet with an `Identity` variable.

You may use the `Set-SPProfileServiceApplication` cmdlet to configure settings on the User Profile Service application using an `Identity` variable, as shown in Listing 14.3.

LISTING 14.3 **Configuring the Profile Service Settings**

```
$profileSA = Get-SPServiceApplication
-Identity 9b6235e4-f306-4ccc-8192-0c83cb679476
Set-SPProfileServiceApplication –Identity $profileSA
-MySiteHostLocation http://sp2010:80/my/
-MySiteManagedPath my/personal
```

The `Set-SPProfileServiceApplication` cmdlet has two parameter sets that are very similar. In addition to the `MySite` parameters, you may also configure various DB credentials. Run `Set-SPProfileServiceApplication -?` to review additional parameters.

Remove Old Comments

Scenario/Problem: You want to remove comments from the User Profile Service application.

Solution: Use the `Remove-SPSocialItemByDate` cmdlet with the `RemoveComments` parameter.

The `Remove-SPSocialItemByDate` cmdlet allows you to remove comments, tags, or ratings prior to a particular required end date. A sample command line is shown in Listing 14.4.

LISTING 14.4 **Removing Old Comments from the Profile Service**

```
$profileProxy = Get-SPServiceApplicationProxy
-Identity 150232e4-a3d2-4915-adb1-155ea77a6dfb
Remove-SPSocialItemByDate
–ProfileServiceApplicationProxy $profileProxy –RemoveComments:$true
–EndDate 12/1/2010 –Confirm:$false
```

The `RemoveComments` parameter is a Boolean parameter, although it looks like a switch parameter. Therefore, either `$true` or `$false` needs to be provided as the Boolean value. The service application proxy for the User Profile Service is needed and can be obtained by reviewing the output from `Get-SPServiceApplicationProxy`.

Remove Old Ratings

Scenario/Problem: You want to remove comments from the User Profile Service application.

Solution: Use the `Remove-SPSocialItemByDate` cmdlet with the `RemoveRatings` parameter.

The `Remove-SPSocialItemByDate` cmdlet allows you to remove comments, tags, or ratings prior to a particular required end date. A sample command line is shown in Listing 14.5.

LISTING 14.5 **Removing Old Ratings from the Profile Service**

```
$profileProxy = Get-SPServiceApplicationProxy
-Identity 150232e4-a3d2-4915-adb1-155ea77a6dfb
Remove-SPSocialItemByDate –ProfileServiceApplicationProxy
$profileProxy –RemoveRatings:$true –EndDate 12/1/2010 –Confirm:$false
```

The `RemoveRatings` parameter is a Boolean parameter, although it looks like a switch parameter. Therefore, either `$true` or `$false` needs to be provided as the Boolean value. The service application proxy for the User Profile Service is needed and can be obtained by reviewing the output from `Get-SPServiceApplicationProxy`.

Remove Old Tags

Scenario/Problem: You want to remove comments from the User Profile Service application.

Solution: Use the `Remove-SPSocialItemByDate` cmdlet with the `RemoveTags` parameter.

The `Remove-SPSocialItemByDate` cmdlet allows you to remove comments, tags, or ratings prior to a particular required end date. A sample command line is shown in Listing 14.6.

LISTING 14.6 **Removing Old Tags from the Profile Service**

```
$profileProxy = Get-SPServiceApplicationProxy
-Identity 150232e4-a3d2-4915-adb1-155ea77a6dfb
Remove-SPSocialItemByDate –ProfileServiceApplicationProxy
$profileProxy –RemoveTags:$true –EndDate 12/1/2010 –Confirm:$false
```

The `RemoveTags` parameter is a Boolean parameter, although it looks like a switch parameter. Therefore, either `$true` or `$false` needs to be provided as the Boolean value. The service application proxy for the User Profile Service is needed and can be obtained by reviewing the output from `Get-SPServiceApplicationProxy`.

Update the Profile Photo Store

Scenario/Problem: You upgraded to SharePoint 2010 and need to update the Profile Photo Store.

Solution: Use the `Update-SPProfilePhotoStore` cmdlet with the `MySiteHostLocation` parameter.

After you upgrade from MOSS 2007 to SharePoint 2010, you need to ensure the profile photo store is compatible with SharePoint 2010. Using the `Update-SPProfilePhotoStore` cmdlet, as shown in Listing 14.7, will ensure that the photo store is updated accordingly.

LISTING 14.7 **Updating the Profile Photo Store**

```
Update-SPProfilePhotoStore –MySiteHostLocation http://mysites
```

This is only needed if you performed an in-place or hybrid upgrade.

Use Ancillary Cmdlets for the User Profile Service

Scenario/Problem: You need to use more advanced cmdlets for the User Profile Service.

Solution: Use the cmdlets listed in this section.

This chapter covers the basic or more common cmdlets for the User Profile Service. For more advance scripting, the following cmdlet nouns are also available:

- `SPSiteSubscriptionProfileConfig`
- `SPProfileServiceApplicationSecurity`
- `SPProfileServiceApplicationProxy`

CHAPTER 15

Business Data Connectivity Service

IN THIS CHAPTER

- Get the Identity of the BCS Service Application
- Get a Specific BCS Service Application Instance
- Configure the BCS Database
- Get a BCS Metadata Object
- Import a BCS Model
- Export a BCS Model

This chapter explores the administrative aspects of the Business Data Connectivity Service (BCS) through the use of SharePoint PowerShell cmdlets. BCS is the replacement of the Business Data Catalog (BDC) from MOSS 2007, although there are remnants of the reference to the Business Data Catalog.

Get the Identity of the BCS Service Application

Scenario/Problem: You need to find the identity of the BCS service application.

Solution: Use the `Get-SPServiceApplication` cmdlet.

In order to get a specific instance of a metadata service application, you need the identity of the service application. The `Set-SPBusinessDataCatalogService Application` cmdlet requires an `Identity` parameter; however, there is no `Get-SPBusinessDataCatalogServiceApplication` cmdlet. Therefore, you need to use the `Get-SPServiceApplication` cmdlet to display all service applications and find the Business Data Connectivity Service. The command line without any parameters is shown in Listing 15.1.

LISTING 15.1 **Displaying the List of Service Applications**

```
Get-SPServiceApplication
```

Get a Specific BCS Service Application Instance

Scenario/Problem: You need to obtain a Secure Store Service application instance reference.

Solution: Use the `Get- SPServiceApplication` cmdlet to assign a variable the results using the `–Identity` parameter.

Providing the identity of the Business Data Connectivity Service application instance with the `Get- SPServiceApplication` cmdlet retrieves a specific Business Data Connectivity Service instance. Use this to assign the results to an object for use in other cmdlets, as shown in Listing 15.2.

LISTING 15.2 **Assigning a Variable to a Business Data Service Application**

```
$bcs = Get-SPServiceApplication
-Identity b7b3652e-6f51-4161-935a-0631a1100ecf
```

There is no cmdlet to get an object type related to a Business Data Connectivity Service application. Any cmdlet using the service application identity uses the `SPServiceApplicationPipeBind` type, so a generic service application object needs to be utilized.

Configure the BCS Database

Scenario/Problem: You want to use a new database for the BCS service application.

Solution: Use the `Set-SPBusinessDataCatalogServiceApplication` cmdlet with the `DatabaseName` parameter.

When the Business Data Connectivity Service application is created, by default the database name is Bdc_Service_DB_<guid>. You may easily change this to a new database using the `DatabaseName` parameter with the `Set-SPBusinessDataCatalogServiceApplication` cmdlet, as shown in Listing 15.3.

LISTING 15.3 **Configuring a New Database for BCS**

```
$bcs = Get-SPServiceApplication
-Identity b7b3652e-6f51-4161-935a-0631a1100ecf
Set-SPBusinessDataCatalogServiceApplication –Identity $bcs
–DatabaseName BCS_DB
```

If the database does not exist, a new database is created. Simply drop the old one with the GUID in the name and make your DBAs happy. By default, the database server is assumed to be the SQL Server instance that SharePoint is using. You may specify the database server by using the `–DatabaseServer` parameter. You only need to specify the database credentials if SQL Authentication is being used.

Get a BCS Metadata Object

Scenario/Problem: You need to obtain a BCS metatdata object reference.

Solution: Use the `Get-SPBusinessDataCatalogMetadataObject` cmdlet.

In order to perform import and export operations, you need the related metadata object that stores or will store the BCS entity. Use the `Get-SPBusinessDataCatalog MetadataObject` cmdlet to set a variable to the metadata object, as shown in Listing 15.4.

LISTING 15.4 **Getting a BCS Metadata Object**

```
$bcsMeta = Get-SPBusinessDataCatalogMetadataObject –ServiceContext
http://sp2010 –BDCObjectType Catalog –Name ApplicationRegistry
```

The `BDCObjectType` may be one of the following values:

- `Catalog`
- `Model`
- `LobSystem`
- `LobSystemInstance`
- `Entity`

The `Name` parameter may be required based on the `BDCObjectType` that is referenced.

Import a BCS Model

Scenario/Problem: You need to import a BCS model.

Solution: Use the `Import-SPBusinessDataCatalogModel` cmdlet.

The `Import-SPBusinessDataCatalogModel` cmdlet allows you to import a model file from a local or shared drive. Use a BCS metadata object variable as the `Identity`, as shown in Listing 15.5.

LISTING 15.5 **Importing a BCS Model**

```
Import-SPBusinessDataCatalogModel –Identity $bcsMeta
–Path E:\BCS_Model.bdcm
```

Several optional switch parameters may be used:

- `Force`—Forces an overwrite if the model exists.
- `ModelsIncluded`—Models are included in the imported XML file.
- `PermissionsIncluded`—The permissions to the model are included.
- `PropertiesIncluded`—Specifies that BCS properties are included.

Export a BCS Model

Scenario/Problem: You need to export a BCS model.

Solution: Use the `Export-SPBusinessDataCatalogModel` cmdlet.

The `Export-SPBusinessDataCatalogModel` cmdlet allows you to export a model file to a local or shared drive. Use a BCS metadata object variable as the `Identity`, as shown in Listing 15.6.

LISTING 15.6 **Exporting a BCS Model**

```
$bcsMeta = Get-SPBusinessDataCatalogMetadataObject –ServiceContext
http://sp2010 –BDCObjectType Catalog –Name ApplicationRegistry
Export-SPBusinessDataCatalogModel –Identity $bcsMeta
–Path E:\BCS_Model.bdcm
```

Several optional switch parameters may be used:

- `Force`—Forces the overwriting of a file if the output file exists.
- `ModelsIncluded`—Models are included in the exported XML file.
- `PermissionsIncluded`—The permissions to the model are included in the export.
- `PropertiesIncluded`—Specifies that BCS properties are included in the export.
- `ProxiesIncluded`—Specifies that proxy information for the BCS service application is included in the export.

CHAPTER 16

Secure Store Service

IN THIS CHAPTER

- Get the Identity of the Secure Store Service Application
- Get a Specific Secure Store Service Application Instance
- Enable Auditing for the Secure Store Service
- Configure the Secure Store Database
- Generate a New Master Key
- Refresh the Encryption Key
- Create Application Fields
- Create a Target Application
- Create a New Application Entry

This chapter explores the administrative aspects of the Secure Store Service through the use of SharePoint PowerShell cmdlets. Only a few cmdlets are used for the actual service application (`SPSecureStoreServiceApplication`), while most of the cmdlets are for a specific application (`SPSecureStoreApplication`) being secured by the Secure Store Service.

Get the Identity of the Secure Store Service Application

Scenario/Problem: You need to find the identity of the Secure Store Service application.

Solution: Use the `Get-SPServiceApplication` cmdlet.

In order to get a specific instance of a Metadata Service application, you need the identity of the service application. The `Set-SPSecureStoreServiceApplication` cmdlet requires an `Identity` parameter; however, there is no `Get-SPSecureStoreServiceApplication` cmdlet. Therefore, you need to use the `Get-SPServiceApplication` cmdlet to display all service applications and find the Secure Store Service. The command line without any parameters is shown in Listing 16.1.

LISTING 16.1 **Displaying the List of Service Applications**

```
Get-SPServiceApplication
```

Get a Specific Secure Store Service Application Instance

Scenario/Problem: You need to obtain a Secure Store Service application instance reference.

Solution: Use the `Get-SPServiceApplication` cmdlet to assign a variable the results using the `–Identity` parameter.

Providing the identity of the Secure Store Service application instance with the `Get-SPServiceApplication` cmdlet retrieves a specific Secure Store Service instance. Use this to assign the results to an object for use in other cmdlets, as shown in Listing 16.2.

LISTING 16.2 **Assigning a Variable to a Specific Metadata Service Application**

```
$sss = Get-SPServiceApplication
-Identity bc4399ed-a2e0-4397-bf07-cd3d207e630e
```

There is no cmdlet to get an object type related to a Secure Store Service application. Any cmdlet using the service application identity uses the `SPServiceApplicationPipeBind` type, so a generic service application object needs to be utilized.

Enable Auditing for the Secure Store Service

Scenario/Problem: You want to enable auditing for the Secure Store Service.

Solution: Use the `Set-SPSecureStoreServiceApplication` cmdlet with the audit parameters.

Two audit parameters may be used with the `Set-SPSecureStoreServiceApplication` cmdlet. The first parameter, `AuditingEnabled`, is a switch parameter that determines if auditing is enabled. The second parameter, `AuditlogMaxSize`, determines how many days to retain the logs. Sample command lines are shown in Listing 16.3.

LISTING 16.3 **Configuring the Secure Store Service Auditing**

```
$sss = Get-SPServiceApplication
-Identity bc4399ed-a2e0-4397-bf07-cd3d207e630e
Set-SPSecureStoreServiceApplication
–Identity $sss –AuditingEnabled –AuditlogMaxSize 15
```

TIP To disable auditing, use `–AuditingEnabled:$false`.

Auditing is disabled by default, and the initial retention value is 30 days.

Configure the Secure Store Database

Scenario/Problem: You want to use a new database for the Secure Store Service application.

Solution: Use the `Set-SPSecureStoreServiceApplication` cmdlet with the `DatabaseName` parameter.

When the Secure Store Service application is created, by default the database name is Secure_Store_Service_<guid>. You may easily change this to a new database using the `DatabaseName` parameter with the `Set-SPSecureStoreServiceApplication` cmdlet, as shown in Listing 16.4.

LISTING 16.4 **Configuring a New Database for the Secure Store**

```
$sss = Get-SPServiceApplication
-Identity bc4399ed-a2e0-4397-bf07-cd3d207e630e
Set-SPSecureStoreServiceApplication –Identity $sss
–DatabaseName SecureStoreDB
```

If the database does not exist, a new database is created. Simply drop the old one with the GUID in the name and make your DBAs happy. By default, the database server is assumed to be the SQL Server instance that SharePoint is using. You may specify the database server by using the `–DatabaseServer` parameter. You only need to specify the database credentials if SQL Authentication is being used.

NOTE At the time of this writing, a new database is only being created when a new database name is specified from within Central Administration. Although the PowerShell cmdlet changes the name, the new database does not appear in SQL Server. However, reviewing the properties of the Secure Store Service application and clicking OK does indeed create the new database.

Generate a New Master Key

Scenario/Problem: You need to create or change the master key for the Secure Store Service.

Solution: Use the `Update-SPSecureStoreMasterKey` cmdlet.

The Secure Store Service needs a master key before target applications can be created. To create or change the master key, use the `Update-SPSecureStoreMasterKey` cmdlet, as shown in Listing 16.5.

LISTING 16.5 **Generating a New Master Key**

```
Update-SPSecureStoreMasterKey
–ServiceApplicationProxy da9247cb-9055-4024-817c-714ef98656e1
–PassPhrase SharePoint1
```

This cmdlet requires the service application proxy GUID for the Secure Store Service. To find the GUID, use the `Get-SPServiceApplicationProxy` cmdlet and locate the Secure Store Service entry. The display name cannot be used here. The passphrase is a plain-text entry and does not need to be encrypted.

Refresh the Encryption Key

Scenario/Problem: You need to create/propagate the master key to all application servers.

Solution: Use the `Update-SPSecureStoreApplicationServerKey` cmdlet.

After a master key (encryption key) is created or updated, all application servers need to be aware of the key. You may propagate the encryption key to all application servers by using the `Update-SPSecureStoreApplicationServerKey` cmdlet, as shown in Listing 16.6.

LISTING 16.6 **Propagating the Encryption Key to All Application Servers**

```
Update-SPSecureStoreApplicationServerKey –ServiceApplicationProxy
da9247cb-9055-4024-817c-714ef98656e1 –PassPhrase SharePoint1
```

This cmdlet requires the service application proxy GUID for the Secure Store Service. To find the GUID, use the `Get-SPServiceApplicationProxy` cmdlet and locate the Secure Store Service entry. The display name cannot be used here. The passphrase is a plain-text entry and does not need to be encrypted; however, it must match the passphrase used in the creation of the master key.

Create Application Fields

Scenario/Problem: You need to create field variables for use in creating a new application in the Secure Store Service.

Solution: Use the `New-SPSecureStoreApplicationField` cmdlet.

When you are creating a new application for the Secure Store Service, several application fields are needed. These application fields can be created using the `New-SPSecureStoreApplicationField` cmdlet. The types of fields are as follows:

- `Username`
- `Password`

- `WindowsUsername`
- `WindowsPassword`
- `Key`
- `Pin`
- `Generic`

Typically you are using the Secure Store Service to provide Single Sign-On using a dedicated service account that has access to the application. Therefore, the most common field types are `WindowsUsername` and `WindowsPassword`. Use variables to create these fields, as shown in Listing 16.7, such that they may be used when you are creating a new application entry.

LISTING 16.7 **Creating Application Fields**

```
$windowsUser = New-SPSecureStoreApplicationField
-Name "SP\AppSvcAccount" -Type WindowsUserName -Masked:$false
$windowsPass = New-SPSecureStoreApplicationField -Name "password"
-Type WindowsPassword -Masked:$true
```

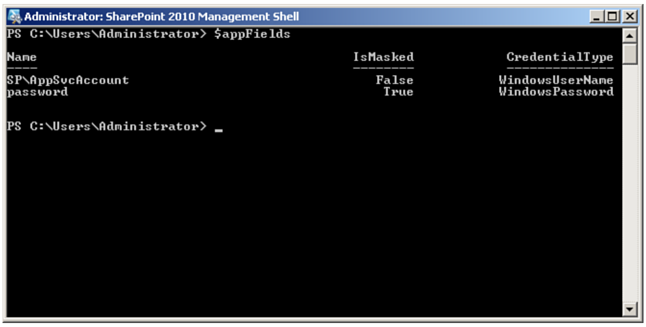
The `Masked` switch parameter is actually required and therefore must have a `$true` or `$false` value. This determines if the field text entry should be masked, as is the case with passwords.

When you are using these fields within other cmdlets, they need to be combined as shown in Listing 16.8.

LISTING 16.8 **Combining Application Fields**

```
$appFields = $windowsUser, $windowsPass
```

Combining the fields produces a fields array, as shown in Figure 16.1. This is required by other cmdlets such as `New-SPSecureStoreApplication`.



```
PS C:\Users\Administrator> $appFields

Name                                    IsMasked          CredentialType
----                                    --------          --------------
SP\AppSvcAccount                           False          WindowsUserName
password                                    True          WindowsPassword

PS C:\Users\Administrator> _
```

FIGURE 16.1 Reviewing the contents of a fields array

Create a Target Application

Scenario/Problem: You need to create a new application entry and need a target application variable.

Solution: Use the `New-SPSecureStoreTargetApplication` cmdlet.

When you are creating a new application for the Secure Store Service, the target application needs to be specified. The target application object can be created using the `New-SPSecureStoreTargetApplication` cmdlet. The type of application can be configured using the `ApplicationType` parameter and one of the following values:

- `Individual`
- `Group`
- `IndividualWithTicketing`
- `GroupWithTicketing`
- `RestrictedIndividual`
- `RestrictedGroup`

Typically you are using the Secure Store Service to provide Single Sign-On using a dedicated service account that has access to the application. Therefore, the most common application type is `Group`. Use a variable to create the target application object, as shown in Listing 16.9, such that the object may be used when you are creating a new application entry.

LISTING 16.9 **Creating the Target Application**

```
$targetApp = New-SPSecureStoreTargetApplication
–Name "SSOApplication" –FriendlyName "SSO Application"
–ApplicationType Group
```

The `Name` entry becomes the application ID for the Secure Store Service application and therefore it is recommended that you provide a name without spaces. Use the `FriendlyName` parameter to provide a proper display name.

Create a New Application Entry

Scenario/Problem: You need to create a new application entry to the Secure Store Service.

Solution: Use the `New-SPSecureStoreApplication` cmdlet.

Creating a new application entry essentially combines several of the previous sections into one script, such that the `New-SPSecureStoreApplication` cmdlet has the necessary values. This involves the target application and application field objects, as shown in Listing 16.10.

LISTING 16.10 **Creating the Secure Store Application Entry**

```
$windowsUser = New-SPSecureStoreApplicationField
–Name "SP\AppSvcAccount" –Type WindowsUserName –Masked:$false
$windowsPass = New-SPSecureStoreApplicationField –Name "password"
–Type WindowsPassword –Masked:$true
$appFields = $windowsUser, $windowsPass
$targetApp = New-SPSecureStoreTargetApplication
–Name "SSOApplication" –FriendlyName "SSO Application"
–ApplicationType Group
New-SPSecureStoreApplication –ServiceContext http://sp2010
–TargetApplication $targetApp –Fields $appFields
```

NOTE Even though the `Administrator` parameter is optional, it may be required depending on the target application configuration.

CHAPTER 17

Metadata Service

IN THIS CHAPTER

- Get the Identity of the Metadata Service Application
- Get a Specific Metadata Service Application Instance
- Configure the Metadata Service Accounts
- Configure the Term Store Database
- Configure the Content Type Hub
- Get the Identity of the Metadata Service Proxy
- Get a Specific Metadata Service Application Proxy Instance
- Configure the Metadata Service Connection Options

This chapter explores the administrative aspects of the Managed Metadata Service through the use of SharePoint PowerShell cmdlets.

Get the Identity of the Metadata Service Application

Scenario/Problem: You need to find the identity of the Metadata Service application.

Solution: Use the `Get-SPServiceApplication` cmdlet.

In order to get a specific instance of a Metadata Service application, you need the identity of the service application. The `Get-SPMetadataServiceApplication` cmdlet requires an `Identity` parameter and therefore does not display all instances. Therefore, you need to use the `Get-SPServiceApplication` cmdlet to display all service applications and find the Managed Metadata Service. The command line without any parameters is shown in Listing 17.1.

LISTING 17.1 **Displaying the List of Service Applications**

```
Get-SPServiceApplication
```

Get a Specific Metadata Service Application Instance

Scenario/Problem: You need to obtain a Metadata Service application instance reference.

Solution: Use the `Get- SPMetadataServiceApplication` cmdlet to assign a variable the results using the `–Identity` parameter.

Providing the identity of the Metadata Service application instance with the `Get-SPMetadataServiceApplication` cmdlet retrieves a specific Metadata Service instance. Use this to assign the results to an object for use in other cmdlets, as shown in Listing 17.2.

LISTING 17.2 **Assigning a Variable to a Specific Metadata Service Application**

```
$metadataApp = Get-SPMetadataServiceApplication
-Identity f3e6d88a-e0a7-455c-bbef-b4942e7607b5
```

Instead of the GUID, you may use the display name. The default display name for the Managed Metadata Service application is “Managed Metadata Service.” Because this is a long name, when the service applications are displayed, the full text is cut off.

Configure the Metadata Service Accounts

Scenario/Problem: You need to configure the service accounts for the Metadata Service application.

Solution: Use the `Set-SPMetadataServiceApplication` cmdlet with the `account` parameters.

There are four service account setting parameters for the metadata service application:

- `AdministratorAccount`
- `FullAccessAccount`
- `RestrictedAccount`
- `ReadAccessAccount`

When one of these parameters is set, the others are removed. Therefore, it is recommended to set all account parameters at the same time, as shown in Listing 17.3.

LISTING 17.3 **Configuring the Metadata Service Accounts**

```
Set-SPMetadataServiceApplication –Identity "Managed Metadata Service"
–AdministratorAccount "SP\Administrator"
-FullAccessAccount "SP\MDSFullAccess"
–RestrictedAccount "SP\MDSRestricted"
–ReadAccessAccount "SP\ReadAccess1,SP\ReadAccess2"
```

You may specify multiple accounts by providing a comma-separated list of the accounts. All accounts specified must exist in the domain.

Configure the Term Store Database

Scenario/Problem: You want to use a new database for the Metadata Service application.

Solution: Use the `Set-SPMetadataServiceApplication` cmdlet with the `DatabaseName` parameter.

When the Metadata Service application is created, by default the database name is Managed Metadata Service_<guid>. You may easily change this to a new database using the `DatabaseName` parameter with the `Set-SPMetadataServiceApplication` cmdlet, as shown in Listing 17.4.

LISTING 17.4 **Configuring the Term Store Database**

```
Set-SPMetadataServiceApplication –Identity "Managed Metadata Service"
 –DatabaseName TermStoreDB
```

If the database does not exist, a new database is created. Simply drop the old one with the GUID in the name and make your DBAs happy. By default the database server is assumed to be the SQL Server instance that SharePoint is using. You may specify the database server by using the `–DatabaseServer` parameter.

Configure the Content Type Hub

Scenario/Problem: You need to set or change the content type hub that should be used by the Metadata Service application.

Solution: Use the `Set-SPMetadataServiceApplication` cmdlet with the `HubUri` parameter.

The content type hub is the site collection you wish to use as your main content type gallery. The Metadata Service application uses the specified content type hub to consume the content types that exist within the site collection. You may set the content type hub using the `HubUri` parameter with the `Set-SPMetadataServiceApplication` cmdlet, as shown in Listing 17.5.

LISTING 17.5 **Configuring the Content Type Hub**

```
Set-SPMetadataServiceApplication –Identity "Managed Metadata Service"
 –HubUri "http://sp2010"
```

When content type hubs are changed, any content type packages are unpublished from the old location. Use the `DoNotUnpublishAllPackages` switch parameter to specify that the content type packages should not be unpublished when the content type hub setting is modified.

Get the Identity of the Metadata Service Proxy

Scenario/Problem: You need to find the identity of the Metadata Service proxy.

Solution: Use the `Get-SPServiceApplicationProxy` cmdlet.

In order to get a specific instance of a Metadata Service application proxy, you need the identity of the service application proxy. The `Get-SPMetadataServiceApplicationProxy` cmdlet requires an `Identity` parameter and therefore does not display all instances. Therefore, you need to use the `Get-SPServiceApplicationProxy` cmdlet to display all service application proxies and find the Managed Metadata Service. The command line without any parameters is shown in Listing 17.6.

LISTING 17.6 **Displaying the List of Service Application Proxies**

```
Get-SPServiceApplicationProxy
```

NOTE Even though the list of proxies looks identical to the service applications, the GUIDs are different. You need to know the Metadata Service proxy for certain configuration operations described in this chapter.

Get a Specific Metadata Service Application Proxy Instance

Scenario/Problem: You need to obtain a Metadata Service application proxy instance reference.

Solution: Use the `Get-SPMetadataServiceApplicationProxy` cmdlet to assign a variable the results using the `–Identity` parameter.

Providing the identity of the proxy with the `Get-SPMetadataServiceApplicationProxy` cmdlet retrieves a specific Metadata Service application proxy instance. Use this to assign the results to an object for use in other cmdlets, as shown in Listing 17.7.

LISTING 17.7 **Assigning a Variable to a Specific Metadata Service Proxy**

```
$metadataProxy = Get-SPMetadataServiceApplication Proxy
-Identity f3e6d88a-e0a7-455c-bbef-b4942e7607b5
```

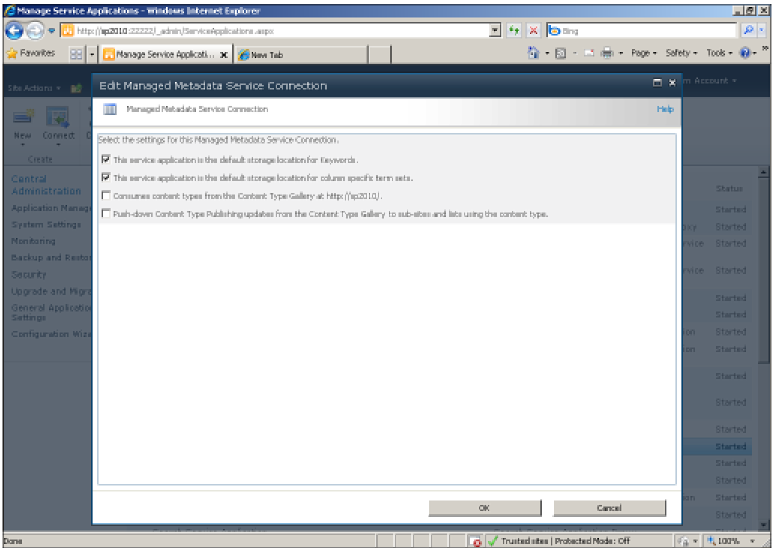
Instead of the GUID, you may use the display name. The default display name for the Managed Metadata Service application proxy is “Managed Metadata Service.” Because this is a long name, when the service application proxies are displayed, the full text is cut off.

Configure the Metadata Service Connection Options

Scenario/Problem: You need to set the Metadata Service connection options.

Solution: Use the `Set- SPMetadataServiceApplicationProxy` cmdlet with the option switch parameters.

The Metadata Service connection settings are configured at the proxy level. Modifying the properties of the Metadata Service proxy allows for four settings, as shown in Figure 17.1.



Edit Managed Metadata Service Connection

Managed Metadata Service Connection — Help

Select the settings for this Managed Metadata Service Connection.

- [✓] This service application is the default storage location for Keywords.
- [✓] This service application is the default storage location for column specific term sets.
- [ ] Consumes content types from the Content Type Gallery at http://sp2010/.
- [ ] Push-down Content Type Publishing updates from the Content Type Gallery to sub-sites and lists using the content type.

FIGURE 17.1 Editing the Managed Metadata Service connection

TIP To get to these properties, from Central Admin, click on the Manage Service Applications link under Application Management. Locate the Managed Metadata Service entry and highlight the connection (the indented instance). Click on the Properties button from the top ribbon bar.

These four options are synonymous, respectively, with the following switch parameters of the `Set-SPMetadataServiceApplicationProxy` cmdlet:

- `DefaultKeywordTaxonomy`
- `DefaultSiteCollectionTaxonomy`
- `ContentTypeSyndicationEnabled`
- `ContentTypePushdownEnabled`

Use these switch parameters with `$true` or `$false` settings (no setting assumes true) as shown in Listing 17.8.

LISTING 17.8 **Configuring the Metadata Service Connection Options**

```
Set-SPMetadataServiceApplicationProxy
-Identity "Managed Metadata Service"
–DefaultKeywordTaxonomy –DefaultSiteCollectionTaxonomy
–ContentTypeSyndicationEnabled:$false
–ContentTypePushdownEnabled:$false
```

CHAPTER 18

Session State and State Services

IN THIS CHAPTER

- ▶ Enable Session State
- ▶ Disable Session State
- ▶ Display Session State Information
- ▶ Configure the Timeout of the Session State
- ▶ Display State Service Applications Configured on the Farm
- ▶ Get a Specific State Service Application
- ▶ Rename a State Service Application
- ▶ Perform a State Service Database Operation

This chapter explores the administrative aspects of session state and the State Service application through the use of SharePoint PowerShell cmdlets.

Enable Session State

Scenario/Problem: You want to enable session state for your SharePoint web applications.

Solution: Use the `Enable-SPSessionStateService` cmdlet.

The Session State Service is not enabled by default and therefore there is no database to manage session state. This is actually the ASP.NET session state that may be leveraged by the SharePoint web applications in your farm.

To enable session state, execute the `Enable-SPSessionStateService` cmdlet with the `DefaultProvision` parameter. This automatically creates the session state database and enables the service using default values. A sample command line is shown in Listing 18.1.

LISTING 18.1 **Enabling Session State Using Defaults**

```
Enable-SPSessionStateService –DefaultProvision
```

The default database is named SessionStateService_<guid>. To make sure that the GUID-based name is not used, you may alternatively provide the database name and server using the `Enable-SPSessionStateService` cmdlet, as shown in Listing 18.2.

LISTING 18.2 **Enabling Session State with a Specific Database**

```
Enable-SPSessionStateService –DatabaseName SessionStateDB
–DatabaseServer SP2010\SQLSharePoint
```

NOTE Session state will be enabled across the entire farm; however, in order for the web applications to use session state, each web.config must be modified such that the `enableSessionState` property of the <pages> element is set to `true`.

Disable Session State

Scenario/Problem: You want to disable session state for your SharePoint web applications.

Solution: Use the `Disable-SPSessionStateService` cmdlet.

Disabling the session state can be performed using the `Disable-SPSessionStateService` cmdlet. Executing this cmdlet will remove the current database and disable the service. A sample command line is shown in Listing 18.3.

LISTING 18.3 **Disabling Session State**

```
Disable-SPSessionStateService
```

Display Session State Information

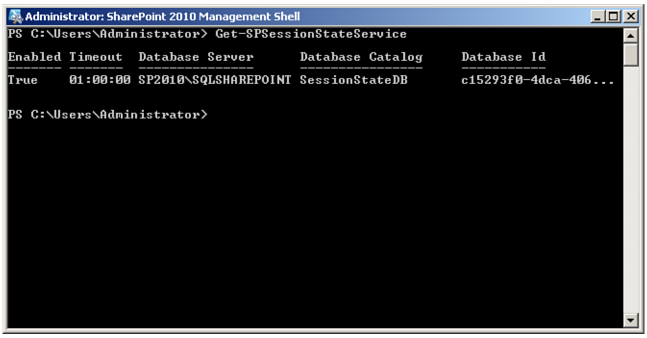
Scenario/Problem: You want to view the current session state configuration for your SharePoint farm.

Solution: Use the `Get-SPSessionStateService` cmdlet.

The `Get-SPSessionStateService` cmdlet provides the current information regarding session state within your SharePoint farm. Execute the cmdlet with no parameters, as shown in Listing 18.4. If the Session State Service is enabled, the database information is displayed, as shown in Figure 18.1. If the Session State Service is not enabled, the database information is blank, as shown in Figure 18.2 (this is the default after a new farm is installed).

LISTING 18.4 **Reviewing the Session State Configuration**

```
Get-SPSessionStateService
```

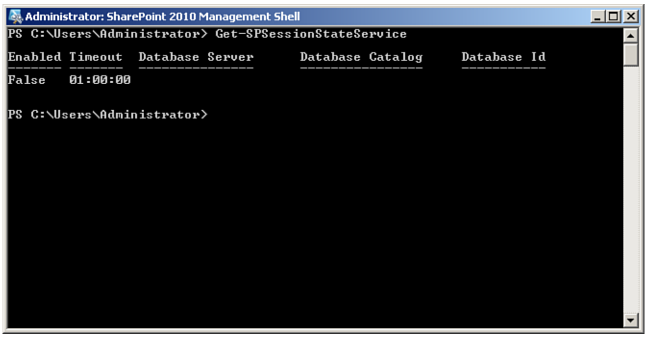


```
PS C:\Users\Administrator> Get-SPSessionStateService

Enabled Timeout  Database Server      Database Catalog     Database Id
------- -------  ---------------      ----------------     -----------
True    01:00:00 SP2010\SQLSHAREPOINT SessionStateDB       c15293f0-4dca-406...

PS C:\Users\Administrator>
```

FIGURE 18.1 Displaying an enabled session state configuration



```
PS C:\Users\Administrator> Get-SPSessionStateService

Enabled Timeout  Database Server      Database Catalog     Database Id
------- -------  ---------------      ----------------     -----------
False   01:00:00

PS C:\Users\Administrator>
```

FIGURE 18.2 Displaying a disabled session state configuration

Configure the Timeout of the Session State

Scenario/Problem: You need to modify the timeout of the Session State Service.

Solution: Use the `Set-SPSessionStateService` cmdlet with the `SessionTimeout` parameter.

One of the only configuration changes you may perform on the Session State Service is changing the timeout value. When viewing the session state configuration (as explained in the previous section), you notice that the timeout is defaulted to one hour (01:00:00). To modify this setting, use the `SessionTimeout` parameter with the `Set-SPSessionStateService` cmdlet, as shown in Listing 18.5.

LISTING 18.5 **Decreasing the Session Timeout to 30 Minutes**

```
Set-SPSessionStateService –SessionTimeout 30
```

The `SessionTimeout` parameter determines the time in minutes when an inactive session should be closed or terminated. Although the default is 60 minutes, some organizations may want to be aggressive and use a lower number such as 20 or 30.

TIP Even though a timeout value is displayed when the session state service is disabled, you may not modify the timeout setting if there is no session state service enabled.

Display State Service Applications Configured on the Farm

Scenario/Problem: You need to obtain the identity of the state service application(s) running on your SharePoint farm.

Solution: Use the `Get-SPStateServiceApplication` cmdlet without any parameters to retrieve all available content deployment paths.

In order to obtain a State Service application reference object, you need to find the identity of the path entry. Use the `Get-SPStateServiceApplication` cmdlet by itself, as shown in Listing 18.6, to list out all available State Service applications.

LISTING 18.6 **Displaying All Available State Service Applications**

```
Get-SPStateServiceApplication
```

Get a Specific State Service Application

Scenario/Problem: You need to obtain a specific state service application reference.

Solution: Use the `Get-SPStateServiceApplication` cmdlet to assign a variable the results using the `–Identity` parameter.

Providing the identity of the State Service application with the `Get-SPStateServiceApplication` cmdlet retrieves a specific State Service application. Use this to assign the results to an object for use in other cmdlets, as shown in Listing 18.7.

LISTING 18.7 **Assigning a Variable to a Specific State Service Application**

```
$stateSA = Get-SPStateServiceApplication
-Identity 5da22672-b2ad-4e68-b725-1e91a7e693eb
```

TIP You may use the display name of the state service application in quotes instead of providing the GUID.

Rename a State Service Application

Scenario/Problem: You want to rename a specific state service application.

Solution: Use the `Set-SPStateServiceApplication` cmdlet with the `Name` parameter.

You may rename a State Service application by using the `Set-SPStateServiceApplication` cmdlet along with the `Name` parameter. The `Identity` parameter must be used to identify the State Service application that should be modified. A sample command line is shown in Listing 18.8.

LISTING 18.8 **Renaming a State Service Application**

```
Set-SPStateServiceApplication
-Identity 5da22672-b2ad-4e68-b725-1e91a7e693eb
–Name "State Service App"
```

TIP You may use the display name of the state service application in quotes instead of providing the GUID. Changing the name changes the display name, so the new name will become the new identity alternative.

Perform a State Service Database Operation

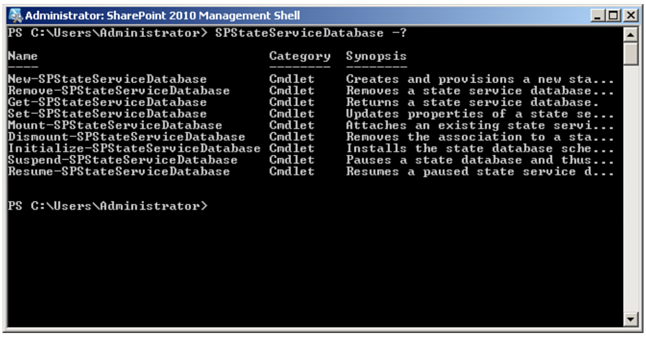
Scenario/Problem: You need to perform various database operations on the underlying state service databases.

Solution: Use the `SPStateServiceDatabase` cmdlets.

Many database operations can be performed on the state service databases. A listing of available cmdlets can be displayed by issuing the command line shown in Listing 18.9. The result is shown in Figure 18.3.

LISTING 18.9 **Command Line to Display the Database Operations**

```
SPStateServiceDatabase -?
```



Administrator: SharePoint 2010 Management Shell

```
PS C:\Users\Administrator> SPStateServiceDatabase -?

Name                              Category  Synopsis
----                              --------  --------
New-SPStateServiceDatabase        Cmdlet    Creates and provisions a new sta...
Remove-SPStateServiceDatabase     Cmdlet    Removes a state service database...
Get-SPStateServiceDatabase        Cmdlet    Returns a state service database.
Set-SPStateServiceDatabase        Cmdlet    Updates properties of a state se...
Mount-SPStateServiceDatabase      Cmdlet    Attaches an existing state servi...
Dismount-SPStateServiceDatabase   Cmdlet    Removes the association to a sta...
Initialize-SPStateServiceDatabase Cmdlet    Installs the state database sche...
Suspend-SPStateServiceDatabase    Cmdlet    Pauses a state database and thus...
Resume-SPStateServiceDatabase     Cmdlet    Resumes a paused state service d...


PS C:\Users\Administrator>
```

FIGURE 18.3 Displaying the available state service database operations

Use any of the provided cmdlets to perform the desired database operation.

PART V

Office Features

CHAPTER 19

Excel Services

IN THIS CHAPTER

- Create a Trusted File Location
- Display All Trusted File Locations
- Get a Specific Trusted File Location
- Remove a Trusted File Location
- Create a Trusted Data Connection Library
- Display All Trusted Data Connection Libraries
- Get a Specific Trusted Data Connection Library
- Remove a Trusted Data Connection Library
- Create a New Safe Data Provider
- Display All Safe Data Providers
- Get a Specific Safe Data Provider
- Remove a Safe Data Provider
- Create a Blocked File Type
- Display All Blocked File Types
- Get a Specific Blocked File Type
- Remove a Blocked File Type
- Create a User-Defined Function Reference
- Display All User-Defined Function References
- Get a Specific User-Defined Function Reference
- Remove a User-Defined Function Reference
- Modify Excel Services Objects

This chapter explores the administrative aspects of Excel Services through the use of SharePoint PowerShell cmdlets.

Create a Trusted File Location

Scenario/Problem: You need to create a new trusted file location for Excel Services.

Solution: Use the `New-SPExcelFileLocation` cmdlet.

The `New-SPExcelFileLocation` cmdlet may be used to create a new trusted location to access Excel workbooks. Provide the address of the location and the Excel Services Application instance, as shown in Listing 19.1.

LISTING 19.1 **Creating a New Trusted File Location**

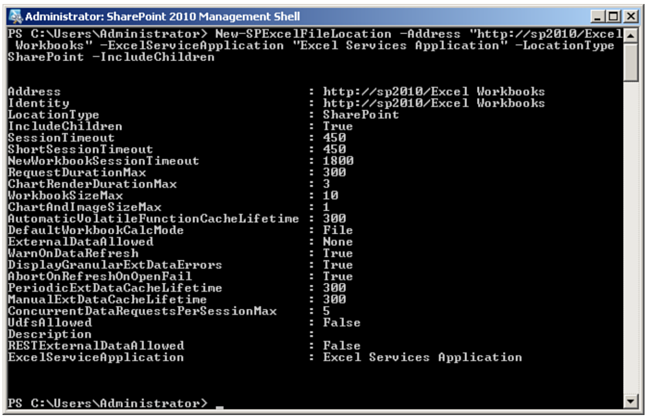
```
New-SPExcelFileLocation -Address "http://sp2010/Excel Workbooks"
-ExcelServiceApplication "Excel Services Application"
–LocationType SharePoint -IncludeChildren
```

The `–LocationType` parameter can be one of the following:

- `SharePoint` (default)
- `UNC`
- `HTTP`

The `–IncludeChildren` switch parameter is used to specify that any folders or locations underneath the address should also be trusted.

Upon successful creation of the new trusted file location, a summary of the properties is displayed, as shown in Figure 19.1.



```
PS C:\Users\Administrator> New-SPExcelFileLocation -Address "http://sp2010/Excel
 Workbooks" -ExcelServiceApplication "Excel Services Application" -LocationType
SharePoint -IncludeChildren


Address                                 : http://sp2010/Excel Workbooks
Identity                                : http://sp2010/Excel Workbooks
LocationType                            : SharePoint
IncludeChildren                         : True
SessionTimeout                          : 450
ShortSessionTimeout                     : 450
NewWorkbookSessionTimeout               : 1800
RequestDurationMax                      : 300
ChartRenderDurationMax                  : 3
WorkbookSizeMax                         : 10
ChartAndImageSizeMax                    : 1
AutomaticVolatileFunctionCacheLifetime  : 300
DefaultWorkbookCalcMode                 : File
ExternalDataAllowed                     : None
WarnOnDataRefresh                       : True
DisplayGranularExtDataErrors            : True
AbortOnRefreshOnOpenFail                : True
PeriodicExtDataCacheLifetime            : 300
ManualExtDataCacheLifetime              : 300
ConcurrentDataRequestsPerSessionMax     : 5
UdfsAllowed                             : False
Description                             :
RESTExternalDataAllowed                 : False
ExcelServiceApplication                 : Excel Services Application


PS C:\Users\Administrator> _
```

FIGURE 19.1 Creating a new trusted file location

TIP The default Excel Services application is named "Excel Services Application." To find the correct name or GUID, use `Get-SPServiceApplication` or `Get-SPExcelServiceApplication`.

Display All Trusted File Locations

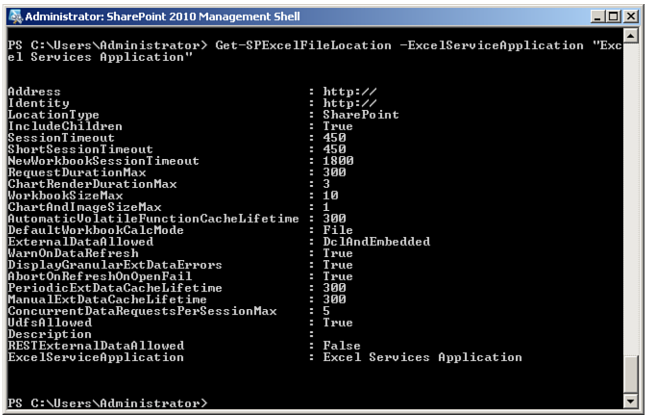
Scenario/Problem: You need to display all trusted file locations for an Excel Services application.

Solution: Use the `Get-SPExcelFileLocation` cmdlet with the `ExcelServiceApplication` parameter.

Using the `Get-SPExcelFileLocation` cmdlet with the `ExcelServiceApplication` parameter (see Listing 19.2) displays all configured trusted file locations, as shown in Figure 19.2.

LISTING 19.2 **Displaying All Trusted File Locations**

```
Get-SPExcelFileLocation
–ExcelServiceApplication "Excel Services Application"
```



```
PS C:\Users\Administrator> Get-SPExcelFileLocation -ExcelServiceApplication "Exc
el Services Application"


Address                                : http://
Identity                               : http://
LocationType                           : SharePoint
IncludeChildren                        : True
SessionTimeout                         : 450
ShortSessionTimeout                    : 450
NewWorkbookSessionTimeout              : 1800
RequestDurationMax                     : 300
ChartRenderDurationMax                 : 3
WorkbookSizeMax                        : 10
ChartAndImageSizeMax                   : 1
AutomaticVolatileFunctionCacheLifetime : 300
DefaultWorkbookCalcMode                : File
ExternalDataAllowed                    : DclAndEmbedded
WarnOnDataRefresh                      : True
DisplayGranularExtDataErrors           : True
AbortOnRefreshOnOpenFail               : True
PeriodicExtDataCacheLifetime           : 300
ManualExtDataCacheLifetime             : 300
ConcurrentDataRequestsPerSessionMax    : 5
UdfsAllowed                            : True
Description                            :
RESTExternalDataAllowed                : False
ExcelServiceApplication                : Excel Services Application


PS C:\Users\Administrator>
```

FIGURE 19.2 The `Get-SPExcelFileLocation` cmdlet displays all trusted file locations.

Get a Specific Trusted File Location

Scenario/Problem: You need to obtain a specific trusted file location reference.

Solution: Use the `Get-SPExcelFileLocation` cmdlet to assign a variable the results using the `–Identity` and `–ExcelServiceApplication` parameters.

Providing the identity of the trusted file location to the `Get-SPExcelFileLocation` cmdlet retrieves a specific trusted file location instance. Use this to assign the results to a variable for use in other cmdlets, as shown in Listing 19.3.

LISTING 19.3 **Assigning a Variable to a Specific Trusted File Location**

```
$trustedLocation = Get-SPExcelFileLocation
-Identity "http://sp2010/Excel Workbooks"
–ExcelServiceApplication "Excel Services Application"
```

TIP When you are creating a new trusted file location, the `Address` becomes the same as the `Identity` property, as shown in the results. Refer to the figures in the previous sections for examples.

Remove a Trusted File Location

Scenario/Problem: You need to remove a trusted file location from Excel Services.

Solution: Use the `Remove-SPExcelFileLocation` cmdlet with a trusted file location variable.

The `Remove-SPExcelFileLocation` cmdlet allows you to remove a specific trusted file location from Excel Services. Use this in conjunction with the `Get-SPExcelFileLocation` cmdlet (explained in a previous section), as shown in Listing 19.4.

LISTING 19.4 **Sample Script for Removing a Trusted File Location**

```
$trustedLocation = Get-SPExcelFileLocation -Identity "http://sp2010/Excel
Workbooks" –ExcelServiceApplication "Excel Services Application"
Remove-SPExcelFileLocation $trustedLocation
–ExcelServiceApplication "Excel Service Application"
```

NOTE Even though the `$trustedLocation` object variable contains the Excel Service Application property, the `–ExcelServiceApplication` parameter is still required. For simplification, you may use the file location identity instead of the variable.

Create a Trusted Data Connection Library

Scenario/Problem: You need to create a new trusted data connection library for Excel Services.

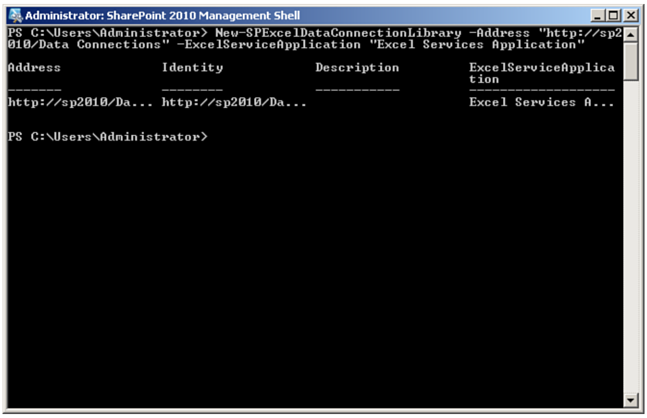
Solution: Use the `New-SPExcelDataConnectionLibrary` cmdlet.

The `New-SPExcelDataConnectionLibrary` cmdlet may be used to create a new trusted data connection location that Excel workbooks can access. Provide the address of the data connection library and the Excel Services Application instance, as shown in Listing 19.5.

LISTING 19.5 **Creating a New Trusted Data Connection Library**

```
New-SPExcelDataConnectionLibrary
-Address "http://sp2010/Data Connections"
-ExcelServiceApplication "Excel Services Application"
```

Upon successful creation of the new trusted data connection library, a summary of the properties is displayed, as shown in Figure 19.3.



```
PS C:\Users\Administrator> New-SPExcelDataConnectionLibrary -Address "http://sp2
010/Data Connections" -ExcelServiceApplication "Excel Services Application"

Address              Identity             Description          ExcelServiceApplica
                                                               tion
-------              --------             -----------          -------------------
http://sp2010/Da... http://sp2010/Da...                        Excel Services A...

PS C:\Users\Administrator>
```

FIGURE 19.3 Creating a new data connection library entry

Display All Trusted Data Connection Libraries

Scenario/Problem: You need to display all trusted data connection libraries configured for an Excel Services application.

Solution: Use the `Get-SPExcelDataConnectionLibrary` cmdlet with the `ExcelServiceApplication` parameter.

Using the `Get-SPExcelDataConnectionLibrary` cmdlet with the `ExcelServiceApplication` parameter, as shown in Listing 19.6, displays all configured trusted data connection libraries.

LISTING 19.6 **Displaying All Trusted Data Connection Libraries**

```
Get-SPExcelDataConnectionLibrary
–ExcelServiceApplication "Excel Services Application"
```

Get a Specific Trusted Data Connection Library

Scenario/Problem: You need to obtain a specific trusted data connection library reference.

Solution: Use the `Get-SPExcelDataConnectionLibrary` cmdlet to assign a variable the results using the `–Identity` and `–ExcelServiceApplication` parameters.

Providing the identity of the trusted data connection library to the `Get-SPExcelDataConnectionLibrary` cmdlet retrieves a specific data connection library instance. Use this to assign the results to a variable for use in other cmdlets, as shown in Listing 19.7.

LISTING 19.7 **Assigning a Variable to a Specific Data Connection Library**

```
$trustedLibrary = Get-SPExcelDataConnectionLibrary
-Identity "http://sp2010/Data Connections"
–ExcelServiceApplication "Excel Services Application"
```

TIP When you are creating a new trusted file location, the `Identity` becomes the same value as the `Address` property, as shown in the results. Refer to the figures in the previous sections for examples.

Remove a Trusted Data Connection Library

Scenario/Problem: You need to remove a trusted data connection library from Excel Services.

Solution: Use the `Remove-SPExcelDataConnectionLibrary` cmdlet with a trusted data connection library variable.

The Remove-SPExcelDataConnectionLibrary cmdlet allows you to remove a specific trusted data connection library from Excel Services. Use this in conjunction with the Get-SPExcelDataConnectionLibrary cmdlet (explained in a previous section), as shown in Listing 19.8.

LISTING 19.8 **Sample Script for Removing a Trusted Data Connection Library**

```
$trustedLibrary = Get-SPExcelDataConnectionLibrary
-Identity "http://sp2010/Data Connections"
–ExcelServiceApplication "Excel Services Application"
Remove-SPExcelDataConnectionLibrary $trustedLibrary
–ExcelServiceApplication "Excel Service Application"
```

NOTE For simplification, you may use the file location identity instead of the $trustedLibrary variable.

Create a New Safe Data Provider

Scenario/Problem: You need to create a new safe data provider for Excel Services.

Solution: Use the New-SPExcelDataProvider cmdlet.

The New-SPExcelDataProvider cmdlet may be used to create a new safe data provider that Excel workbooks can access via a data connection. Provide the provider information and the Excel Services Application instance, as shown in Listing 19.9.

LISTING 19.9 **Creating a New Safe Data Provider**

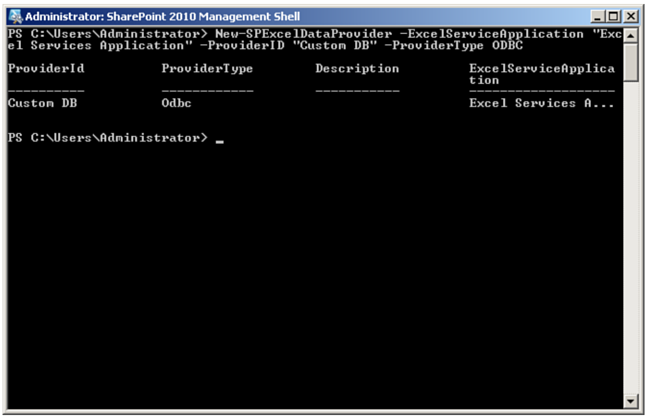
```
New-SPExcelDataProvider
-ExcelServiceApplication "Excel Services Application"
–ProviderID "Custom DB" –ProviderType ODBC
```

The –ProviderID is an identifier for the data provider. The –ProviderType can be any of the following values:

- OLEDB
- ODBC
- ODBCDSN

TIP Verify third-party data provider documentation for the proper provider ID and provider type to use.

Upon successful creation of the new safe data provider, a summary of the properties is displayed, as shown in Figure 19.4.



```
Administrator: SharePoint 2010 Management Shell
PS C:\Users\Administrator> New-SPExcelDataProvider -ExcelServiceApplication "Exc
el Services Application" -ProviderID "Custom DB" -ProviderType ODBC

ProviderId          ProviderType        Description         ExcelServiceApplica
                                                            tion
----------          ------------        -----------         -------------------
Custom DB           Odbc                                    Excel Services A...


PS C:\Users\Administrator> _
```

FIGURE 19.4 Creating a new data provider entry

Display All Safe Data Providers

Scenario/Problem: You need to display all safe data providers configured for an Excel Services application.

Solution: Use the `Get-SPExcelDataProvider` cmdlet with the `ExcelServiceApplication` parameter.

Using the `Get-SPExcelDataProvider` cmdlet with the `ExcelServiceApplication` parameter, as shown in Listing 19.10, displays all configured trusted data connection libraries.

LISTING 19.10 **Displaying All Trusted Data Connection Libraries**

```
Get-SPExcelDataProvider
-ExcelServiceApplication "Excel Services Application"
```

Get a Specific Safe Data Provider

Scenario/Problem: You need to obtain a specific safe data provider reference.

Solution: Use the `Get-SPExcelDataProvider` cmdlet with a piped `where` clause.

The `Get-SPExcelDataProvider` cmdlet does not have an `Identity` or `ProviderID` parameter and therefore only returns the entire list of safe data providers. Therefore, you need to pipe the output to a `where` clause that finds the particular data provider based on the provider ID. In order to accomplish this filtering while providing the required Excel Service Application object, you first need to pipe the Excel Service Application object into the `Get-SPExcelDataProvider` cmdlet.

This sounds more complicated than it actually is implemented. An example is shown in Listing 19.11.

LISTING 19.11 **Assigning a Variable to a Specific Data Provider**

```
$dataProvider = Get-SPExcelServiceApplication
-Identity "Excel Services Application" | Get-SPExcelDataProvider
 | where {$_.ProviderID -eq "Custom DB"}
```

Remove a Safe Data Provider

Scenario/Problem: You need to remove a safe data provider from Excel Services.

Solution: Use the `Remove-SPExcelDataProvider` cmdlet with a safe data provider variable.

The `Remove-SPExcelDataProvider` cmdlet allows you to remove a specific safe data provider from Excel Services. Use this in conjunction with the `Get-SPExcelDataProvider` cmdlet (explained in a previous section), as shown in Listing 19.12.

LISTING 19.12 **Sample Script for Removing a Safe Data Provider**

```
$dataProvider = Get-SPExcelServiceApplication
-Identity "Excel Services Application" | Get-SPExcelDataProvider
 | where {$_.ProviderID -eq "Custom DB"}
Remove-SPExcelDataProvider $dataProvider
```

The `Remove-SPExcelDataProvider` cmdlet doesn't require an Excel Service Application parameter, which makes up for the trouble of having to generate the proper data provider object.

Create a Blocked File Type

Scenario/Problem: You need to block a file type for Excel Services.

Solution: Use the `New-SPExcelBlockedFileType` cmdlet.

The `New-SPExcelBlockedFileType` cmdlet may be used to block a specific type of Excel workbook from Excel Services. Provide the file type and the Excel Services Application instance, as shown in Listing 19.13.

LISTING 19.13 **Creating a New Blocked File Type**

```
New-SPExcelBlockedFileType
-ExcelServiceApplication "Excel Services Application" –FileType XLSB
```

The `–FileType` can be one of the following:

- `XLSX`
- `XLSM`
- `XLSB`

Essentially you are blocking Excel 2010 Workbooks, Macro-Enabled Workbooks, or Binary-Enabled Workbooks (or some combination thereof).

Display All Blocked File Types

Scenario/Problem: You need to display all blocked file types for an Excel Services application.

Solution: Use the `Get-SPExcelBlockedFileType` cmdlet with the `ExcelServiceApplication` parameter.

Using the `Get-SPExcelBlockedFileType` cmdlet with the `ExcelServiceApplication` parameter, as shown in Listing 19.14, displays all configured blocked file types.

LISTING 19.14 **Displaying All Blocked File Types**

```
Get-SPExcelBlockedFileType
–ExcelServiceApplication "Excel Services Application"
```

Get a Specific Blocked File Type

Scenario/Problem: You need to obtain a specific trusted file location reference.

Solution: Use the `Get-SPExcelBlockedFileType` cmdlet to assign a variable the results using the `–Identity` and `–ExcelServiceApplication` parameters.

Providing the identity of the blocked file type to the `Get-SPExcelBlockedFileType` cmdlet retrieves a specific blocked file type instance. Use this to assign the results to a variable for use in other cmdlets, as shown in Listing 19.15.

LISTING 19.15 **Assigning a Variable to a Specific Blocked File Type**

```
$blockedFileType = Get-SPExcelBlockedFileType
-Identity "XLSB" –ExcelServiceApplication "Excel Services Application"
```

TIP When you are creating a new blocked file type, the `Identity` becomes the same value as the `File Type` property.

Remove a Blocked File Type

Scenario/Problem: You need to remove a trusted file location from Excel Services.

Solution: Use the `Remove-SPExcelBlockedFileType` cmdlet with a blocked file type variable.

The `Remove-SPExcelBlockedFileType` cmdlet allows you to remove a specific blocked file type from Excel Services. Use this in conjunction with the `Get-SPExcelBlockedFileType` cmdlet (explained in a previous section), as shown in Listing 19.16.

LISTING 19.16 **Sample Script for Removing a Blocked File Type**

```
$blockedFileType = Get-SPExcelBlockedFileType -Identity "XLSB"
–ExcelServiceApplication "Excel Services Application"
Remove-SPExcelBlockedFileType $blockedFileType
–ExcelServiceApplication "Excel Service Application"
```

NOTE Even though the `$blockedFileType` object variable contains the Excel Service Application property, the `–ExcelServiceApplication` parameter is still required. For simplification, you may use the file type identity instead of the variable.

Create a User-Defined Function Reference

Scenario/Problem: You need to configure a user-defined function reference for use in Excel Services.

Solution: Use the `New-SPExcelUserDefinedFunction` cmdlet.

The `New-SPExcelUserDefinedFunction` cmdlet may be used to create a reference to an assembly that can be used to call user-defined functions within Excel workbooks running under Excel Services. Provide the assembly information and the Excel Services Application instance, as shown in Listing 19.17.

LISTING 19.17 **Creating a New User-Defined Function Reference**

```
New-SPExcelUserDefinedFunction -ExcelServiceApplication "Excel Services
Application" –Assembly "MyCompany.ExcelServices.UDF,
Version=1.0.0.0, Culture=neutral, PublicKeyToken=82e6dce111e9429c"
–AssemblyLocation GAC
```

The `–AssemblyLocation` can be one of the following:

- `GAC`
- `LocalFile`

For security as well as central access, user-defined functions should be compiled into a DLL and placed in the GAC.

Display All User-Defined Function References

Scenario/Problem: You need to display all user-defined function references configured for an Excel Services application.

Solution: Use the `Get-SPExcelUserDefinedFunction` cmdlet with the `ExcelServiceApplication` parameter.

Using the `Get-SPExcelUserDefinedFunction` cmdlet with the `ExcelServiceApplication` parameter, as shown in Listing 19.18, displays all configured user-defined functions.

LISTING 19.18 **Displaying All User-Defined Function References**

```
Get-SPExcelUserDefinedFunction
–ExcelServiceApplication "Excel Services Application"
```

Get a Specific User-Defined Function Reference

Scenario/Problem: You need to obtain a specific user-defined function reference.

Solution: Use the `Get-SPExcelUserDefinedFunction` cmdlet to assign a variable the results using the `–Identity` and `–ExcelServiceApplication` parameters.

Providing the identity of the user-defined function to the `Get-SPExcelUserDefinedFunction` cmdlet retrieves a specific user-defined function instance. Use this to assign the results to a variable for use in other cmdlets, as shown in Listing 19.19.

LISTING 19.19 **Assigning a Variable to a Specific User-Defined Function**

```
$udfReference = Get-SPExcelUserDefinedFunction
–ExcelServiceApplication "Excel Services Application"
-Identity "Microsoft.Office.Server.WebAnalytics.UI, Version=14.0.0.0,
Culture=neutral, PublicKeyToken=71e9bce111e9429c"
```

TIP When you are creating a new user-defined function reference, the `Identity` becomes the same value as the `Assembly` property.

Remove a User-Defined Function Reference

Scenario/Problem: You need to remove a user-defined function reference from Excel Services.

Solution: Use the `Remove-SPExcelUserDefinedFunction` cmdlet with a user-defined function variable.

The `Remove-SPExcelUserDefinedFunction` cmdlet allows you to remove a specific user-defined function from Excel Services. Use this in conjunction with the `Get-SPExcelUserDefinedFunction` cmdlet (explained in a previous section), as shown in Listing 19.20.

LISTING 19.20 **Sample Script for Removing a User-Defined Function**

```
$udfReference = Get-SPExcelUserDefinedFunction
–ExcelServiceApplication "Excel Services Application"
-Identity "Microsoft.Office.Server.WebAnalytics.UI, Version=14.0.0.0,
Culture=neutral, PublicKeyToken=71e9bce111e9429c"
Remove-SPExcelUserDefinedFunction $udfReference
–ExcelServiceApplication "Excel Service Application"
```

NOTE Even though the `$udfReference` object variable contains the Excel Service Application property, the `–ExcelServiceApplication` parameter is still required. For simplification, you may use the user-defined function identity instead of the variable.

Modify Excel Services Objects

Scenario/Problem: You need to modify properties of various Excel Services objects.

Solution: Use the cmdlets listed in this section to change the configuration of existing trusted file locations, trusted data connection libraries, safe data providers, blocked file types, user-defined function references, and the Excel Services application itself.

This chapter covers the basic or more common cmdlets for creating and using Excel Services objects. For more advance scripting and modifications of these objects, the following cmdlets are available:

- `Set-SPExcelServiceApplication`
- `Set-SPExcelDataConnectionLibrary`
- `Set-SPExcelDataProvider`
- `Set-SPExcelFileLocation`
- `Set-SPExcelUserDefinedFunction`

CHAPTER 20

PerformancePoint Services

IN THIS CHAPTER

- Configure the Unattended Service Account
- Display the Unattended Service Account
- Create a Trusted Content Location
- Create a Trusted Data Source Location
- Display All Trusted Content Locations
- Display All Trusted Data Source Locations
- Display Details of Trusted Locations
- Get a Specific Trusted Location
- Remove a Trusted Location
- Configure PerformancePoint Services to Enforce Trusted Locations
- Configure PerformancePoint Services Application Settings

This chapter explores the administrative aspects of PerformancePoint Services (PPS) through the use of SharePoint PowerShell cmdlets. These are probably the longest cmdlets for SharePoint and do behave differently when compared to other service applications such as Excel Services.

Configure the Unattended Service Account

Scenario/Problem: You need to configure the unattended account for PerformancePoint Services.

Solution: Use the `Set-SPPerformancePointSecureDataValues` cmdlet.

One of the first things you need to do in order to use PerformancePoint Services within SharePoint 2010 is to set the unattended service account. This account will be used to access the data sources for PPS. This can be easily facilitated using the `Set-SPPerformancePointSecureDataValues` cmdlet, as shown in Listing 20.1.

LISTING 20.1 **Setting the Unattended Service Account**

```
Set-SPPerformancePointSecureDataValues  -ServiceApplication
"PerformancePoint Service Application"
–DataSourceUnattendedServiceAccount (Get-Credential)
```

The default PerformancePoint Services application is named “PerformancePoint Service Application.” To find the correct name or GUID, use `Get-SPServiceApplication` or `Get-SPPerformancePointServiceApplication`.

TIP Use inline credentials instead of `(Get-Credential)`. See Chapter 2, “PowerShell Basics,” for more details.

Display the Unattended Service Account

Scenario/Problem: You need to see which account is configured as the unattended service account.

Solution: Use the `Get-SPPerformancePointSecureDataValues` cmdlet.

To review the unattended service account that is configured for PPS data sources, use the `Get-SPPerformancePointSecureValues` cmdlet with the `-ServiceApplication` parameter, as shown in Listing 20.2.

LISTING 20.2 **Displaying the Unattended Service Account**

```
Get-SPPerformancePointSecureDataValues –ServiceApplication
"PerformancePoint Service Application"
```

The account name will be displayed but not the password. The default PerformancePoint Services application is named "PerformancePoint Service Application." To find the correct name or GUID, use `Get-SPServiceApplication` or `Get-SPPerformancePointServiceApplication`.

Create a Trusted Content Location

Scenario/Problem: You need to create a new trusted content location for PerformancePoint Services.

Solution: Use the `New-SPPerformancePointServiceApplicationTrusted Location` cmdlet.

The `New-SPPerformancePointServiceApplicationTrustedLocation` cmdlet may be used to create a new trusted content location to access PPS content. Provide the address (URL) of the location, the type, and the PerformancePoint Services application instance, as shown in Listing 20.3.

LISTING 20.3 **Creating a New Trusted Content Location**

```
New-SPPerformancePointServiceApplicationTrustedLocation
-Url "http://sp2010/Lists/PPS Content" -ServiceApplication
"PerformancePoint Service Application" –Type List
–TrustedLocationType Content
```

The `–Type` parameter may be one of the following for a content location:

- `SiteCollection`
- `Site`
- `List`

Passing in `Content` with the `–TrustedLocationType` parameter determines that this location is a content location for PPS. Although `DocumentLibrary` is a valid `–Type`, it is not valid for content locations.

Create a Trusted Data Source Location

Scenario/Problem: You need to create a new trusted data source location for PerformancePoint Services.

Solution: Use the `New-SPPerformancePointServiceApplicationTrustedLocation` cmdlet.

The `New-SPPerformancePointServiceApplicationTrustedLocation` cmdlet may be used to create a new trusted data source location such that PPS may access data connections. Provide the address (URL) of the location, the type, and the PerformancePoint Services application instance, as shown in Listing 20.4.

LISTING 20.4 **Creating a New Trusted Data Source Location**

```
New-SPPerformancePointServiceApplicationTrustedLocation -Url "http://
sp2010/PPS DataSource" -ServiceApplication
"PerformancePoint Service Application" –Type DocumentLibrary
–TrustedLocationType DataSource
```

The `–Type` may be one of the following for creating a content location:

- `SiteCollection`
- `Site`
- `DocumentLibrary`

Passing in `DataSource` with the `–TrustedLocationType` parameter determines that this location is a data source location for PPS. Although `List` is a valid `–Type`, it is not valid for data source locations.

Display All Trusted Content Locations

Scenario/Problem: You need to display all trusted content locations configured for a PerformancePoint Services application.

Solution: Use the `Get-SPPerformancePointServiceApplicationTrustedLocation` cmdlet with a piped `where` clause.

In order to filter the list of trusted locations, you need to use a piped `where` clause with the `Get-SPPerformancePointServiceApplicationTrustedLocation` cmdlet.

Because this cmdlet doesn't show detailed information by default, the `–Identity` parameter must be used. The service application is therefore piped into this cmdlet. A sample command line is shown in Listing 20.5.

LISTING 20.5 **Displaying All Trusted Content Locations**

```
Get-SPPerformancePointServiceApplication
"Performance Point Service Application" |
Get-SPPerformancePointServiceApplicationTrustedLocation
-Identity $_.ID | where {$_.FileType -eq "Content"}
```

Display All Trusted Data Source Locations

Scenario/Problem: You need to display all trusted data source locations configured for a PerformancePoint Services application.

Solution: Use the `Get-SPPerformancePointServiceApplicationTrusted Location` cmdlet with a piped `where` clause.

In order to filter the list of trusted locations, you need to use a piped `where` clause with the `Get-SPPerformancePointServiceApplicationTrustedLocation` cmdlet. Because this cmdlet doesn't show detailed information by default, the `–Identity` parameter must be used. The service application is therefore piped into this cmdlet. A sample command line is shown in Listing 20.6.

LISTING 20.6 **Displaying All Trusted Data Source Locations**

```
Get-SPPerformancePointServiceApplication
"Performance Point Service Application"
| Get-SPPerformancePointServiceApplicationTrustedLocation
-Identity $_.ID | where {$_.FileType -eq "DataSource"}
```

Display Details of Trusted Locations

Scenario/Problem: You need to display the underlying details and identities of trusted locations.

Solution: Use the `Get-SPPerformancePointServiceApplicationTrusted Location` cmdlet with the `–Identity` parameter piped to a `select *`.

To view all the details of the trusted locations, such as the identity GUID, you need to use the `–Identity` parameter with `Get-SPPerformancePointServiceApplication` while piping the results to a `select *` directive. The service application needs to be piped into the `Get-SPPerformancePointServiceApplication` cmdlet, as shown in Listing 20.7.

LISTING 20.7 **Displaying All Details of the Trusted Locations**

```
Get-SPPerformancePointServiceApplication
"Performance Point Service Application" |
Get-SPPerformancePointServiceApplicationTrustedLocation
-Identity $_.ID | select *
```

Once the details are displayed, you may fine-tune your `select` directive to only retrieve the information you need (for example, `select url, id`).

Get a Specific Trusted Location

Scenario/Problem: You need to obtain a specific trusted location reference.

Solution: Use the `Get-SPPerformancePointServiceApplicationTrusted Location` cmdlet to assign a variable the results using the `–Identity` and `–ServiceApplication` parameters.

Providing the identity of the trusted data connection library to the `Get-SPPerformancePointServiceApplicationTrustedLocation` cmdlet retrieves a specific data connection library instance. Use this to assign the results to a variable for use in other cmdlets, as shown in Listing 20.8.

LISTING 20.8 **Assigning a Variable to a Specific Trusted Location**

```
$trustedLocation =
Get-SPPerformancePointServiceApplicationTrustedLocation -Identity
fab6396b-5cd3-4f17-be60-57ae7d18c8a6
```

Remove a Trusted Location

Scenario/Problem: You need to remove a trusted location from PerformancePoint Services.

Solution: Use the `Remove-SPPerformancePointServiceApplicationTrusted Location` cmdlet with a trusted data location variable.-

The `Remove-SPPerformancePointServiceApplicationTrustedLocation` cmdlet allows you to remove a specific trusted location from PerformancePoint Services. Use this in conjunction with the `Get-SPPerformancePointServiceApplication TrustedLocation` cmdlet (explained in the previous section), as shown in Listing 20.9.

LISTING 20.9 **Sample Script for Removing a Trusted Location**

```
$trustedLocation =
Get-SPPerformancePointServiceApplicationTrustedLocation
-Identity fab6396b-5cd3-4f17-be60-57ae7d18c8a6
Remove-SPPerformancePointServiceApplicationTrustedLocation
$trustedLocation
```

NOTE For simplification, you may use the trusted location identity instead of the `$trustedLocation` variable.

Configure PerformancePoint Services to Enforce Trusted Locations

Scenario/Problem: You need to modify the PerformancePoint Service application to use the added trusted locations.

Solution: Use the `Set-SPPerformancePointServiceApplication` cmdlet with the trusted location parameters.

By default, all SharePoint locations are trusted as for content and data sources. Even if you added new locations using the cmdlets in the previous sections, they will be added but not enforced.

To restrict PerformancePoint Services to only use the configured trusted locations, use the `Set-SPPerformancePointServiceApplication` cmdlet with the trusted location parameters, as shown in Listing 20.10 and Listing 20.11.

LISTING 20.10 **Enforcing Trusted Content Locations**

```
Set-SPPerformancePointServiceApplication -Identity
"PerformancePoint Service Application"
–TrustedContentLocationsRestricted $true
```

LISTING 20.11 **Enforcing Trusted Data Source Locations**

```
Set-SPPerformancePointServiceApplication -Identity
"PerformancePoint Service Application"
–TrustedDataSourceLocationsRestricted $true
```

Use `$false` to disable the enforcement of the trusted locations and allow all SharePoint locations to be acceptable.

Configure PerformancePoint Services Application Settings

Scenario/Problem: You need to modify the PerformancePoint Service application settings.

Solution: Use the `Set-SPPerformancePointServiceApplication` cmdlet with the desired parameters.

The `Set-SPPerformancePointServiceApplication` cmdlet allows you to configure various settings within the PerformancePoint Service application specified. Pass in the name or the ID of the service application instance using the `–Identity` parameter and use one or more of the optional parameters to modify the appropriate settings. A sample modification of the PerformancePoint Service application settings is shown in Listing 20.12.

LISTING 20.12 **Modifying the PerformancePoint Service Application Settings**

```
Set-SPPerformancePointServiceApplication -Identity
"PerformancePoint Service Application" –CommentsDisabled $false
–CommentsScorecardMax 100
```

CHAPTER 21

Access Services

IN THIS CHAPTER

- Configure the Application Log Size
- Configure the Cache Timeout
- Modify the Maximum Columns in a Query
- Modify the Maximum Calculated Columns in a Query
- Configure the Maximum Order By Clauses
- Configure the Maximum Number of Rows in a Query
- Configure the Maximum Number of Records in an Access Table
- Configure the Maximum Number of Sources in a Query
- Enable and Disable the Use of Outer Joins
- Allow or Restrict Non-Remotable Queries
- Allow or Restrict Non-Remotable Queries
- Throttle Access Services Memory Utilization
- Throttle Session Memory Utilization
- Configure User Sessions
- Limit Template Sizes

This chapter explores the administrative aspects of Access Services through the use of SharePoint PowerShell cmdlets. Access Services is pretty light in regard to its associated PowerShell commands. Essentially there is only one cmdlet that would be used most often to configure Access Services settings. However, because many options can be configured separately, the scenarios for these settings are described in this chapter.

Configure the Application Log Size

Scenario/Problem: You need to modify the application log size for Access Services.

Solution: Use the `ApplicationLogSize` parameter with the `Set-SPAccessServiceApplication` cmdlet.

The `ApplicationLogSize` parameter determines the maximum number of records that Access Services should log. A -1 entry means unlimited, and 0 means none. The maximum entry is the maximum integer size (which varies by environment). The default is 3, records. A sample modification of the application log size is shown in Listing 21.1.

LISTING 21.1 **Setting the Application Log Size**

```
Set-SPAccessServiceApplication –Identity "Access Services"
–ApplicationLogSize 5
```

TIP The default identity of the Access Service application is “Access Services.” To find the correct name or GUID, use `Get-SPServiceApplication` or `Get-SPAccessServiceApplication`.

Configure the Cache Timeout

Scenario/Problem: You need to modify the cache timeout value for Access Services.

Solution: Use the `CacheTimeout` parameter with the `Set-SPAccessService Application` cmdlet.

The `CacheTimeout` parameter determines the maximum number of seconds that Access Services should cache data without user activity. A -1 entry means never timeout. The valid range otherwise is 1 to 2073600 (that is, 2,073,600 seconds, equating to 24 days). The default is 300 seconds (5 minutes). A sample modification of the cache timeout is shown in Listing 21.2.

LISTING 21.2 **Setting the Cache Timeout to 10 Minutes**

```
Set-SPAccessServiceApplication –Identity "Access Services"
–CacheTimeout 600
```

Modify the Maximum Columns in a Query

Scenario/Problem: You need to modify the maximum amount of columns in a query.

Solution: Use the `ColumnsMax` parameter with the `Set-SPAccessService Application` cmdlet.

The `ColumnsMax` parameter has a default value of 40 and determines the maximum number of columns that can participate in a query and/or be contained in the output of a query. The valid range of values for this parameter is 1 to 255. A sample modification of the maximum amount of columns is shown in Listing 21.3.

LISTING 21.3 **Changing the Maximum Amount of Columns**

```
Set-SPAccessServiceApplication –Identity "Access Services"
–ColumnsMax  20
```

Although you may need this to increase the amount of columns allowed, queries typically perform better when there are fewer columns to deal with. If anything, attempt to reduce the amount of columns needed in your Access Services queries.

Modify the Maximum Calculated Columns in a Query

Scenario/Problem: You need to modify the maximum amount of calculated columns in a query.

Solution: Use the `OutputCalculatedColumnsMax` parameter with the `Set-SPAccessServiceApplication` cmdlet.

The `OutputCalculatedColumnsMax` parameter has a default value of 10 and determines the maximum number of calculated columns that can be contained in the output of a query. This value does not include any calculated columns that may be part of SharePoint lists. The valid range of values for this parameter is 0 to 32. A sample modification of the maximum amount of calculated columns is shown in Listing 21.4.

LISTING 21.4 **Changing the Maximum Amount of Columns**

```
Set-SPAccessServiceApplication –Identity "Access Services"
–OutputCalculatedColumnsMax 5
```

TIP Calculated columns may improve performance because values are precalculated and then cached. However, having many calculated columns with dynamic data may present performance issues and/or stale data.

Configure the Maximum Order By Clauses

Scenario/Problem: You need to modify the maximum amount of Order By clauses in a query.

Solution: Use the `OrderByMax` parameter with the `Set-SPAccessServiceApplication` cmdlet.

The `OrderByMax` parameter has a default value of 4 and determines the maximum number of Order By clauses that can be contained within a query. The valid range of values for this parameter is 0 to 8. A sample modification of the maximum amount of Order By clauses is shown in Listing 21.5.

LISTING 21.5 **Changing the Maximum Amount of Order By Clauses**

```
Set-SPAccessServiceApplication –Identity "Access Services"
–OrderByMax 2
```

Order By clauses instantiate sorting algorithms within the query engine. Although this may be optimized with indexes, multiple Order By clauses could place a strain on the query performance. Typically, you only need one Order By to sort the results of your query; however, complex queries may require more than one.

Configure the Maximum Number of Rows in a Query

Scenario/Problem: You need to modify the maximum amount of rows in a query.

Solution: Use the `RowsMax` parameter with the `Set-SPAccessServiceApplication` cmdlet.

The `RowsMax` parameter has a default value of 25 and determines the maximum number of rows a list that participates in a query may contain and/or the maximum number of rows that may be contained in the results of the query. The valid range of values for this parameter is 1 to 200. A sample modification of the maximum amount of rows is shown in Listing 21.6.

LISTING 21.6 **Changing the Maximum Number of Rows**

```
Set-SPAccessServiceApplication –Identity "Access Services"
–RowsMax 10
```

Although lists and libraries have been optimized in SharePoint 2010 to allow an “unlimited” amount of items, if you need to deal with entities that contain thousands of rows, you may want to consider migrating the list to an Access table or the entire set of entities to a full-fledged RDBMS such as SQL Server.

Configure the Maximum Number of Records in an Access Table

Scenario/Problem: You need to modify the maximum amount of records allowed in an Access table.

Solution: Use the `RecordsInTableMax` parameter with the `Set-SPAccessServiceApplication` cmdlet.

The `RecordsInTableMax` parameter has a default value of 500 and determines the maximum number of records that may be contained within an Access table used within Access Services. An entry of -1 means unlimited, whereas the valid range of values for this parameter is 1 to the maximum integer value allowed. A sample modification of the maximum amount of records is shown in Listing 21.7.

LISTING 21.7 **Changing the Maximum Number of Records in a Table**

```
Set-SPAccessServiceApplication –Identity "Access Services"
–RecordsInTableMax 100
```

If you need to manage tens of thousands of records within Access tables, you may want to consider a migration of the database to a full-fledged RDBMS such as SQL Server.

Configure the Maximum Number of Sources in a Query

Scenario/Problem: You need to modify the maximum number of sources allowed in a query within Access Services.

Solution: Use the `SourcesMax` parameter with the `Set-SPAccessServiceApplication` cmdlet.

The `SourcesMax` parameter has a default value of 12 and determines the maximum number of lists that may be used in a query within Access Services. The valid range of values for this parameter is 1 to 20. A sample modification of the maximum amount of sources is shown in Listing 21.8.

LISTING 21.8 **Changing the Maximum Number of Sources in a Query**

```
Set-SPAccessServiceApplication –Identity "Access Services"
–SourcesMax 8
```

The more lists that are used within a query could affect performance. Limiting the amount of lists is a good idea. If you see the need to use many lists, you may want to reevaluate the list structures or query.

Enable and Disable the Use of Outer Joins

Scenario/Problem: You need to enable or disable the use of outer joins within queries.

Solution: Use the `OuterJoinsAllowed` switch parameter with the `Set-SPAccessServiceApplication` cmdlet.

By default, the Access Services service application is configured to allow outer joins within queries. Because `OuterJoinsAllowed` is a switch parameter, you may use it with a `$false` setting to disable outer joins, as shown in Listing 21.9.

LISTING 21.9 **Disabling Outer Joins in Queries**

```
Set-SPAccessServiceApplication –Identity "Access Services"
–OuterJoinsAllowed:$false
```

You may use `–OuterJoinsAllowed` by itself to enable this setting or with a `$true` value, as shown in Listing 21.10.

LISTING 21.10 **Enabling Outer Joins in Queries**

```
Set-SPAccessServiceApplication –Identity "Access Services"
–OuterJoinsAllowed:$true
```

Allow or Restrict Non-Remotable Queries

Scenario/Problem: You need to allow or restrict the use of non-remotable queries.

Solution: Use the `NonRemotableQueriesAllowed` switch parameter with the `Set-SPAccessServiceApplication` cmdlet.

Non-remotable queries are queries that cannot be remotely executed within the database. By default, the Access Services service application is configured to allow non-remotable queries. Because `NonRemotableQueriesAllowed` is a switch parameter, you may use it with a `$false` setting to disable non-remotable queries, as shown in Listing 21.11.

LISTING 21.11 **Disabling Non-Remotable Queries**

```
Set-SPAccessServiceApplication –Identity "Access Services"
–NonRemotableQueriesAllowed:$false
```

You may use `–NonRemotableQueriesAllowed` by itself to enable this setting or with a `$true` value, as shown in Listing 21.12.

LISTING 21.12 **Enabling Non-Remotable Queries**

```
Set-SPAccessServiceApplication –Identity "Access Services"
–NonRemotableQueriesAllowed:$true
```

NOTE When you are reviewing this setting in the Access Services service application settings within Central Administration, the check box label states “Remotable Queries Allowed.” However, if this box is checked, it means that non-remotable queries are indeed allowed.

Throttle Access Services Memory Utilization

Scenario/Problem: You want to modify the amount of memory that Access Services can utilize.

Solution: Use the `PrivateBytesMax` parameter with the `Set-SPAccessServiceApplication` cmdlet.

By default, Access Services is configured to use up to 50% of the physical memory of the application server that is hosting the service application. This is denoted by a -1 value for the Maximum Private Bytes setting. To change this value, you may use the `PrivateBytesMax` parameter with a value between 1 and the maximum amount of physical memory available. A sample modification of memory usage using the `Set-SPAccessServiceApplication` cmdlet is shown in Listing 21.13.

LISTING 21.13 **Modifying the Memory Usage of Access Services**

```
Set-SPAccessServiceApplication –Identity "Access Services"
–PrivateBytesMax 1024
```

If Access Services is heavily used within your SharePoint environment, you may want to allocate more memory to the service application. If the server that hosts Access Services also hosts other service applications, you may consider adding more memory to that server or splitting out services onto dedicated servers. This all depends on your overall utilization and SharePoint topology.

Throttle Session Memory Utilization

Scenario/Problem: You want to modify the amount of memory that Access Services can utilize per session.

Solution: Use the `SessionMemoryMax` parameter with the `Set-SPAccessServiceApplication` cmdlet.

By default, Access Services is configured to use up to 64MB of memory per session. To change this value, you may use the `SessionMemoryMax` parameter with a value between 1 and 4095. A sample modification of session memory usage using the `Set-SPAccessServiceApplication` cmdlet is shown in Listing 21.14.

LISTING 21.14 **Modifying the Session Memory Usage of Access Services**

```
Set-SPAccessServiceApplication –Identity "Access Services"
–SessionMemoryMax 50
```

This is the maximum amount of memory that a session will use. Normally only a small amount is needed, so the default limit of 64MB should be plenty. If sessions are using more memory than expected, you may want to investigate what operations are being performed during typical sessions.

Configure User Sessions

Scenario/Problem: You want to modify the amount of sessions per user within Access Services.

Solution: Use the `SessionsPerUserMax` or `SessionsPerAnonymousUserMax` parameter with the `Set-SPAccessServiceApplication` cmdlet.

The `SessionsPerUserMax` and `SessionsPerAnonymousUserMax` settings determine the maximum amount of sessions that can be created per user. Once the limit is reached, older sessions are deleted to make room for new ones. The default per user is 10, whereas the default per anonymous user is 25. An entry of -1 denotes unlimited, and the valid range otherwise is 1 to the maximum integer value allowed. An example of setting both user session values is shown in Listing 21.15.

LISTING 21.15 **Modifying the Amount of Sessions Per User**

```
Set-SPAccessServiceApplication –Identity "Access Services"
–SessionsPerUserMax 5 –SessionsPerAnonymousUserMax 5
```

Stale sessions will indeed continue to take up resources until the cache timeout is reached. If users are constantly instantiating new sessions, you may want to decrease these maximum session settings.

Limit Template Sizes

Scenario/Problem: You want to modify the size limit the Access templates uploaded to SharePoint.

Solution: Use the `TemplateSizeMax` parameter with the `Set-SPAccessServiceApplication` cmdlet.

The `TemplateSizeMax` parameter may be used to configure the maximum size an Access template may be. The default is 30MB. An entry of -1 denotes no limit, and a valid entry is any positive integer. An example for modifying the template size limit is shown in Listing 21.16.

LISTING 21.16 **Modifying the Template Size Limit**

```
Set-SPAccessServiceApplication –Identity "Access Services"
–TemplateSizeMax 20
```

Similar to web applications limiting documents over 50MB, the Template Size setting, by default, limits the uploading of Access templates that are more than 30MB. Typically, Access templates svhould not become that large because usually no data is included.

This setting only deals with the uploading of Access templates that are .accdt files. This has nothing to do with Access Web Database files (.accdb) that may be published to Access Services from Access 2010.

CHAPTER 22

Visio Graphics Services

IN THIS CHAPTER

- Configure the Unattended Service Account
- Create a New Safe Data Provider
- Display All Safe Data Providers
- Get a Specific Safe Data Provider
- Remove a Safe Data Provider
- Set the Description of a Data Provider
- Configure Visio Performance Settings

This chapter explores the administrative aspects of Visio Graphics Services through the use of SharePoint PowerShell cmdlets.

Configure the Unattended Service Account

Scenario/Problem: You need to configure the unattended service account for Visio Graphics Services.

Solution: Use the `Set-SPVisioExternalData` cmdlet.

Visio Graphics Services uses the Secure Store Service to facilitate the use of an unattended service account. You must first create a Secure Store Service entry with an Application ID for Visio Graphics Services.

Once an Application ID has been established, you may use the `Set-SPVisioExternalData` cmdlet to configure the unattended service account for Visio Graphic Services, as shown in Listing 22.1.

LISTING 22.1 **Configuring the Unattended Service Account**

```
Set-SPVisioExternalData -VisioServiceApplication
"Visio Graphics Service" –UnattendedServiceAccountApplicationID
"SSS Visio Graphics"
```

TIP Use the `Get-SPVisioExternalData` cmdlet to verify the unattended service account Application ID entry.

Create a New Safe Data Provider

Scenario/Problem: You need to create a new safe data provider for Visio Graphics Services.

Solution: Use the `New-SPVisioSafeDataProvider` cmdlet.

The `New-SPVisioSafeDataProvider` cmdlet may be used to create a new safe data provider that Excel workbooks can access via a data connection. Supply the provider information and the Visio Graphics Service instance, as shown in Listing 22.2.

LISTING 22.2 **Creating a New Safe Data Provider**

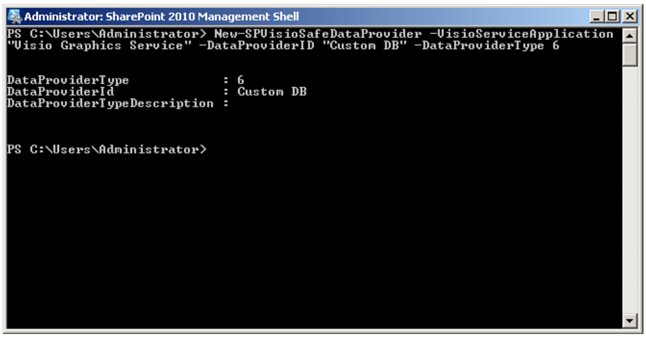
```
New-SPVisioSafeDataProvider -VisioServiceApplication
"Visio Graphics Service"
-DataProviderID "Custom DB" -DataProviderType 6
```

The `-DataProviderID` parameter is an identifier for the data provider. The `-DataProviderType` parameter can be any of the following integer values:

- 1—OLE DB
- 2—SQL Server
- 3—ODBC
- 4—ODBC with DSN
- 5—SharePoint List
- 6—Custom

TIP Verify third-party data provider documentation for the proper Data Provider ID and Data Provider Type to use.

Upon successful creation of the new safe data provider, a summary of the properties are displayed, as shown in Figure 22.1.



Administrator: SharePoint 2010 Management Shell

```
PS C:\Users\Administrator> New-SPVisioSafeDataProvider -VisioServiceApplication
"Visio Graphics Service" -DataProviderID "Custom DB" -DataProviderType 6


DataProviderType            : 6
DataProviderId              : Custom DB
DataProviderTypeDescription :


PS C:\Users\Administrator>
```

FIGURE 22.1 Creating a new safe data provider entry

Notice the `DataProviderTypeDescription` is blank. The `New-SPVisioSafeDataProvider` cmdlet does not have a parameter to include a description. To update the description, you need to use the `Set-SPVisioSafeDataProvider` cmdlet.

Display All Safe Data Providers

Scenario/Problem: You need to display all safe data providers configured for a Visio Graphics Service.

Solution: Use the `Get-SPVisioSafeDataProvider` cmdlet with the `VisioServiceApplication` parameter.

Using the `Get-SPVisioSafeDataProvider` cmdlet with the `VisioServiceApplication` parameter, as shown in Listing 22.3, displays all configured safe data providers.

LISTING 22.3 **Displaying All Safe Data Providers**

```
Get-SPVisioSafeDataProvider –VisioServiceApplication
"Visio Graphics Service"
```

Get a Specific Safe Data Provider

Scenario/Problem: You need to obtain a specific safe data provider reference.

Solution: Use the `Get-SPVisioSafeDataProvider` cmdlet.

The `Get-SPVisioSafeDataProvider` cmdlet can be used to retrieve a specific safe data provider using the `VisioServiceApplication` and data provider parameters, as shown in Listing 22.4.

LISTING 22.4 **Assigning a Variable to a Specific Data Provider**

```
$dataProvider =
Get-SPVisioSafeDataProvider –VisioServiceApplication
"Visio Graphics Service" –DataProviderID "Custom DB"
–DataProviderType 6
```

This cmdlet uses both the `DataProviderID` and the `DataProviderType` to identify the safe data provider. If one or the other is excluded, no results will be returned.

Remove a Safe Data Provider

Scenario/Problem: You need to remove a safe data provider from Visio Graphics Services.

Solution: Use the `Remove-SPVisioSafeDataProvider` cmdlet.

The `Remove-SPVisioSafeDataProvider` cmdlet allows you to remove a specific safe data provider from Visio Graphics Services. This cmdlet uses both the `DataProviderID` and the `DataProviderType` to identify the safe data provider, as shown in Listing 22.5.

LISTING 22.5 **Sample Script for Removing a Safe Data Provider**

```
Remove-SPVisioSafeDataProvider –VisioServiceApplication
"Visio Graphics Service" –DataProviderID "Custom DB"
–DataProviderType 6
```

Set the Description of a Data Provider

Scenario/Problem: You need to enter a description for a safe provider entry.

Solution: Use the `Set-SPVisioSafeDataProvider` cmdlet.

The only purpose of the `Set-SPVisioSafeDataProvider` cmdlet is to add or change the description of a specific safe data provider for a Visio Graphics Service application. This cmdlet uses both the `DataProviderID` and the `DataProviderType` to identify the safe data provider, as shown in Listing 22.6.

LISTING 22.6 **Setting the Description of a Safe Data Provider**

```
Set-SPVisioSafeDataProvider –VisioServiceApplication
"Visio Graphics Service" –DataProviderID "Custom DB"
–DataProviderType 6 –Description "Custom DB Provider"
```

TIP Use the `Get-SPVisioSafeDataProvider` cmdlet to verify the description entry or modification.

Configure Visio Performance Settings

Scenario/Problem: You need to modify performance properties of Visio Graphics Services.

Solution: Use the `Set-SPVisioPerformance` cmdlet to configure the performance settings.

One of the unique things about the `Set-SPVisioPerformance` cmdlet is that all of the parameters for configuring the Visio Graphics Service are required. Therefore, you need to provide a value for each setting, as shown in Listing 22.7.

LISTING 22.7 **Configuring the Visio Performance Settings**

```
Set-SPVisioPerformance –VisioServiceApplication
"Visio Graphics Service" –MaxDiagramCacheAge 30 -MaxDiagramSize 5
-MaxRecalcDuration 60 -MinDiagramCacheAge 10
```

The `–MaxDiagramCacheAge` parameter determines the maximum amount of minutes a diagram should be cached. A longer cache age reduces memory and CPU utilization because the diagrams are not refreshed as often. The default value is 60 minutes; however, the valid range of values for this parameter is between 0 and 34,560 minutes (equating to 24 days).

The `–MaxDiagramSize` parameter determines the largest amount of megabytes a Visio diagram can be to be processed by the Visio Graphics Service. The default value is 5 megabytes; however, the valid range of values for this parameter is between 1 and 50 megabytes.

The `–MaxRecalcDuration` parameter determines the maximum amount of seconds that the Visio Graphics Service should spend recalculating a diagram. The default is 60 seconds, while the parameter range can be from 1 to 120 seconds. Any recalculations taking longer than this setting will time out.

The `–MinDiagramCacheAge` parameter determines the minimum amount of minutes a diagram should be cached. A longer cache age reduces memory and CPU utilization because the diagrams are not refreshed as often. The default value is 5 minutes; however, the valid range of values for this parameter is between 0 and 34,560 minutes (equating to 24 days).

TIP To force a fixed cache duration, set the `MaxDiagramCacheAge` and `MinDiagramCacheAge` parameters to the same value.

CHAPTER 23

Word Services

IN THIS CHAPTER

- Configure the Conversion Processes
- Configure Conversion Throughput
- Configure Supported Document Formats for Conversion
- Modify Database Information
- Modify Job Monitoring
- Modify Conversion Timeouts
- Modify the Maximum Conversion Attempts
- Modify the Maximum Memory Usage
- Disable Word 97–2003 Document Scanning
- Disable Embedded Fonts in Conversions
- Modify the Recycle Threshold

This chapter explores the administrative aspects of Word Services through the use of SharePoint PowerShell cmdlets. Technically the service application is Word Automation Services, but to be consistent with the other Office applications, I am calling this Word Services.

Similar to Access Services, Word Services is pretty light in regard to its associated PowerShell commands. Essentially there is only one cmdlet that would be used most often to configure Word Services settings. However, because many options can be configured separately, the scenarios for these settings are described in this chapter.

Configure the Conversion Processes

Scenario/Problem: You need to modify the number of conversion processes on each server running Word Services.

Solution: Use the `ActiveProcesses` parameter with the `Set-SPWordConversionServiceApplication` cmdlet.

The `ActiveProcesses` parameter determines the how many active conversion processes can occur on each server running the Word Services service application. The default is only one process. The valid entries range from 1 to 1. A sample modification of the active conversion processes is shown in Listing 23.1.

LISTING 23.1 **Setting the Active Conversion Processes**

```
Set-SPWordConversionServiceApplication
–Identity "Word Automation Services" –ActiveProcesses 5
```

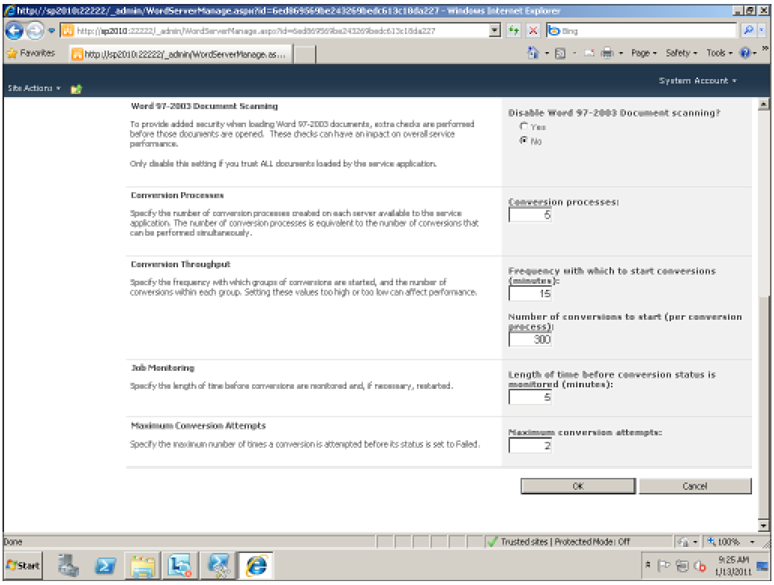
TIP The default identity of the Word Service application is “Word Automation Services.” To find the correct name or GUID, use `Get-SPServiceApplication`.

Configure Conversion Throughput

Scenario/Problem: You need to configure the throughput for the Word Services conversions.

Solution: Use the `TimerJobFrequency` and `ConversionsPerInstance` parameters with the `Set-SPWordConversionServiceApplication` cmdlet.

The conversion throughput parameters `TimerJobFrequency` and `ConversionsPerInstance` correspond to the Word Services configuration settings, as shown in Figure 23.1.


http://sp2010:22222/_admin/WordServerManage.aspx?id=6ed869569be243269bedc613c18da227 - Windows Internet Explorer
Favorites
Page
Safety
Tools
Site Actions
System Account
Done
Trusted sites | Protected Mode: Off
100%
Start
9:25 AM
1/13/2011

Word 97-2003 Document Scanning

To provide added security when loading Word 97-2003 documents, extra checks are performed before those documents are opened. These checks can have an impact on overall service performance.

Only disable this setting if you trust ALL documents loaded by the service application.

Disable Word 97-2003 Document scanning?
Yes [✓ No]

Conversion Processes

Specify the number of conversion processes created on each server available to the service application. The number of conversion processes is equivalent to the number of conversions that can be performed simultaneously.

Conversion processes: 5

Conversion Throughput

Specify the frequency with which groups of conversions are started, and the number of conversions within each group. Setting these values too high or too low can affect performance.

Frequency with which to start conversions (minutes): 15

Number of conversions to start (per conversion process): 300

Job Monitoring

Specify the length of time before conversions are monitored and, if necessary, restarted.

Length of time before conversion status is monitored (minutes): 5

Maximum Conversion Attempts

Specify the maximum number of times a conversion is attempted before its status is set to Failed.

Maximum conversion attempts: 2

OK Cancel

FIGURE 23.1 Conversion throughput configuration settings in Central Administration

TIP To get to these properties, from Central Admin, click on the Manage Service Applications link under Application Management. Locate the Word Services entry and highlight the connection (the indented instance) and then click on the Properties button from the top ribbon bar.

The `TimerJobFrequency` parameter determines how often the conversion process should be started using the corresponding service timer job. The default is 15 minutes, and the valid range may be any number of minutes between 1 and 59.

The `ConversionsPerInstance` parameter determines how many conversions may be processed per instance. The number of instances can be configured using the `ActiveProcesses` parameter, as explained in the previous section. The default value is 300 conversions.

A sample command line for setting the conversion throughput values is shown in Listing 23.2.

LISTING 23.2 **Setting the Conversion Throughput Values**

```
Set-SPWordConversionServiceApplication
-Identity "Word Automation Services"
-TimerJobFrequency 5 -ConversionsPerInstance 100
```

The values for throughput should be modified based on usage of Word Services within your organization. Heavy usage may warrant more frequent timer job execution. Increasing the amount of minutes for the timer job may warrant an increase in conversions per instance; however, increasing these values could cause performance issues. Use these values along with the active processes to balance the usage and optimize throughput.

Configure Supported Document Formats for Conversion

Scenario/Problem: You need to modify the supported document formats for Word Services.

Solution: Use the `AddSupportedFormats` and `RemoveSupportedFormats` parameters with the `Set-SPWordConversionServiceApplication` cmdlet.

By default, all available document file formats are supported within Word Services. You may use the `AddSupportedFormats` or `RemoveSupportedFormats` parameter with the `Set-SPWordConversionServiceApplication` cmdlet to modify which set of formats is supported.

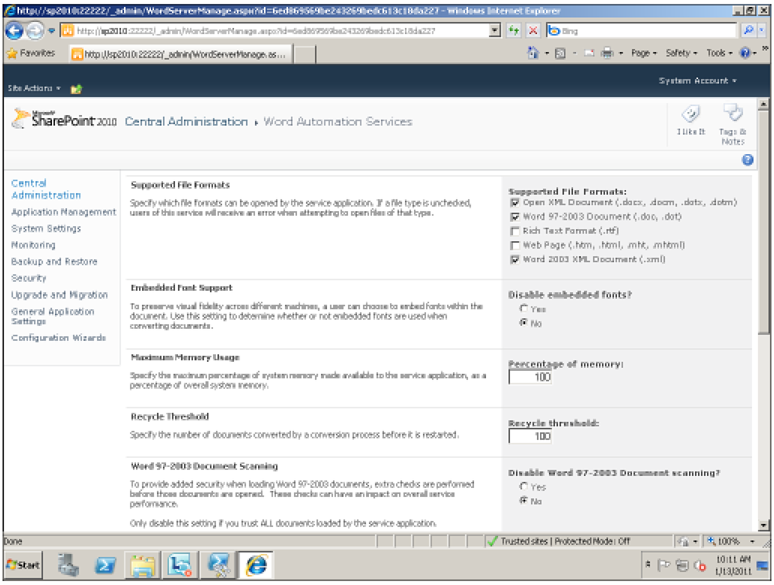
Both of these parameters take a comma-separated list of valid document formats. The recognized values are as follows:

- **docx**—Open XML Document. This essentially is any Word 2010 or Word 2007 document and includes .docx, .docm, .dotx, and .dotm documents.
- **doc**—Word 97-2003 Documents. Provides support for older Word documents with .doc or .dot extensions.
- **rtf**—Rich text Format. Only supports .rtf files.
- **mht**—Web Page. Any HTML-based web page, including .htm, .html, .mht, and .mhtml documents.
- **xml**—Word 2003 XML Document. Any Word 2003 document saved as an .xml file.

As an example of how these parameters operate, removing "doc" also removes support for .dot documents. The supported formats are grouped together and cannot be broken apart. These groups correspond to the available options available through Central Administration, as shown in Figure 23.2. Sample command lines are shown in Listing 23.3.

LISTING 23.3 **Modifying the Supported File Formats**

```
Set-SPWordConversionServiceApplication
-Identity "Word Automation Services" -RemoveSupportedFormats rtf,mht
Set-SPWordConversionServiceApplication
-Identity "Word Automation Services" -AddSupportedFormats docx,doc
```



Supported File Formats

Specify which file formats can be opened by the service application. If a file type is unchecked, users of this service will receive an error when attempting to open files of that type.

Supported File Formats:
- ☑ Open XML Document (.docx, .docm, .dotx, .dotm)
- ☑ Word 97-2003 Document (.doc, .dot)
- ☐ Rich Text Format (.rtf)
- ☐ Web Page (.htm, .html, .mht, .mhtml)
- ☑ Word 2003 XML Document (.xml)

Embedded Font Support

To preserve visual fidelity across different machines, a user can choose to embed fonts within the document. Use this setting to determine whether or not embedded fonts are used when converting documents.

Disable embedded fonts? Yes [✓ No]

Maximum Memory Usage

Specify the maximum percentage of system memory made available to the service application, as a percentage of overall system memory.

Percentage of memory: 100

Recycle Threshold

Specify the number of documents converted by a conversion process before it is restarted.

Recycle threshold: 100

Word 97-2003 Document Scanning

To provide added security when loading Word 97-2003 documents, extra checks are performed before those documents are opened. These checks can have an impact on overall service performance.

Only disable this setting if you trust ALL documents loaded by the service application.

Disable Word 97-2003 Document scanning? Yes [✓ No]

Done | Trusted sites | Protected Mode: Off | 100% | 10:11 AM 1/13/2011

FIGURE 23.2 Supported document options for Word Services

You may use quotes around a single file format; however, when you are providing a comma-separated list of formats, quotes cannot be used. No errors will be presented, although the operation will not modify the settings.

TIP When scripting out supported file format operations, use the `ClearSupportedFormats` parameter first to start with a clean slate; then just add the supported types. This ensures that any unsupported types do not remain active.

Modify Database Information

Scenario/Problem: You need to modify the database that Word Services should use.

Solution: Use the database parameters with the `Set-SPWordConversionServiceApplication` cmdlet.

Three database parameters may be used to configure the database settings for Word Services: `DatabaseServer`, `DatabaseName`, and `DatabaseCredential`. The `DatabaseCredential` parameter is only used if the authorization is performed through SQL Authentication. Even though all of these parameters are optional overall, if the `DatabaseCredential` parameter is used, then the `DatabaseServer` and `DatabaseName` become required. A sample modification of the database settings is shown in Listing 23.4.

LISTING 23.4 **Changing the Database Configuration for Word Services**

```
Set-SPWordConversionServiceApplication
–Identity "Word Automation Services" –DatabaseServer SP2010
–DatabaseName WordServices –DatabaseCredential (Get-Credential)
```

TIP Use inline credentials instead of `(Get-Credential)`. See Chapter 2, “PowerShell Basics,” for more details.

If the database that is described within the `DatabaseName` parameter does not exist, it will be created automatically!

TIP Use the database parameters to create a new database without the GUID. Delete the old WordAutomationServices_GUID database afterward.

Modify Job Monitoring

Scenario/Problem: You need to modify the job monitoring timeout value.

Solution: Use the `ConversionTimeout` parameters with the `Set-SPWord ConversionServiceApplication` cmdlet.

The job monitoring configuration setting is modified using the `ConversionTimeOut` parameter. This value is the number of minutes a conversion should be in progress before the timer job restarts the conversion. The default is 5 minutes. The valid range of values is 1 to 60 minutes. A sample modification of the job monitoring setting is shown in Listing 23.5.

LISTING 23.5 **Changing the Job Monitoring Timeout Setting**

```
Set-SPWordConversionServiceApplication
–Identity "Word Automation Services"
–ConversionTimeout 10
```

This is different from the actual configurations that determine how long a conversion can be nonresponsive or how long a conversion should run. Those values can be modified using the parameters explained in the next section.

Modify Conversion Timeouts

Scenario/Problem: You need to modify conversion timeout values either for running or nonresponsive conversions.

Solution: Use the `KeepAliveTimeout` or `MaximumConversionTime` parameter with the `Set-SPWordConversionServiceApplication` cmdlet.

To make things confusing, the `ConversionTimeout` parameter explained in the previous section deals with the job monitoring and not the timeouts of conversion processing. Use the `KeepAliveTimeout` parameter to determine how many seconds a process can be nonresponsive before it is terminated. The valid values for this setting are from 10 to 60 seconds and the default is 30 seconds.

Use the `MaximumConversionTime` parameter to determine how long a conversion should continue running. The default is 300 seconds, whereas the range starts at 60 seconds and can go up to the maximum allowed integer value. A sample modification of the conversion timeout setting is shown in Listing 23.6.

LISTING 23.6 **Changing the Conversion Timeout Settings**

```
Set-SPWordConversionServiceApplication –Identity "Word Automation
Services" –KeepAliveTimeout 45 –MaximumConversionTime 120
```

These values cannot be configured through the service application settings in Central Administration. As of this writing, they are only available via the `Set-SPWordConversionServiceApplication` cmdlet.

Modify the Maximum Conversion Attempts

Scenario/Problem: You need to modify the maximum conversion attempts in Word Services.

Solution: Use the `MaximumConversionAttempts` parameter with the `Set-SPWordConversionServiceApplication` cmdlet.

Use the `MaximumConversionAttempts` parameter to configure how many attempts should be taken before a conversion is considered a failure. The default is 2 (attempts) and the range of valid values is 1 to 10. A sample modification of the maximum conversion attempts setting is shown in Listing 23.7.

LISTING 23.7 **Changing the Maximum Conversion Attempts Setting**

```
Set-SPWordConversionServiceApplication –Identity "Word Automation Ser-
vices" –MaximumConversionAttempts 3
```

The default is two attempts but three seems like a normal setting. Anything more than five is probably too optimistic. If the conversion fails after three times, there is a 90% chance it will continue to fail.

Modify the Maximum Memory Usage

Scenario/Problem: You need to modify the maximum memory usage of Word Services.

Solution: Use the `MaximumMemoryUsage` parameter with the `Set-SPWordConversionServiceApplication` cmdlet.

Use the `MaximumMemoryUsage` parameter to configure how much memory Word Services can use on the server. The default is 100 (percent), whereas the range of valid values is 10 to 100. A sample modification of the maximum memory usage setting is shown in Listing 23.8.

LISTING 23.8 **Changing the Maximum Memory Usage to 50%**

```
Set-SPWordConversionServiceApplication
–Identity "Word Automation Services" –MaximumMemoryUsage 50
```

Most service applications like to use as much memory as possible (as defined by their default memory settings). Although most settings determine the maximum amount that can be used, you should monitor usage and available memory on your application servers to determine how to balance these settings across service applications.

Disable Word 97–2003 Document Scanning

Scenario/Problem: You want to disable extra checks on Word 97–2003 documents loaded by Word Services.

Solution: Use the `DisableBinaryFileScan` parameter with the `Set-SPWordConversionServiceApplication` cmdlet.

`DisableBinaryFileScan` is a switch parameter. By default it is false, and Word Services performs additional checks on legacy Word documents (Word 97–2003). If you trust all sources of Word documents, you may disable this additional scanning by using `Set-SPWordConversionServiceApplication` with the `DisableBinaryFileScan` switch parameter, as shown in Listing 23.9.

LISTING 23.9 **Disabling the Scanning of Word 97–2003 Documents**

```
Set-SPWordConversionServiceApplication
–Identity "Word Automation Services" –DisableBinaryFileScan
```

As long as all sources are trusted, it is a good idea to disable scanning to improve performance of Word Services against those documents. To enable scanning, use the same parameter with a `$false` value, as shown in Listing 23.10.

LISTING 23.10 **Enabling the Scanning of Word 97–2003 Documents**

```
Set-SPWordConversionServiceApplication
–Identity "Word Automation Services" –DisableBinaryFileScan:$false
```

Disable Embedded Fonts in Conversions

Scenario/Problem: You want to disable the use of embedded fonts within Word document conversions.

Solution: Use the `DisableEmbeddedFonts` parameter with the `Set-SPWordConversionServiceApplication` cmdlet.

`DisableEmbeddedFonts` is a switch parameter. By default it is false, and Word Services supports the conversion of embedded fonts within Word documents. You may disable this support by using `Set-SPWordConversionServiceApplication` with the `DisableEmbeddedFonts` switch parameter, as shown in Listing 23.11.

LISTING 23.11 **Disabling Embedded Font Support**

```
Set-SPWordConversionServiceApplication
–Identity "Word Automation Services" –DisableEmbeddedFonts
```

Unless you are experiencing issues with embedded fonts, you should be able to keep this setting enabled. To enable embedded font support, use the same parameter with a `$false` value, as shown in Listing 23.12.

LISTING 23.12 **Enabling Embedded Font Support**

```
Set-SPWordConversionServiceApplication
–Identity "Word Automation Services" –DisableEmbeddedFonts:$false
```

Modify the Recycle Threshold

Scenario/Problem: You need to modify the recycle threshold of the conversion processes in Word Services.

Solution: Use the `RecycleProcessThreshold` parameter with the `Set-SPWordConversionServiceApplication` cmdlet.

The `RecycleProcessThreshold` parameter allows you to specify the number of documents each conversion process can convert before the process is recycled. The default value is 100 (documents), whereas the valid range of values is 1 to 1. A sample threshold modification is shown in Listing 23.13.

LISTING 23.13 **Changing the Recycle Threshold**

```
Set-SPWordConversionServiceApplication
–Identity "Word Automation Services" –RecycleProcessThreshold 200
```

PART VI

Appendices

APPENDIX A

stsadm vs. SharePoint PowerShell

This appendix presents a guide to the previous stsadm operations and the new SharePoint PowerShell cmdlet replacements. Because stsadm is being phased out, the new PowerShell cmdlets will become the standard administration method.

A Operations

stsadm Operation	SharePoint 2010 Management Shell Cmdlet
Activatefeature	Enable-SPFeature
Activateformtemplate	Enable-SPInfoPathFormTemplate
Addalternatedomain	New-SPAlternateUrl
Addcontentdb	Mount-SPContentDatabase
	New-SPContentDatabase
Adddataconnectionfile	Install-SPDataConnectionFile
Add-ecsfiletrustedlocation	New-SPExcelFileLocation
Add-ecssafedataprovider	New-SPExcelDataProvider
Add-ecstrusteddataconnectionlibrary	New-SPExcelDataConnectionLibrary
Add-ecsuserdefinedfunction	New-SPExcelUserDefinedFunction
Addexemptuseragent	Add-SPInfoPathUserAgent
Addpath	New-SPManagedPath
Addsolution	Add-SPSolution
Addtemplate	Install-SPWebTemplate
Adduser	New-SPUser
Addwppack	Install-SPWebPartPack
Addzoneurl	New-SPAlternateUrl
Allowuserformwebserviceproxy	Set-SPInfoPathWebServiceProxy
Allowwebserviceproxy	Set-SPInfoPathWebServiceProxy
Authentication	Set-SPWebApplication

B Operations

stsadm Operation	SharePoint 2010 Management Shell Cmdlet
Backup	Backup-SPConfigurationDatabase Backup-SPFarm Backup-SPSite
Backuphistory	Get-SPBackupHistory

C Operations

stsadm Operation	SharePoint 2010 Management Shell Cmdlet
Createadminvs	New-SPCentralAdministration
Createsite	New-SPSite
Createsiteinnewdb	New-SPSite
Createweb	New-SPWeb

D Operations

stsadm Operation	SharePoint 2010 Management Shell Cmdlet
Deactivatefeature	Disable-SPFeature
Deactivateformtemplate	Disable-SPInfoPathFormTemplate
Deletealternatedomain	Remove-SPAlternateUrl
Deleteconfigdb	Remove-SPConfigurationDatabase
Deletecontentdb	Dismount-SPContentDatabase
Deletepath	Remove-SPManagedPath
Deletesite	Remove-SPSite
Deletesolution	Remove-SPSolution
Deletetemplate	Uninstall-SPWebTemplate
Deleteuser	Remove-SPUser
Deleteweb	Remove-SPWeb
Deletewppack	Uninstall-SPWebPartPack
Deletezoneurl	Remove-SPAlternateUrl
Deploysolution	Install-SPSolution
Deploywppack	Install-SPWebPartPack
Displaysolution	Get-SPSolution

E Operations

stsadm Operation	SharePoint 2010 Management Shell Cmdlet
Editcontentdeploymentpath	Set-SPContentDeploymentPath
Enumalternatedomains	Get-SPAlternateURL
Enumcontentdbs	Get-SPContentDatabase
Enumdataconnectionfiledependants	Get-SPDataConnectionFileDependent
Enumdataconnectionfiles	Get-SPDataConnectionFile
Enumexemptuseragents	Get-SPInfoPathUserAgent
Enumformtemplates	Get-SPInfoPathFormTemplate
Enumservices	Get-SPServiceInstance
Enumsites	Get-SPSite
Enumsolutions	Get-SPSolution
Enumsubwebs	Get-SPWeb
Enumtemplates	Get-SPWebTemplate
Enumusers	Get-SPUser
Enumwppacks	Get-SPWebPartPack
Enumzoneurls	Get-SPAlternateURL
Execadmsvcjobs	Start-SPAdminJob
Export	Export-SPWeb
Extendvs	New-SPWebApplication
Extendvsinwebfarm	New-SPWebApplicationExtension

G Operations

stsadm Operation	SharePoint 2010 Management Shell Cmdlet
Getadminport	Get-SPWebApplication
Getdataconnectionfileproperty	Get-SPDataConnectionFile
Getformtemplateproperty	Get-SPInfoPathFormTemplate
Getproperty	Get-SPFarmConfig Get-SPTimerJob Disable-SPTimerJob Enable-SPTimerJob Set-SPTimerJob Start-SPTimerJob
Getsitelock	Get-SPSiteAdministration
Geturlzone	Get-SPAlternateURL

I Operations

stsadm Operation	SharePoint 2010 Management Shell Cmdlet
Import	Import-SPWeb
Installfeature	Install-SPFeature

L Operations

stsadm Operation	SharePoint 2010 Management Shell cmdlet
Listlogginglevels	Get-SPLogLevel
Listregisteredsecuritytrimmers	Get-SPEnterpriseSearchSecurityTrimmer

M Operations

stsadm Operation	SharePoint 2010 Management Shell Cmdlet
Mergecontentdbs	Move-SPSite
Migrateuser	Move-SPUser

O, P, and Q Operations

stsadm Operation	SharePoint 2010 Management Shell Cmdlet
Osearch	SPEnterpriseSearchService SPEnterpriseSearchServiceInstance
Osearchdiacriticsensitive	SPEnterpriseSearchServiceApplication
Provisionservice	Start-SPServiceInstance
Quiesceformtemplate	Stop-SPInfoPathFormTemplate

R Operations

stsadm Operation	SharePoint 2010 Management Shell Cmdlet
Reconvertallformtemplates	Update-SPInfoPathFormTemplate
Registersecuritytrimmer	New-SPEnterpriseSearchSecurityTrimmer
Removedataconnectionfile	Uninstall-SPDataConnectionFile
Remove-ecsfiletrustedlocation	Remove-SPExcelFileLocation
Remove-ecssafedataprovider	Remove-SPExcelDataProvider
Remove-ecstrusteddataconnectionlibrary	Remove-SPExcelDataConnectionLibrary
Remove-ecsuserdefinedfunction	Remove-SPExcelFileLocation
Removeexemptuseragent	Remove-SPInfoPathUserAgent

stsadm Operation	SharePoint 2010 Management Shell Cmdlet
Removeformtemplate	Uninstall-SPInfoPathFormTemplate
Renameserver	Rename-SPServer
Renamesite	Set-SPSite
Renameweb	Set-SPWeb
Restore	Restore-SPFarm Restore-SPSite
Retractsolution	Uninstall-SPSolution
Runcontentdeploymentjob	Start-SPContentDeploymentJob

S Operations

stsadm Operation	SharePoint 2010 Management Shell Cmdlet
Scanforfeatures	Install-SPFeature
Setadminport	Set-SPCentralAdministration
Setconfigdb	Connect-SPConfigurationDatabase
Setcontentdeploymentjobschedule	Set-SPContentDeploymentJob
Setdataconnectionfileproperty	Set-SPDataConnectionFile
Set-ecsexternaldata	Set-SPExcelFileLocation
Set-ecsloadbalancing	Set-SPExcelServiceApplication
Set-ecsmemoryutilization	Set-SPExcelServiceApplication
Set-ecssecurity	Set-SPExcelServiceApplication
Set-ecssessionmanagement	Set-SPExcelServiceApplication
Set-ecsworkbookcache	Set-SPExcelServiceApplication
Setformtemplateproperty	Set-SPInfoPathFormTemplate
Setlogginglevel	Set-SPLogLevel
Setproperty	SPFarmConfig SPTimerJob
Setsitelock	Set-SPSiteAdministration
Setsiteuseraccountdirectorypath	Get-SPSiteSubscription v-SPSiteSubscription Remove-SPSiteSubscription
Setworkflowconfig	Set-SPWorkflowConfig
Siteowner	Set-SPSiteAdministration
Syncsolution	Install-SPSolution

U Operations

stsadm Operation	SharePoint 2010 Management Shell Cmdlet
Unextendvs	Remove-SPWebApplication
Uninstallfeature	Uninstall-SPFeature
Unquiesceformtemplate	Start-SPInfoPathFormTemplate
Unregistersecuritytrimmer	Remove-SPEnterpriseSearchSecurityTrimmer
Updateaccountpassword	Set-SPManagedAccount
Upgradeformtemplate	Install-SPInfoPathFormTemplate
Upgradesolution	Update-SPSolution
Uploadformtemplate	Install-SPInfoPathFormTemplate
Userrole	SPUser

V Operations

stsadm Operation	SharePoint 2010 Management Shell Cmdlet
Verifyformtemplate	Test-SPInfoPathFormTemplate

APPENDIX B

Upgrade/Migration

This appendix presents a couple SharePoint cmdlets that you may use during the upgrade and migration process when migrating from MOSS 2007 to SharePoint 2010.

Test-SPContentDatabase

When you're performing the database-attach migration strategy, part of the operation is to back up the MOSS 2007 content database(s) and then restore it on your SharePoint 2010 SQL Server. You then should test the content database(s) against your new SharePoint 2010 web application to determine if there are any issues.

The syntax for this scenario is as follows:

```
Test-SPContentDatabase –Name <name of content database>
-WebApplication <name of web application to test against>
```

By default, the current SQL Server instance will be used, but you may specify an instance by using the `–ServerInstance` parameter and passing in the SQL Server instance.

Upgrade-SPContentDatabase

Once the content database has been tested and all issues have been resolved, it then can be attached to the SharePoint farm using the `Mount-SPContentDatabase` cmdlet. The `upgrade` command will not execute until the content database is actually attached. Once it is attached, you may execute the `upgrade` command using the following syntax:

```
Upgrade-SPContentDatabase –Name <name of content database>
-WebApplication <name of web application hosting content
database>
```

Again, by default, the current SQL Server instance will be used, but you may specify an instance by using the `–ServerInstance` parameter and passing in the SQL Server instance.

A `switch` parameter is available that performs the visual upgrade during the content database upgrade process. Using `–UpdateUserExperience` with the `Upgrade-SPContentDatabase` cmdlet updates the content database to the new SharePoint 2010 look and feel. However, it is recommended that you migrate the content first, review it within the SharePoint 2010 farm, and then perform the visual upgrade manually from within SharePoint.

Index