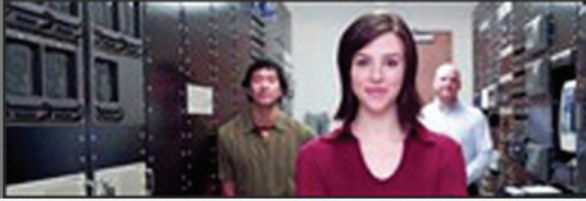


ENSIM PRO - WINDOWS



Ensim[®] Pro 10.x.x for Windows[®] 2003 Power Tools Developer Guide



Published: August 14, 2006

© 2006 Ensim Corporation. All rights reserved.

This document contains information proprietary to Ensim Corporation and its receipt or possession does not convey any rights to reproduce, disclose, manufacture, or sell anything it might describe. Reproduction, disclosure, or use without Ensim's specific written authorization is strictly forbidden.

Ensim Corporation makes no representations or warranties with respect to the contents or use of this document. It also reserves the right to revise this publication and make changes to the content at any time, without the obligation to notify any person or entity of such revisions or changes.

Further, Ensim Corporation assumes no responsibility or liability for any errors or inaccuracies, makes no warranty of any kind (express, implied or statutory) with respect to the contents or use of the information, and expressly disclaims any and all warranties of merchantability, fitness for particular purposes, and non-infringement of third party rights.

Ensim and the Ensim logo are registered trademarks of Ensim Corporation.

All other trademarks are the property of their respective owners.

CORPORATE HEADQUARTERS

Ensim Corporation
1366 Borregas Avenue
Sunnyvale, CA 94089
(408) 745-3300

www.ensim.com

Contents

About this guide

Introduction	v
Intended audience and required skills.....	v
Related documentation	v
Document conventions.....	vi
Support and feedback	vi

Chapter 1

Preparing for the integration

Introduction	1-1
Overview of the integration process	1-1
Software requirements	1-2

Chapter 2

Integrating Web applications with Ensim Pro

Introduction	2-1
Creating the Web application directory structure	2-1
Creating the DSP.xml file	2-2
Generating the DSP2.xml file	2-2
Creating the DAPI.cs file	2-2
Copying files to the Ensim Pro server	2-3
Registering the Web application with Ensim Pro	2-4
Verifying the success of the integration	2-5

Chapter 3

Contents of a sample DSP.xml file

Introduction	3-1
Contents of the DSP.xml file	3-1

Chapter 4

Contents of a sample DAPI.cs file

Introduction	4-1
Contents of the DAPI.cs file	4-1

Appendix A

Troubleshooting integration issues

Introduction	A-1
Compilation errors.....	A-1
Runtime issues.....	A-3

About this guide

Introduction

This guide provides instructions for integrating custom Web applications with Ensim® Pro 10.x.x for Windows® 2003.



Note

For simplicity, Ensim Pro 10.x.x for Windows 2003 is referred to as “Ensim Pro” throughout this document.

Chapters in this guide include:

- [Chapter 1, “Preparing for the integration”](#). Overview of the integration process and the software requirements.
- [Chapter 2, “Integrating Web applications with Ensim Pro”](#). Instructions for integrating Web applications with Ensim Pro.
- [Chapter 3, “Contents of a sample DSP.xml file”](#). Contents of a sample **DSP.xml** file.
- [Chapter 4, “Contents of a sample DAPI.cs file”](#). Contents of a sample **DAPI.cs** file.
- [Appendix A, “Troubleshooting integration issues”](#). Information about resolving integration issues.

Intended audience and required skills

This guide is intended for developers who want to integrate custom Web applications with Ensim Pro. To use this document, you need to be familiar with Microsoft® Windows Server™ 2003, Ensim Pro, and C#.

Related documentation

The following documents provide additional information about applications for Ensim Pro:

- *Ensim Pro 10.x.x for Windows 2003 Release Notes*. Important information about Ensim Pro 10.x.x release.
- *Ensim Pro 10.x.x for Windows 2003 Online Help*. The Ensim Pro control panel provides comprehensive online Help designed to answer questions and help you find the information you need, when you need it. To access online Help, log in to the control panel, then click the Help links on the lower left navigation panel.

Documentation is also available in the Ensim Pro section of the Ensim Support site, <http://support.ensim.com>. All customers receive passwords that allow access to this site. If you do not have a password, contact your site’s liaison to Ensim.

Document conventions

Throughout this guide, specific fonts are used to identify computer input, output, and interface elements. Table 1 lists the typographic conventions used in this guide.

Table 1. Typographic conventions

Appearance of text	How it is used
Narrow bold text (sans serif font)	Used for button names, column names, field names, file names, keystrokes, menu items, and path names. <i>Example:</i> Click Start , point to Settings , then click Control Panel .
Regular bold text (serif font)	Used for information you type. <i>Example:</i> On the command line, type continue
<i>Bold, italic text (serif font)</i>	Used for variables you replace with appropriate information. <i>Example:</i> Type <i><servername></i> where <i><servername></i> is the host name of your server.
Courier font	Used for system messages and screen text. <i>Example:</i> The following message is displayed: The server has been added.



Tip

In addition, key information is sometimes displayed using special headings and formats, such as this one, to make it stand out from regular text.

Support and feedback

For Ensim online support or feedback, use the following links:

- <http://support.ensim.com> to access product downloads and documentation
- <https://onlinesupport.ensim.com> to file a support ticket or request online support
- http://onlinesupport.ensim.com/kb_search_sln.asp to search the knowledge base
- <http://www.ensim.com/about/feedback.asp> to provide feedback to Ensim



Note

All customers receive passwords that allow access to the Ensim Support site. If you do not have a password, contact your site's liaison to Ensim.

Preparing for the integration

Introduction

This chapter provides an overview of integrating custom Web applications with Ensim Pro and the software requirements. Topics include:

- [“Overview of the integration process” on page 1-1](#)
- [“Software requirements” on page 1-2](#)

Overview of the integration process

Ensim provides a collection of Web applications with Ensim Pro. These Web applications are called Power Tools. Some of these applications are packaged by Ensim and are automatically installed and configured during the installation of Ensim Pro. The existing Web applications packaged with Ensim Pro include:

- Gallery (a tool to create and maintain photo albums)
- PHPBB (a discussion/bulletin board)
- WordPress (a news and Web log tool)
- DotNetNuke (a content management system)
- Community Server (a knowledge management and collaboration platform)

In addition to these Web applications, you can integrate custom Web applications with Ensim Pro by following the instructions given in this guide. Integrating a custom Web application with Ensim Pro involves:

- 1 Creating the Web application directory structure on the development server
- 2 Creating the **DSP.xml** file
- 3 Generating the **DSP2.xml** file
- 4 Creating the **DAPI.cs** file
- 5 Copying files from the Web application directory on the development server to the Ensim Pro server
- 6 Registering the Web application on Ensim Pro as a Power Tool

For instructions, see [“Integrating Web applications with Ensim Pro” on page 2-1](#).

Software requirements

You need the following software for integrating custom Web applications with Ensim Pro:

- **Power Tools Developer Kit.** The Power Tools Developer Kit comprises the following files: **PreprocessDspXML.exe**, **dsp_options.xml**, **DSP.xml**, and **DAPI.cs**. You must download these files from Ensim's FTP server.
- **Ensim Pro 10.x.x for Windows 2003.** The Power Tools Developer Kit requires version 10.x.x of Ensim Pro on the Windows 2003 server.
- **Custom Web application files.** These files are used by the Power Tools Developer Kit for integrating the Web application with Ensim Pro.
- **Third-party applications and their add-ons.** Certain Web applications require third-party applications for successful implementation. Some of the third-party applications required include MySQL, ColdFusion, Urchin. For such Web applications, you must install the third-party applications and their corresponding add-ons on the Ensim Pro server.
- **Scripting languages.** Ensim Pro provides default support for ActivPerl 5.61 and PHP 4.0.23. Ensure that a compatible version of Perl and PHP required by the Web application is installed on the Ensim Pro server.
- **Microsoft® .NET.** Ensim Pro provides default support for .NET 1.1. For the Web applications requiring .NET 2.0, you need to install .NET 2.0 on the Ensim Pro server.
- **Development environment for creating C# files.** To integrate custom Web applications, you must set up a development environment for creating C# files. Ensim recommends Microsoft Visual Studio.NET® for creating C# files.

Integrating Web applications with Ensim Pro

Introduction

This chapter provides instructions for integrating a custom Web application with Ensim Pro. This involves:

- 1 Creating the Web application directory structure
- 2 Creating the DSP.xml file
- 3 Generating the DSP2.xml file
- 4 Creating the DAPI.cs file
- 5 Copying files to the Ensim Pro server
- 6 Registering the Web application with Ensim Pro
- 7 Verifying the success of the integration

Creating the Web application directory structure

This section describes the directory structure you need to create on the development server for storing the Web application files.

Procedure

To create the directory structure for the Web application files:

- 1 Log in to the development server as the administrator.
- 2 Create a directory **<powertool_name>** on this server, where **<powertool_name>** is the name of the Web application, for example, C:\mypowertool.
- 3 Browse to this directory **<powertool_name>** and perform the following steps:
 - a Create a .gif file of size 150 x 50 pixels and save it in the directory as **<powertool_name>.gif**, for example, C:\mypowertool\mypowertool.gif. This image identifies the Web application and is displayed on the *List Power Tools* page in the site administrator control panel.
 - b Create another sub-directory **<powertool_name>** in the **<powertool_name>** directory. For example, C:\mypowertool\mypowertool.
 - c Save all the Web application files in the sub-directory **<powertool_name>**.

- d Save the sub-directory `<powertool_name>` as a zip archive `<powertool_name>_<powertool_version>.zip`, for example, `C:\mypowertool\mypowertool_1.0.zip`. Ensure that you extract the files in the parent directory `<powertool_name>`.

Important

If you have archived the files in any other format such as .tar, .tar.gz, make sure you convert it to a .zip archive.

The Web application directory structure is created.

Creating the DSP.xml file

The **DSP.xml** file contains the configuration information required to integrate a Web application with Ensim Pro. It also contains certain preprocessor instructions, which are used to create the final .xml configuration file for the Power Tool.

Download the sample **DSP.xml** file from the Power Tools Developer Kit available on Ensim's FTP server. For a complete definition of the **DSP.xml** file, refer to [Chapter 3, "Contents of a sample DSP.xml file"](#).

Generating the DSP2.xml file

After creating the **DSP.xml** file, generate the **DSP2.xml** file from the **DSP.xml** file using the pre-processor file **PreprocessDspXML.exe**. This is required to consolidate the installation and configuration information into a single output file.

Procedure

To generate the **DSP2.xml** file from **DSP.xml** file:

- 1 Open a command window and change to the directory where the file **PreprocessDspXML.exe** is located.
- 2 Run the following command:


```
PreprocessDspXML.exe <path_DSP_options.xml> <path_DSP.xml> > <path_DSP2.xml>
```

 where `<path_DSP.xml>` is the complete path where you have saved the **DSP.xml** file
`<path_DSP_options.xml>` is the complete path of the XSL file used with the preprocessor file **PreprocessDspXML.exe**
`<path_DSP2.xml>` is the complete path where you want to save the **DSP2.xml** file.
 The **DSP2.xml** file is now generated.

Creating the DAPI.cs file

The **DAPI.cs** file contains the implementation required to install, remove, import and export a Power Tool instance. To include any other custom operations, you need to add the corresponding code in the **DAPI.cs** file. This file uses data from the **DSP.xml** and **DSP2.xml** files.

Download the sample **DAPI.cs** file from the Power Tools Developer Kit available on Ensim's FTP server. For complete definition of the **DAPI.cs** file, refer to [Chapter 4, "Contents of a sample DAPI.cs file"](#).

Copying files to the Ensim Pro server

This section provides instructions for copying the Web application files from the development server to the Ensim Pro server. Before copying the files, ensure that you have downloaded the Web application files on the development server.

Procedure

To copy the Web application files to the Ensim Pro server:

- 1 Log in to the Ensim Pro server as the administrator.
- 2 Copy the Web application files from the development server to the Ensim Pro server as follows:
 - a Browse to the directory `<install_dir>\pe\PowerTools`, where `<install_dir>` is the complete path where you installed Ensim Pro. If the default settings were used during the Ensim Pro installation, the directory path is `C:\Program Files\Ensim\WEBpliance\pe\PowerTools`.
 - b Create a directory `<powertool_shortcode>` in the directory `<install_dir>\pe\PowerTools`, where `<powertool_shortcode>` is the short name of the Web application specified in the **DSP.xml** file. For example, `C:\Program Files\Ensim\WEBpliance\pe\PowerTools\mypowertool`.
 - c Create a directory `<powertool_version>` in the directory `<install_dir>\pe\PowerTools\<powertool_shortcode>`, where `<powertool_version>` is the version of the Web application specified in the **DSP.xml** file. For example, `C:\Program Files\Ensim\WEBpliance\pe\PowerTools\mypowertool\1.0`
 - d Log in to the development server, and copy the following files located in the parent directory `<powertool_name>` to the directory `<install_dir>\pe\PowerTools\<powertool_shortcode>\<powertool_version>` on the Ensim Pro server:
 - ❑ `<powertool_name>.gif`
 - ❑ `<powertool_name>_<powertool_version>.zip`
 - ❑ **DSP.xml**
 - ❑ **DSP2.xml**
 - ❑ **DAPI.cs**

The Web application files are now available on the Ensim Pro server.

Registering the Web application with Ensim Pro

This section provides instructions for registering the Web application with Ensim Pro.

Important

Registry entries are case-sensitive. Ensure that you use lower-case characters for registry variables.

Procedure

To register the Web application:

- 1 Log in to the Ensim Pro server as the administrator.
- 2 Open the registry editor and expand **My Computer > HKEY_LOCAL_MACHINE > SOFTWARE > Ensim > WEBppliance > serviceComponents > powertools > tools**.
- 3 Right-click **tools**, select **New**, then click **Key**.
- 4 Rename the existing key entry as **<powertool_shortname>**. For example, mypowertool.
- 5 Right-click **<powertool_shortname>**, select **New**, then click **String Value**. This is a registry variable of type RE_SZ.
- 6 Rename this registry variable as **name**.
- 7 Double-click **name**, type **<powertool_shortname>** in the **Value data** text box, then click **OK**.
- 8 Right-click **<powertool_shortname>**, select **New**, then click **String Value**.
- 9 Rename this registry variable as **version**.
- 10 Double-click **version**, type **<powertool_version>** in the **Value data** text box, then click **OK**.
- 11 Register the Web application with Ensim Pro as follows:
 - a Open a command window and go to the directory **<install_dir>\pe\PowerTools**.
 - b Run the following command:


```
cscript registerTool.vbs
<install_dir>\pe\PowerTools\<powertool_shortname>\<powertool_version>\dsp2.xml
```

The Web application is registered with Ensim Pro.

Verifying the success of the integration

Verify that the Web application is successfully integrated with Ensim Pro.

Procedure

To verify the integration:

- 1 Log in to the Ensim Pro control panel as the administrator.
- 2 Add a site with the Web application.
- 3 In the *Services* page, expand the **power tools** icon. The newly registered application should now be visible in the list of Power Tools.
- 4 Log in to this site as the site administrator and install an instance of this Power Tool.
- 5 On the *List Installations* page, click the URL for the instance that you installed. The browser should display the Web site for the Web application.

The Web application is successfully integrated with Ensim Pro.

Contents of a sample DSP.xml file

Introduction

This chapter contains a sample **DSP.xml** file provided by Ensim. You may use the sample file as a reference for creating your own **DSP.xml**.



Important

Do not change the values of the XML tags that are displayed in bold.

Contents of the DSP.xml file

This section describes the contents of a sample **DSP.xml** file:

```
<?xml version="1.0" encoding="utf-8" ?>

<dsp>

  <dspDescription>

    <shortname>mypowertool</shortname> <!-- Name used internally by Ensim Pro for
    referring to the Power Tool.-->

    <longname>My Powertool</longname><!-- A short description of the Power Tool displayed
    on the List Power Tools page in the site administrator control panel.-->

    <version>1.0</version> <!-- Version of the Power Tool.-->

    <url>http://www.mypowertool.com/</url> <!-- Web site URL of the Power Tool.-->

    <type>Portal</type> <!-- Type of the Power Tool. Example: Weblog, Bulletin Board.-->

    <shortDescription>My Power Tool is a content management system.</
    shortDescription> <!-- A short description of the Power Tool displayed on the List Power Tools
    page in the site administrator control panel.-->

    <icon> <!--Power Tool image that is displayed on the List Power Tools page in the site administrator
    control panel.-->

    <filename>mypowertool.gif</filename> <!--File name of the Power Tool image.-->

    <mimeType>image/gif</mimeType> <!-- Type of the Power Tool image.-->

  </icon>

  <dapiversion>1.0</dapiversion> <!-- Do not change this value.-->

  <dapirelease>1</dapirelease> <!-- Do not change this value.-->
```

```

<upgrades><!-- This element is currently not being used but should be included in this XML file for
successful compilation. This element will be used in future releases-->

<oldversion>

<shortname>mypowertool</shortname> <!--Short name of the Power Tool must be same as
above.-->

<version>1.0</version> <!-- Version of the Power Tool, must be same as above.-->

<dapiversion>1.0</dapiversion> <!--Do not change this value.-->

<dapirelease>1</dapirelease> <!-- Do not change this value.-->

</oldversion>

</upgrades>

<dependinglist>

<!-- List of Ensim Pro service components required by the Power Tool for successful installation. The
list below includes all the service component names. You can remove the service component tags which
are not required for this Power Tool installation.-->

<servicecomponent name="w3svc"> <!-- Include this tag if this Power Tool installation
requires IIS Web Service.-->

<config name="EnableDotNet" value="true" /> <!-- Include this tag if this Power Tool
installation requires the .NET framework.-->

</servicecomponent>

<servicecomponent name="msde" /> <!-- Include this tag if this Power Tool installation
requires a MSDE database.-->

<servicecomponent name="mysql" /> <!-- Include this tag if this Power Tool installation
requires a MySQL database.-->

<servicecomponent name="coldfusion" /> <!--Include this tag if this Power Tool
installation requires ColdFusion.-->

<servicecomponent name="frontpage" /> <!--Include this tag if this Power Tool
installation requires Microsoft FrontPage.-->

<servicecomponent name="msdns" /> <!--Include this tag if this Power Tool installation
requires MS DNS.-->

<servicecomponent name="msftpsvc" /> <!--Include this tag if this Power Tool
installation requires MS FTP Service.-->

<servicecomponent name="odbc" /><!--Include this tag if this Power Tool installation
requires ODBC.-->

<servicecomponent name="perl" /> <!-- Include this tag if this Power Tool installation
requires Perl.-->

<servicecomponent name="php" /> <!-- Include this tag if this Power Tool installation
requires PHP.-->

</dependinglist> <!-- Values listed hereafter are processed by the "PreprocessDspXML.exe"
tool which will be executed later in the Power Tool creation process. -->

<commonOption name="domain_to_path" /> <!-- Do not change this value.-->

```



```

<commonOption name="url_net_required" required="yes" /> <!-- Do not change
this value.-->

<commonOption name="url_path" required="yes" /> <!-- If you want to let the user
specify the name of the virtual directory where an instance of this Power Tool will be installed, then set
the value of the attribute 'required' to "yes".-->

<commonOption name="url_loc" /> <!-- Do not change this value.-->

<commonOption name="install_loc" /> <!--Do not change this value.-->

<commonOption name="dbname" required="yes" /> <!--If this Power Tool requires a
database, like MSDE or MYSQL, then set the value of the attribute 'required' to "yes"-->

<commonOption name="sqldbuser" required="yes" /> <!-- If this Power Tool requires
a database, like MSDE or MYSQL, then set the value of the attribute 'required' to "yes"-->

<commonOption name="sqldbpass" required="yes" /> <!--If this Power Tool requires
a database, like MSDE or MYSQL,then set the value of the attribute 'required' to "yes" -->

<commonOption name="username" requiried="yes" /> <!--If this Power Tool requires a
tool owner, then set the value of the attribute 'required' to "yes" -->

<commonOption name="instance_admin" required="yes" /> <!-- If this Power Tool
requires a tool owner, then set the value of the attribute 'required' to "yes"-->

<commonOption name="instance_pass1" required="yes" /> <!-- If this Power Tool
requires a tool owner, then set the value of the attribute 'required' to "yes" -->

<commonOption name="instance_pass2" required="yes" /> <!-- If this Power Tool
requires a tool owner and you want to confirm the password, then set the value of the attribute
'required' to "yes"-->

<commonOption name="instance_admin_email" /> <!-- If this Powertool requires a tool
owner and you want to accept email for this user, then set the value of the attribute 'required' to "yes"--
>

<commonOption name="syncfs" /> <!-- Do not change this value.-->

<commonOption name="overwritefiles" /><!--Do not change this value.-->

<commonOption name="applypatches" /> <!-- Do not change this value.-->

<commonOption name="archive_name" file="mypowertool_1.0.zip" /> <!--The
'file' attribute should specify the name of the zip file, which contains the actual Power Tool files. The file
name should be <powertool shortname>_<powertool version>.zip. -->

<commonOption name="archive_target" dir="mypowertool" /> -<!-- The 'dir'
attribute should specify the name of the directory to which the above zipped files should be extracted.
Must be the powertool short name.-->

<commonOption name="archive_type" type="zip" /> <!--The 'type' attribute should
specify the type of the archive specified above. Must have the value "zip". -->

</dspDescription>

</dsp>

```


Contents of a sample DAPI.cs file

Introduction

This chapter contains a sample **DAPI.cs** file provided by Ensim. You may use the sample file as a reference for creating your own **DAPI.cs** file.

Contents of the DAPI.cs file

This section describes the contents of a sample **DAPI.cs** file:

//Include the following namespaces to implement you Power Tool.

//-----

```
using System;
using System.Collections;
using System.Collections.Specialized;
using Ensim.WVH;
using Ensim.Services.PowerTools;
using EnsimBackendLib;
```

//-----

//Power Tool main class must be in the namespace "Ensim.Services.PowerTools"

```
namespace Ensim.Services.PowerTools
{
```

//Power Tool class name must be the same as the shortname specified in the dsp.xml.

//Also this class must inherit from "Powertool" class and must implement the "IDAPIInterface" interface.

```
public class mypowertool : PowerTool, IDAPIInterface
{
```

```

//Constructors must have same declaration as below.
//-----

    public mypowertool(SiteInfo site, PowerToolXMLData data):
        base(site, data)
    {
//Custom code goes here
    }

    public mypowertool(SiteInfo site, PowerToolXMLData data,
        IDictionary instanceData) : base(site, data, instanceData)
    {
//Custom code goes here
    }
//-----

/// <summary>
/// Method declaration is mandatory. Implement method to install Power Tool instance.
/// </summary>

    public void Install()
    {
//Use code below to install a Power Tool instance
// Use code below if Power Tool uses any database

        /*instanceData.Add("dbuser", dbuser);*/

//Create actions collection

        /*ActionCollection actions = new ActionCollection();*/

//Use code below to create MSDE database

        /*actions += new CreateMSDEDB();*/

//Use code below to create MYSQL database

        /*actions += new CreateDB();*/

//Use code below to unpack the archive (zip file). This is the same zip file that contains the Power Tool
files.

        /*actions += new UnpackArchive();*/

//Use the code below, if you need to set any file permissions on the Power Tool install location.

        /*PermissionsInfo[] pi = new PermissionsInfo[1];

        pi[0] = new PermissionsInfo(this.anonUser, AccessMask.Write |
            AccessMask.Read | AccessMask.Modify);

```

```

        actions += new
        SetPermission(instanceData["install_loc"].ToString(), pi,
        PermissionActionType.ADD, true);*/

//Use the code below to create a web directory for the Power Tool at the install location

        /*actions += new CreateWebDir(instanceData["url_path"].ToString(),
        true);*/

//Use the code below to move any files between two locations.

//First argument is source and second is destination

        /*actions += new MoveFile(instanceData["install_loc"].ToString() +
        "\\release.config",

        instanceData["install_loc"].ToString() + "\\web.config");*/

//Use the code below to replace key-value pairs in any file.

//Note: File must contain key-value pairs, for example web.config file.

        /*string origKey, newKey;

        Hashtable ht = new Hashtable();

        origKey = "<add
        key=\"SiteSqlServer\" value=\"Server=(local);Database=mypowertool;u
        id=;pwd=;\" />";

        newKey = "<add
        key=\"SiteSqlServer\" value=\"Server=(local);Database="+
        instanceData["dbname"]+";uid="+
        instanceData["sqldbuser"]+";pwd="+instanceData["sqldbpass"]+";\" /
        >";

        ht.Add(origKey, newKey);

        actions += new UpdateFileContent("Web.Config", ht);*/

//Use the code below, if Power Tool needs to set any IIS meta properties.

//For example, if the tool needs to add default.aspx to the "defaultdoc" meta property

        /*Hashtable appendProperties = new Hashtable();

        appendProperties.Add("DefaultDoc", "default.htm");

        actions += new
        SetDefaultProperties(instanceData["url_path"].ToString(), appendPro
        perties, null);*/

//Use the code below to remove any files not required after instance is created

//Pass comma-separated values, values ending with \ are directories and rest are files.

        /*actions += new RemoveFiles(@"Installer\",
        @"App_GlobalResources\Locales.Portal.xml.resources",
        @"App_GlobalResources\Locales.xml.resources");*/

```

```

//Use code below to execute all the above actions

    /*ExecuteActions(actions, ActionType.INSTALL, true);*/
}

/// <summary>
/// When the site administrator user's password changes, then the ChangePassword method is
/// implemented to handle this password change in the Power Tool.
/// </summary>
/// <param name="username">username whose password has changed.</param>
/// <param name="newpassword">the new password</param>
    public void ChangePassword(string username,string newpassword)
    {
    }

/// <summary>
/// Implement this method if you want to reconfigure the Power Tool.
/// </summary>
    public void Reconfigure()
    {
    }

/// <summary>
/// Implement this method if you want to upgrade the Power Tool.
/// This is currently not being used. This method will be used in future releases.
/// </summary>
    public void Upgrade()
    {
    }

/// <summary>
/// Method declaration is mandatory. Implement method to remove Power Tool instance.
/// </summary>
    public void remove()
    {

//use code below to remove the Power Tool instance

        /*ActionCollection actions = new ActionCollection();
        actions += new RemoveWebDir( instanceData["url_path"] );
        actions += new RemoveToolFiles();*/

```

```

//Use code below if the tool uses MSDE database.

    /*actions += new RemoveMSDEDB();*/

//Use code below if the tool uses MYSQL database.

    /*actions += new RemoveDB();*/

//Use code below to execute all above actions

    /*ExecuteActions(actions, ActionType.REMOVE, false);*/

}

/// <summary>
/// This methods is used to get the configuration values for the Power Tool.
/// </summary>

/// <returns>Returns an IDictionary object which contains a key-value pair of the Power Tool
configuration. These configuration keys are the same as those defined in the dsp.xml file.
/// </returns>

    public IDictionary GetInstanceConfiguration() {

/*For example, if method needs to return configuration values for keys "dbname", "sqldbuser",
"sqldbpass", "username" then use the commented code below. Note: The return value cannot be NULL
for successful Power Tool instantiation.*/

        //IDictionary instance_conf = this.DSPFillInstance(new
        // string[] { "dbname", "sqldbuser", "sqldbpass", "username" } );

        //return instance_conf;

    }

/// <summary>
/// Implement this method if the Power Tool supports export/import functionality.
/// Declaration of this method is mandatory.
/// </summary>

    /// <param name="datafolderpath">Power Tool export location.</
    param>

    public void Export(string datafolderpath)
    {

/*For example, if powertool uses MSDE databases, then use the commented code below.*/

        //ActionCollection actions = new ActionCollection();

        //actions += new ExportMSDEDB(datafolderpath );

        //ExecuteActions(actions, ActionType.EXPORT, false);

    }

```

```

    /// <summary>
    /// Implement this method if Power Tool supports export/import functionality.
    /// Declaration of this method is mandatory.
    /// </summary>
    /// <param name="datafolderpath">Power Tool import location.</param>
    public void Import(string datafolderpath)
    {
        /*For example, to import a powertool using MSDE/MYSQL, use the below commented code*/
        //Use code below to import MSDE databases

        /*ActionCollection actions = new ActionCollection();
        actions += new CreateMSDEDB();

        actions += new ImportMSDEDB( datafolderpath,
        instanceData["sqldbuser"].ToString());*/

        //Use code below to import MYSQL databases

        *string oldptdatapath = String.Format(@"{0}\{1}", datafolderpath,
        instanceData["dbname"].ToString());

        if(Directory.Exists(oldptdatapath))
        {
            actions += new CreateDB();
            actions += new ImportTables( datafolderpath );
        }*/

        //Use the code below to remove the Power Tool files, if any, copied by the w3svc service.

        /*actions += new RemoveWebDir(instanceData["url_path"] );
        actions += new RemoveToolFiles();*/

        //Use the code below to restore Power Tool files, from import location to the site data location

        /*actions += new CreateWebDir(instanceData["url_path"].ToString(),
        true );

        actions += new CopyToolFiles( datafolderpath );*/

        //Use the code below, if Power Tool needs to set any IIS meta properties while importing.
        //For example, if the tool needs to add default.aspx to the "defaultdoc" meta property
        //Note: Every instance of the Power Tool will add one entry of default.aspx into the site properties in IIS.

        /*Hashtable appendProperties = new Hashtable();
        appendProperties.Add("DefaultDoc", "default.aspx");

```



```

        actions += new
        SetDefaultProperties(instanceData["url_path"].ToString(),
        appendProperties, null);*/

//Use the code below, if Power Tool needs to set any file permissions on the Power Tool install location.

        /*PermissionsInfo[] pi = new PermissionsInfo[1];

        pi[0] = new PermissionsInfo(this.anonUser, AccessMask.Write |
        AccessMask.Read | AccessMask.Modify);

        actions += new
        SetPermission(instanceData["install_loc"].ToString(), pi,
        PermissionActionType.ADD, true);*/

//Use code below to execute all above actions

        /*ExecuteActions(actions, ActionType.IMPORT, false);*/
    }

/// <summary>

/// If the Power Tool requires .Net, then the implementation to handle ASP.Net version change is as
/// follows:

/// </summary>

    /// <param name="Add">If ASP.Net version is changed to
    "1.1.4322.0" the value is true.

    /// If ASP.Net version is changed to "2.0.50727.0" the value is
    false.

    /// </param>

    public void HandleASPNetVersionChange(bool Add)
    {
    }
}

```


Troubleshooting integration issues

Introduction

This chapter provides information for troubleshooting issues that you may encounter while integrating custom Web applications with Ensim Pro.

Topics include:

- “Compilation errors” on page A-1
- “Runtime issues” on page A-3

Compilation errors

Table A-1 lists the issues that you may encounter while generating the **DSP2.xml** file from the **DSP.xml** file.

Table A-1. Compilation errors

Error message	Possible cause	Solution
Usage : PreprocessDspXML <stylesheet> <xmlfile>	This error occurs if the command used to generate the DSP2.xml file is incorrect.	Ensure that you have typed the command in the command window, in the directory where the file PreprocessDspXML.exe is located as follows: PreprocessDspXML.exe <path_DSP_options.xml> <path_DSP.xml> > <path_DSP2.xml> For example, PreprocessDspXML.exe dsp_options.xml d:\X3\dsp.xml > d:\x3\dsp2.xml

Table A-1. Compilation errors (continued)

Error message	Possible cause	Solution
<pre> Unhandled Exception: System.Xml.XmlException: This is an invalid comment syntax. Expected '-->'. Line 58, position 58. at System.Xml.XmlScanner .ScanComment() at System.Xml.XmlScanner .ScanMarkup() </pre>	<p>This error occurs if the DSP.xml file contains a syntax error.</p>	<p>Verify that the XML syntax is correct at the line number specified in the exception for the DSP.xml file.</p>
<pre> File .\X\3.2.1\dsp2.xml does not exist </pre>	<p>This error occurs if you have entered an incorrect path in the command while registering the Power Tool with Ensimg Pro.</p>	<p>Ensure that you have entered the correct path <code><install_dir>\pe\PowerTools\<powertool_shortname>\<powertool_version>\dsp2.xml</code> for the DSP2.xml file in the command window from the directory <code><install_dir>\pe\PowerTools</code> For example: <pre> cscript registerTool.vbs .\X3\3.2.1\dsp2.xml </pre> </p>

Runtime issues

Table A-2 lists the issues that you may encounter while provisioning or managing the Power Tool through the Ensim Pro control panel.

Table A-2. Runtime issues

Description	Possible cause	Solution
Generic exception while creating a Power Tool instance	<ul style="list-style-type: none"> The registry entries are typed using incorrect case. The Power Tool short directory name, long directory name, and/or version number are not the same in the DSP.xml and DAPI.cs files, and the registry on the Ensim Pro server. 	Ensure that: <ul style="list-style-type: none"> The registry entries are typed using lower-case characters. The Power Tool short directory name, long directory name and/or version number are the same in the DSP.xml and DAPI.cs files and the registry on the Ensim Pro server.
Exception while provisioning the Power Tool to a site.	Certain applications required by the Power Tool are not provisioned for the site. For example, PHP or Perl required by the Power Tool is not installed.	Ensure that: <ul style="list-style-type: none"> All dependencies specified in the DSP.xml file are met while provisioning the site with the Power Tool. Interdependencies between other service components should also be also satisfied.
Unable to delete a site with a Power Tool instance.	The Remove () method in the DAPI.cs file is marked as a comment.	Ensure that the Remove () method is not marked as a comment in the DAPI.cs file.
Unable to delete a Power Tool instance from a site.	The Remove () method in the DAPI.cs file is marked as a comment.	Ensure that the Remove () method is not marked as a comment in the DAPI.cs file.
The Power Tool is provisioned more than once on a site.	The command for registering the Power Tool with Ensim Pro has been executed more than once.	Ensure that you run the following command only once in the command window from the directory <install_dir>\pe\PowerTools on the Ensim Pro server. For example: <pre>cscript registerTool.vbs .\X3\3.2.1\dsp2.xml</pre>

Table A-2. Runtime issues (continued)

Description	Possible cause	Solution
Multiple default.aspx entries in the site configuration file on the IIS server.	<p>The following code is executed for every Power Tool instance and adds the default.aspx entry in the default document list each time:</p> <pre> Hashtable appendProperties = new Hashtable(); appendProperties.Add("DefaultDoc", "default.aspx"); actions += new SetDefaultProperties(instanceData["url_path"].ToString(), appendProperties, null); </pre>	<p>Ensure that the following lines in the DAPI.cs file are marked as a comment:</p> <pre> Hashtable appendProperties = new Hashtable(); appendProperties.Add("DefaultDoc", "default.aspx"); actions += new SetDefaultProperties(instanceData["url_path"].ToString(), appendProperties, null); </pre>
Mandatory database-specific fields for provisioning the Power Tool for a site	<p>The following XML options have the required attribute value as "yes" in the DSP.xml file:</p> <pre> "dbname" "sqldbuser" "sqldbpass" "username" "instance_admin" "instance_pass1" "instance_pass2" </pre>	<p>Remove the required attribute from the following XML options in the DSP.xml file:</p> <pre> "dbname" "sqldbuser" "sqldbpass" "username" "instance_admin" "instance_pass1" "instance_pass2" </pre>

ENSIM PRO - WINDOWS

ENSIM CORPORATION

1366 Borregas Avenue
Sunnyvale, California 94089
www.ensim.com

