Lab 6: Using the Official File API

**Lab Time: 45 Minutes**

**Lab Directory: ECM401.OfficialFileSubmit**

**Lab Overview:**

In this lab, you will create a custom **STSADM** command that will allow content administrators to send files to the Litware records repository from the command line. It will also enable them to enumerate the available routing types in the repository.

# Exercise 1: Create a command to enumerate routing types

1. Start by creating a new **Class Library** project in Visual Studio. Give it the name **ECM401.OfficialFileSTSADMCommand** .
2. Add a reference to the **Microsoft.SharePoint** assembly. Right click on the References node and select **Add Reference…** from the context menu. From the Add Reference dialog, scroll to the bottom and select **Windows SharePoint Services** from the list.
3. Add a second reference to the **Microsoft.Office.Policy** assembly. Click **Add Reference…** again and click **Browse** at the top of the dialog. Navigate to the SharePoint **ISAPI** folder and double-click the **Microsoft.Office.Policy.dll** file.

Note: The ISAPI folder is located at **c:\Program Files\Common Files\Microsoft Shared\web server extensions\12\ISAPI**

1. Rename the auto-generated **class1.cs** file to **EnumRoutingTypes.cs** and open the file for editing.
2. Add the following lines of code at the top of the file.

using Microsoft.SharePoint;  
using Microsoft.SharePoint.StsAdmin;  
using Microsoft.Office.RecordsManagement.RecordsRepository;

1. Delete everything inside the **ECM401.OfficialFileSTSADMCommand** namespace and create a new class named **EnumRoutingTypes** derived from **ISPStsadmCommand** .
2. Enter the following code to provide a help message for users.

Code Snippet: 'ISPStsadmCommand.GetHelpMessage'

string ISPStsadmCommand.GetHelpMessage(string command)  
 {  
 string msg = "Enumerates the available routing types in a records repository site.";  
 msg += "\n-url\t&lt;url&gt;  
 \t\tthe url of the record center site";  
 return msg;  
 }

1. Implement the Run method as follows:

Code Snippet: 'ISPStsadmCommand.Run'

int ISPStsadmCommand.Run(string command,  
 System.Collections.Specialized.StringDictionary keyValues,  
 out string output)  
 {  
 int result = 1;  
 string sRepositoryUrl = string.Empty;  
 try  
 {  
 // validate the arguments  
 if (null == (sRepositoryUrl = keyValues["url"])  
 || sRepositoryUrl.Length == 0)  
 throw new ApplicationException("No url specified.");  
  
 // open the website  
 using (SPSite site = new SPSite(keyValues["url"]))  
 {  
 using (SPWeb web = site.OpenWeb())  
 {  
 // Enumerate the routing types using  
 // the RecordSeriesCollection wrapper.  
 RecordSeriesCollection routingTypes =  
 new RecordSeriesCollection(web);  
  
 if (null == routingTypes || routingTypes.Count == 0)  
 {  
 output = "No routing types were found.";  
 }  
 else  
 {  
 output = string.Format(  
 "Found {0} routing types:", routingTypes.Count);  
 for (int i=0; i &lt; routingTypes.Count; i++)  
 {  
 RecordSeries routingType = routingTypes[i];  
 output += string.Format(  
 "\n{0}\n\t{1}\n", routingType.Name,  
 routingType.Description);  
 }  
 }  
 }  
 }  
 }  
 catch (Exception x)  
 {  
 output = x.ToString();  
 }  
 return result;  
 }

1. Sign the assembly using the **ECM401.snk** file located in your **Student\Resources** folder.
2. Add a new **XML File** item to the project named **stsadmcommands.records.xml** . This file will contain the CAML code needed to tell STSADM that your custom command exists and where to load the ISPStsadmCommand implementation you have provided. Open the file and add the following lines of XML code.

XML Snippet: 'EnumRoutingTypes Command Definition'

PublicKeyToken\*\*\*INSERT PUBLIC KEY TOKEN\*\*\*  
 <?xml version="1.0" encoding="utf-8" ?>  
 <commands>  
 <command name="enumroutingtypes"  
 class="ECM401.OfficialFileSTSADMCommand.EnumRoutingTypes, ECM401.OfficialFileSTSADMCommand, Version=1.0.0.0, Culture=neutral, PublicKeyToken=$PublicKeyToken$"/>  
 </commands>

Note: The PublicKeyToken value must match the actual public key token of the assembly, or STSADM will be unable to locate your command implementation.

1. Save the file and open the project property pages. Navigate to the **Build Events** page and enter the following commands into the **Post-Build Events** window.

gacutil -f -i "$(TargetPath)"  
copy "$(ProjectDir)stsadmcommands.records.xml" "C:\Program Files\Common Files\Microsoft Shared\Web Server Extensions\12\Config"

1. These commands will register your assembly in the Global Assembly Cache and copy your custom command definition file into the SharePoint **Config** folder. Now you are ready to build the project.
2. Open a command prompt window and type the following command:stsadm –help enumroutingtypes
3. You should see the following screen:
4. Now type in the command to enumerate the routing types in the Litware Record Center:stsadm –o enumroutingtypes –url http://localhost:9997
5. You should see the following screen:
6. Next, you will extend the solution to include another command to submit files to the record center.

# Exercise 2: Add a command to submit files to the records center

1. In this exercise, you will add a second **STSADM** command to enable administrators to select files and send them to the records center directly from the command line.
2. Create a new class file and give it the name **SubmitOfficialFile.cs** . Add the following lines of code at the top of the file.

using System.IO;  
using Microsoft.SharePoint;  
using Microsoft.SharePoint.StsAdmin;  
using OfficialFile=Microsoft.Office.RecordsManagement.RecordsRepository;

1. Implement the **GetHelpMessage** method as follows to provide guidance for the user.

Code Snippet: 'SubmitOfficialFile.GetHelpMessage'

string ISPStsadmCommand.GetHelpMessage(string command)  
 {  
 string msg = “stsadm -o submitfiles”;  
 msg += "\nSubmits one or more files to an official records repository site.";  
 msg += "\n-url\t<url> \t\tthe url of the record center site";  
 msg += "\n-path\t<file mask=""> \tthe wildcard mask for the files to be submitted";  
 msg += "\n-type\t<routing type=""> \tthe routing type name";  
 return msg;  
 }

1. Next, implement the run command as follows:

Code Snippet: 'SubmitOfficialFile.Run'

int ISPStsadmCommand.Run(string command,  
 System.Collections.Specialized.StringDictionary keyValues,  
 out string output)  
 {  
 int result = 1;  
  
 string sRoutingType=string.Empty;  
 string sFileMask=string.Empty;  
 string sRepositoryUrl=string.Empty;  
 StringBuilder sbResult = new StringBuilder();  
  
 try  
 {  
 // validate the arguments  
 if (null == (sRepositoryUrl = keyValues["url"])  
 || sRepositoryUrl.Length == 0)  
 throw new ApplicationException("No url specified.");  
  
 if (null == (sFileMask = keyValues["path"]) || sFileMask.Length == 0)  
 throw new ApplicationException("No files specified.");  
  
 if (null == (sRoutingType = keyValues["type"]))  
 {  
 sbResult.AppendLine("No routing type specified - using default.");  
 sRoutingType = string.Empty;  
 }  
  
 // open the website  
 using (SPSite site = new SPSite(keyValues["url"]))  
 {  
 using (SPWeb web = site.OpenWeb())  
 {  
 OfficialFile.RecordsRepositoryProperty[] props =  
 new OfficialFile.RecordsRepositoryProperty[1];  
  
 // Process each file specified.  
 foreach (FileInfo info in ExpandWildcards(sFileMask))  
 {  
 string strResultInfo="";  
 sbResult.AppendLine(info.Name);  
  
 OfficialFile.OfficialFileResult rDisposition =  
 OfficialFile.OfficialFileCore.SubmitFile(web,  
 File.ReadAllBytes(info.FullName),  
 props, sRoutingType, info.FullName,  
 out strResultInfo);  
  
 sbResult.AppendFormat("--&gt; {0}\n", rDisposition.ToString());  
 }  
 }  
 }  
 output = sbResult.ToString();  
 result = 0;  
 }  
 catch (Exception x)  
 {  
 output = x.ToString();  
 }  
 return result;  
 }  
  
 /// <summary>  
 /// Reduces a wildcard mask to an array of FileInfo objects.  
 /// </summary>  
 static FileInfo[] ExpandWildcards(string mask)  
 {  
 string folderName = Path.GetDirectoryName(mask);  
 if (null == folderName || folderName.Length == 0)  
 {  
 folderName = Environment.CurrentDirectory;  
 }  
 DirectoryInfo folder = new DirectoryInfo(Path.GetFullPath(folderName));  
 return folder.GetFiles(Path.GetFileName(mask));  
 }

1. Save the file.
2. Now you need to go back and edit the STSADM **command definition file** you created earlier. Open the **stsadmcommands.records.xml** file for editing. Add the following XML code above the closing </commands> tag.

XML Snippet: 'SubmitFiles STSADM Command Definition'

PublicKeyTokenReplace with the actual public key token for the assembly  
 <command name="submitfiles"  
 class="ECM401.OfficialFileSTSADMCommand.SubmitOfficialFile, ECM401.OfficialFileSTSADMCommand, Version=1.0.0.0, Culture=neutral, PublicKeyToken=$PublicKeyToken$"/>

1. Save the file and rebuild the project.
2. Open a command prompt and type the following command:stsadm –help submitfiles
3. You should see the following screen:
4. Now submit some files using the command line. The following screen shows the result of sending two new contracts to the record center:
5. Open the browser and navigate to the record center site to verify the results.

**This concludes the lab exercises.**