Lab 13: Implementing and Customizing SharePoint Search

**Lab Time:** 60 minutes

**Lab Overview:** In this lab you will get your first experience to Office SharePoint Server 2007 search capabilities. First you'll setup and configure search to index your site once it has been filled with some content. Then you'll see how easy it is to customize the OOTB search results page as well as adding additional fields to the search results and even introducing pivots for search refinements. Lastly, you'll learn how to interact with SharePoint search via the robust and powerful API when you have more complex business requirements that could be solved using search.

Exercise 1: Configuring Office SharePoint Server 2007 Search

In this first exercise you will configure SharePoint search to index and query your existing Publishing site. First you'll need to add some content.

1. Before configuring search you need content in the site. For the best results, add two types of content. First, upload documents to the root site collection's **Documents** library.
2. Select **Site Actions** » **Manage Content & Structure**. Then use the ECB menu to open the **Documents** library in a new window, making it easier to upload multiple files.  
     
   *Before uploading files, you might want to make it easier on yourself by changing the versioning settings of the list (from* ***Settings*** *»* ***Document Library Settings*** *» Version Settings. Set the list to* ***\*not\**** *require* ***Content Approval*** *and* ***\*not\**** *require* ***Check Out****.*
3. Upload the sample documents (all **\*.docx** files), all SharePoint 2007 technical papers, can be found in the **c:\Student\Labs\10\_Search\Resources** folder. Verify that all files are checked in & published (something you need to do manually if you didn't disable checkout & approval).

*If possible, add some content logged in as one user and then login as a different user. This will make the author pivot in a later exercise easier to see the results.*

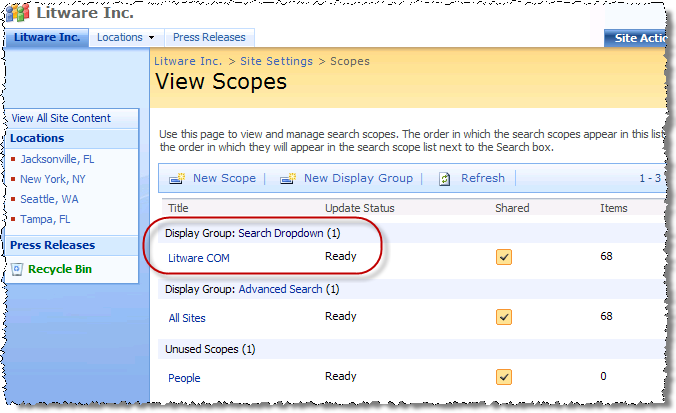
1. Now add some HTML content. Create a new subsite named **Locations** and add four new pages using the locations saved as **\*.MHT** files in the **c:\Student\Labs\10\_Search\Resources** folder.

At this point the site has four (4) content pages in the **Locations** subsite and ten (10) documents in the **Documents** list in the root of the site collection. Now it is time to configure search.

1. Open **Central Administration** and select the **Litware SSP** in the left-hand column to administer the search settings as search is tied to a specific SSP.
2. From the **Litware SSP** administration page, pick **Search Settings** under the **Search** section.
3. First configure the content source. While the default content source **Local Office SharePoint Server Sites** will crawl & index our new content, it also indexes all the site collections in all Web Applications this SSP is configured with. Instead, create a new content source.  
     
   From the **Configure Search Settings** page, select **Content sources and crawl rules**.
4. Before creating the new content source, you first need to remove the **http://wcm.litwareinc.com** URL from the **Local Office SharePoint Server Sites** content source. Do this by selecting the content source and deleting the URL from the **Start Address** field.
5. From the **Manage Content Sources** page, select **New Content Source** and use the following information to create a new content source:
   * **Name:** LitwareCOM
   * **Content Source Type:** SharePoint Sites
   * **Start Address:** http://wcm.litwareinc.com
6. With the content source created, kick off a **Full Crawl** from the **Litware COM** ECB menu.
7. Now create a new search scope that will include only our content from the http://wcm.litwareinc.com site. From the **Configure Search Settings** page, select **View scopes** from the **Scopes** section.
8. From the **View Scopes** page, select **New Scope** and give it a name of **Litware COM**.
9. Next, from the **View Scopes** page, select **Add Rule** on the **Litware COM** scope. Use the following details to create the new rule:
   * **Scope Rule Type:** Web Address
   * **Web Address:** Domain/Subdomain & wcm.litwareinc.com
   * **Behavior:** Include
10. With the rule created you can either wait for the scopes to get updated, or go back to the **Configure Search Settings** page and click **Start** **Update Now** in the **Scopes** section. In a few seconds, the scopes will be updated and upon viewing them you should see a bunch of items in the scope.

At this point the search crawler and scope have been setup. Now it's time to configure the Litware Publishing site to use the new scope.

1. Navigate to the Litware Publishing site (http://wcm.litwareinc.com) and **select Site Actions** » **Site Settings** » **Modify All Site Settings**. Next select the **Search scopes** link under the **Site Collection Administration** link.
2. Change the default search by selecting the **Search Dropdown** display group, uncheck all options except the Litware COM content source as shown in the following figure:



1. Lastly, test the search configuration. Go back to the site and enter **Seattle** in the search box in the upper right corner of the user interface to see the results.

The next thing to do is to customize the search results.

Exercise 2: Customizing the Out-of-the-Box Search Results by Adding Custom Formatting and Refinement Pivots

In this exercise you will take the out of the box (OOTB) search results and customize them to suit your business needs.

1. One common request to customize the search results is to mimic a feature the top Internet search engines have: apply specific formatting to the words matching the keyword query. To do this, execute a search for **SharePoint**.
2. Next, switch the page into edit mode by selecting **Site Actions** » **Edit Page**.
3. Select the **Edit** menu on the **Search Core Results Web Part** and select **Modify Shared Web Part**.
4. Click the **XSL Editor** button to display the OOTB XSL for the Web Part. Copy all of the XSL to the clipboard.
5. Launch **SharePoint Designer 2007**, open the **http://wcm.litwareinc.com** Publishing site, create a new file in the **/Style Library/XSL Style Sheets** folder named **defaultresults.xsl** and paste the XSL from the previous step into the file. Save the file, check it in and publish a major version.
6. Go back to the search results page in the browser, still in edit mode, and enter the path **/Style Library/XSL Style Sheets/defaultresults.xsl** into the **XSL Link** property in the **Miscellaneous** section. Click **OK** to save your changes. The page should show the same results as the XSL has not changed, just the location has changed.
7. Now, go back to **SharePoint Designer 2007**, check out the **/Style Library/XSL Style Sheets/defaultresults.xsl**.
8. Find the markup that looks like the following, around **line 183**:

<xsl:template match="c0">

<b><xsl:value-of select="." /></b>

</xsl>

1. Change the markup to the following to add yellow highlighting to the search results:

<xsl:template match="c0">

<span style="background-color:yellow; font-weight:bold"><xsl:value-of select="." /></span>

</xsl>

1. Save the changes and refresh the search results page.

Now add a extra functionality to the search results. Give the users the ability to filter by a specific author.

1. In **SharePoint Designer 2007**, open the **/Style Library/XSL Style Sheets/defaultresults.xsl** file and check it out if it isn't already.
2. First create the XSL template that will add the filter text to each search result hit. Add the following markup to the end of the XSL file just before the closing **</xsl:stylesheet>** element:

<xsl:template name="SearchAuthor">

<xsl:param name="str" />

<xsl:if test='string-length($str) &gt; 0'>

- <a title="Filter by author" href="javascript:window.location='?k=' +getParameter(window.location.search, 'k') +' +author:{$str}'"><xsl:value-of select="$str" /></a>

</xsl:if>

</xsl:template>

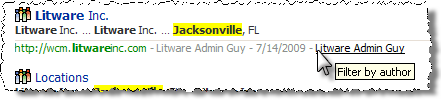
1. Next, add the following markup to call the new template to the **Results** template, just before the closing **</p>** element near **line 171**:

<xsl:call-template name="SearchAuthor">

<xsl:with-param name="str" select="author" />

</xsl:call-template>

1. Save your changes and refresh the page to see the new author pivot. Upon clicking the author name, the page will refresh and the query will have been modified:



Exercise 3: Working with the Office SharePoint Server 2007 Search API

In this exercise you'll learn how to use the SharePoint API to leverage SharePoint search programmatically.

1. Open **Visual Studio** and create a new **C# Console Application** project named **Lab1**. You should create the project within the following path so the project files will reside within the same directory structure as all the other lab exercises:

c:\Student\Labs\13\_Search\Lab

1. In this lab you will be working with common SharePoint objects as well as objects specific to the Microsoft.Office.Server namespace. In order to do this, you will need to add two assembly references to the project: **Windows SharePoint Services (Microsoft.SharePoint.dll)** which you will find under the **.NET** tab in the **Add Reference** dialog, **Microsoft.Office.Server.dll** and **Microsoft.Office.Server.Search.dll** (no component name specified). To add the **these two Microsoft.Office.\*** references you will need to pick the file from the following location on the file system using the **Browse** tab in the **Add Reference** dialog: **c:\Program Files\Common Files\Microsoft Shared\web server extensions\12\ISAPI**.
2. Make your life a little easier by adding the following using statements to the top of the **Program.cs** file:

using Microsoft.SharePoint;

using Microsoft.Office.Server.Search.Query;

using System.Data;

1. Next, add the following using statement to get the context of our site collection:

static void Main(string[] args)

{

using (SPSite siteCollection = new SPSite("http://wcm.litwareinc.com"))

{

}

}

1. Within the previous using statement, add another to create and configure the FullTextSqlQuery search object:

using (FullTextSqlQuery ftQuery = new FullTextSqlQuery(siteCollection))

{

ftQuery.StartRow = 0;

ftQuery.RowLimit = 10;

ftQuery.TrimDuplicates = true;

ftQuery.ResultTypes = ResultType.RelevantResults;

}

1. Now add the following code to enter the query that will search for a specific term using a specific scope and then sort the results in reverse order just as any search results page would:

string queryString = "SELECT title, path, rank FROM Scope() WHERE (\"scope\" ='Litware COM') AND FREETEXT(defaultproperties,'Jacksonville') ORDER BY rank desc";

ftQuery.QueryText = queryString;

ResultTableCollection results = ftQuery.Execute();

ResultTable relevantResults = results[ResultType.RelevantResults];

1. The relevantResults object now contains a collection of all the results. The next step is to convert these results to an ADO.NET DataTable object and write out the results:

DataTable resultTable = new DataTable();

resultTable.Load(relevantResults, LoadOption.OverwriteChanges);

foreach (DataRow row in resultTable.Rows)

{

Console.WriteLine("title:" + row["title"].ToString());

Console.WriteLine("path:" + row["path"].ToString());

Console.WriteLine();

}

Console.ReadLine();

1. Press F5 to run the code and you should see the same results you get in the browser!