



An Article on Custom Enterprise Search in
MOSS 2007
[By: MOSS Team METAOPTION]



METAOPTION LLC, 574 NEWARK AVENUE, SUITE 210
JERSEY CITY, NJ 07306; PH: 201-377-3150
EMAIL: info@metaoption.com ; WEB SITE: www.metaoption.com



I have used Microsoft Visual Studio 2005 to create a Web Control Library that contains a Web Part that uses the Enterprise Author Search functionality included with Office Share Point Server 2007. The sample Web Part accepts a search term via its user interface and then performs a query that searches for results in which the **Author** property matches the search term. Creating the Web Part requires four major steps. They are:

Step 1.

1. Creating a Web Control Library project in Visual Studio 2005.
2. Adding references to the required assemblies to the Visual Studio project.
3. Adding code that implements the Enterprise Search functionality.
4. Deploying the Web Part to an Office Share Point Server 2007 site.

Creating a Web Control Library Project in Visual Studio

First, you create a Web Control Library project in Visual Studio.

To create a Web Control Library project in Visual Studio

1. Start Visual Studio 2005.
2. On the **File** menu, point to **New**, and then click **Project**.
3. In the **New Project** dialog box, in the **Project Types** pane, expand **Visual C#** or **Visual Basic**, and then select **Windows**.
4. In the **Templates** pane, select **Web Control Library**.
5. Type **AuthorCustomSearchWebParts** for the name of the project.
6. Specify a location for the project, and then click **OK**.

Visual Studio generates a Web Control Library project that contains a single source file named **WebCustomControl1.cs** or **WebCustomControl1.vb**, depending on the language you selected in

Step 2.

In Solution Explorer, right-click **WebCustomControl1.cs** or **WebCustomControl1.vb** source file, and then click **Rename**. Name the file **AuthorSearchWebPart.cs** or **AuthorSearchWebPart.vb**, depending on the language you are using.

Adding References to the Required Assemblies

The sample code in this article uses classes provided with the **Microsoft.Office.Server.Search.Query** namespace, and the **Microsoft.Office.Server.Search** or **Microsoft.SharePoint** namespace. You must add references to necessary assemblies to the project so that you can use these objects.

To add references to the required assemblies

- If you are running Visual Studio on a server that has Office Share Point Server 2007 installed:





1. In Solution Explorer, right-click the **AuthorCustomSearchWebParts** project, and then click **Add Reference**.
2. On the **.NET** tab of the **Add Reference** dialog box, press and hold the CTRL key, and select the following components:
Microsoft Office Server ([Microsoft.Office.Server.dll](#))
Microsoft Office Share Point Server Search ([Microsoft.Office.Server.Search.dll](#))
Windows Share Point Services ([Microsoft.SharePoint.dll](#))
3. Click **OK** to add the references to the project.

If you are running Visual Studio on a computer that does not have Office Share Point Server 2007 installed, the necessary assemblies are not available. In this case:

4. Copy the [Microsoft.Office.Server.dll](#) file, the [Microsoft.Office.Server.Search.dll](#) file, and the [Microsoft.SharePoint.dll](#) file from a server running Office Share Point Server 2007 to a local project folder on the development computer. The following is the default assembly location on a computer that is running Office Share Point Server 2007:
%ProgramFiles%\Common Files\Microsoft Shared\Web Server Extensions\12\ISAPI
5. In Solution Explorer, right-click the **AuthorCustomSearchWebParts** project, and then click **Add Reference**.
6. In the **Add Reference** dialog box, on the **Browse** tab, navigate to the local folder that contains the copies of the assembly files.
7. Press and hold the CTRL key, and select [Microsoft.Office.Server.dll](#), [Microsoft.Office.Server.Search.dll](#), and [Microsoft.SharePoint.dll](#).
8. Click **OK** to add the references to the project.

Step 3.

The sample code in this article also uses classes defined in the [System.Data](#) namespace and the [System.XML](#) namespace. You must add the necessary references to the Visual Studio project so that you can use these classes.

To add references to System.Data and System.XML

1. In Solution Explorer, right-click the **AuthorCustomSearchWebParts** project, and then click **Add Reference**.
2. On the **.NET** tab of the **Add Reference** dialog box, press and hold the CTRL key, and select the following components:
System.Data ([System.Data.dll](#))
System.Xml ([System.XML.dll](#))





3. Click **OK** to add the references to the project.

Implementing the Author Search Web Part

Functionality

After you add the references to the required assemblies, you add the code that provides the ability for the Web Part to perform a search for items where the **Author** property matches a search term.

To implement the Author search Web Part functionality

1. Add the following **Imports** or **using** statements to the top of the `AuthorSearchWebPart.cs` source file. Add the statements after the **Imports** or **using** statements that Visual Studio generated when you created the project.

Below are code sample that I have used in C#

```
using System.Drawing;
using System.Data;
using System.Xml.Serialization;
using Microsoft.SharePoint.WebPartPages;
using Microsoft.Office.Server;
using Microsoft.Office.Server.Search.Query;
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;
using System.Web.UI;
using System.Web.UI.WebControls;
```

1. The **Imports** and **using** statements enable you to use the classes and types defined in the referenced namespaces without needing to use fully qualified namespace paths.
2. Replace the version of the `AuthorSearchWebPart` class definition that Visual Studio generated with the following code.

```
[XmlRoot(Namespace = "CustomSearch")]
public class AuthorSearchWebPart : WebPart
{
    Button cmdSearch;
    TextBox txtQueryText;
    Label lblQueryResult;
    DataGrid grdSearchResults;

    protected override void CreateChildControls()
    {
        Controls.Clear();

        txtQueryText = new TextBox();
        this.Controls.Add(txtQueryText);

        cmdSearch = new Button();
        cmdSearch.Text = "Search";
        cmdSearch.Click += new EventHandler(cmdSearch_Click);
        this.Controls.Add(cmdSearch);

        lblQueryResult = new Label();
        this.Controls.Add(lblQueryResult);
    }

    void cmdSearch_Click(object sender, EventArgs e)
    {
        // Search logic
    }
}
```





```
{
    if (txtQueryText.Text != string.Empty)
    {
        keywordQueryExecute(txtQueryText.Text);
    }
    else
    {
        lblQueryResult.Text = "Search parameters are required." ;
    }
}

private void keywordQueryExecute(string strQueryText)
{
    KeywordQuery kRequest = new KeywordQuery(ServerContext.Current)
    {
        QueryText = "author:" + strQueryText;
    };
    // configure the query to return relevant results.
    kRequest.ResultTypes |= ResultType.RelevantResults;
    // execute the query
    ResultTableCollection resultTbls = kRequest.Execute();

    if ((int)ResultType.RelevantResults != 0)
    {
        ResultTable tblResult = resultTbls[ResultType.RelevantResults];

        if (tblResult.TotalRows == 0)
        {
            lblQueryResult.Text = "<br>No Search Results Returned." ;
        }
        else
        {
            ReadResultTable(tblResult);
        }
    }
}

void ReadResultTable(ResultTable rt)
{
    DataTable relResultsTbl = new DataTable();
    relResultsTbl.TableName = "Relevant Results" ;
    DataSet ds = new DataSet("resultsset");
    ds.Tables.Add(relResultsTbl);
    ds.Load(rt, LoadOption.OverwriteChanges, relResultsTbl);

    // Add the results to the DataGrid.
    fillResultsGrid(ds);
}
}
```





```

private void fillResultsGrid(DataSet grdDs)
{
    // Create an instance of the DataGrid and set its
    // DataSource property to the supplied DataSet.
    grdSearchResults = new DataGrid();
    grdSearchResults.DataSource =
    grdDs;

    // Set the display properties for the DataGrid.
    grdSearchResults.GridLines = GridLines.None;
    grdSearchResults.CellPadding = 4;
    grdSearchResults.Width = Unit.Percentage(100);
    grdSearchResults.ItemStyle.ForeColor = Color.Red;
    grdSearchResults.ItemStyle.BackColor = Color.AliceBlue;
    grdSearchResults.ItemStyle.Font.Size =
    FontUnit.Small;
    grdSearchResults.ItemStyle.Font.Name = "Courier New" ;
    grdSearchResults.HeaderStyle.BackColor = Color.Navy;
    grdSearchResults.HeaderStyle.ForeColor = Color.Green;
    grdSearchResults.HeaderStyle.Font.Bold = true ;
    grdSearchResults.HeaderStyle.Font.Name = "Courier New" ;
    grdSearchResults.HeaderStyle.Font.Size =
    FontUnit.Medium;

    // Turn off AutoGenerate for the columns so that the DataGrid
    // does not automatically bind to all of the columns in the
    // search results set. Then create and configure the DataGrid
    // to display only the desired columns.

    grdSearchResults.AutoGenerateColumns = false ;
    HyperLinkColumn colTitle = new HyperLinkColumn();
    colTitle.DataTextField = "Title" ;
    colTitle.HeaderText = "Title" ;
    colTitle.DataNavigateUrlField = "Path" ;
    grdSearchResults.Columns.Add(colTitle);
    BoundColumn colAuthor = new BoundColumn();
    colAuthor.DataField = "Author" ;
    colAuthor.HeaderText = "Author" ;
    grdSearchResults.Columns.Add(colAuthor);

    // Bind the data to the DataGrid.
    grdSearchResults.DataBind();

    //Add the DataGrid to the controls.
    Controls.Add(grdSearchResults);
}
}

```

3. Build the **CustomSearchWebParts** project.

Step 4.

Deploying the Web Part to Share Point Server 2007

Use the following procedure to deploy the sample Web Part to an Office Share Point Server site.





To deploy the sample Web Part to an Office Share Point Server site

1. Determine the path to the **_app_bin** folder for the Office Share Point Server site:
 - a. In Internet Information Services (IIS), in the IIS Manager console, expand the **Sites** node. **Web**
 - b. Right-click the application for the site, and then click **Properties**.
 - c. On the **Home Directory** tab, the **Local path** displays the path to the application.
2. Copy the **CustomSearchWebParts.dll** assembly file to the **_app_bin** folder of the Office Share Point Server site.
3. Register the sample Web Part as a safe control:
 - a. Open the **web.config** file for the site that you are adding the Web Part to.
The **web.config** file is in the root folder for the site.
 - b. Add the following `<SafeControl/` tag to the `<SafeControls></SafeControls` section of the **web.config** file.

Example: Add in web.config

```
<SafeControl Assembly="CustomSearchWebParts,
Version=1.0.0.0,
Culture=Neutral,
PublicKeyToken=null"
Namespace="CustomSearchWebParts"
Type="CustomSearchWebParts" />
```

1. Restart IIS.
2. Open a **Command Prompt** window on the server. At the command prompt, type **iisreset**.
3. Create a Web Part definition file for the sample Web Part:
 - a. Open a new file in a text editor such as Notepad, and then add the following XML code to the file.


```
<?xml version="1.0"? >
<WebPart xmlns="http://schemas.microsoft.com/WebPart/v2" >
  <Assembly >CustomSearchWebParts, Version=1.0.0.0,
  Culture=Neutral,
  PublicKeyToken=nul </ Assembly >
  <TypeName >CustomSearchWebParts.AuthorSearchWebPart </ TypeName >
  <Title >Custom Author Search Web </ Title >
  <Description >A custom author search web </ Description >
</ WebPart >
```
 - b. Now, name the file **CustomAuthorSearchWebPart.dwp**, and then save it to a folder that you can access from Office Share Point Server.
4. Here, I'm going to show how to add this custom control in MOSS site





- a. On the **Site Actions** menu, click **Site Settings**, and then click **Modify All Site Settings**.
- b. On the **Site Settings** page, in the **Galleries** section, click **Web Parts**.
- c. On the **Web Part Gallery** page, click **Upload**.
- d. On the **Upload Web Part: Web Part Gallery** page, browse to and select the **CustomAuthorSearchWebPart.dwp** file that you created previously. Select **Overwrite existing file(s)**, and then click **OK**.
The **CustomAuthorSearchWebPart.dwp** file provisions the **CustomSearchWebPart** page. **Web Part Gallery:**
- e. In the **Group** section, in the drop-down list, select **Default Web Parts**.
- f. If you want to include the Web Part in the **Quick Groups** list, select **Add Quick Groups**. If you want to leave the custom Web Part out of the **Default** list, clear the check boxes. Click **OK**.

After you deploy the Web Part to the Office Share Point Server site, you should test it.

To test the deployed Web Part

1. On the Search Center site, click **Site Actions**, and then click **Create Page**.
2. Type **Test Page for Custom Author Search Web Part** for the **Title**.
3. In the **Page Layout** list, click **(Welcome Page) Search Page**.
4. To create the page, click **Create**.
5. With **TestPageforCustomAuthorSearchWebPart.aspx** open in the browser, in the Top Zone, click **Add a Web Part**.
6. In the **Add Web Parts to Top Zone Custom Search Web Part** dialog box, in the **Default Web Parts** category, select **Custom Search Author**.
7. Click **Add** to add the Web Part to the zone.
8. To return those items that were authored by the specified user, in the **Web Part**, type a user name in the text box control, and then click **Start**.

You will get the all matched user name.

