

Creating a Web Part That Uses Word Automation Services in SharePoint 2010

(Microsoft Corporation)

This walkthrough shows how to initiate Word Automation Services document conversions using a SharePoint Server 2010 Web Part. During the walkthrough, you learn the following:

- Set up a Visual Studio 2010 project for Word Automation Services development.
- Create a SharePoint Server 2010 Web Part.
- Create and configure the settings for a Word Automation Services conversion job.
- Deploy the Web Part.

Prerequisites

To complete this walkthrough, you must have the following:

- A working installation of SharePoint Server 2010 Enterprise or Standard edition.
- Microsoft Visual Studio 2010 installed on one of the SharePoint Server 2010 Web front-end servers in the SharePoint Server 2010 farm.
- Appropriate permissions. (You must have the permissions to make the SharePoint Server 2010 Web Part that you create available sites on the farm).

Step 1: Set Up the Visual Studio 2010 Project For the Web Part Solution

This topic shows how to set up for the Word Automation Services Web Part example solution.

Setting Up For the Word Automation Services Web Part Solution Example

The custom application that you create in this walkthrough takes a given set of files from the specified SharePoint Document Library as input, uses Word Automation Services to convert the files to a specified output format, and then places the converted files into the output SharePoint Document Library.

In the following procedure, you create SharePoint Document Libraries for the input and output of a Word Automation Services conversion job.

To create the input and output SharePoint document libraries

1. Navigate to a site on your SharePoint Server 2010 server for which you have permissions to create a document library.
2. Click **Site Actions** and then click **New Document Library**.
3. Name the document library Input.
4. Click **Create**. This document library is the file input location for the Word Automation Services conversion job. For example, *http://wordserver/demo/input/*.
5. Click **Site Actions** and then click **New Document Library**.
6. Name the document library Output.
7. Click **Create**. This document library is the file output location for the Word Automation Services conversion job. For example, *http://wordserver/demo/output/*.

This walkthrough shows how to initiate a Word Automation Services conversion job using a Web Part. To create the Web Part, use the Visual Studio 2010 Visual Web Part template.

Creating a Web Part That Uses Word Automation Services in SharePoint 2010

(Microsoft Corporation)

To create the Visual Studio 2010 Visual Web Part project

1. Start Visual Studio 2010 as an administrator.

Important: You must start Visual Studio 2010 as an administrator in order to successfully publish the project to the SharePoint 2010 farm.

2. Click **File**, point to **New**, and then click **Project...**
3. In the **New Project** dialog box, under **Installed Templates**, expand **Visual C#** (if it is not already expanded), expand **SharePoint**, and then click **2010**.
4. In the list of template types, select **Visual Web Part**.
5. Give the project a name, for example, ConvertDocuments.
6. Click **OK** to begin creating the project.
7. On the **Specify the site and security level for debugging** page of the **SharePoint Customization Wizard** verify the SharePoint site for the solution.
8. Also on the **Specify the site and security level for debugging** page of the **SharePoint Customization Wizard**, ensure that **Deploy as a farm solution** is selected.

Important: You must select the option **Deploy as a farm solution** because the Word Automation Services object model cannot be accessed from partially-trusted code.

9. Click **Finish** to complete creating the project.

To use the Word Automation Services object model in code, add a reference to the Word Automation Services assembly.

To add a reference to Word Automation Services

1. In **Solution Explorer**, right-click **References** and then click **Add Reference...**
2. In the **Add Reference** dialog box, click the **Browse** tab. The Microsoft.Office.Word.Server.dll assembly is located in the SharePoint Server 2010 \ISAPI folder, which usually has the following path:

```
%systemdrive%\Program Files\Common Files\Microsoft Shared\Web Server Extensions\14\ISAPI
```

3. Select **Microsoft.Office.Word.Server.dll** and then click **OK** to add the reference.

Note: If you receive a warning that Microsoft.Office.Word.Server.dll requires a newer version of the .NET Framework, click **Yes** to add the reference. Then, add a reference to System.Web.DataVisualization, which usually has the following path:

```
%windir%\Assembly\GAC_MSIL\System.Web.DataVisualization\3.5.0.0__31bf3856ad364e35\
```

Creating a Web Part That Uses Word Automation Services in SharePoint 2010

(Microsoft Corporation)

Step 2: Code the Web Part

This topic shows how to write code that initiates Word Automation Services document conversion using a SharePoint Server 2010 Web Part. An end user can convert documents in the specified SharePoint Document Library by clicking a button on a Web Part.

Write Code to Convert the Document Library

The following procedures add a button to the Web Part and then add code in the Click event for the button that initiates a Word Automation Services document conversion.

To add a button to the Web part

1. In the Visual Studio 2010 project for the Web Part solution example, in **Solution Explorer**, expand **VisualWebPart1**, right-click **VisualWebPart1UserControl.ascx** and then click **View Designer**.
2. From **Toolbox**, drag a button control to the designer surface.
3. With the button selected in the designer, in **Properties**, double-click in the **(ID)** field, type btnSubmit, and then press **ENTER**.
4. In **Properties**, scroll down to **Text**, double-click the **Text** field, type Convert Document Library, and then press **ENTER**.

To convert the documents

1. With **VisualWebPart1UserControl.ascx** open in the designer, ensure that the button on the Web Part surface is selected.
2. In **Properties**, click the button with the lightning bolt icon to move to the Events view, and then double-click in the **Click** event field. The btnSubmit_Click event handler is created in **VisualWebPart1UserControl.ascx.cs**.
3. In **VisualWebPart1UserControl.ascx.cs**, add a using directive for Word Automation Services.

```
using Microsoft.Office.Word.Server.Conversions;
```

The primary objects used to perform Word Automation Services document conversion are in the Microsoft.Office.Word.Server.Conversions namespace.

4. In the btnSubmit_Click method, add the following code.

```
ConversionJob myJob = new ConversionJob("Word Automation Services");
```

The string argument passed to the ConversionJob¹ constructor must be the name of the service application instance for Word Automation Services, as described in the topic [Configuring Word Automation Services for Development](#)². The service application instance name is visible on the **Manage Service Applications** page in **SharePoint Central Administration**.

5. Next, set properties on the ConversionJob¹ by adding the following code.

```
myJob.Settings.OutputFormat = SaveFormat.PDF;  
myJob.Settings.OutputSaveBehavior = SaveBehavior.AppendIfPossible;
```

Creating a Web Part That Uses Word Automation Services in ShaePoint 2010

(Microsoft Corporation)

The two settings in the code specify:

- The output format should be in PDF format.
- The output files should be appended as a new version to existing files when versioning is turned on, and replace those existing files otherwise.

6. Set the credentials to present when running the conversion job.

Important: This setting specifies that all reading/writing of documents should use the credentials of the user that clicks the button. By default, Word Automation Services uses the anonymous context, so it is important to set this property.

```
myJob.UserToken = SPContext.Current.Web.CurrentUser.UserToken;
```

7. Next, specify the input library that contains the files to convert and the output library for the converted files.

```
SPWeb myWebSite = SPContext.Current.Web;  
SPList inputLibrary = myWebSite.Lists["Input"];  
SPList outputLibrary = myWebSite.Lists["Output"];  
myJob.AddLibrary(inputLibrary, outputLibrary, true);
```

8. Finally, complete the code for the btnSubmit_Click method by adding code that starts the Word Automation Services conversion job.

```
myJob.Start();
```

When a user clicks the button on the Web Part, the code initiates a conversion job for the files in the input library.

Step 3: Build and Deploy the Web Part Solution

This topic shows how to deploy and use the Word Automation Services Web Part solution example.

Building and Deploying the Solution

Once the solution has been deployed, add the Web Part to a SharePoint Server page and click the button on the Web Part to convert all Microsoft Word documents in the input library to PDF files in the output library.

To build and deploy the solution

1. With the **ConvertDocuments** solution selected in **Solution Explorer** in Visual Studio 2010, click **Build** and then click **Deploy Solution**.
2. On the SharePoint Server site where the solution is deployed, navigate to the input document library created in Step 1 of this walkthrough. For example, *http://wordserver/demo/input/*.
3. Click **Add document** and then add 1 or more Word documents to the document library.

Creating a Web Part That Uses Word Automation Services in SharePoint 2010

(Microsoft Corporation)

4. Click **Site Actions** on the SharePoint Server 2010 ribbon, and then click **New Page**.
5. In the **New Page** dialog, under **New page name:**, enter a name for the custom page, for example Convert Document Library.
6. Click **Create** to add the custom page. The page opens in editing mode.
7. Under the **Editing Tools** tab group, click **Insert** and then click **Web Part**.
8. In the **Categories** list, click **Custom**. Under **Web Parts**, **VisualWebPart1** appears.
9. Select **VisualWebPart1** and then click **Add**.
10. Click the **Save & Close** button next to the **Navigate Up** button on the ribbon.
11. When the page refreshes, click **Convert Document Library** on the Web Part.
12. Navigate to the output library for the solution created in Step 1 of this walkthrough. For example, *<http://wordserver/demo/output/>*.

When the solution runs, documents in the input document library are converted to PDF. If the input library contains subfolders, they are replicated in the output library so that files are put in the same relative locations as the original files.