By Kevin Idzi, Software Development Engineer, Microsoft Corporation

Introduction

Integration Services package developers often ask about the best way to get data out of or into a SharePoint list. The SharePoint List Source and Destination Sample available on the Microsoft SQL Server Community Samples: Integration Services page on Codeplex provides an optimized solution with an easy-to-use interface. The sample also includes an API for accomplishing these tasks efficiently outside of Integration Services.

There a few different ways to extract or load SharePoint data:

- Use the SharePoint APIs to add or remove items one by one.
- Use the SharePoint Batch API to generate XML and submit the XML.
- Use the Lists Web service, which uses the same XML as the Batch API.

Calling the Web service is a powerful way to transfer data to or from SharePoint, whether or not you have extensive rights in your SharePoint environment, because it does not alter the SharePoint server itself in any way. The Web service also uses the Batch XML structure, which provides better performance than the server APIs for extracting or loading data.

Features of the SharePoint List Source and Destination

The SharePoint List source and destination use public SharePoint Web services and have several features that enhance their performance and their ease of use:

- Only the fields that you want are returned from SharePoint.
- Large lists are not transferred all at once. They are paged in batches, with a configurable batch size.
- Column type information from SharePoint is used for mapping to Integration Services data types
- CAML queries can be added to the query to filter the rows to be returned.
- Update and Delete operations from an Integration Services package are simple.
- Important custom properties of the source and destination can be set by using Integration Services expressions.
- Supports SQL Server 2005, 2008, 2008 R2, and later.
- Supports x64 and x86 Processors (choose the appropriate installer package)

Preparing to Use the SharePoint List Source and Destination

To prepare to use the SharePoint List source and destination:

1. Download the appropriate installer package from the Codeplex site: Microsoft SQL Server Community Samples: Integration Services - Release: SharePoint List Source and Destination.

2. Run the installer package. Select Complete Install to have the installer install connectors for every version of SSIS currently on the target machine.

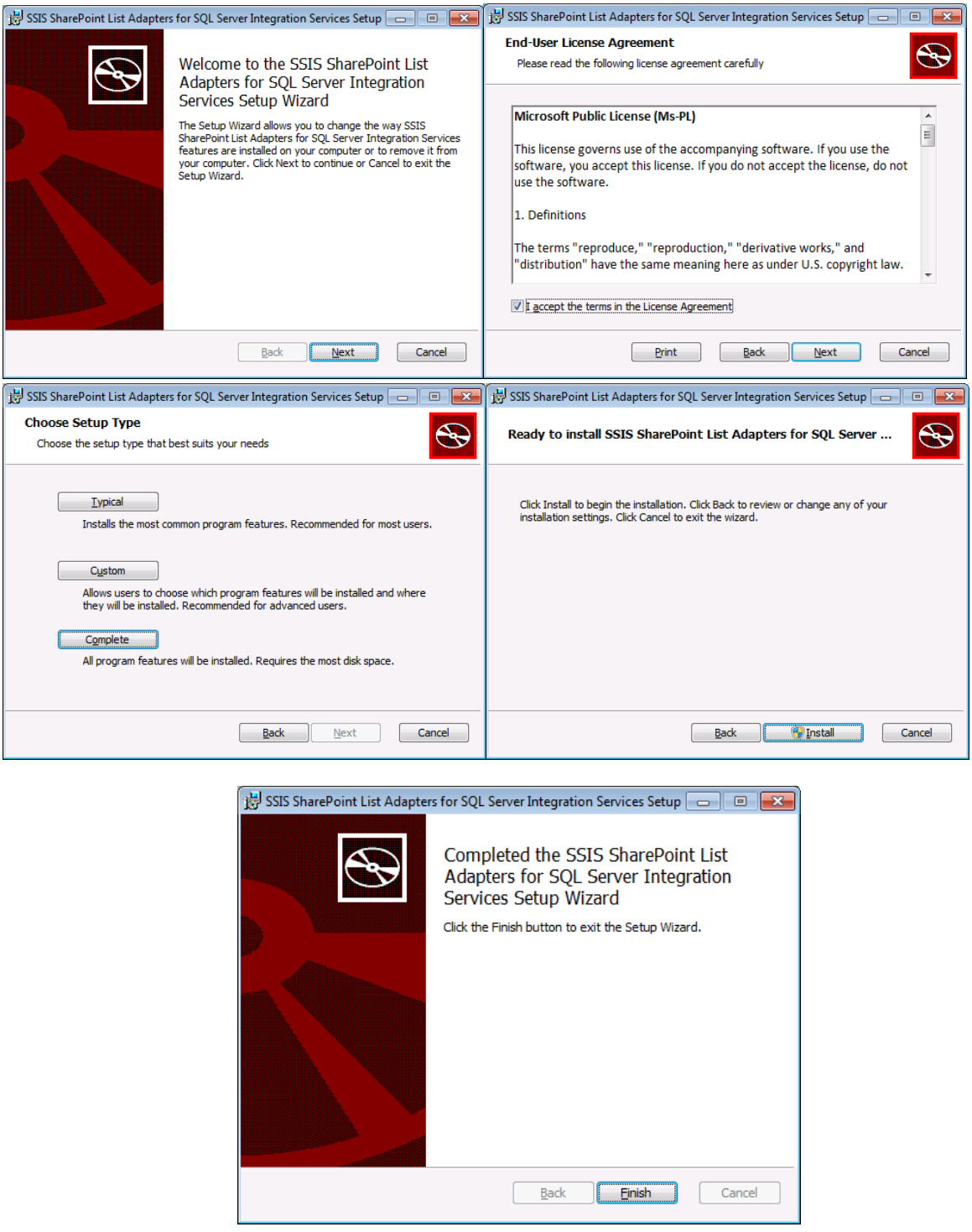


Figure 1. Pages of the Setup utility.

3. Add the source and destination to the Toolbox:

a. Open Business Intelligence Development Studio or Visual Studio.

b. Create a new Integration Services project, or open an existing one.

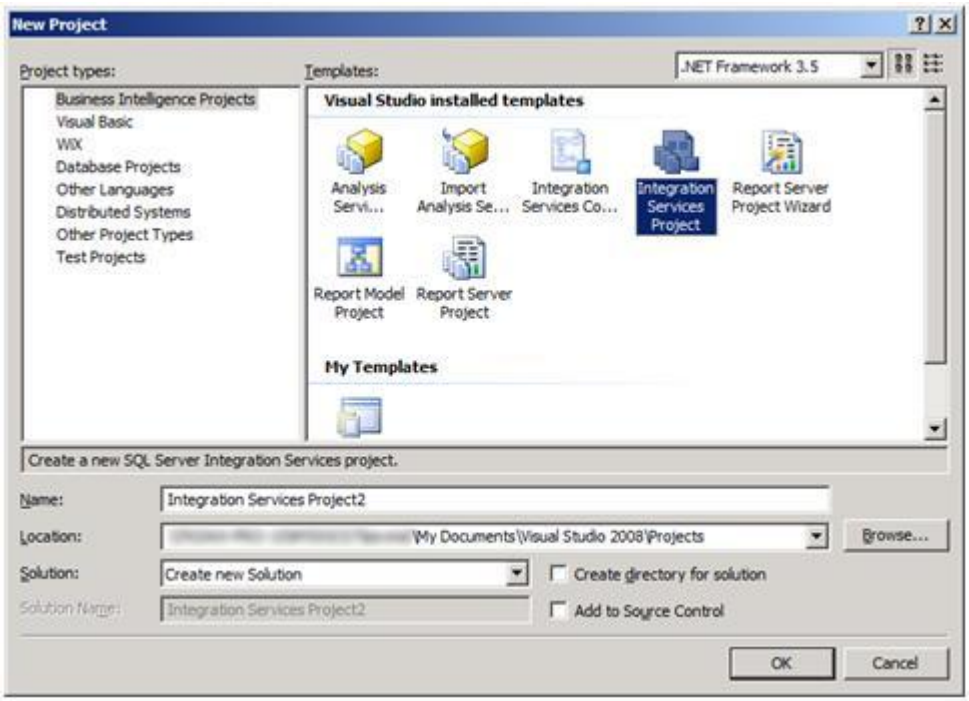


Figure 2. Creating a new Integration Services project.

c. On the **Tools** menu, select **Choose Toolbox Items**.

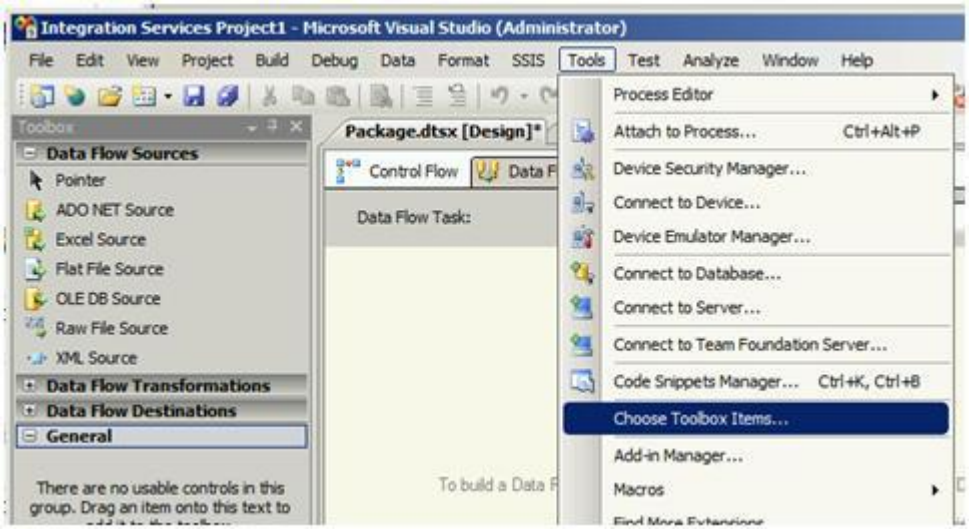


Figure 3. The **Choose Toolbox Items** option on the **Tools** menu.

d. On the **SSIS Data Flow Items** tab of the **Choose Toolbox Items** dialog box, select the **SharePoint List Source** and the **SharePoint List Destination** by checking the boxes. Then click **OK**.

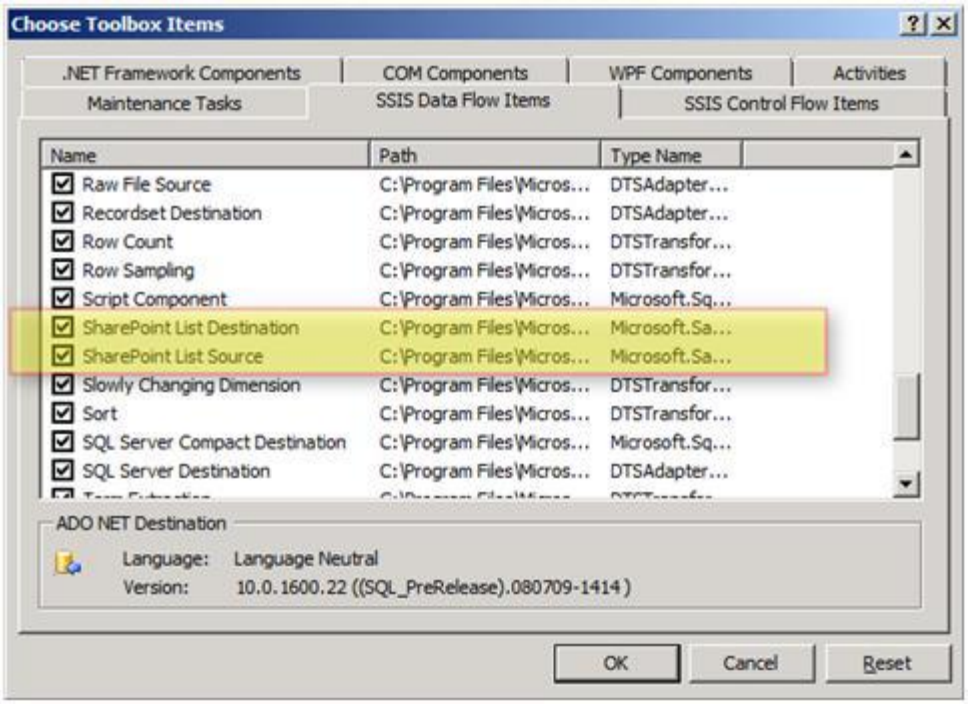


Figure 4. The **SSIS Data Flow Items** tab of the **Choose Toolbox Items** dialog box

e. If you find that the source and destination have been added to the **General** list in the toolbox, drag the source to the **Data Flow Sources** list, and drag the destination to the **Data Flow Destinations** list.

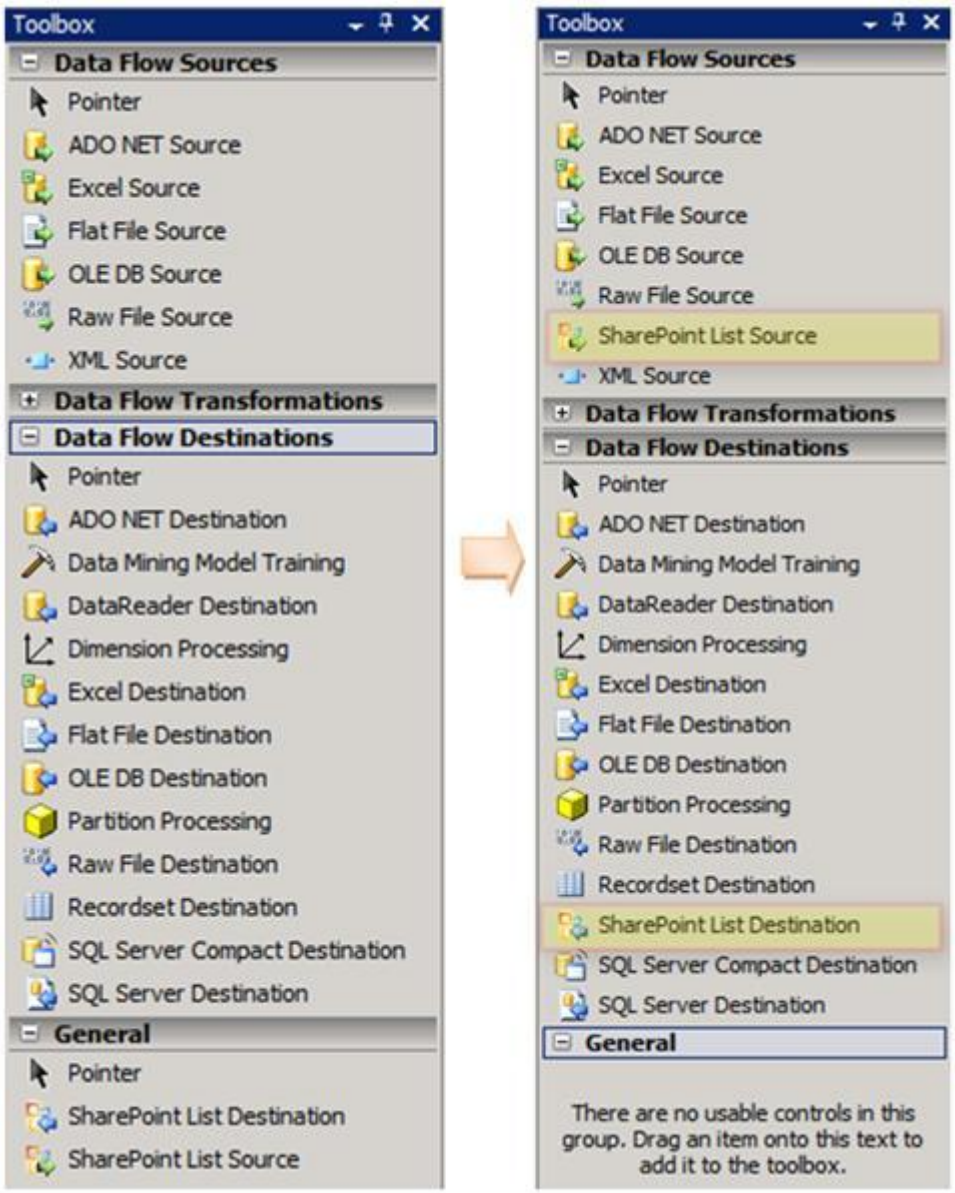


Figure 5. Moving the source and destination to the appropriate categories in the Toolbox.

Prepare your SharePoint list for testing.

Create List and Data:

Name: Test SharePoint List

Columns:

- SomeOtherColumn (Type = Single Line of Text, 255 Chars)
- Number Data (Type = Number)
- Date Data (Type = Date & Time, Format = Date & Time)

- YesNo Data (Type = Yes/No)
- Choice Data (Type = Choice, Options: First, Second, Third, Default=First)

Title	SomeOtherColumn	Number Data	Date Data	YesNo Data	Choice Data
Test Item 1 NEW		123	12/4/2011 4:20 AM	Yes	First
Test Item 2 NEW	Other Data	555,111	12/5/2011 12:00 AM	No	Second
Test Item 3 NEW				No	Third

Figure 6.1. A sample SharePoint list.

Create a Standard View:

Name: Test View

Columns: All from standard view, remove the Number Data and YesNo Data columns.

Filter: Show items when YesNo Data column is equal to Yes

Title	SomeOtherColumn	Date Data	Choice Data
Test Item 1 NEW		12/4/2011 4:20 AM	First

Figure 6.2. A sample SharePoint view showing the data.

Extracting Data from SharePoint by Using the SharePoint List Source

Add the SharePoint Connection

The SharePoint adapters use a Connection Manager to allow SharePoint lists to be accessed in different ways. By default, the credentials of the process that runs the SSIS Package will be used. Using Custom credentials can be set for the development environment by updating the editor shown below

NOTE: SSIS will NOT save the password for use outside of the designer. Read the following article for some methods to securely store the password for executing your package:

http://support.microsoft.com/kb/918760

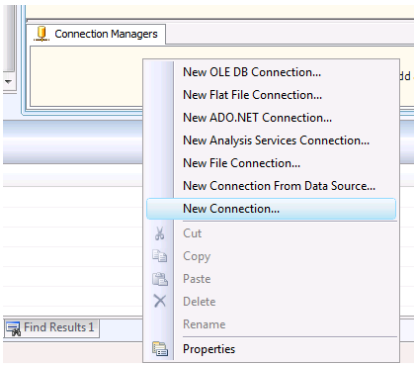


Figure 7.1. Sample showing the right-click menu from the SSIS Connection Manager

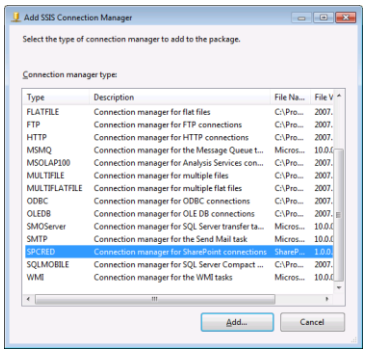


Figure 7.2. Sample showing the list of Connections

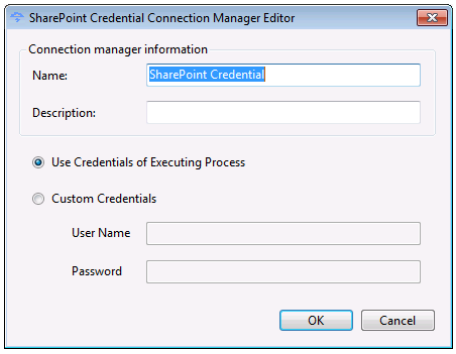


Figure 7.3. Sample showing the Connection Manager Editor

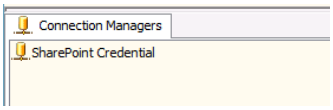


Figure 7.4. Sample showing the newly added SharePoint credential.

Add Components to SSIS Data Flow

Here is the visual representation of the SharePoint List source after you add it to the data flow of an Integration Services package.

Figure 8. The SharePoint List source in the data flow.

Setting the Properties of the SharePoint List Source

You can configure the following custom properties of the SharePoint List source in the **Advanced Editor**.

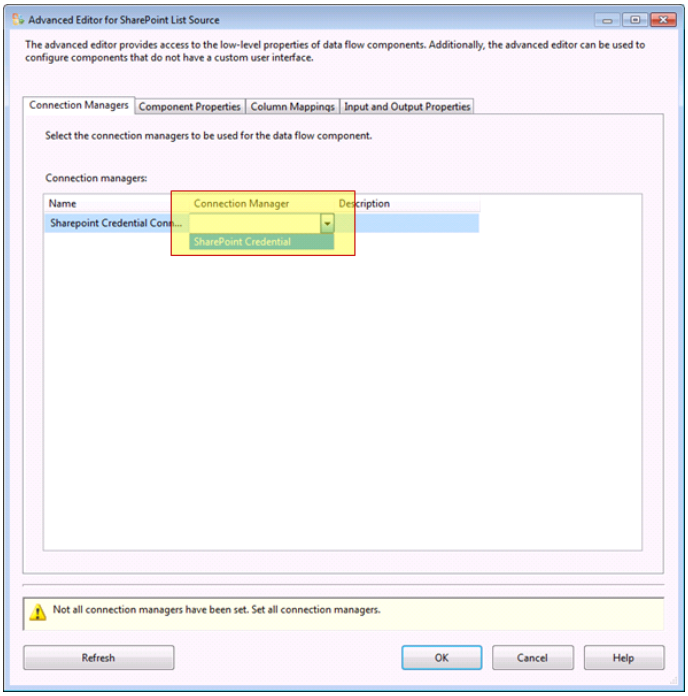


Figure 9.1. Select the Connection Manager created in previous step.

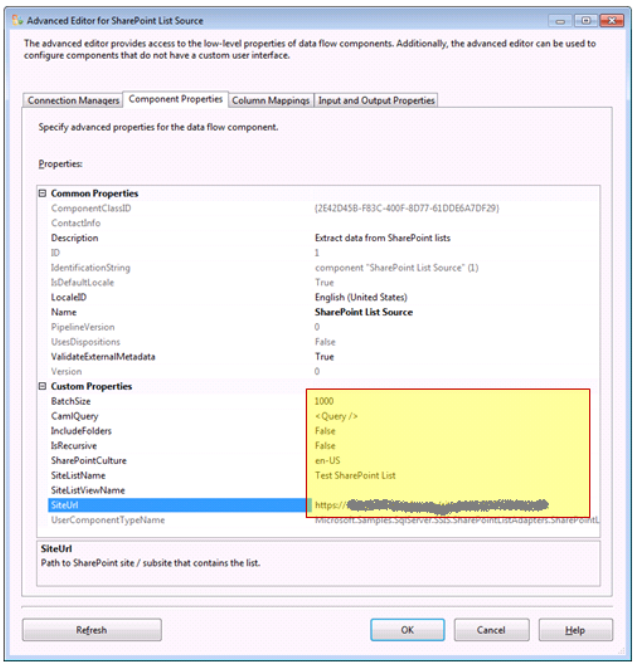


Figure 9.2. The custom properties of the SharePoint List source displayed in the **Advanced Editor**.

BatchSize

The SharePoint List source retrieves items from SharePoint in batches. Try 1,000 items at a time as a starting point – this should work well with a typical 15-column SharePoint list. You can vary this number based on the number of columns that you want to retrieve and the width of the columns.

If you only request a few small columns, then you can increase this number for better performance. If you request a lot of columns, or wide columns, then you may need to reduce the batch size.

Determining the best batch size may require some trial and error. Keep in mind that SharePoint builds and sends its response in XML format. If the XML response is too large, an exception with an uninformative message will be raised.

If you have problems with the batch size, set the batch size to match the paging size used for the SharePoint list view before you try a larger number. If your results include text fields of widely varying lengths, decide whether it's necessary to include those columns.

CamlQuery

You can use a CAML query to filter the data returned by the server. You can create a dynamic CAML query by using an Integration Services expression to set this property. This can be key to making truly incremental loading packages by loading only items created after a particular date/time, for example.

IncludeFolders

By default, folders that are found in a list are not returned. However, you can change this setting.

IsRecursive

By default, only the list items at the top level are returned. If the list contains folders that contain other items, you can change this setting to load both the items in the top-level folder, and the items in all child folders.

SharePointCulture

Used to format the dates and numeric values when SharePoint is installed using an international culture.

SiteListName

The name of the SharePoint list, as seen on the SharePoint page.

SiteListViewName

The name of the SharePoint list view from which you want to retrieve data. You can use a list view to pre-filter rows and eliminate the need to add dynamic CAML queries to your package. This does NOT affect columns, only data.

If you do not specify a list view name, then the default list view is used. The default list view is the one that appears first in the **View** dropdown list, which may not show all of the items in the list. Leaving this option blank is the same as entering the default list view, which is typically **All Items** if the default has not been changed.

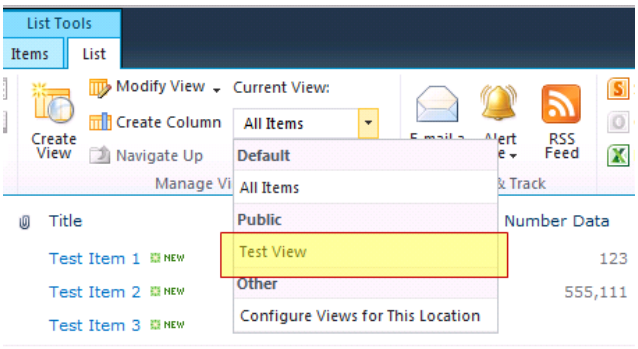


Figure 9.3. Selecting the Test View in a SharePoint list.

SiteURL

The URL for the primary site on which the list is found. Do not include any other subfolders or list paths, or the location of an .asmx file.

Selecting Columns

Selecting Columns by Using the Column Mappings Page

After you configure the custom properties for the SharePoint List source, go to the **Column Mappings** page of the **Advanced Editor** to select the columns from the list that you want to include in the data flow. To improve the performance of the extraction from SharePoint, select **Ignore** for any columns that you don't need in your data flow.

Selecting Columns by Using the Input and Output Properties Page

When you remove columns by using the **Column Mappings** page, the columns contain no data, but they still consume space in the buffers of data that pass through the data flow. To remove unwanted columns completely and optimize the use of memory for buffers, remove the columns from the **Output Columns** list on the **Input and Output Properties** page.

When a list is initially synched, SharePoint may have the column multiple times (note "Title" is found three times below for the Link Titles). Remove any extraneous fields.

New columns can be added by clicking 'Refresh'. Any removed or modifications to existing columns should remain even if refreshed.

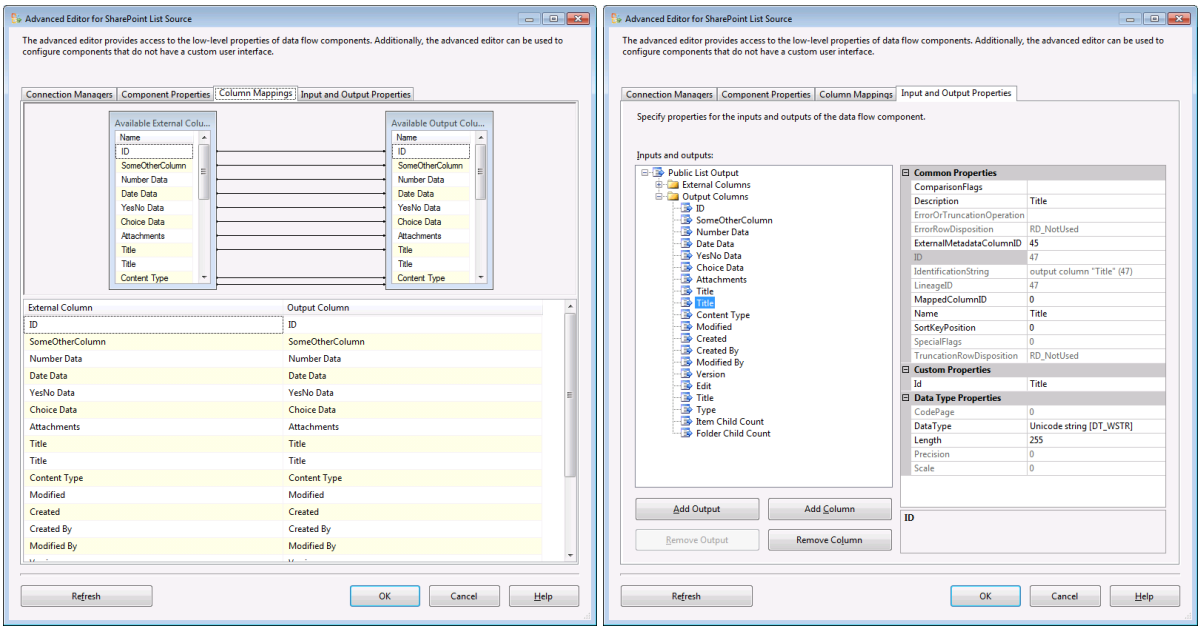


Figure 10. The list of columns on the **Column Mappings** page of the **Advanced Editor** and **Input and Output Properties Tab.**

Writing Data to SharePoint by Using the SharePoint List Destination

Here is the visual representation of the SharePoint List destination after you add it to the data flow of an Integration Services package.

Drop a SharePoint List Destination component from the toolbox onto the Dataflow. Connect a line from the Source to the Destination control before entering properties.

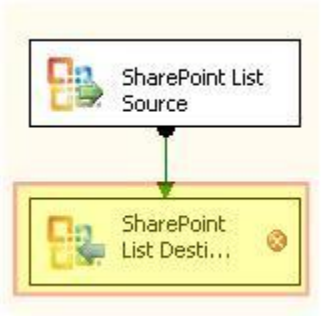


Figure 11. The SharePoint List destination in the data flow.

Setting the Properties of the SharePoint List Destination

You can configure the following custom properties of the SharePoint List destination in the **Advanced Editor**.

If using the same connection manager, select it as with the Source Component.

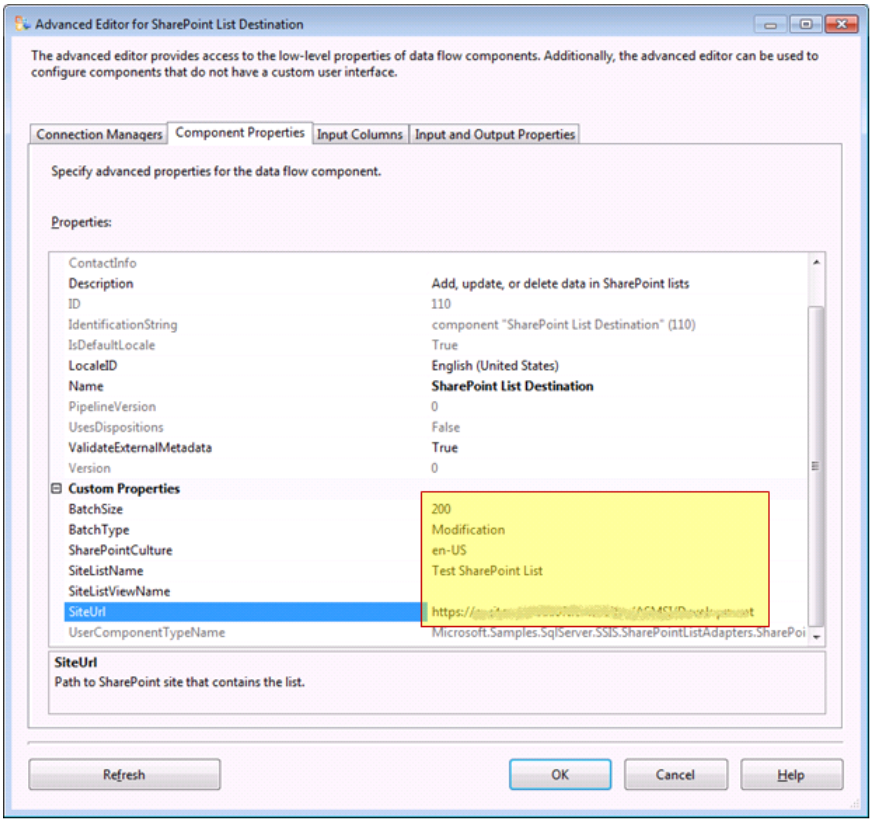


Figure 12.1. The custom properties of the SharePoint List destination displayed in the **Advanced Editor**.

BatchSize

Since updates or deletes add to the size of the XML message, you have to use a smaller batch size for updates and deletes than for retrieving list items. Try 200 updates at a time as a starting point, and adjust this value as necessary.

BatchType

The SharePoint List destination can create, update or delete rows of data.

- Create a Row (Modification Batch Type). A row is created on the server when you select this batch type and do not specify an ID value (NULL) for a given row of data. Any columns that are not included receive the default values/empty on the server.
- Update a Row (Modification Batch Type). A row is updated on the server when you select this batch type and specify the SharePoint ID of an existing row. Only the columns contained in the data flow are modified. If you specify this batch type, but the data flow does not contain any updated rows, an error is raised.
- Delete a Row (Deletion Batch Type). A row is deleted on the server when you select this batch type and specify the SharePoint ID of an existing row. If you specify this batch type, but the data flow does not contain any deleted rows, an error is raised.

SharePointCulture

Used to format the dates and numeric values when SharePoint is installed using an international culture.

SiteListName

This property behaves the same as the property of the same name for the SharePoint List source.

SiteListViewName

This property behaves the same as the property of the same name for the SharePoint List source. If a row being modified or deleted is present in the view, then the action will not occur and will cause the data flow to raise an error.

SiteURL

This property behaves the same as the property of the same name for the SharePoint List source.

Selecting Input Columns

After you configure the custom properties for the SharePoint List destination, go to the **Input Columns** page of the **Advanced Editor** to select the columns from the data flow that you want to load into SharePoint. Any columns that exist on the server but not in the data flow will be ignored during the loading of data.

You may need to click “Refresh” at the bottom of this tab to load in the target columns.

Only Editable columns will be available from SharePoint for mapping.

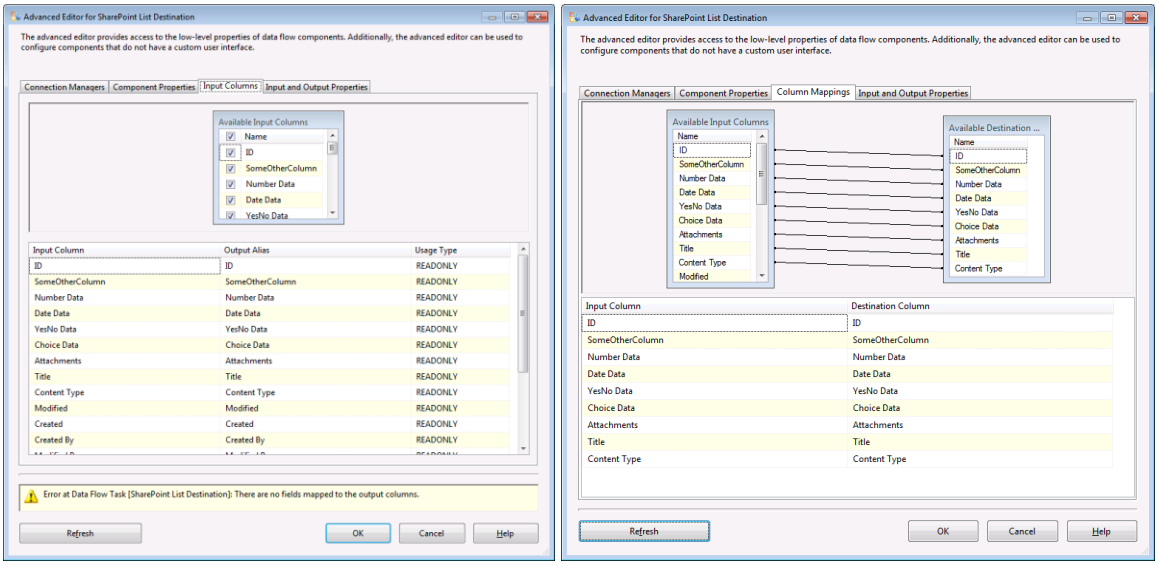


Figure 12.2. Showing before and after clicking “Refresh” button to load in the column mappings.

Working with SharePoint Data in the Data Flow

Looking Up Values in a SharePoint List

If you have to look up a value in a SharePoint list, you can use the Lookup transformation in your data flow, and use the SharePoint List source to load the lookup table. You may have to add a Derived Column transformation or a Script component that splits data in the lookup column on the ";#" delimiter to separate the ID value from the description.

If you are replacing values in your data with the values that you look up in the list, then loading the changed data back into SharePoint, you only have to include the ID from the lookup column. SharePoint ignores the description if you include it.

Understanding the Data Types of Columns

The data types of columns that are loaded by the SharePoint List source are derived from columns in the SharePoint list.

SharePoint supplies NULL values when data has not been entered and a column does not have a default value. When a new column is added to a list, all existing rows contain NULL in that column by default. The logic of your package has to recognize the possibility of NULL values.

Previewing Data

You can easily preview the data in your data flow by adding a Data Viewer.

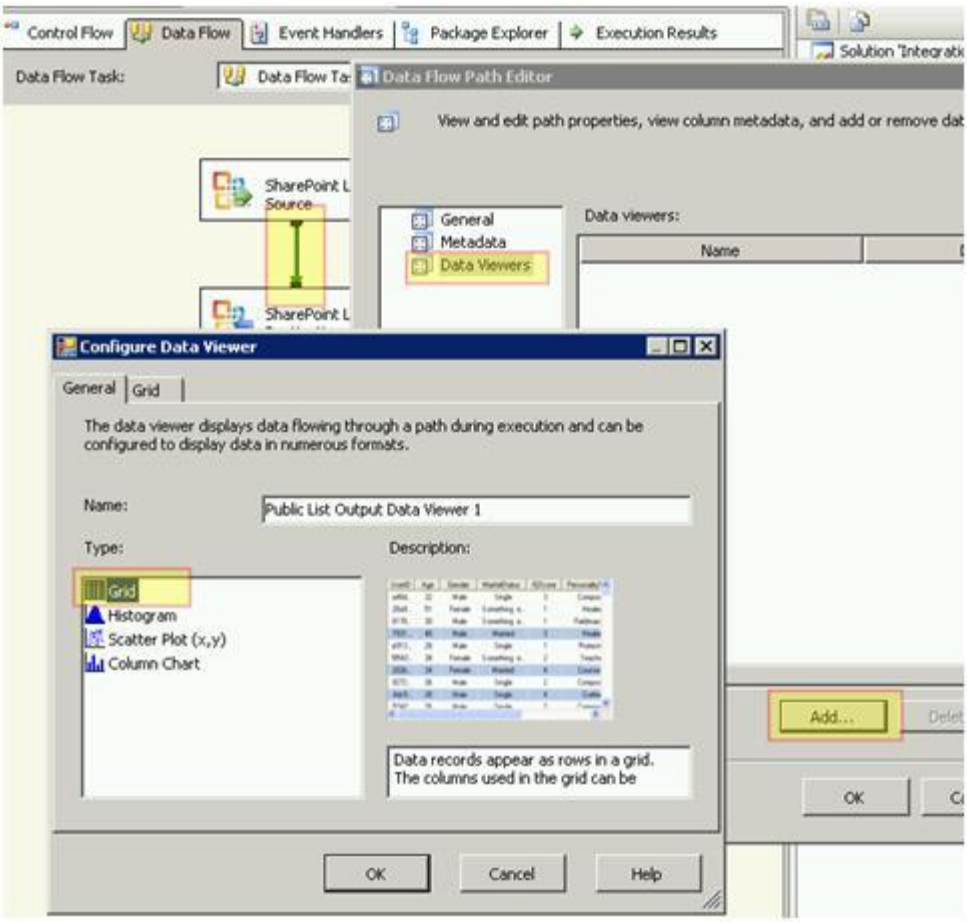


Figure 13. Configuring a Data Viewer to preview data.

To preview data by adding a Data Viewer to the data flow:

1. Double-click on the path (that is, the connecting arrow) between the two components at the point in the data flow where you want to view the data.

2. In the **Data Flow Path Editor**, select **Data Viewers**, then click **Add**.

3. In the **Configure Data Viewer** dialog box, select **Grid**, then click **OK** to close the dialog box. Click **OK** again to close the **Data Flow Path Editor**.

When you run the package, you will see a grid that shows the current state of the data as it flows from the source to the destination. This can help you to ensure that you have correctly configured the properties of the components and the logic of the package.

Setting Properties by Using SSIS Expressions during Runtime

You can set many of the custom properties of the SharePoint List source and destination by using Integration Services expressions. Keep in mind that, in the data flow, you have to create these expressions on the containing Data Flow task, and not on the individual data flow component.

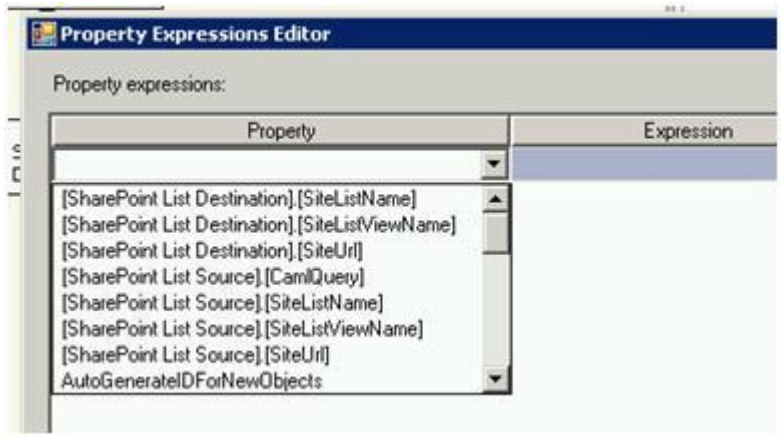


Figure 14. Using expressions to set the properties of the SharePoint List source and destination.

You can use expressions to set the following properties of the SharePoint List source:

- SharePointCulture
- SiteUrl
- SiteListName
- SiteListViewName
- CamlQuery

You can use expressions to set the following properties of the SharePoint List destination:

- SharePointCulture
- SiteUrl
- SiteListName
- SiteListViewName

There are several ways to set an expression that offer considerable flexibility to the package developer:

- You can use a package variable to set an expression. The expression can take its value from a package variable, which might also contain an expression. This expression can construct CAML dynamically, or contain the results of a query sent to the SharePoint site by an Execute SQL task.
- You can use a package configuration to set an expression. Configurations simplify deployment by extracting values that you may want to change from the package into a separate file or storage location.

Using SharePointUtility.dll Outside of Integration Services

The SharePoint List source and destination install and use an assembly (**SharePointUtility.dll**) of utility functions which can also be used outside of Integration Services for accessing SharePoint Lists.

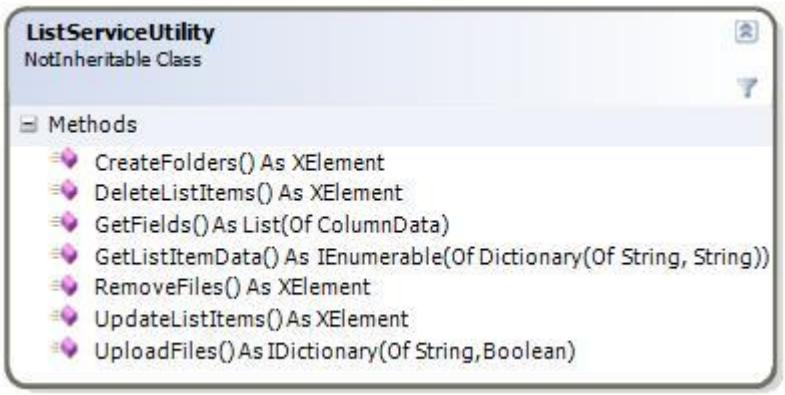


Figure 15. The methods of the **ListServiceUtility** class in the **SharePointUtility.dll** assembly.

Code for the below is stored in the Test Project, available on Codeplex. For the below samples, the following constants are shared. Every Method can take a Credential to use for connections as an override. By default, the current credentials are used.

```
    ''' <summary>
    ''' Folder where files will be loaded from for testing
    ''' </summary>
    ''' <remarks></remarks>
    Private testLocalFolderWithFiles As String = "C:\source\Code"

    ''' <summary>
    ''' SharePoint list which has at least one element used for testing
    ''' </summary>
    ''' <remarks></remarks>
    Private testSitePath As String = "http://TESTSERVER"
    Private testSiteListName As String = "Some Crazy List"
    Private testSiteListViewName As String = "Other View"
    Private testSiteDocumentLibraryName As String = "Documents"
    Private testSiteListSurveyName As String = "Test Survey"
```

CreateFolders

The unique data this takes is a list of folder names to create.

```
      Dim TEST_SHAREPOINTSITE_URL As Uri = New Uri(testSitePath + "/_vti_bin/lists.asmx")
        Dim TEST_LIST_NAME As String = testSiteListName

        Dim folderList As String() = {"test2", "teast"}
        Dim actual As XElement
        actual = ListServiceUtility.CreateFolders(TEST_SHAREPOINTSITE_URL, TEST_LIST_NAME, Nothing, fo
lderList)
        Assert.IsTrue(actual.Elements.Count <> 0, "# of items from sharepoint should not be empty.")
        If ((From item In actual...<action> Where item.Value = "Failure").Count() = 0) Then
```

```
            Assert.Fail("Did not receive an expected error updating the id on sharepoint.")
        End If
```

DeleteListItems

This method takes a list of IDs to remove. Other methods can be chained to get specific IDs.

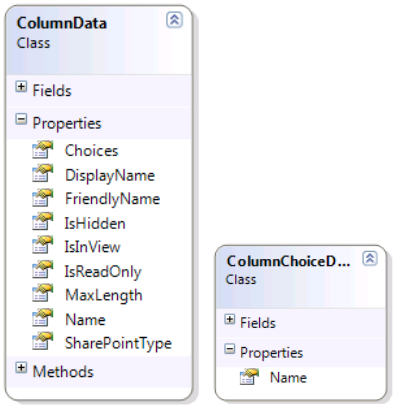
```
Dim TEST_SHAREPOINTSITE_URL As Uri = New Uri(testSitePath + "/_vti_bin/lists.asmx")
Dim TEST_LIST_NAME As String = testSiteListName

Dim fields = From f In ListServiceUtility.GetFields(TEST_SHAREPOINTSITE_URL, TEST_LIST_NAME, Nothing)
_
             Where f.IsHidden = False _
             Select f.Name

Dim dataForDelete = ListServiceUtility.GetListItemData(TEST_SHAREPOINTSITE_URL, TEST_LIST_NAME, Nothin
g, fields, <Query/>, False, 50)
Dim deleteIds = New List(Of String)
deleteIds.Add(dataForDelete(dataForDelete.Count - 1)("ID"))
ListServiceUtility.DeleteListItems(TEST_SHAREPOINTSITE_URL, TEST_LIST_NAME, Nothing, deleteIds)
```

GetFields

This method returns all of the columns from the SharePoint list into a dictionary using a Column object. The Choices are also returned for the fields which have a domain of multiple possible entries.




```
         Dim TEST_SHAREPOINTSITE_URL As Uri = New Uri(testSitePath & "/_vti_bin/lists.asmx")
        Dim TEST_LIST_NAME As String = testSiteListName

        Dim listName As String = TEST_LIST_NAME
        Dim actual As List(Of ColumnData)

        actual = ListServiceUtility.GetFields(TEST_SHAREPOINTSITE_URL, listName, Nothing)
        Assert.IsTrue(actual.Count <> 0, "# of items from sharepoint should not be empty.")
        If ((From fld In actual Where fld.Name = "Id").Count() = 1) Then
            Assert.Fail("ID Field Column has not been found.")
        End If
        Dim column = (From a In actual Where a.Name = "ID").Single()
        Assert.AreEqual(column.IsHidden, False)
        Assert.AreEqual(column.IsReadOnly, True)
        Assert.AreEqual(column.MaxLength, 25)
```

```

```

GetListItemData

This gets the data for the SharePoint list. This returns data into a list of dictionaries (Key/Value). Note that the CAML code and fields are passed in, this allows the server to filter out fields not needed or perform server side filtering to reduce the query time.

```
        Dim TEST_SHAREPOINTSITE_URL As Uri = New Uri(testSitePath & "/_vti_bin/lists.asmx")
        Dim TEST_LIST_NAME As String = testSiteListName

        Dim listName As String = TEST_LIST_NAME
        Dim fieldNames As IEnumerable(Of String) = New List(Of String)(New String() {"Id"})
        Dim isRecursive As Boolean = False
        Dim pagingSize As Short = 50
        Dim actual As IEnumerable(Of Dictionary(Of String, String))

        actual = ListServiceUtility.GetListItemData(TEST_SHAREPOINTSITE_URL, listName, Nothing, fieldN
ames, <Query/>, isRecursive, pagingSize)
        Assert.IsTrue(actual.Count <> 0, "# of items from sharepoint should not be empty.")
        If (actual(0).ContainsKey("Id")) Then
            Assert.Fail("ID Field Column has not been found.")
        End If
```

RemoveFiles

This method can be used to remove files from Document Libraries. The file name list can be any enumerable List or string Array.

```
         Dim TEST_SHAREPOINTSITE_URL As Uri = New Uri(testSitePath & "/_vti_bin/lists.asmx")
        Dim TEST_LIST_NAME As String = testSiteDocumentLibraryName

        ' First we need to upload the files
        UploadFilesTest()

        ' Now remove them.
        Dim fullPathList = New List(Of String)(System.IO.Directory.GetFiles(testLocalFolderWithFiles))
        Dim fileNameList = From f In fullPathList _
                           Select System.IO.Path.GetFileName(f)
        Dim removeResults = ListServiceUtility.RemoveFiles(TEST_SHAREPOINTSITE_URL, TEST_LIST_NAME, No
thing, fileNameList.ToArray())
        If ((From item In removeResults...<action> Where item.Value = "Failure").Count() > 0) Then
            Assert.Fail("Updating the title on sharepoint failed.")
        End If
        Assert.IsTrue(removeResults.Elements.Count <> 0, "# of items from sharepoint should not be emp
ty.")
```

UpdateListItems

Lists can be updated, however the key (ID) must be a field in the dictionary of values. If that is present, then SharePoint will update. If the value does NOT match a target row of SharePoint, then it will error. If the field is not present or null, then the row will automatically be added by SharePoint.

```
Dim TEST_SHAREPOINTSITE_URL As Uri = New Uri(testSitePath & "/_vti_bin/lists.asmx")
Dim TEST_LIST_NAME As String = testSiteListName
```

```
Dim fields = From f In ListServiceUtility.GetFields(TEST_SHAREPOINTSITE_URL, TEST_LIST_NAME, Nothing)
_
                     Where f.IsHidden = False _
                     Select f.Name
Dim data = ListServiceUtility.GetListItemData(TEST_SHAREPOINTSITE_URL, TEST_LIST_NAME, Nothing, fields
, <Query/>, False, 50)
data(0)("Title") = "Updated title: " + New Random().Next().ToString()
Dim updateOutput = ListServiceUtility.UpdateListItems(TEST_SHAREPOINTSITE_URL, TEST_LIST_NAME, Nothing
, data, 2)
If ((From item In updateOutput...<action> Where item.Value = "Failure").Count() > 0) Then
      Assert.Fail("Updating the title on SharePoint failed.")
End If
Assert.IsTrue(updateOutput.Elements.Count <> 0, "# of items from sharepoint should not be empty.")
```

UploadFiles

Files can be easily added to document libraries using this method and a list of locally accessible files.

```
Dim TEST_SHAREPOINTSITE_URL As Uri = New Uri(testSitePath & "/_vti_bin/lists.asmx")
Dim TEST_LIST_NAME As String = testSiteDocumentLibraryName

Dim fullPathList = New List(Of String)(System.IO.Directory.GetFiles(testLocalFolderWithFiles))
Dim uploadResults = ListServiceUtility.UploadFiles(TEST_SHAREPOINTSITE_URL, TEST_LIST_NAME, fullPathLi
st)
If ((From result In uploadResults Where result.Value = False).Count() > 0) Then
        Assert.Fail("Uploads to SharePoint had a failure.")
End If
Assert.IsTrue(uploadResults.Count <> 0, "# of items from SharePoint should not be empty.")
```

The **ListServiceUtility** class in the **SharePointUtility.dll** assembly has the following features:

- Although the methods of the **ListServiceUtility** class require parameter values, the parameters do not use GUID values and do not expose SharePoint XML.
- When you request list data, the methods of the class return rows with a dictionary for the columns and values. When you update rows, you have to provide a list of dictionaries in return, where the names in the dictionary are column names.
- When you work with the methods of the **ListServiceUtility** class, you never see the ows_ prefix, and spaces are converted from their XML representation (_x0020_) to a space character.
- Files can be uploaded and removed easily. The methods also support SharePoint on a port other than port 80. (Older versions of Windows only support port 80 for WebDAV).
- XML results are reformatted for update operations. If an error occurs during the update, a different XML structure that contains a description is returned, as in the following examples:

Normal result of an update

```
<result ID=”1” >
```

```
<action>Success</action>

<row [attributes are columns, and their values]></row>

</result>
```

Result of an update after an error

```
<result ID=”1” >

      <action>Failure</action>

      <errorCode>0x81020015</errorCode>

      <errorDescription>The changes requested conflict with those made by another client.</errorDescription>

<row [attributes are columns, and their values]></row>

</result>
```

Conclusion

The SharePoint List Source and Destination Sample makes it easy to get data out of or into a SharePoint list from an Integration Services package. The sample also includes a utility library that simplifies working with SharePoint lists outside of Integration Services.

If you have feedback or questions about these components, please visit the Codeplex site at http://www.codeplex.com/SQLSrvIntegrationSrv.