




---

## SharePoint Integration Guide

---

The DBXL Administration Tool (DAT) allows you to integrate a DBXL Document Type and a SharePoint Form Library. Whenever possible, Qdabra Software recommends this integrated approach, and this will be the main focus of this document.

However, a SharePoint mapping may not be possible for every scenario. For example:

- SharePoint mapping maps all XML documents into the Form Library, which means that users can see all documents. If you need DBXL permissions enforced, a data view is recommended.
- If you have a form with code, or a form with Qdabra's qRules, and need end-users to open the forms in the browser (IPFS), you won't be able to use a SharePoint mapping unless you are using SharePoint 2010.

These additional scenarios will be briefly discussed in this document, with links provided as necessary.

Add a SharePoint mapping to your document type .....	2
Workflows .....	3
Propagate deletes .....	3
Form naming .....	4
Web Service .....	4
Conditional Sharepoint mapping .....	5
Create a SharePoint data view that enforces DBXL permissions .....	6
Side by Side .....	13
Solutions that contains code .....	13
Create a SharePoint Data View .....	14
Support .....	14



## ADD A SHAREPOINT MAPPING TO YOUR DOCUMENT TYPE

These steps will use the QdExpenseReport that is included in all installations of DBXL.

This first scenario assumes the following setup:

- If using SharePoint 2007, the InfoPath form (XSN) does not contain code and does not utilize qRules.
- If using SharePoint 2010, the InfoPath form (XSN) may contain code or qRules.
- DBXL and SharePoint are installed on the same site, and the DBXL application pool is using the same identity as the SharePoint app pool.
- A Document Type admin cannot add SharePoint mappings, as this is an operation that involves the server. To create a SharePoint mapping, you'll need to be a DBXLAdmin.

Follow these steps to add a SharePoint mapping:

1. For the Expense Report document type, switch to the **SharePoint** tab.
2. Click **Insert Mapping**.
3. In the **URL** field, enter the URL to a form library that does not exist.

Creating a SharePoint mapping using Object Model requires that DBXL be installed on the same server as SharePoint. Therefore, your SharePoint form library will exist on the same server where DBXL is installed.

4. For **Method**, select **Object Model**.
5. Check the checkboxes for **Publish Library and XSN** and **Use Forms Services**.

If users will open the form in the InfoPath client, do not check Use Forms Services.

6. Click **Save** and then click **OK** in the confirmation dialog.

**SharePoint List or Library (i.e. Collection)**

Specify SharePoint lists and/or libraries that you would like to integrate with this doctype. By integrating, you can enable SharePoint features such as Workflow and Form Server for your database documents. After the user clicks submit from InfoPath or browser (browser based forms), the Qdabra Web Service will try to copy the document to the SharePoint sites in the order specified below.

Mappings Allowed	4	Remaining	3
------------------	---	-----------	---

URL	Method
http://qshp2010/ExpenseReport	Object Model

Expression	Operator
<input checked="" type="checkbox"/> Publish Library and XSN <input checked="" type="checkbox"/> Use Forms Services <input type="checkbox"/> Propagate Deletes <input type="checkbox"/> Asynch <input type="checkbox"/> Content Type	

Insert Mapping

7. Use Internet Explorer to navigate to the form library URL.
8. Click **Add Document**.
9. Fill out the form and submit.

The form library in Internet Explorer will refresh and you'll see the new document. Congratulations! You have created a SharePoint mapping.



Type	Name	Modified	Modified By	Checked Out To	DBXL ID	DBXL Version	Author	Description	Title
	2012-03-18 	3/18/2012 7:20 PM	AUTONOMYSYSTEMS\ErnestoM		6467	1	Michael Jones	This is a test	2012-03-18

[Add document](#)

What about the existing documents, though? Simple – just click **Reshred All** in the **Documents** tab.

Type	Name	Modified	Modified By	Checked Out To	DBXL ID	DBXL Version
	2012-03-18 (6461) 	3/18/2012 7:21 PM	AUTONOMYSYSTEMS\ErnestoM		6461	1
	2012-03-18 (6462) 	3/18/2012 7:21 PM	AUTONOMYSYSTEMS\ErnestoM		6462	1
	2012-03-18 (6463) 	3/18/2012 7:21 PM	AUTONOMYSYSTEMS\ErnestoM		6463	1
	2012-03-18 (6464) 	3/18/2012 7:21 PM	AUTONOMYSYSTEMS\ErnestoM		6464	1
	2012-03-18 (6465) 	3/18/2012 7:21 PM	AUTONOMYSYSTEMS\ErnestoM		6465	3

## WORKFLOWS

If you wish to use SharePoint workflows, please refer to the steps outlined in this document:  
[http://www.infopathdev.com/files/folders/integration\\_with\\_sharepoint/entry67954.aspx](http://www.infopathdev.com/files/folders/integration_with_sharepoint/entry67954.aspx)

## PROPAGATE DELETES

In general, deleting documents is not a DBXL best practice. Qdabra recommends leveraging the power of a “Status” field to tag documents as “Rejected” or “Closed” instead.

However, if you have created a SharePoint mapping DAT and you wish to delete documents, some considerations are worth mentioning. The following list assumes that you have an Object Model (i.e. same machine) SharePoint mapping for your Document Type.

- If you delete from the Documents tab in the DBXL Administration Tool (DAT), the document will be deleted from both DBXL and SharePoint.
- If you want to allow users to delete from the SharePoint form library, you must set up Qdabra’s SharePoint listener. For details, click here:  
[http://www.infopathdev.com/files/folders/integration\\_with\\_sharepoint/entry65676.aspx](http://www.infopathdev.com/files/folders/integration_with_sharepoint/entry65676.aspx)

If the SharePoint Listener is NOT set up, and users attempt to delete a document from the SharePoint Form Library, the document will be removed from SharePoint, but not DBXL. To force the document to be re-mapped to SharePoint, use the Reshred button in the Documents tab in DAT.



<http://www.qdabra.com>

Last updated on 2/19/2013 2:52 PM

Copyright © 2011 Autonomy Systems, LLC. All rights reserved.

If you have a SharePoint mapping that uses Web Service (i.e. different machine), then an administrator will need to manually delete from both the Documents tab in the DBXL Administration Tool AND from the SharePoint Form Library.

The same manual steps for deletion apply for side-by-side.

## FORM NAMING

Starting in DBXL v2.5, the DBXL web service expects the DBXL::Name token to be unique. This value is used as the Title in the SharePoint mapping. If the value of DBXL::Name is blank, the web service will use the DocID, and if there are duplicates in the DBXL::Name, the web service will append the DocID to distinguish between them.

Starting in DBXL v2.7, you can add a setting in web.config to modify the way DBXL maps XML forms to SharePoint. If the key is set to true, the DBXL::Name is ignored and the DocID is used instead. This is not, by default, included in DBXL's web.config and must be manually added above the closing appSettings tag.

```
<add key="UseDocIdForSharePointFileName" value="true" />
<add key="UseDocIdForSharePointFileName-YourDocTypeNameHere" value="true" />
```

Another option is to modify the link manually; for details on this please see this document: [http://www.infopathdev.com/files/folders/integration\\_with\\_sharepoint/entry81327.aspx](http://www.infopathdev.com/files/folders/integration_with_sharepoint/entry81327.aspx)

## WEB SERVICE

You may have noticed that DAT provides a second option when creating a SharePoint mapping. We used Object Model because DBXL and SharePoint are on the same site in the same server.

With Object Model, DBXL uses the SharePoint object model library to save to SharePoint. This normally gives the best control of SharePoint, but it does have some restrictions. When using Object Model, the DBXL service account must have permissions to the SharePoint content databases. This is normally satisfied by using the same account as the SharePoint app pool. Using the SharePoint app pool account prevents issues with Kerberos.

Another side-effect of using the SharePoint Object Model is that the web.config must be updated to prevent issues with workflows (<http://msdn.microsoft.com/en-us/library/bb727787.aspx>). Object Model will only work when DBXL is on the SharePoint web front end. Using Object Model gives DBXL the ability to set some SharePoint properties that it cannot control using the Web Service, such as Modified By, Modified Date, Created By. This is helpful when reshredding so the modified by does not change based on who reshredded.

If DBXL and SharePoint were NOT on the same site, we would need to select the Web Service option instead and configure SSO. Selecting Web Service reveals two additional fields: **SSO** and **SSO Application Name**.

When using the SharePoint Web Service, requests go through the SharePoint web service layer rather than directly to the SharePoint database and internal APIs. Performance will be a little slower, but functionality should be essentially the same. Impersonation can sometimes be tricky using web service. If the web service call appears



to cross the machine boundary, impersonation will fail. This can often be corrected by updating the server's hosts file to use the loopback address for the site.

The use of IPFS-enabled solutions is not supported if the installations of SharePoint and DBXL lie on different machines/sites. If your solution requires IPFS, we recommend installing SharePoint and DBXL on the same machine, and using the Object Model method when creating your mapping.

## CONDITIONAL SHAREPOINT MAPPING

Now you want to map forms only when a certain condition is satisfied. This can be useful if you would like some XML to be saved to one library, and some to another, for example, or if you only want to save XML at a certain status or completeness to SharePoint, while keeping all XML in progress in DBXL.

1. Back in the DBXL Administration Tool, in the SharePoint tab, click **Insert Expression**.

The screenshot shows the 'SharePoint List or Library (i.e. Collection)' configuration window in the DBXL Administration Tool. The 'URL' field contains 'http://qshp2010/ExpenseReport' and the 'Method' is set to 'Object Model'. The 'Expression' field is highlighted, and the 'Insert Expression' button is being clicked. Other options include 'Publish Library and XSN', 'Use Forms Services', 'Propagate Deletes', 'Asynch', and 'Content Type'.

2. In the Expression field, enter

**`/my:expenseReport/my:expenseCode != ""`**

3. Click **Save**, and then **OK** in the confirmation dialog.
4. Back in Internet Explorer, click **Add Document**.
5. Fill out the form and submit. Make sure to leave the expenseCode blank.

When the form submits and the form library refreshes, you'll notice that the form you just submitted does not show up! This is because it did not satisfy the condition.

6. In DAT, switch to the Documents tab, to verify that the form you just submitted is listed.
7. Back in Internet Explorer, click **Add Document**.
8. Fill out the form and submit.

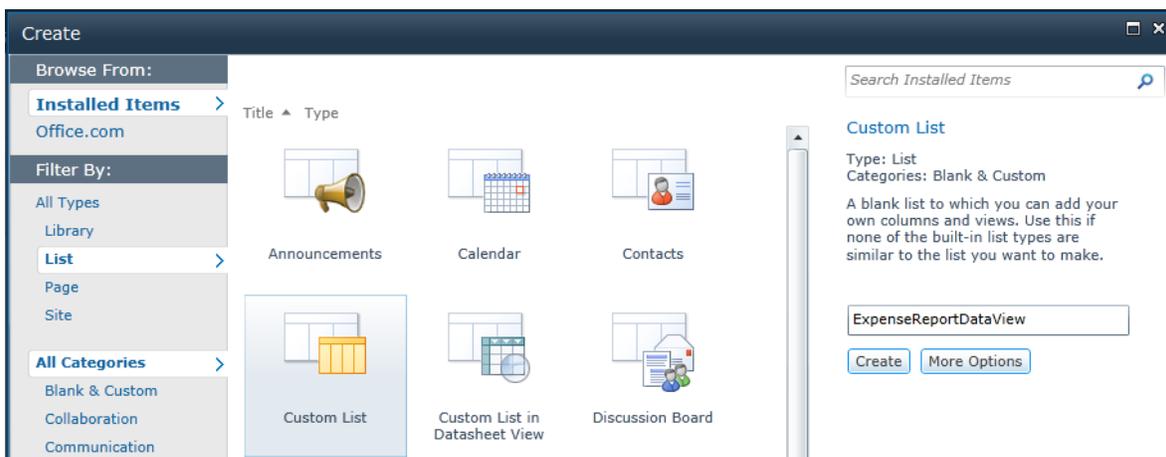
Refresh the form library and the **Documents** tab in DAT to verify that the forms with an expenseCode are all mapped to SharePoint correctly.



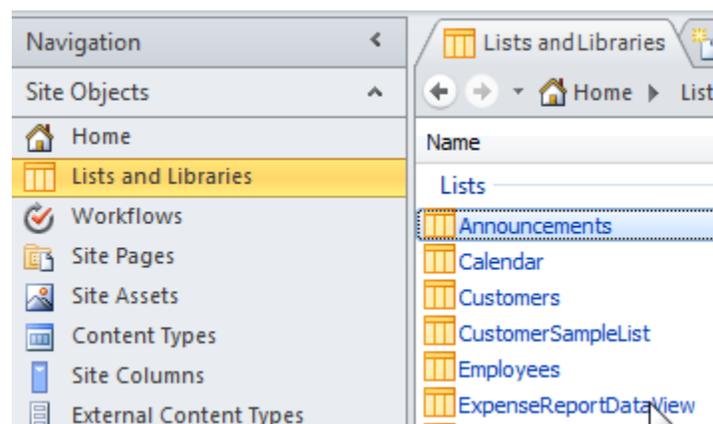
## CREATE A SHAREPOINT DATA VIEW THAT ENFORCES DBXL PERMISSIONS

SharePoint Designer allows you to receive data from DBXL's web methods, and one method in DBXL is particularly interesting because it allows you to create a SharePoint Data View. The DBXLDocumentService web service contains GetListItems, which retrieves promoted properties in the form and displays them as columns in SharePoint. GetListItems has an added bonus: it will respect DBXL permissions.

1. Navigate to the site where you wish to add this data view. Click on **All Site Content** and then click on **Create**.
2. Select **Custom List**, enter a name for the list and click **Create**. We are using a custom list here just for simplicities sake; the data view web part could be added to an existing page in the site, or to a web part page.



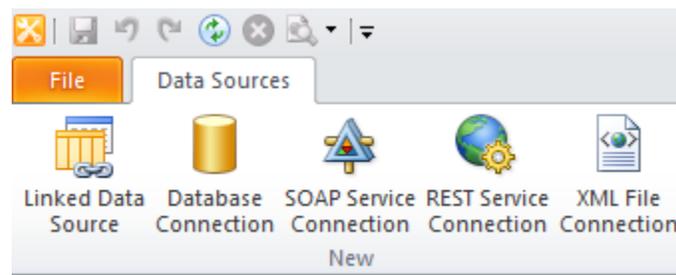
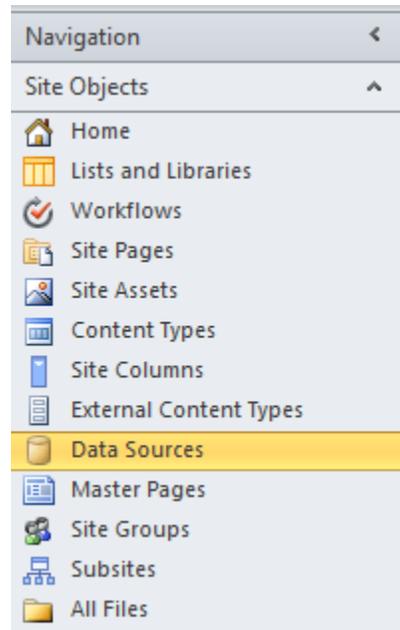
3. Launch SharePoint Designer 2010.
4. Go to **File > Open Site**. Enter the URL to your site and click **Open**. The site will load and might take a few seconds depending on the size of your site.
5. In the **Navigation** frame, select **List and Libraries**. This will load the list of available lists and libraries.



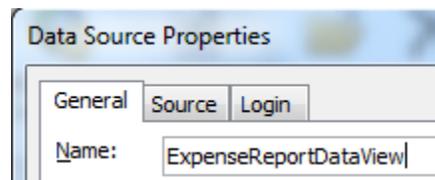
6. Click on the list you previously created to open it.



7. Under **Views**, click on **All Items** to open this default view in the editor.
8. In the **Navigation** pane, select **Data Sources**.
9. Select **Data Sources** in the Ribbon and then select **SOAP Service Connection**.



10. In the **General** tab, enter a name for your data source.



11. In the **Source** tab, enter the web service URL:  
***http://<servername>/qdabrawebservice/dbxldocumentservice.asmx***
12. Click on **Connect Now**.
13. From the **Operation** dropdown select **GetListItem**s.
14. Double click on the **listName** and **viewName** parameters, and enter the name of the DBXL Document Type you wish to query, in this case, ExpenseReport. Click **OK**.



15. **GetListItems** limits the results returned in your SharePoint list to the first 100 documents. In order to increase this, set the rowLimit to more than 100 rows.

**Data Source Properties**

General Source Login

Service description location:  
   
 Example: http://example.com?WSDL

Select which data command to configure:

Select Connection Info

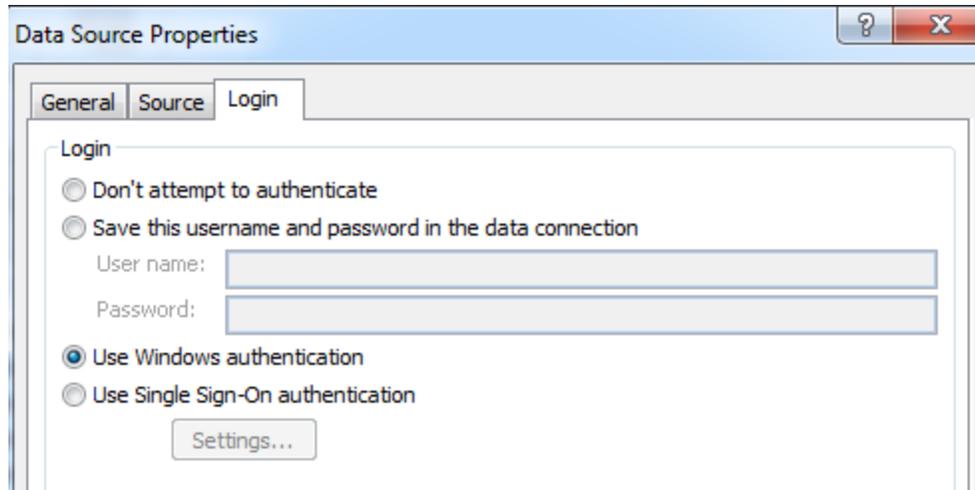
Service Name: DbxlDocumentService  
 Port: IDbxlDocumentService  
 Address: http://qshp2010/qdabrawebsevice/dbxldocumentservice.asmx  
 Operation: GetListItems

Parameters (\* required):

Name	Value	Element	Type	Runtim...
listName	ExpenseReport		s:string	<input type="checkbox"/>
viewName	ExpenseReport		s:string	<input type="checkbox"/>
query				<input type="checkbox"/>
viewFields				<input type="checkbox"/>
rowLimit	1000		s:string	<input type="checkbox"/>
queryOptions				<input type="checkbox"/>

16. Under the **Login** tab, select **Windows Authentication**.





17. Click **OK**. This creates the Data Source.
18. Return to the **AllItems.aspx** edit view. You'll see navigation tabs at the top.

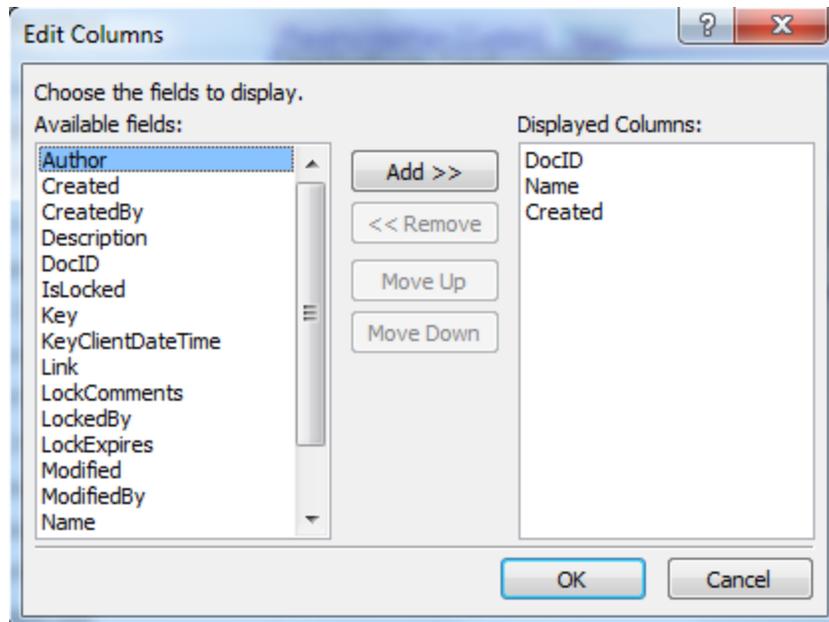


SharePoint adds some generic layout tables and controls to this view, which you can remove or modify as desired.

19. Place the cursor on the location where you wish to enter the data view.
20. From the ribbon option **Insert**, select **Data View** and then select the data source we created above. If it doesn't display among the options, you might need to select **More Data Sources**. If the current data source does not display in the right hand taskpane, click on **Refresh data source** to query it again.
21. Modify the available columns by going to **Options > Add/Remove columns** in the ribbon. Add/remove columns as desired. You may also choose to reorder the columns using the **Move Up** and **Move Down** buttons.

**Note:** You can only show columns for fields that have been promoted. If you don't see the field you want to display as a column, please verify that it has been added as a promoted property to the form and the form republished.





22. Select the column item you wish to use as a hyperlink, and then select **Insert > Hyperlink** from the ribbon. In our case we will add the link to the DocID column.

HomeExpenseReportDataView Default ▾

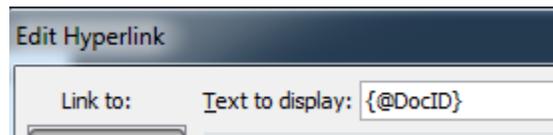
Home BCS Search GMO Kiewit CAP Cynthia Webcasts GIS InfoPath Pilot Webinars Northw

Royal British Legion ACE PlaceholderMain (Custom) GalleryWebPart Mel's Test Site PS Main

td.ms-vb	Name	Created
6461	2012-03-18	2012-03-18T05:02:06-07:00
6462	2012-03-18	2012-03-18T06:08:55-07:00
6463	2012-03-18	2012-03-18T06:13:43-07:00
6464	2012-03-18	2012-03-18T06:35:26-07:00
6467	2012-03-18	2012-03-18T07:20:37-07:00



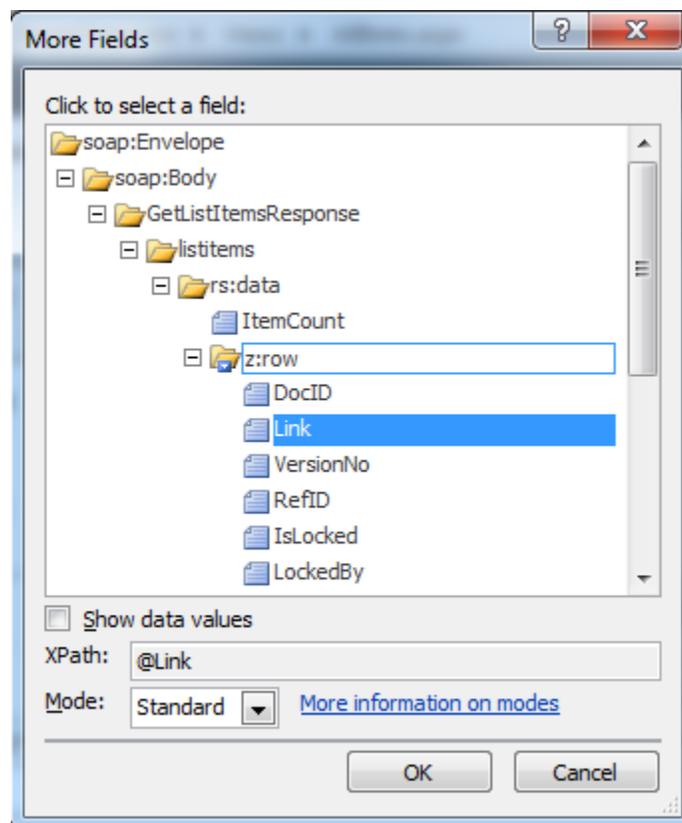
23. Notice that the **Edit Hyperlink** window, shown below, has the Text to display set to the DocID.



24. Make sure that the **Address** box is empty (SharePoint Designer may try to load a value into that box, but we don't need it.)

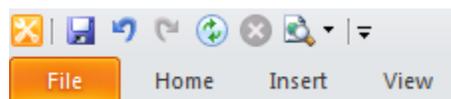
25. Next to the **Address** box, click the **fx** button (  ).

26. Select the **Link** field and click **OK**.



27. Click **OK**.

28. Click the **Save** icon to save the changes to your SharePoint List.



29. Return to Internet Explorer and navigate to the page you have been designing. You might need to refresh it to load the design changes. You'll see that it contains a list of the available forms.

DocID	Name	Created
6034	ErnestoM - 2012-02-02T18:03:48	2012-02-02T01:04:13-08:00
6035	ErnestoM - 2012-02-02T18:04:26	2012-02-02T01:04:28-08:00

You have now created a simple, SharePoint data view and have learned the basics of SharePoint Designer. The links will open the forms that are stored in DBXL, and the DBXLDocumentService will respect DBXL permissions.

However, the steps so far use the DBXL link, which will open the form in InfoPath Filler. What if you want the form to open in the browser?

30. Return to the Form Library created in the previous section, and click on any of the existing documents. Make a note of the URL.

Let's understand that URL:

`http://<servername>/_layouts/FormServer.aspx?XmlLocation=/ExpenseReport/2012-03-18%20(6462).xml&ClientInstalled=true&Source=<encoded URL>&DefaultItemOpen=1`

It is made up of the following essential portions:

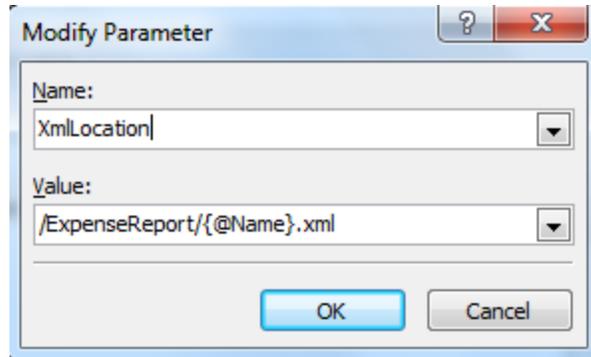
- `http://<servername>/_layouts/FormServer.aspx` (where `<servername>` will be replaced with the name of your server)
- `XmlLocation=/<form library name>/<DBXL Name>.xml` (where `<form library name>` will be replaced with the name of your form library)
- `Source=<encoded URL>` (the URL where users will be returned after closing the form)
- `DefaultItemOpen` will be replaced by a different parameter, per [Microsoft's documentation](#).

Your URL may also contain a site or subsite.

We'll need to recreate this URL in SharePoint designer.

31. Reselect the column in the data view that contains the link.
32. Click **Insert Hyperlink** in the ribbon.
33. Clear out the contents of **Address**.
34. Click **Parameters**.
35. For **Path**, enter `http://<servername>/_layouts/FormServer.aspx`
36. Click **Add**.
37. For **Name**, enter **XmlLocation**.
38. For **Value**, enter `/<form library name>/{@Name}.xml`





39. Click **Add** again.
40. For **Name**, enter **OpenIn**.
41. For **Value** enter **Browser**.
42. Click **Add** again.
43. For **Name**, enter **Source**.
44. For **Value** enter the URL for the page users should be returned to after filling out the form. NOTE: This URL must be in the same site collection or an error will be returned.
45. Click OK to dismiss all windows.
46. Click **File > Save**.
47. Refresh the data view page on your browser. Try the new link – it should open the forms in the browser!

## SIDE BY SIDE

Side-by-Side refers to the architecture in which forms can be codeless and stand alone. The forms won't have a DBXL DocID reference in them – they can live in SharePoint or on someone's local disk. This is an interesting scenario if:

- DBXL is not installed on the SharePoint 80 site
- You have an existing form library with existing forms and are unable to migrate to the integrated scenario.

To simplify the form changes, Qdabra created the DBXL Side-By-Side template part. Users can add the XTP in InfoPath, and then just drag and drop to the bottom of the form. The XTP verifies that a form is not out-of-date (more recent form exists), checks if a form is locked for editing by another user (thus preventing simultaneous edits and/or submits) and retrieves DBXL information about the form.

This document details the instructions on how to add and use the DBXL Side-By-Side template part in your DBXL solutions. [http://www.infopathdev.com/files/folders/side\\_by\\_side/entry74086.aspx](http://www.infopathdev.com/files/folders/side_by_side/entry74086.aspx)

## SOLUTIONS THAT CONTAINS CODE



<http://www.qdabra.com>

Last updated on 2/19/2013 2:52 PM

Copyright © 2011 Autonomy Systems, LLC. All rights reserved.

In general, the steps discussed so far will not work if your InfoPath form uses code (or Qdabra's qRules).

If using qRules v2.4 (or newer) and SharePoint 2010, you should follow the steps in this document:

<http://www.infopathdev.com/files/folders/qrules/entry62469.aspx>

If your form will be opened in the InfoPath client, and not the browser, please try the steps outlined here:

[http://www.infopathdev.com/files/folders/integration\\_with\\_sharepoint/entry74101.aspx](http://www.infopathdev.com/files/folders/integration_with_sharepoint/entry74101.aspx)

There are two additional ways you can integrate your browser-enabled qRules form (or forms that have custom code) with DBXL:

1. Use the content type mapping in DBXL. For details on this scenario, please refer to this document [http://www.infopathdev.com/files/folders/integration\\_with\\_sharepoint/entry76111.aspx](http://www.infopathdev.com/files/folders/integration_with_sharepoint/entry76111.aspx). This scenario does work for codeless forms as well; for details, see this document [http://www.infopathdev.com/files/folders/integration\\_with\\_sharepoint/entry74106.aspx](http://www.infopathdev.com/files/folders/integration_with_sharepoint/entry74106.aspx).
2. Use the qRules command called SwapDOM. <http://www.infopathdev.com/files/folders/qrules/entry53171.aspx>

## CREATE A SHAREPOINT DATA VIEW

SharePoint Designer 2007 and 2010 allow you to create and customize data views. There are two ways in which DBXL can take advantage of data views.

1. Using the GetListItems web service, you can create a data view that is able to retrieve xml documents. The promoted properties are displayed as columns in the SharePoint data view and DBXL permissions are respected. There is a second web service, called GetMyListItems, designed to respect DBXL assignments (if your solution uses QdFlow assignments).
  - a. 2010: [http://www.infopathdev.com/files/folders/integration\\_with\\_sharepoint/entry65678.aspx](http://www.infopathdev.com/files/folders/integration_with_sharepoint/entry65678.aspx)
  - b. 2007: [http://www.infopathdev.com/files/folders/integration\\_with\\_sharepoint/entry74098.aspx](http://www.infopathdev.com/files/folders/integration_with_sharepoint/entry74098.aspx)
2. You can also create a data view that obtains its data from SQL. We can accomplish this because DBXL allows you to map your xml data into SQL tables. Once that SQL mapping is created, you can create a SharePoint data view that feeds on SQL data. For more information, please refer to a full tutorial in this URL: [http://www.infopathdev.com/files/folders/integration\\_with\\_sharepoint/entry74100.aspx](http://www.infopathdev.com/files/folders/integration_with_sharepoint/entry74100.aspx)

## SUPPORT

In this document we have covered various methods which allow you to integrate your DBXL document types with SharePoint. Though there may be more possibilities for integration, the above give you a range of possibilities. Should you come across a situation that is not satisfied by the scenarios above, please contact Qdabra for assistance.



<http://www.qdabra.com>

Last updated on 2/19/2013 2:52 PM

Copyright © 2011 Autonomy Systems, LLC. All rights reserved.

If you have questions about the information in this document, please contact us for assistance.

Licensed customers can contact us via [Support@Qdabra.com](mailto:Support@Qdabra.com).

Alternatively, please use the [InfoPathDev.com Qdabra Product support forums](http://InfoPathDev.com/Qdabra/Product/support/forums) to request help.



<http://www.qdabra.com>

Last updated on 2/19/2013 2:52 PM

Copyright © 2011 Autonomy Systems, LLC. All rights reserved.