



Product: Database Accelerator (DBXL) v2.4

Title: Submit data to DBXL from a non-InfoPath form

DBXL provides a set of web services to submit and update data. This document provides instructions for creating an application that accesses the **DBXLDocumentService** Web service directly without using InfoPath.

Below you will find an outline of this document's contents.

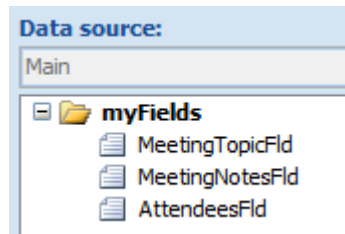
Pre-requisites.....	2
Execution steps.....	3
Support	11



PRE-REQUISITES

For this tutorial you will need to have a server with **DBXL** installed. We assume that you have some basic DBXL knowledge. Please refer to the DBXL v2.4 Product page on Qdabra.com for a full list of documentation and resources. Finally, you need to have **Visual Studio 2005** or a later version on your machine.

- 1) Create an XSN that defines the schema of your web form. This tutorial uses the below schema, where all fields are of data type *string*.



- 2) Create a DAT configuration (this tutorial uses one by the name of **submitTemplate**) with the schema defined in the above step.

DATABASE ACCELERATOR
Administration Tool

Document Type: submitTemplate
 Save
 Save As...
 Catalog

General
Database
Permissions
SharePoint
Documents
Taxonomy Tree

Document Type Details

Name	Solution Group
submitTemplate	▼
Description	
Last Modified By	
varunaa	

InfoPath Form Template

XSN File	Form Template Properties
WebServiceTemplate.xsn Microsoft Office InfoPath Form Template 3.72 KB Open	Form Version: 1.0.0.2 Last Modified: 4/19/2010 10:34:59 AM ☐ Form template is signed ☐ URN-based solution
URL to Form Template	
HTTP:// /QdabraWebService/Forms/ID/474/template.xsn	

Ref. ID

Format for Ref. ID	Base Reference #	XPath to Inject Ref. ID Into (Leave Blank to Disable)



<http://www.qdabra.com>

Last updated on 6/29/2010 12:05 PM

Copyright © 2006-2010 Autonomy Systems, LLC. All rights reserved.

- 3) Define a database mapping in DAT.

DATABASE ACCELERATOR Administration Tool QDABRA SOFTWARE

Document Type: submitTemplate Save Save As... Catalog

General Database Permissions SharePoint Documents Taxonomy Tree

Database Configuration

Connection String Test Connection

Integrated Security=SSPI;Data Source=;Initial Catalog=Aruna_TestDB

Submit Options

☒ Enable History ☒ Report Shredding Errors to the Form User

Database Map Reshred All Documents Refresh Database

Table Name (3 Columns) ▼	Node Path			
MeetingTable	/my:myFields			
Column Name	DB Type	Key	Node Path	
Attendees	String ▼	☐	my:AttendeesFld	
MtgNotes	String ▼	☐	my:MeetingNotesFld	
MtgTopic	String ▼	☒	my:MeetingTopicFld	

☒ Insert Table or Column
☒ Insert Table or Column

EXECUTION STEPS

Here is a simple walkthrough to illustrate how to submit data to DBXL using an ASP.NET web page. This example uses steps and screenshots from Visual Studio 2008.

1. Open Visual Studio and create a new project by selecting **New -> Project...** from the **File** menu.



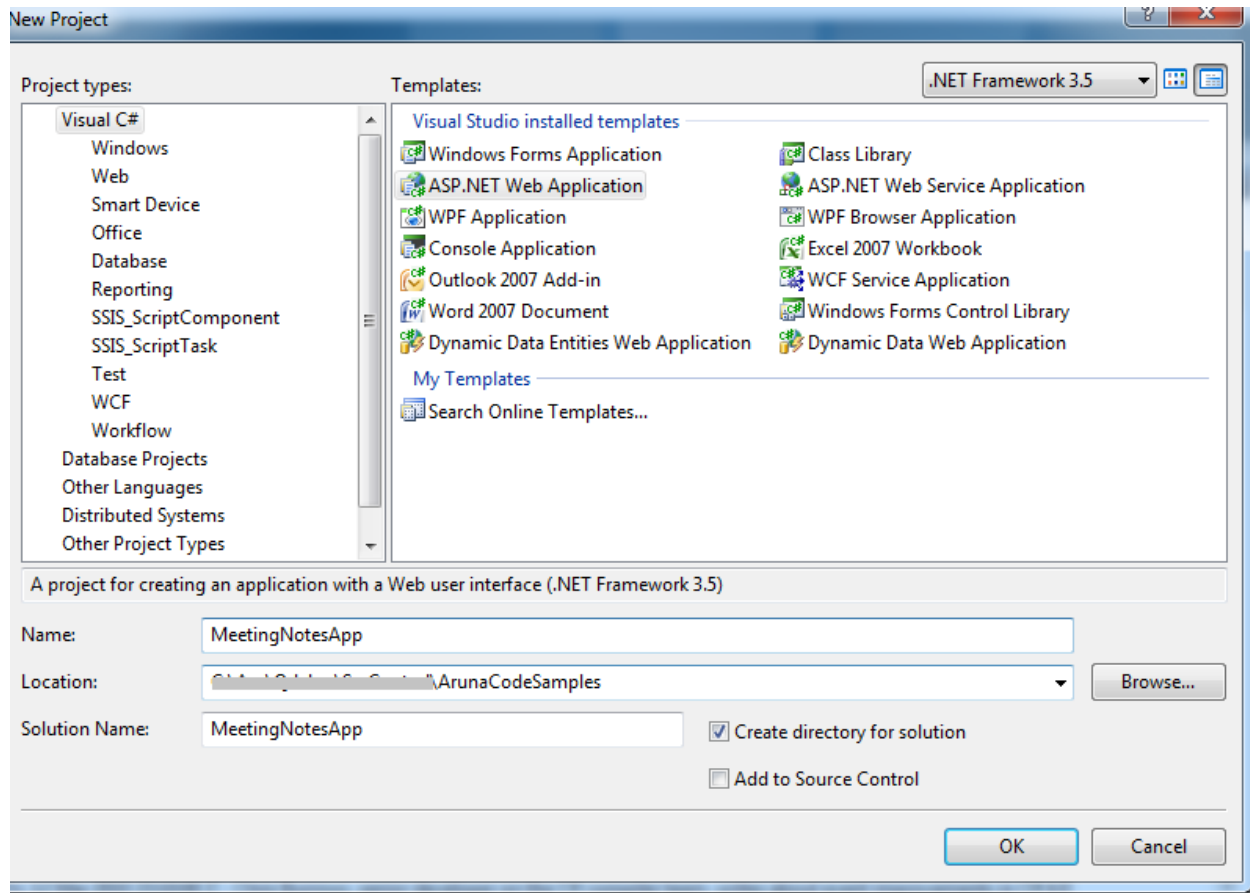
2. Select **Visual C#** under **Project types** and then select **ASP.NET Web Application** under **Templates**. This tutorial uses **MeetingNotesApp** as the project name. Click the **OK** button.



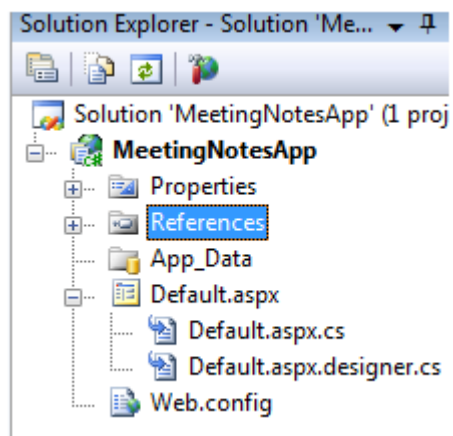
<http://www.qdabra.com>

Last updated on 6/29/2010 12:05 PM

Copyright © 2006-2010 Autonomy Systems, LLC. All rights reserved.

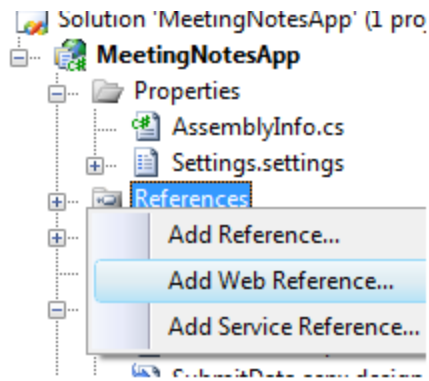


3. Click the **OK** button. A new project is created with a page called **Default.aspx**.

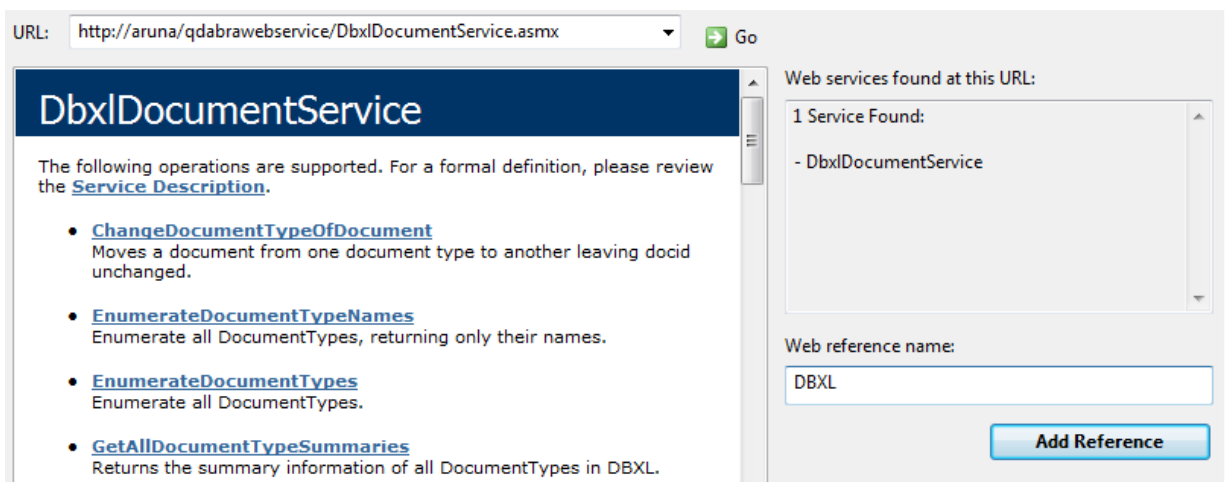


4. Give **Default.aspx** a meaningful name (this tutorial uses **SubmitData.aspx**)
5. Add a web reference: In the Solution Explorer window of Visual Studio, right click on **References...** in the **MeetingNotesApp** project and select **Add Web Reference...** from the context menu.

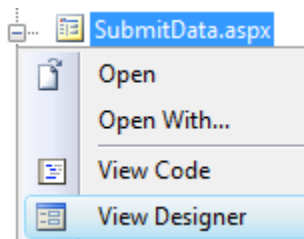




6. Enter the web service URL (<http://<servername>/qdabrawebservice/DbxlDocumentService.asmx>) and click on the **Go** button to navigate to a web service URL and click on the **Add Reference** button to list add all the available services. Give this web reference a meaningful name (this tutorial uses **DBXL**).



7. Open SubmitData.aspx in design mode and add four textbox controls, and name them **idFld**, **mtgTopicFld**, **mtgNotesFld**, and **mtgAttendeesFld**. Add a button and name it **SubmitBtn**.



Id:	
Author:	
Meeting Topic:	
Meeting Notes:	
Attendees:	
Submit	

8. Open **SubmitData.aspx.cs** file. After all of the **using** statements at the top, add a web reference namespace, **MeetingNotesApp.DBXL**.

```

MeetingNotesApp.SubmitData
1 using System;
2 using System.Collections.Generic;
3 using System.Linq;
4 using System.Xml;
5 using System.Web;
6 using System.Windows.Forms;
7 using System.Web.UI;
8 using System.Web.UI.WebControls;
9
10 using MeetingNotesApp.DBXL;

```

9. Now we declare the local variables and constant strings.
- Open the schema file created in earlier steps in preview mode and save the file locally. Open the saved xml file in a text editor and copy the XML to the clipboard.
 - Declare a constant string that holds the template xml as below (Note: the value to the **Template_XML** is the value that was saved in clipboard in the previous step).

```

private const string TEMPLATE_XML = @"<?xml version='1.0' encoding='UTF-
8'??>

    <?mso-infoPathSolution
    solutionVersion='1.0.0.4' productVersion='12.0.0'
    PIVersion='1.0.0.0'
    href='manifest.xsf' ?>
    <?mso-application progid='InfoPath.Document'
versionProgid='InfoPath.Document.2'??>
    <my:myFields xmlns:xsi='http://www.w3.org/2001/XMLSchema-
instance'

xmlns:my='http://schemas.microsoft.com/office/infopath/2003/myXSD/2010-04-
19T17:27:00'

```



```

xmlns:xd='http://schemas.microsoft.com/office/infopath/2003' xml:lang='en-us'>
    <my:MeetingTopicFld></my:MeetingTopicFld>
    <my:MeetingNotesFld></my:MeetingNotesFld>
    <my:AttendeesFld></my:AttendeesFld>
</my:myFields>";

```

- c. Create an instance to **DBXLDocumentService**

```

private DbxlDocumentService docService = new
    DbxlDocumentService();

```

- d. Declare member variables that matches the schema.

```

Private string author;
private string mtgTopic;
private string mtgNotes;
private string mtgAttendees;

```

- e. Declare the below variables.

```

private XmlDocument document;
private XmlNamespaceManager nsmgr;
private const string c_Namespace =
"http://schemas.microsoft.com/office/infopath/2003/myXSD/2010-04-19T17:27:00";

```

10. Create a static constructor. Add code that calls **CreateXmlDocument** (note: **CreateXmlDocument** method is defined in the next step).

```

static SubmitData()
{
    document = CreateXmlDocument(TEMPLATE_XML, "my", c_Namespace,
out nsmgr);
}

```

11. Create a method to create an XML document.

```

/// <summary>
/// This method creates an xml document
/// </summary>
/// <param name="xml">xml string</param>
/// <param name="prefix">prefix</param>
/// <param name="ns">namespace</param>
/// <param name="namespaceManager">namespacemanager</param>
/// <returns></returns>
internal static XmlDocument CreateXmlDocument(string xml, string
prefix, string ns, out XmlNamespaceManager namespaceManager)
{

```



```

        XmlDocument doc = CreateXmlDocument(xml);
        namespaceManager = new XmlNamespaceManager(doc.NameTable);
        namespaceManager.AddNamespace(prefix, ns);

        return doc;
    }

    /// <summary>
    /// This method creates an xml document
    /// </summary>
    /// <param name="xml">xml string</param>
    /// <returns></returns>
    internal static XmlDocument CreateXmlDocument(string xml)
    {
        XmlDocument doc = new XmlDocument();
        doc.LoadXml(xml);

        return doc;
    }

```

12. Now create a submit button click event to submit data. This event invokes **SubmitDocument** method from **DBXLDocument** web service.

```

/// <summary>
/// This method submits data using DBXLDocument service.
/// </summary>
/// <param name="sender"></param>
/// <param name="e"></param>
protected void SubmitBtn_Click(object sender, EventArgs e)
{
    XmlDocument doc = GetClone(document);

    int docId;
    string refId;

    // check if author field is null or not.
    if (!string.IsNullOrEmpty(authorFld.Text))
    {
        author = authorFld.Text;
    }

    // check if meeting topic field is null or not.
    if (!string.IsNullOrEmpty(mtgTopicFld.Text))
    {
        mtgTopic = mtgTopicFld.Text;
    }

    // check if meeting notes field is null or not.
    if (!string.IsNullOrEmpty(mtgNotesFld.Text))
    {
        mtgNotes = mtgNotesFld.Text;
    }
}

```



```

        // check if meeting attendees field is null or not.
        if (!string.IsNullOrEmpty(mtgAttendeesFld.Text))
        {
            mtgAttendees = mtgAttendeesFld.Text;
        }

        // modify the document with values.
        UpdateXmlDoc(ref doc, mtgTopic, mtgNotes, mtgAttendees);

        // always use default credentials.
        docService.UseDefaultCredentials = true;

        StatusInfo status =
        docService.SubmitDocument("submitTemplate", document.InnerXml, mtgTopic,
        author, mtgNotes, out docId, out refId);

        if (status.Success)
        {
            idFld.Text = docId.ToString();
            MessageBox.Show("The data is submitted successfully!");
        }
        else
        {
            MessageBox.Show("The data failed to submit!");
        }
    }

    /// <summary>
    /// Creates a clone of the specified XmlDocument that can be
    freely modified without affecting the original one.
    /// </summary>
    /// <param name="original">A valid XmlDocument</param>
    /// <returns></returns>
    private static XmlDocument GetClone(XmlDocument original)
    {
        return (XmlNode)original.Clone();
    }

    /// <summary>
    /// Updates a given xml doc with new values.
    /// </summary>
    /// <param name="doc">XML Document</param>
    /// <param name="topic">Meeting Topic Field</param>
    /// <param name="notes">Meeting Notes Field</param>
    /// <param name="attendees">Attendees Field</param>
    private static void UpdateXmlDoc(ref XmlDocument doc, string topic, string
    notes, string attendees)

```



```

    {
        XmlNode myFields = doc.SelectSingleNode("my:myFields", nsmgr);
        myFields.SelectSingleNode("my:MeetingTopicFld", nsmgr).InnerText
= name;
        myFields.SelectSingleNode("my:MeetingNotesFld", nsmgr).InnerText
= author;
        myFields.SelectSingleNode("my:AttendeesFld", nsmgr).InnerText =
desc;
    }

```

13. If you carried out all of the above steps successfully then your submit data code should be in place and ready to test. Now all you have to do is compile and run the program. Enter data in the fields (**Author**, **Meeting Topic**, **Meeting Notes**, and **Attendees**) and click on the **Submit** button.

Id:	
Author:	John Smith
Meeting Topic:	InfoPath bugs triage
Meeting Notes:	Code complete: 5/14; Bug bash: 5/8; Bug triage: 5/9; Discuss blocking issues.
Attendees:	John Smith; Kevin Harris; Cliff Hord;
Submit	

14. The form is submitted and returns the corresponding document id back in the form. Also, the data

Id:	707
Author:	John Smith
Meeting Topic:	InfoPath bugs triage
Meeting Notes:	Code complete: 5/14; Bug bash: 5/8; Bug triage: 5/9; Discuss blocking issues.
Attendees:	John Smith; Kevin Harris; Cliff Hord;
Submit	



Administration Tool

Document Type: submitTemplate Save Save As... Catalog

General Database Permissions SharePoint Documents **Taxonomy Tree**

Export Documents

Import Documents

Query Documents

To limit the number of documents shown, enter a starting Doc. ID and the number of documents to show, and then click query. You can then jump forward or backward by the number of shown documents, or click Query All to show all documents. To search for a Document with a specific Ref. ID, type the ID into the field, and then click Query.

Starting Doc. ID: 707 Ref. ID: Range: 20 Previous Next Query Query All

Refresh

Doc. ID	Ref. ID	Version #	Name	Author	Description	New
707	32	1	InfoPath bugs triage	John Smith	Code complete: 5/14; Bug bash: 5/8; Bug triage: 5/9; Discuss blocking issues.	Open

15. Open SQL Server and verify shredded data.

```
Select * From MeetingTable
```

MtgTopic	MtgNotes	Attendees
1 InfoPath bugs triage	Code complete: 5/4; Bug bash: 5/8; Bug triage: 5/9; Discuss blocking issues.	John Smith; Kevin Harris; Cliff Hord;

SUPPORT

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

Licensed customers can contact us via Support@Qdabra.com. Alternatively, please use the InfoPathDev.com [Qdabra](http://Qdabra.com) Product support forums to request help.



<http://www.qdabra.com>

Last updated on 6/29/2010 12:05 PM

Copyright © 2006-2010 Autonomy Systems, LLC. All rights reserved.