Developing an app using Web Services, DB2, and .Net

Presented by the DB2 Developer Domain

http://www7b.software.ibm.com/dmdd/

Table of contents

If you're viewing this document online, you can click any of the topics below to link directly to that section.

1. Before you start ................................................................................... 2
2. Overview .............................................................................................. 4
3. Getting started ..................................................................................... 6
4. Setting up the database ....................................................................... 8
5. Working with the GetRateWS function .............................................. 10
6. Creating a customer application ........................................................ 13
7. How it all works .................................................................................. 19
8. Summary and resources ................................................................... 20
Section 1. Before you start

About this tutorial

In this tutorial, you will build a sample customer application that invokes Web services as user defined functions. This tutorial demonstrates how Web services calls can be easily combined with SQL to create DB2 applications on the Windows platform.

You will use IBM DB2 Add-ins for Visual Studio .Net to create this application. IBM DB2 Add-ins provide rapid application development for DB2 Universal Database on Microsoft Visual Studio .NET.

Should I take this tutorial?

You should take this tutorial if:

- You want to develop windows applications to invoke Web services using DB2 and SQL.
- You want to learn more about using the DB2 Development Add-ins for Visual Studio .NET to rapidly create such Windows applications.

Prerequisites

To complete the steps in this tutorial, you need the following software from IBM and Microsoft:

- IBM DB2 Universal Database V8.1.2 Application Development Client
- Microsoft Visual Studio .NET

About the author

Sonali Surange is an Advisory Software Engineer working at the IBM San Francisco lab, on Data Management Application Development Tools. Currently, Sonali is working on providing tools for seamless integration with IBM Servers on Visual Studio .Net.
Trademarks

IBM, DB2 and DB2 Universal Database are registered trademarks of the IBM Corporation in the United States and/or other countries.

Microsoft and Windows are registered trademarks of Microsoft Corporation in the United States, other countries, or both.

Other company, product and service names may be trademarks or service marks of others.
Section 2. Overview

Web services functions

DB2 Universal Database can optimize access to Web services by using SQL statements to consume Web services data.

DB2 table or scalar functions are created from WSDL interfaces that define the Web service. The data returned from the Web service can then be used within the context of your SQL, and in your DB2 applications.

Sample customer application

In this sample customer application, you will create a customer table in the DB2 database using the SQL scripts provided as part of the download for this tutorial. The customer table contains customer orders; each order contains a customer number and order in USD.

Using the SQL scripts provided, you will also create a Web service user defined function. This user defined function will consume a currency conversion Web service from www.xmethods.net. You will use this function to obtain the live conversion rate from USD to EUROS. You will need a live connection to the Internet.

Finally, you will update the converted currency amount in the database.

The main steps are as follows:
○ Enable DB2 for Web service user defined functions.
○ Set up tracing to view soap request and response messages (optional).
○ Create and initialize the customer table and create the GetRateWS Web service user defined function.
○ View and run the Web service user defined function using IBM Explorer.
○ Create Windows applications to invoke the Web services user defined function.
○ From the windows application, update the DB2 table with the values obtained.

This tutorial demonstrates:

○ Creating the Web Service user defined function using IBM DB2 Projects.
○ Working with the created function using IBM Explorer to:
  1. View the source
  2. Execute the function
○ Viewing the soap request and response messages using tracing.
○ Creating a Windows application to invoke the above function.
○ Updating the data received back in the DB2 table.
Section 3. Getting started

Enable DB2 for Web service user defined functions

To use Web services consumer user defined functions, you first have to create helper functions on your database that are required for this functionality.

Follow the steps below.

1. Enable DB2 for XML Extender.

To enable the sample database for XML Extender, run the following command from a DB2 command prompt: `dxxadm enable_db sample`

Replace `sample` with the appropriate name if you choose to work with another database.

2. Enable DB2 for Web service user defined functions.

To enable the sample database for Web service user defined functions, run the following command from a DB2 command prompt: `db2enable_soap_udf -n sample -u db2admin -p db2admin`.

Replace `sample, username, and password` with an appropriate database name, your DB2 username and password.

Verify that the database was enabled

To make sure the database was enabled:

1. Click **View => IBM Explorer** to open the IBM Explorer window.
2. Add a connection to the sample database, or the appropriate database enabled in the earlier step.
3. Under the Functions folder in IBM Explorer, you will see the DB2 user defined functions required for the XML extender, as shown below.
4. Under the Functions folder in IBM Explorer, you will see the DB2 user defined functions required for the Web services functionality, as shown below.

Set up tracing to view soap messages

You can optionally trace soap request and response messages when the Web service is invoked. These messages are saved into a file on your machine.

To set up tracing, set the system environment variable DB2SOAP_TRACE to a fully qualified file name.

DB2SOAP_TRACE=c:\db2vsnetsap_trace\db2vsnetsap_trace.xml.
Section 4. Setting up the database

Create database objects

Download the sql scripts included with this tutorial. Use the IBM DB2 projects to create these objects.

---

View the user defined function

Using IBM Explorer, view the function that was created in the earlier step.

1. Add a filter to the functions folder to view all functions for your schema. Click Functions => Filter.
2. Enter a schema name in the Schema field.
Section 5. Working with the GetRateWS function

Executing the GetRateWS function

Using IBM Explorer, you can perform tasks such as executing the GetRateWS function or viewing the source of this function.

1. Right-click the GetRateWS function and select **Run function** from the menu.

![IBM Explorer](image)

2. Add parameters to run this function. For example, add **US** and **EURO** to obtain conversion rate from USD to EUROS.

3. Click OK.

![Parameter Values](image)

4. You will see the conversion rate in the grid. The GetRateWS function has obtained the live conversion rate from the Web service running at www.xmethods.net.
Viewing the soap messages in trace

Viewing the source

To view the source for the GetRateWS function:

1. Right-click GetRateWS and select View Source.

2. View the source of this function in the IBM DB2 editor.
CREATE FUNCTION SSURANCE.GETRATESUS(
    COUNTRY1 VARCHAR(100),
    COUNTRY2 VARCHAR(100)
) RETURNS DOUBLE LANGUAGE SQL CONTAINS SQL
EXTERNAL ACTION NOT DETERMINISTIC
RETURN with
    soap_input (in)
AS
VALUES varchar(
    '<?xml:namespace prefix="m" '
    xmlns:m="urn:xmethods-CurrencyExchange"
    SOAP-ENV:encodingStyle="
        http://schemas.xmlsoap.org/soap/encoding/"'>
    <m:getRate xmlns:m=""
        xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
        xsi:schemaLocation="http://services.xmethods.net:80/soap"
        xmlns:soap="http://schemas.xmlsoap.org/soap/encoding/">
    <country1 xsi:type="xsd:string">'
    |  country11 | '</country1>
    |  <country2 xsi:type="xsd:string">'
    |  country21 | '</country2>
    |  '</m:getRate>' },
soap_output (out)
AS
VALUES db2xml.soaphttpv(
    'http://services.xmethods.net:80/soap', '',
    (SELECT in FROM soap_input))
)
select
    db2xml.extractDouble(db2xml.xmlclob(x.out), '/^\*/')
from soap output x
Section 6. Creating a customer application

Create a Windows application

Using the IBM DB2 Add-ins, create a Windows application to invoke the getRateWS Web service created in the previous section.

To create a data-bound Windows application:

1. Create a Windows C# application and name it customerApp.
2. Open Form1.cs in design mode.
3. Drag and drop a Windows data grid control on the form. Use this to data-bind the customer table.
4. Drag and drop the customer table from IBM Explorer. This creates db2DataAdapter1.
5. Click on db2DataAdapter1 and generate DataSet. Name the dataset DB2DataSet_customer.

6. Set the DataSource property of the dataGrid1 to DB2DataSet_customer.CUSTOMER. This binds the data grid to the
dataset at design time.

7. Add the following code (bold) in the beginning of Form1.cs:
   ```csharp
   using IBM.Data.DB2;
   ```

8. Write the following code (bold) in the form load event handlers for the form:
   ```csharp
   private void form1_load(object sender, System.EventArgs e)
   {
      this.db2DataAdapter1.Fill(this.DB2DataSet_customer);
   }
   ```

---

Update the currency amount

To update the currency amount:

1. Open Form1.cs in design mode.
2. Drag and drop a button onto the form and set the Name property to `updateEuroAmt` and the text property to "Update Amount in Euros".
3. Click on `db2DataAdapter1` and in the properties section, select the **Update Command** property. Change the command text for the update command to the SQL (bold) listed below. Replace the schema name `ssurange` with the schema name used to create `getRateWS` and `customer`.

   ```sql
   UPDATE customer
   ```
update SSURANGE.CUSTOMER SET euroamt=decimal(getRateWS('US','EURO')
*decimal(usdamt,7,2),7,2)

4. Add the following code (bold) in the click event handler for the
updateEuroAmt button.

```csharp
private void updateEuroAmt_Click(object sender, System.EventArgs e)
{
    try
    {
        this.db2DataAdapter1.UpdateCommand.Connection.Open();
        int count = this.db2DataAdapter1.UpdateCommand.ExecuteNonQuery();
        if (count > 0)
        {
            MessageBox.Show("Euro amount successfully updated
using a live conversion rate!");
        }
    }
    catch (DB2Exception exp)
    {
        MessageBox.Show(exp.Message);
    }
}
```

Refresh values from the database

To view the updated currency values from the database:

1. Open Form1.cs in design mode.
2. Drag and drop a button onto the form and set the Name property to
   refresh and its text property to "Refresh From Database".
3. Write the following code (bold) in the refresh button click event handler for the form:

```csharp
private void refresh_click(object sender, System.EventArgs e)
{
    this.db2DataAdapter1.Fill(this.DB2DataSet_customer);
}
```

4. Right click on the customerApp project and mark it as a Start-up project.

5. Build the customerApp project.

---

**Testing the customer application**

To test the customer application:

1. Run the customer application. View the values of the customer's order in USD from the customer table.
2. Click **Update Euro Amount**. View the message, "Euro amount successfully updated using a live conversion rate!"

3. Click **Refresh** to view the values in Euros in the grid. The values are USD amount converted to EURO amount, based on the conversion rate at the time the update button is clicked.
<table>
<thead>
<tr>
<th>CUSTOMER</th>
<th>ORDERNO</th>
<th>USDAMT</th>
<th>EUROAMT</th>
</tr>
</thead>
<tbody>
<tr>
<td>000001</td>
<td>000001</td>
<td>10.5</td>
<td>9.11</td>
</tr>
<tr>
<td>000001</td>
<td>000002</td>
<td>200.82</td>
<td>174.27</td>
</tr>
<tr>
<td>000001</td>
<td>000003</td>
<td>0.76</td>
<td>0.65</td>
</tr>
</tbody>
</table>
Section 7. How it all works

How it all works

A customer table is created with customer order amounts in USD. This is data bound to the data grid using the data adapter db2DataAdapter1, which is created for the customer table. When the application starts, the select statement of the data adapter is executed and the existing orders with USD amounts are displayed.

The update command for db2DataAdapter1 is used to calculate and update the corresponding EURO amounts. The conversion rate is obtained by running the GetRateWS user defined function; that in turn invokes the currency conversion Web service from www.xmethds.net. The rate obtained is applied to each of the USD amounts to obtain their corresponding EURO amounts. All of the above is achieved by running one SQL statement as listed below:

```sql
update SSURANGE.CUSTOMER SET euroamt=decimal(getRateWS('US','EURO')
*decimal(usdamt,7,2),7,2)
```
Section 8. Summary and resources

Summary

You used the IBM DB2 Add-ins for Microsoft Visual Studio .Net to create tables, and DB2 Web Service User Defined functions.

You used IBM Explorer to run the Web service user defined function to get a live currency conversion rate, and to view the source for this function.

You also used soap tracing to view the soap request and response sent and received when the Web service user defined function was executed.

Using IBM Explorer, you successfully created the sample windows application to view and update the DB2 table information using SQL to invoke Web services.

Resources

Refer to the following resources for more information:

- IBM DB2 Development Add-ins for Visual Studio .Net
- Developing DB2 UDB Database Projects in Visual Studio .Net
- Exploring DB2 Data Connections in Visual Studio .Net
- Developing Web Services and Messaging DB2 Applications
- Developing an application using Websphere MQ and Visual Studio .Net

Feedback

Colophon

This tutorial was written entirely in XML, using the developerWorks Toot-O-Matic tutorial generator. The open source Toot-O-Matic tool is an XSLT stylesheet and several XSLT extension functions that convert an XML file into a number of HTML pages, a zip file, JPEG heading graphics, and two PDF files. Our ability to generate multiple text and binary formats from a single source file illustrates the power and flexibility of XML. (It also saves our production team a great deal of time and effort.)