

Accessing DB2 from Ruby

Contents

1. Introduction	2
2. Prerequisites	2
3. Setup	2
Populate DB2 Database	2
Set up ODBC Access	2
4. Discovery.....	3
4.1. Web Service Creation using SOA Gateway	3
4.2. Accessing the WSDL	5
5. Accessing a Web Service with Ruby.....	7
Ruby Editor.....	8
Ruby Code	8
Debugging Code	8
6. Conclusion.....	9
7. Appendix	10

1. Introduction

In this tutorial we will show you how to build a Ruby application to access DB2 via the SOA Gateway.

2. Prerequisites

It is assumed that you are running the 3 components, DB2, Ruby and the SOA Gateway on Windows.

It is assumed you already have a SOA Gateway server and Control Centre installed. See [here](#) for more info about installing the SOA Gateway.

3. Setup

Download the latest binaries for Ruby from the [Ruby downloads page](#), install it according to the install.txt document in the Ruby distribution library. For this tutorial I downloaded Ruby 1.8.6 One-Click Installer and followed the default options.

You will also need a DB2 database. IBM provides a downloadable edition of DB2, called “DB2 Express-C”. See [this link](#) for the DB2 Express-C homepage. Download and install a version of DB2 Express.

Populate DB2 Database

Now that you’ve got DB2 installed, we need to populate it with some demo data. For this we’ll use the Risarisk Bank Demo, which is available [here](#). Save this file to “c:\Temp\RisariskBank_db2.sql”.

- Open the DB2 Control Centre under “IBM DB2”, “General Admin Tools” in the Start Menu.
- Right click “All databases” and select “Create Database”, “Standard”.
- Name your new database “RISBANK”. All other options can be left as default, so click “Finish”.
- Open a DB2 command shell by typing “db2cmd” in a DOS box.
- From the db2 command shell, change directory to where you downloaded the RisariskBank_db2.sql. E.g “cd \temp”.
- Populate the RISBANK database by running the command “db2 -f RisariskBank_db2.sql”. Note you may see errors about “SYSTEM.CUSTOMERINFORMATION is an undefined name”. These occur because the RisariskBank_db2.sql attempts to drop any existing tables before creating new ones. To prove this, you can run the same command again, and the errors will disappear.
- You can now return to the DB2 Control Centre and view the newly created tables in the RISBANK database.

Set up ODBC Access

The final thing to do with your DB2 Database is to set up an ODBC DSN which will be used by the SOA Gateway to access this database.

Click Start, Control Panel, Administrative Tools, Data Sources (ODBC)

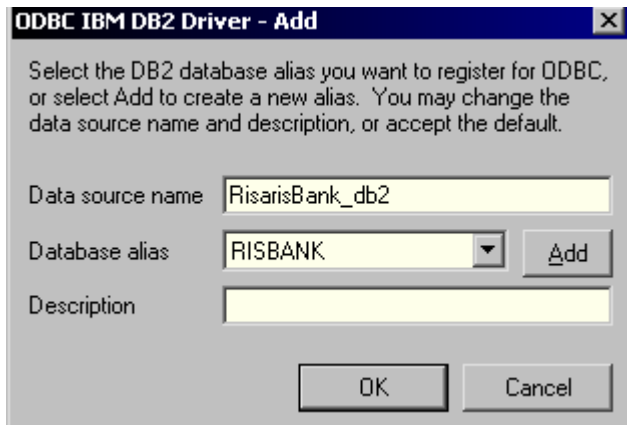
From the resulting screen, choose the “System DSN” Tab.

Click Add

From the list of data source drivers, select “IBM DB2 ODBC DRIVER”, and click “Finish”.

Enter “RisarisBank_db2” as the Data source Name.

Ensure that the Database Alias is RISBANK, and click “OK”.



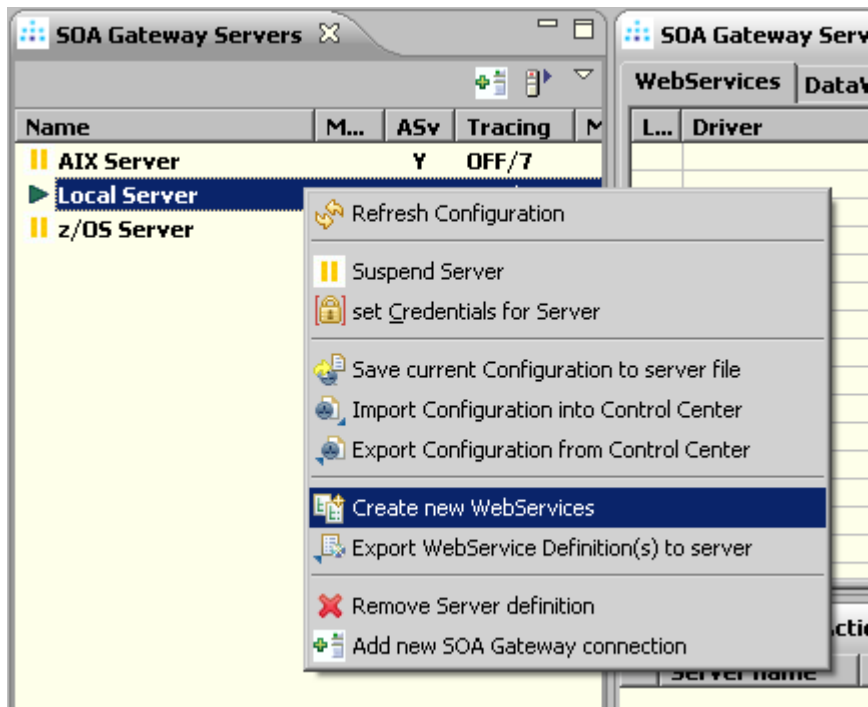
4. Discovery

At this stage you’ve got Ruby installed, and a DB2 database with some sample data in it. In this section we’ll show you how to create web services from each of the DB2 tables. These web services can be used by the Ruby language (and many others) to give you direct real-time access to your DB2 Data.

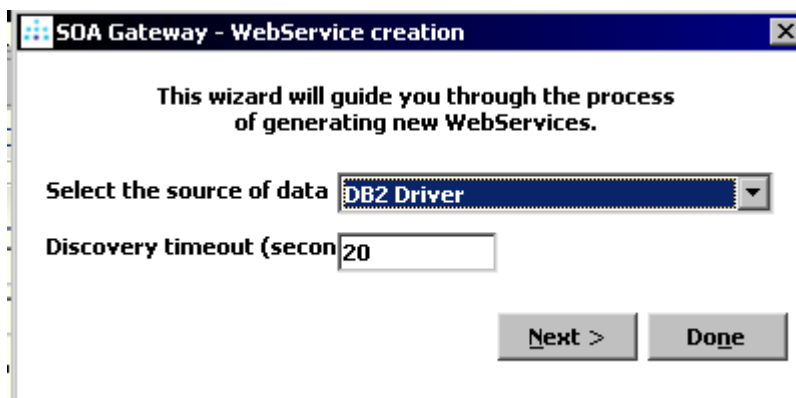
4.1. Web Service Creation using SOA Gateway

Start your SOA Gateway Control Centre. See [here](#) for an introduction to the Control Centre.

In your servers view, right click the entry which represents your local SOA Gateway Server. Select “Create New Web Services”.



From the next dialog, choose “DB2 Driver”. If you do not see have a DB2 Driver in the list, see how to create one [here](#).



Click Next.

The next screen gives you the ability to add information about your DSN

For each of the 8 web services you've created in the previous section, the SOA Gateway provides you with a WSDL to describe the Web Service. The WSDL itself is usually interpreted by a web service client, such as Ruby, but it is useful to know where to find the WSDL for each of your Web Services.

As WSDL is XML-based, it will open in your browser of choice. To see the WSDL for one of your Risar Bank web services, do the following in your SOA Gateway Control Centre:

- Click on the web service you are interested in, for example the "branch" web service.
- The properties for this web service should appear in your [Properties View](#). If you do not see the Properties view, select Window -> Show View -> Other -> General -> Properties and click OK.
- In the properties view, there is a link to your WSDL. Click it to open the WSDL in a browser.

The screenshot displays the SOA Gateway Server Configuration interface. It is divided into three main sections:

- WebServices:** A table listing various web services. The 'BRANCH_SYSTEM' service is highlighted in blue.

Mod	Driver	WebService	DataSource Id
DB2.	DB2 Driver	ACCOUNTSMOVEMENTS_SY...	odbcDsn=RisarisBank_db2, schen
DB2.	DB2 Driver	AUDIT_SYSTEM	odbcDsn=RisarisBank_db2, schen
DB2.	DB2 Driver	BRANCH_SYSTEM	odbcDsn=RisarisBank_db2, schen
DB2.	DB2 Driver	CURRENTACCOUNT_SYSTEM	odbcDsn=RisarisBank_db2, schen
DB2.	DB2 Driver	CUSTOMERACCOUNTXREF_S...	odbcDsn=RisarisBank_db2, schen
DB2.	DB2 Driver	CUSTOMERINFORMATION_S...	odbcDsn=RisarisBank_db2, schen
DB2.	DB2 Driver	DEPOSITACCOUNT_SYSTEM	odbcDsn=RisarisBank_db2, schen
DB2.	DB2 Driver	TELLERTABLE_SYSTEM	odbcDsn=RisarisBank_db2, schen
- SOA Gateway Action Log:** Shows a message: "Local Server ODBC discovery completed, 8 WebService(s) generated".
- Properties:** The 'WebService properties' view for 'BRANCH_SYSTEM' is shown. A green arrow points to the 'Driver' dropdown menu. The WSDL URL is http://localhost:56000/BRANCH_SYSTEM?WSDL. Below, the 'WebService Identification and options' section shows:
 - odbcDsn: RisarisBank_db2
 - schemaName: SYSTEM
 - tableName: BRANCH

You can view the WSDL for the other web services by clicking the link from their properties view.

This WSDL is the starting point for using Web Services, and can be used time and again by different web service clients.

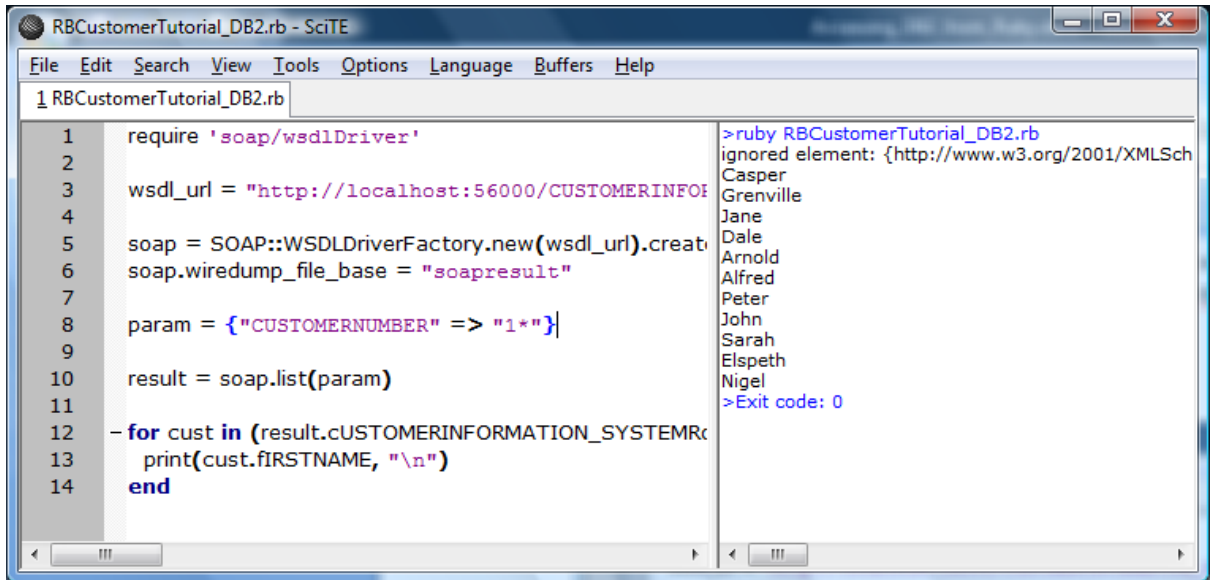
5. Accessing a Web Service with Ruby

We will use a Ruby script to access our new Risaris Bank Web Services via the WSDL. Those familiar with Ruby will probably want to change the script provided but it can be run with minimum changes described below. The whole script is provided in APPENDIX A.

Ruby Editor

Use the editor provided, SciTE.exe, which is located in folder scite where you installed Ruby. Copy the code from Appendix A and save the code as the filename RBCustomerTutorial_DB2.rb (as in example below).

Hit F5 and the results will appear in the right-hand pane:



The screenshot shows the SciTE editor window titled 'RBCustomerTutorial_DB2.rb - SciTE'. The left pane contains the following Ruby code:

```
1 require 'soap/wsdlDriver'
2
3 wsdl_url = "http://localhost:56000/CUSTOMERINFORMATION_SYSTEM?WSDL"
4
5 soap = SOAP::WSDLDriverFactory.new(wsdl_url).create_driver
6 soap.wiredump_file_base = "soapresult"
7
8 param = {"CUSTOMERNUMBER" => "1*"}
9
10 result = soap.list(param)
11
12 - for cust in (result.CUSTOMERINFORMATION_SYSTEMRoot.CUSTOMERINFORMATION_SYSTEMGroup)
13   print(cust.FIRSTNAME, "\n")
14 end
```

The right pane shows the output of the script:

```
>ruby RBCustomerTutorial_DB2.rb
ignored element: {http://www.w3.org/2001/XMLSchema}
Casper
Grenville
Jane
Dale
Arnold
Alfred
Peter
John
Sarah
Elspeth
Nigel
>Exit code: 0
```

Ruby Code

Note the WSDL URL http://localhost:56000/CUSTOMERINFORMATION_SYSTEM?WSDL. Again depending on your SOA Gateway configuration you may need to change the server/port number.

In this example provided we are issuing a **list** request on the customer information table in the RirisBank database. The list request can handle a wildcarded customer number key:

```
param = {"CUSTOMERNUMBER" => "1*"}
```

```
result = soap.list(param)
```

Finally we can query the result for the customer's first name:

```
for cust in
(result.CUSTOMERINFORMATION_SYSTEMRoot.CUSTOMERINFORMATION_SYSTEMGroup)

  print(cust.FIRSTNAME, "\n")

end
```

Obviously this can be replaced by any of the column names in the customer information table. N.B. If adding any further columns to the results note that Ruby dictates the first letter is in lowercase e.g. `FIRSTNAME` vs `firstname`, `SURNAME` vs `surname` etc.

Debugging Code

The code provided should work off the bat.

The `p` and `puts` functions are useful for checking the content of variables. For example `p result` after the `result = soap.list(param)` call gives the following information and is useful for checking column names for adding new code.

```
#<SOAP::Mapping::Object:0x32da5ca
{}CUSTOMERINFORMATION_SYSTEMRoot=#<SOAP::Mapping::Object:0x32da4d0
{}CUSTOMERINFORMATION_SYSTEMGroup=[#<SOAP::Mapping::Object:0x32da3d6
{}CUSTOMERNUMBER="1" {}FIRSTNAME="Casper" {}SURNAME="Ankergren"
{}ADDRESSLINE1="34 Green Street" {}ADDRESSLINE2="Crosses Street" {}CITY="Mansfield"
{}POSTCODE="MA5 9AJ" {}DATEOFBIRTH="30/05/1978">, #<SOAP::Mapping::Object:0x32d8edc
{}CUSTOMERNUMBER="10" {}FIRSTNAME="Grenville" {}SURNAME="Dekker"
{}ADDRESSLINE1="22 Uttoxter New Road" {}ADDRESSLINE2="Chaddesden" {}CITY="Leeds"
{}POSTCODE="LS4 9WB" {}DATEOFBIRTH="21/10/1974">, #<SOAP::Mapping::Object:0x32d7b04
{}CUSTOMERNUMBER="11" {}FIRSTNAME="Jane" {}SURNAME="Freeman" {}ADDRESSLINE1="3
Portland Street" {}ADDRESSLINE2="Lichfield" {}CITY="Gloucester" {}POSTCODE="GL4 8KD"
{}DATEOFBIRTH="26/10/1980">, #<SOAP::Mapping::Object:0x32d6736
{}CUSTOMERNUMBER="12" {}FIRSTNAME="Dale" {}SURNAME="Rolling" {}ADDRESSLINE1="The
Hunters Rest" {}ADDRESSLINE2="Branston" {}CITY="Preston" {}POSTCODE="PR4 4HG"
{}DATEOFBIRTH="06/12/1989">, #<SOAP::Mapping::Object:0x32d5368
{}CUSTOMERNUMBER="13" {}FIRSTNAME="Arnold" {}SURNAME="Corrigan"
{}ADDRESSLINE1="14 Anderson Street" {}ADDRESSLINE2="Allestree" {}CITY="Derby"
{}POSTCODE="DE6 8FT" {}DATEOFBIRTH="10/06/1970">, #<SOAP::Mapping::Object:0x32d3f9a
{}CUSTOMERNUMBER="14" {}FIRSTNAME="Alfred" {}SURNAME="Summerscale"
{}ADDRESSLINE1="3701 S. George Mason" {}ADDRESSLINE2="Melbourne" {}CITY="Bath"
{}POSTCODE="BA6 8UU" {}DATEOFBIRTH="07/08/1982">, #<SOAP::Mapping::Object:0x32d2bcc
{}CUSTOMERNUMBER="15" {}FIRSTNAME="Peter" {}SURNAME="Spiegel"
{}ADDRESSLINE1="11663 Charter Oak Co" {}ADDRESSLINE2="Foremark" {}CITY="Manchester"
{}POSTCODE="M98 4GT" {}DATEOFBIRTH="29/12/1971">, #<SOAP::Mapping::Object:0x32d17fe
{}CUSTOMERNUMBER="16" {}FIRSTNAME="John" {}SURNAME="Sviridov"
{}ADDRESSLINE1="3319 Rosemere Court" {}ADDRESSLINE2="Brailsford" {}CITY="Leeds"
{}POSTCODE="LS8 5QQ" {}DATEOFBIRTH="12/06/1984">, #<SOAP::Mapping::Object:0x32d0430
{}CUSTOMERNUMBER="17" {}FIRSTNAME="Sarah" {}SURNAME="Hall" {}ADDRESSLINE1="4
Denmark Lane" {}ADDRESSLINE2="East Bridgford" {}CITY="Portsmouth" {}POSTCODE="PO9 7QN"
{}DATEOFBIRTH="12/02/1983">, #<SOAP::Mapping::Object:0x32cf062
{}CUSTOMERNUMBER="18" {}FIRSTNAME="Elspeth" {}SURNAME="Jones"
{}ADDRESSLINE1="12375 W. Ohio Circle" {}ADDRESSLINE2="Risley" {}CITY="Birmingham"
{}POSTCODE="B93 1IO" {}DATEOFBIRTH="28/08/1986">, #<SOAP::Mapping::Object:0x32cdc94
{}CUSTOMERNUMBER="19" {}FIRSTNAME="Nigel" {}SURNAME="Wyllis" {}ADDRESSLINE1="14
Borough Road" {}ADDRESSLINE2="Ockbrook" {}CITY="Leeds" {}POSTCODE="LS8 2WL"
{}DATEOFBIRTH="29/04/1981">>>
```

6. Conclusion

This tutorial shows how to access DB2 from Ruby using the SOA Gateway. As you can see, you have built a powerful application that uses Web Services to retrieve information in real-time.

7. Appendix

```
require 'soap/wsdlDriver'
```

```
wsdl_url = "http://localhost:56000/CUSTOMERINFORMATION_SYSTEM?WSDL"
```

```
soap = SOAP::WSDLDriverFactory.new(wsdl_url).create_rpc_driver
```

```
soap.wiredump_file_base = "soapresult"
```

```
param = {"CUSTOMERNUMBER" => "1*"}
```

```
result = soap.list(param)
```

```
for cust in (result.CUSTOMERINFORMATION_SYSTEMRoot.CUSTOMERINFORMATION_SYSTEMGroup)
```

```
  print(cust.FIRSTNAME, "\n")
```

```
end
```