

README

SAS[®] BI Web Services Wizard

SAS BI Web Services Wizard is a code generating wizard for Microsoft Visual Studio .NET, Visual Studio .NET 2003, and Visual Studio 2005. It simplifies the process of calling a SAS stored process through SAS BI Web Services for Java or .NET. The wizard will generate a class with static methods to call any stored process that can be discovered through SAS BI Web Services.

Installation

The SAS BI Web Services Wizard will run on any version of Windows where you can install all of the required software. This is typically Windows 2000, Windows XP, and Windows Server 2003.

The following software must be installed to use the SAS BI Web Services Wizard:

- Microsoft Visual Studio .NET, Visual Studio .NET 2003, or Visual Studio 2005
- Microsoft .NET Framework (1.0, 1.1, or 2.0)

You will also need a working install of SAS BI Web Services for Java or .NET.

Installation of SAS BI Web Services Wizard is very simple. The only option available is the installation folder for the required files. The installer will automatically detect which versions of Visual Studio .NET are installed, and copy the required files to the appropriate locations for each installed version. If there are no versions of Visual Studio .NET installed, the installer will display an error message and terminate.

After installation, you should have the following files in your installation folder (by default C:\Program Files\SAS\SAS BI Web Services Wizard):

- XmlaWizard.dll - assembly containing all SAS BI Web Services Wizard classes
- XmlaWizard.ico - icon for the SAS BI Web Services Wizard
- WizardDoc.html - this documentation file
- Images folder - contains images displayed by this documentation
- InstallClass.dll - assembly used by Microsoft Installer
- installclass.InstallState - InstallClass state

Also, depending on which versions of Visual Studio .NET are installed, you will have a number of new folders created inside the C# and VB.NET product directories each containing XmlaWizard.ico, XmlaWizard.vmdir, and XmlaWizard.vsz. These files are copied to the appropriate locations to make the SAS BI Web Services Wizard available in the list of new items.

Usage

Generating a New Class

After installation, you will be able to launch SAS BI Web Services Wizard from either a VB.NET or C# project. To launch the SAS BI Web Services Wizard:

1. In Visual Studio .NET, create a new VB.NET or C# project.
2. On the **Project** menu, click **Add New Item...**
3. On the **Add New Item** dialog, you should notice a new subfolder named "SAS Project Items." Select this folder and the SAS BI Web Services Wizard icon will be available in the list of available templates.
4. Double-click the SAS BI Web Services Wizard icon to launch the wizard.
5. After the SAS BI Web Services Wizard is launched, the welcome dialog opens. Click **Next** to continue.
6. A **Enter Required Information** dialog opens, in which you should enter the following:

- URL of your SAS BI Web Service WSDL file

This is the URL where clients can go to retrieve information about the Web service interface. It is used to create a Web service proxy class to call SAS BI Web Services. Some default URLs are listed in the drop down list, but you may enter a different URL if necessary.

- Class Name

This is the name of the class SAS BI Web Services Wizard is going to generate for you. If you already have a class with this name in your namespace, the wizard will generate it anyway and you must edit the name to correct this problem. The name of the file generated is based on this class name. If a file already exists with the name of the class specified, a unique file name will be generated for you.

- Namespace

This is the namespace that your new class will belong to. This field should be populated with the default namespace for your solution. You may only change the namespace if you are adding the class to C# project. Due to a limitation of VB.NET, this field cannot be changed if you are adding the class to a VB.NET project. After the code is generated, you are free to modify the namespace as you like.

After filling in the fields, click **Next**.

7. The **Verify Web Services URL** dialog opens, where you will need to verify the URL for your SAS BI Web Services installation. The URL displayed was parsed out of the WSDL file that was entered previously. In most situations this URL should be correct. If the URL contains `localhost`, we recommend that you specify the full DNS name of the machine instead of `localhost`. After you have verified your Web service URL is correct, click **Next**.
8. At this point, the Discover Web method of the specified SAS BI Web Services installation will be called to determine the SAS stored processes available. A list of the available stored processes will then be displayed in the **Select the SAS Stored Processes to Execute** dialog. Select the stored processes you want to execute from your class by checking the checkbox by the stored process name. Click **Next** to confirm your selections.

9. The **Confirmation** dialog opens. After you have confirmed that all the entered information is correct, click **Finish**.

A thread will now execute in the background to discover all the required information about the selected stored processes. Using this information, a class will be generated based on the parameters and streams of each stored process. There will be one static method per stored process in the class. You may notice that your file does not appear immediately if your Web service is slow. The time to generate the class is mostly limited by the time it takes to discover the stored process parameters.

Using the New Class in an Application

After you have created a new class using the SAS BI Web Services Wizard, you should be able to easily execute any of the stored processes you selected by calling the appropriate static method. The name of the static method is based on the full name of the stored process. For example, if the name of the stored process is `/Samples/StoredProcesses/Sample: MEANS Procedure Web Service`, the corresponding static method in the generated class would be

```
SamplesStoredProcessesSampleMEANSProcedureWebService().
```

The name of the static method was generated by stripping out all non-alphanumeric characters from the full name.

The stored process mentioned above is a sample stored process created as part of the SAS Web Infrastructure Kit install. It will also be installed if you do a SAS Software Navigator project install that creates a SAS Stored Process Server for you. If you need to create the metadata for the MEANS Procedure Web Service stored process, you can follow the directions to create it on the SAS BI Web Services Site, at

http://support.sas.com/rnd/itech/doc9/dev_guide/websrvcs/sample.html.

The `Sample: Means Procedure Web Service (means)` stored process takes one parameter, `tablename`, and one input stream, `instream`. `instream` is the input XML stream that we will be using as input data for PROC MEANS. `tablename` is the name of the table in the input stream that PROC MEANS will run against.

To execute the stored process with the class SAS BI Web Services Wizard generated, you would write code similar to the following:

C#

```
System.Data.DataSet ds = new System.Data.DataSet();
ds.ReadXml("c:\\mydoc.xml");
System.Xml.XmlDocument doc = new System.Xml.XmlDocument();
doc.LoadXml(ds.GetXml());

string tablename = "mydata";

System.Xml.XmlElement result =
    XMLA.SamplesStoredProcessesSampleMEANSProcedureWebService(
        tablename, doc.DocumentElement);
```

VB.NET

```
Dim ds As System.Data.DataSet = New System.Data.DataSet
ds.ReadXml("c:\\mydoc.xml")
Dim doc As System.Xml.XmlDocument = New System.Xml.XmlDocument
doc.LoadXml(ds.GetXml())
```

```
Dim tablename As String = "mydata"  
Dim result As System.Xml.XmlElement = _  
    XMLA.SamplesStoredProcessesSampleMEANSProcedureWebService( _  
        tablename, doc.DocumentElement)
```

First, create an XmlDocument object to hold your input stream data. You can get this input XML from any data source available to your application. Typically, in a .NET application, the source of this data would be an ADO.NET DataSet. In this case, the XML is loading from the file `c:\mydoc.xml`. The name of the table in the XML is mydata. Next MEANS is executed by simply calling the static method passing in the `tablename` and `instream` parameters. When the method returns, the results will be available in the XmlElement result. The XML in result can now be used in your application:

- create a new ADO.NET DataSet to use for data binding
- transform the XML into HTML using XSLT to generate custom reports
- create a graph or chart using GDI+

Note that you can modify the code generated by SAS BI Web Services Wizard as you see fit. You may want to modify the default PropertyList, or you may want to specify a different URL for the Web service. If you somehow break the code, you can simply re-run the wizard to create a new working copy.

Known Issues

SAS BI Web Services Wizard is not available as a new item in a Web project in Visual Studio 2005 or Visual Web Developer. One workaround is to invoke the Wizard in a separate class library project and then manually move the generated code to your Web project. You will also need to manually add the Web Reference to your Web project. Another workaround is to mark the generated static methods as public and then add a reference to the assembly to your Web project.

Comments or Suggestions

If you find the SAS BI Web Services Wizard helpful or if you have suggestions for enhancing this product, we would like to hear from you. Direct emails regarding the SAS BI Web Services Wizard to webservices@sas.com.

SAS and all other SAS Institute product or service names are registered trademarks or trademarks of SAS Institute Inc. in the USA and other countries. Other brand and product names are registered trademarks or trademarks of their respective companies.

® indicates USA registration.

Copyright © 2006 SAS Institute Inc. Cary, NC, USA. All rights reserved.