Writing Applications - Broker ActiveX Control

This chapter covers the following topics:

- Calling a Broker Function
- Viewing the Type Library
- Adding the Broker ActiveX Control Component to Visual Studio
- Using Internationalization with Broker ActiveX Control
- Using the Property Pages

Calling a Broker Function

Setting the Broker ActiveX Properties

You can set the Broker ActiveX properties either in the program or in the property pages.

Specifying the Send Parameters

Before executing a `send` function, specify the send parameters with the method `SetSendDataLong(String bsData, Long DataLen)` or `SetSendData(String bsData, Short DataLen)`.

This method sets only the send buffer.

The first parameter specifies the buffer that has to be sent to the server. The second parameter specifies the number of bytes to be transferred.

The following rules apply to the `SetSendData` method:

- The `DataLen` bytes of the string `bsData` are copied to the internal send buffer.
- A byte copy is performed (not a string character copy), which means that the string `bsData` can contain zero bytes.
- The function `BOOL SetSendData( String bsData, Short DataLen )` can be used if the send buffer is smaller than 32 KB.

Calling the Broker Function

- Set the required properties.
- When you use the `send` function, use the method `SetSendData` to set up the send buffer.

- When you use the `receive` function, use the property `ReceiveBufferSize` to set up the size of the internal receive buffer.
- Use the static automation method to call the Broker functions:

```
BOOL InvokeBrokerFunction()
```

This method executes the Broker function defined by the current value of the property `Function`. Depending on the function, the required Broker parameters are taken from the current values of the corresponding properties.

If the Broker call is successful:

- The function returns TRUE.
- The `ErrorCode` property is set to ’00000000’ and the `ErrorMsg` property is empty.

If the Broker call is a `Send` or `Receive` function, this call may also update the `ConvID` property.

If the Broker call is a `Receive` function and asterisks were specified for ServerClass, ServerName and Service, the call updates the `ServerClass`, `ServerName` and `Service` properties.

If the Broker call is a `Receive` or `Send` with implicit `Receive` (Wait > 0), the number of bytes received is stored in the property `ReturnDataLength` and the returned data can be retrieved with the `GetReceiveData` method.

If the Broker call fails:

- The function returns FALSE.
- The `ErrorCode` and `ErrorMsg` properties contain the corresponding error reason.

The error code has two parts:

- error class (first four digits), which provides information for the application on how to react to the returned error, and
- error number (last four digits), which indicates the reason for the error.

The `GetErrorText` method is still available and returns the value of the `ErrorMsg` property.

For more information see *Error Messages and Codes*.

Getting the Contents of the Receive Buffer

If a `Receive` function was executed, the receive buffer can be retrieved with the function

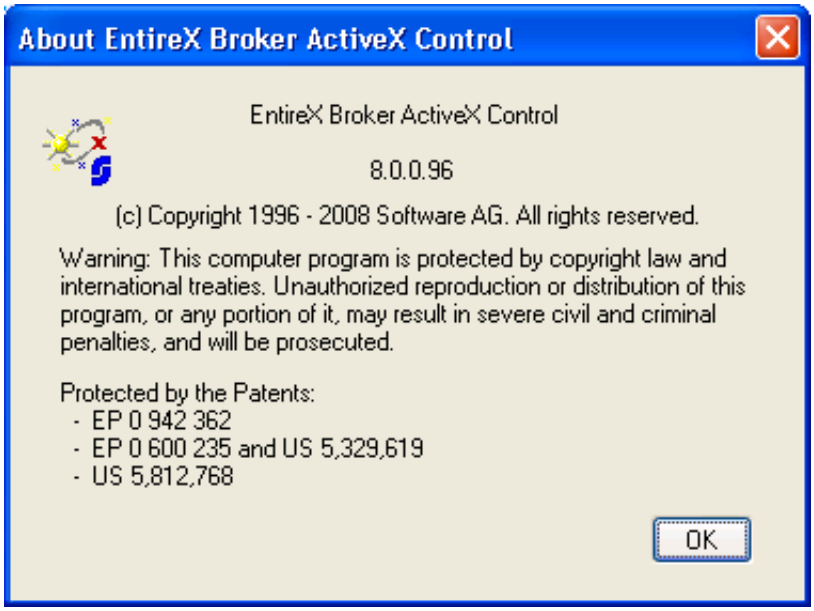
```
STRING GetReceiveData()
```

AboutBox

The `AboutBox` method is used to show the version of Broker ActiveX Control.

A message box will be displayed containing the **About** information.

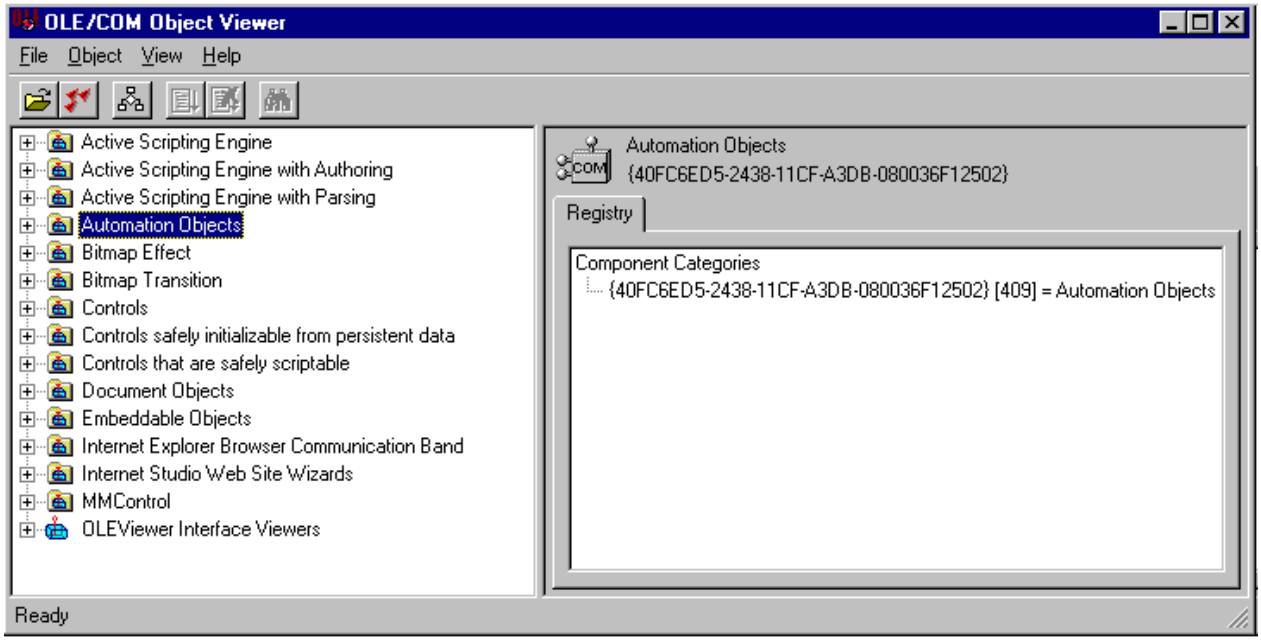
```
AboutBox ()
```



Viewing the Type Library

To view the Type Library of Broker ActiveX Control

- Use the OLE/COM Object Viewer (choose **EntireX Broker ActiveX Control** and choose **View Type Information**).

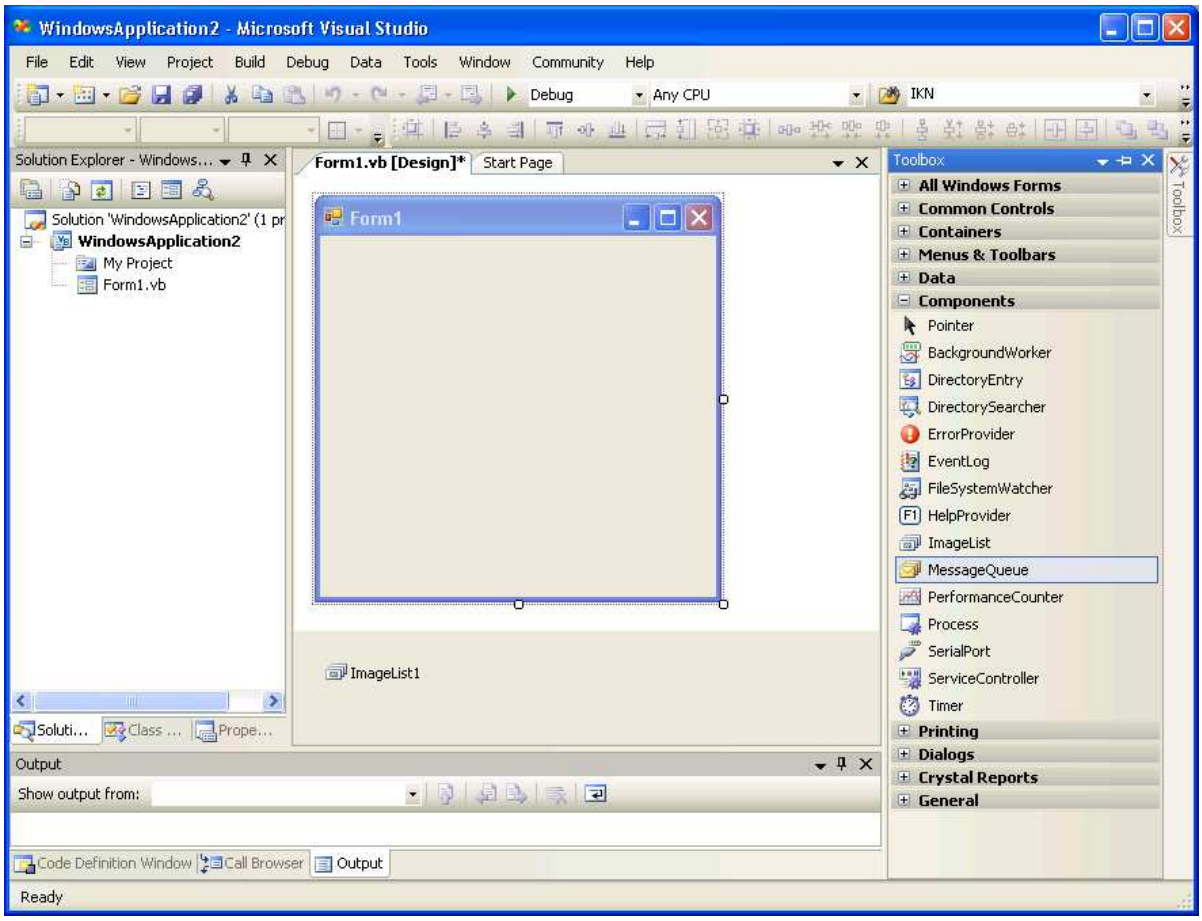


To do this with Visual Basic, see *Using Broker ActiveX Control as an Automation Server*.

Adding the Broker ActiveX Control Component to Visual Studio

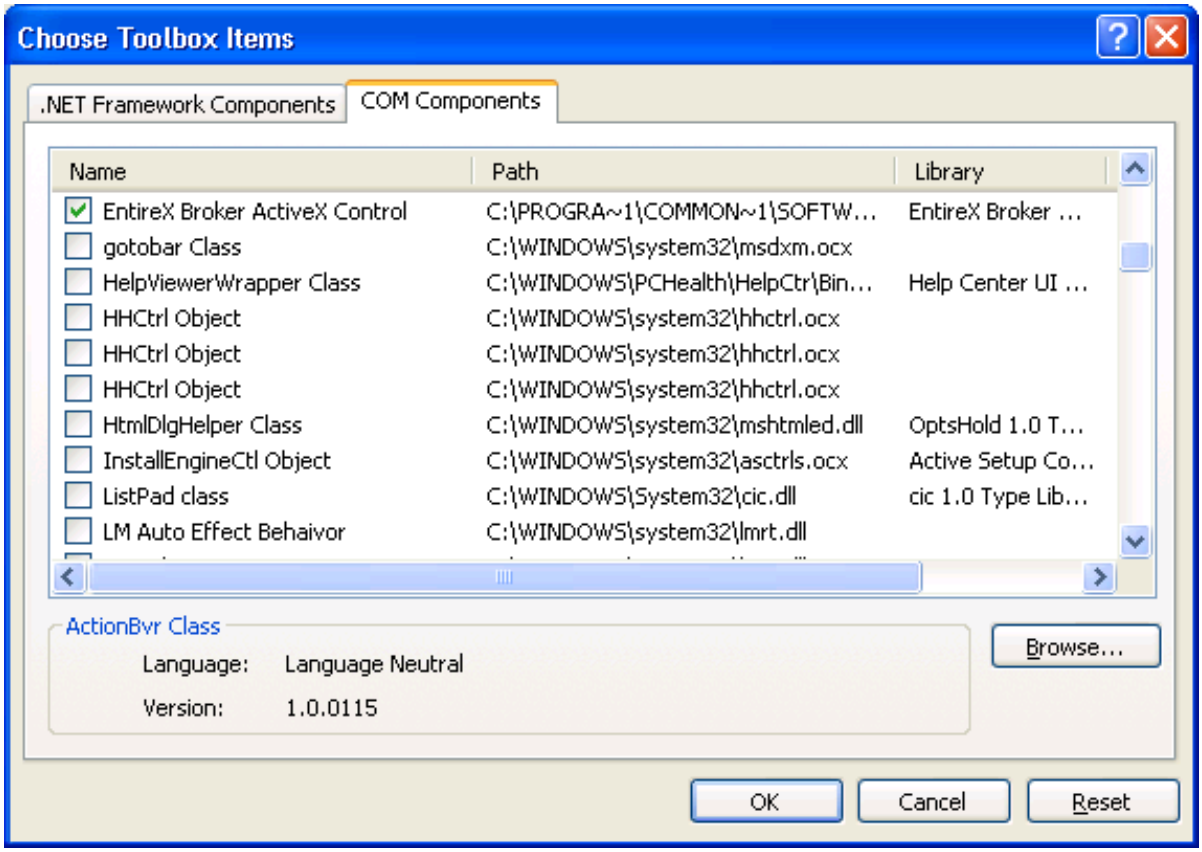
To add the Broker ActiveX Control component to Visual Studio

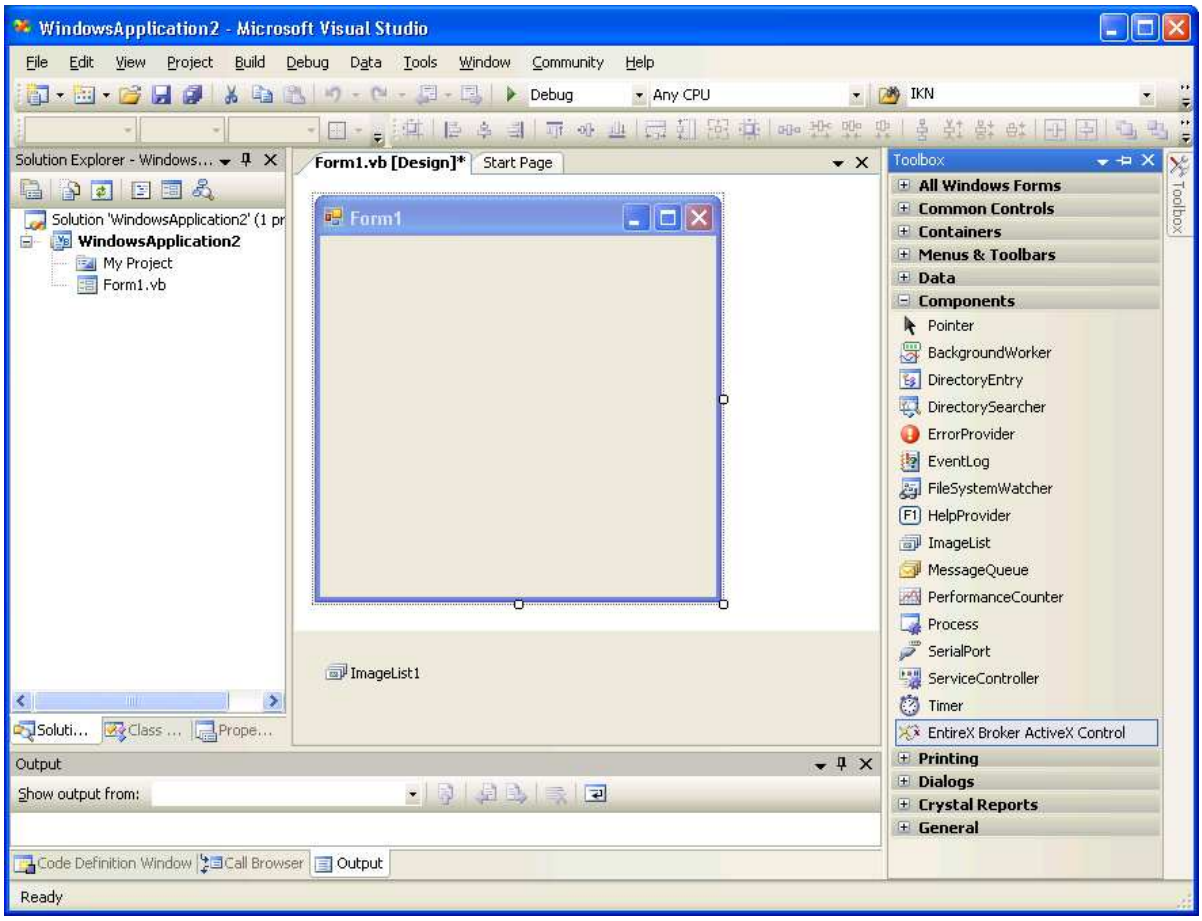
1. In Visual Studio, choose **Toolbox** > **Components**.



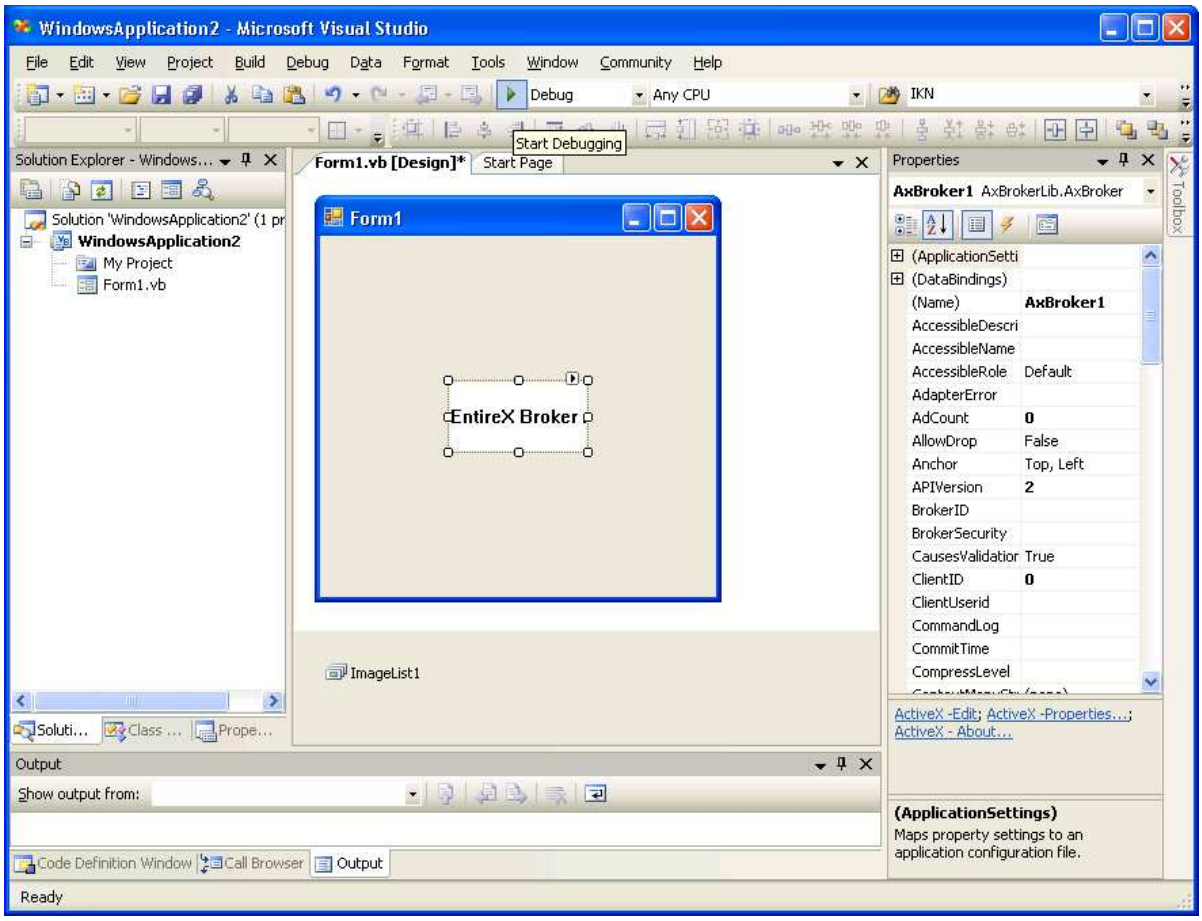
2. From the context menu, choose **Choose Item**.

3. In the **Choose Toolbox Items** dialog under **COM Components**, check "EntireX Broker ActiveX Control".


Choose Toolbox Items
.NET Framework Components
COM Components
Name
Path
Library
EntireX Broker ActiveX Control
C:\PROGRA~1\COMMON~1\SOFTW...
EntireX Broker ...
gotobar Class
C:\WINDOWS\system32\msdxm.ocx
HelpViewerWrapper Class
C:\WINDOWS\PCHealth\HelpCtr\Bin...
Help Center UI ...
HHCtrl Object
C:\WINDOWS\system32\hhctrl.ocx
HHCtrl Object
C:\WINDOWS\system32\hhctrl.ocx
HHCtrl Object
C:\WINDOWS\system32\hhctrl.ocx
HtmlDlgHelper Class
C:\WINDOWS\system32\mshtmled.dll
OptsHold 1.0 T...
InstallEngineCtl Object
C:\WINDOWS\system32\asctrls.ocx
Active Setup Co...
ListPad class
C:\WINDOWS\System32\cic.dll
cic 1.0 Type Lib...
LM Auto Effect Behaivor
C:\WINDOWS\system32\lmrt.dll
ActionBvr Class
Language: Language Neutral
Version: 1.0.0115
Browse...
OK
Cancel
Reset



EntireX Broker ActiveX Control is now known to Visual Studio. It can be copied and pasted into the new form for later use.



Using Internationalization with Broker ActiveX Control

It is assumed that you have read the document *Internationalization with EntireX* and are familiar with the various internationalization approaches described there.

By default, Broker ActiveX Control uses the Windows ANSI codepage to convert the Unicode (UTF-16) representation within BSTRINGS to the multibyte or single-byte encoding sent to or received from the broker. This codepage is also transferred as part of the locale string to tell the broker the encoding of the data.

If you want to adapt the Windows codepage, see the Regional Settings in the Windows Control Panel and your Windows documentation.

With the property `LocaleString` (see LocaleString in *Reference - Broker ActiveX Control*) you can prevent a locale string from being sent if communicating with broker version 7.1.x and below (blank out the property for this purpose).

Restrictions

- Only the codepage configured for Windows in the Regional Settings can be used. It is not possible to use any codepage other than the codepage configured for Windows in the Regional Settings. Only LOCAL or blank is allowed as a value for the property. See *Using the Abstract Codepage Name LOCAL* for more information.

- No TOR file property is available. When you are using the TOR interface, you can set this property as usual in your own application.
- The Windows codepage used by Broker ActiveX Control must also be a codepage supported by the broker, depending on the internationalization approach. See *Locale String Mapping* for information on how the broker derives the codepage from the locale string.

Using the Property Pages

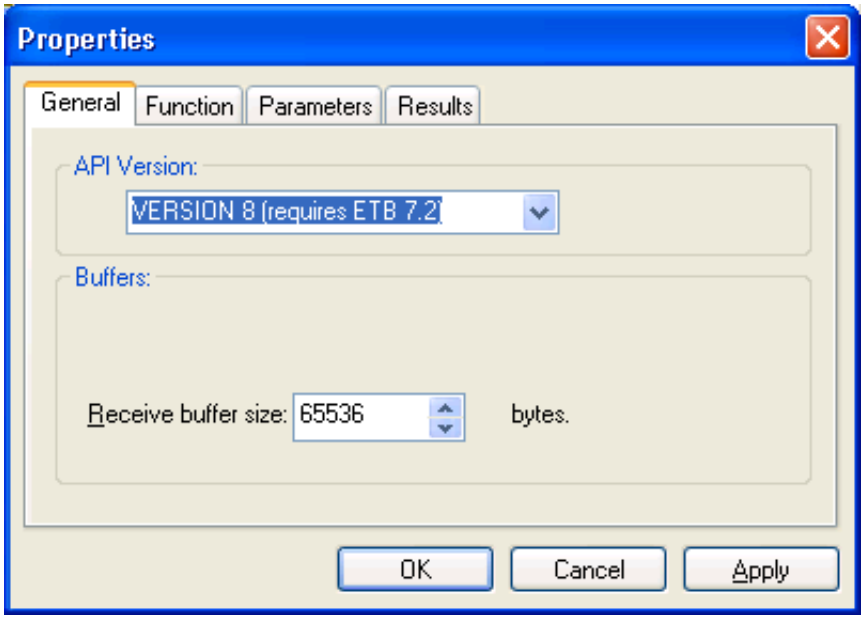
If you do not use Transaction Object Repository (TOR) files, you can also supply the properties using the property sheet of Broker ActiveX Control. (If you use Broker ActiveX Control as an automation server, the property pages are not available.)

The property sheet contains the following:

- General Page
- Function Page
- Parameters Page
- Results Page

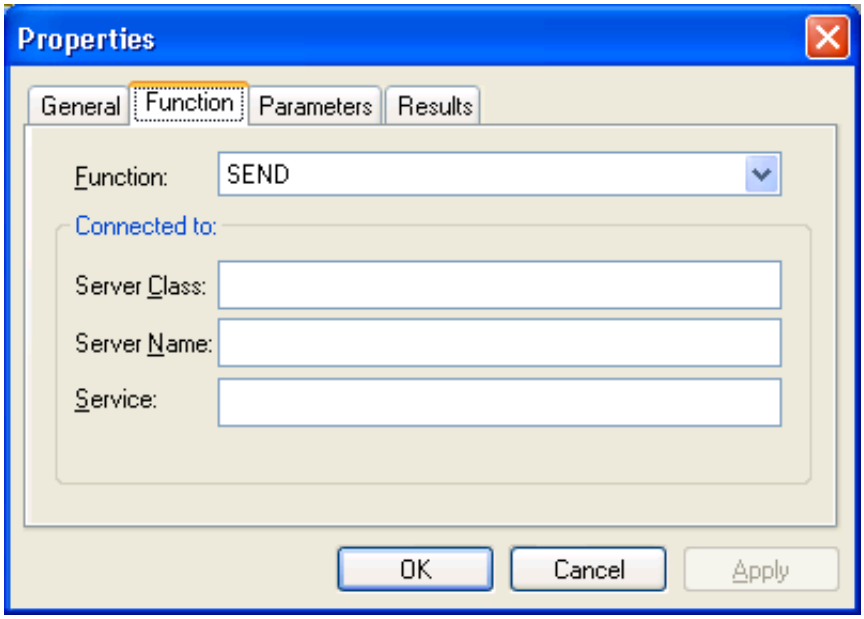
General Page

With this page you can specify the API version and the size of the receive buffer.



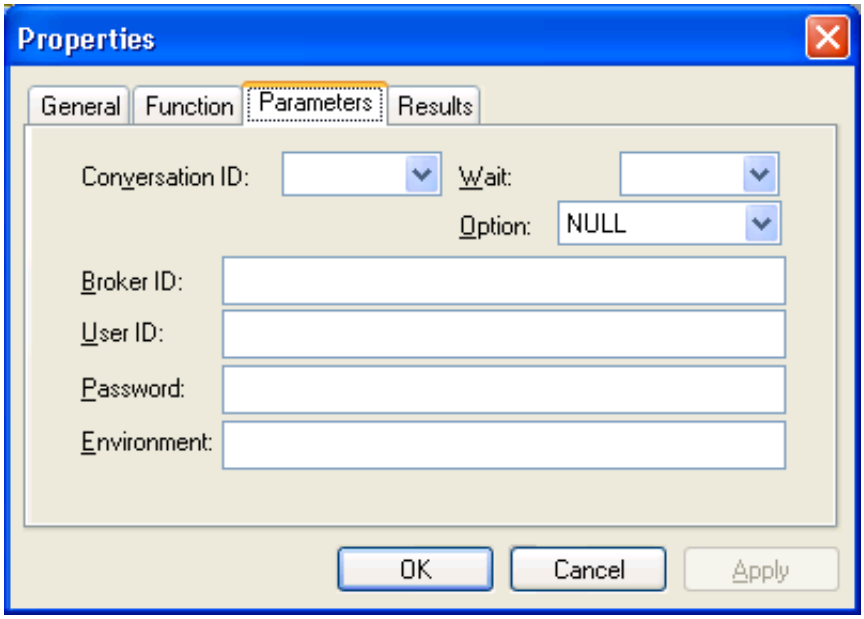
Function Page

With this page you can specify the function to be called and Service, Server Class and Server Name.



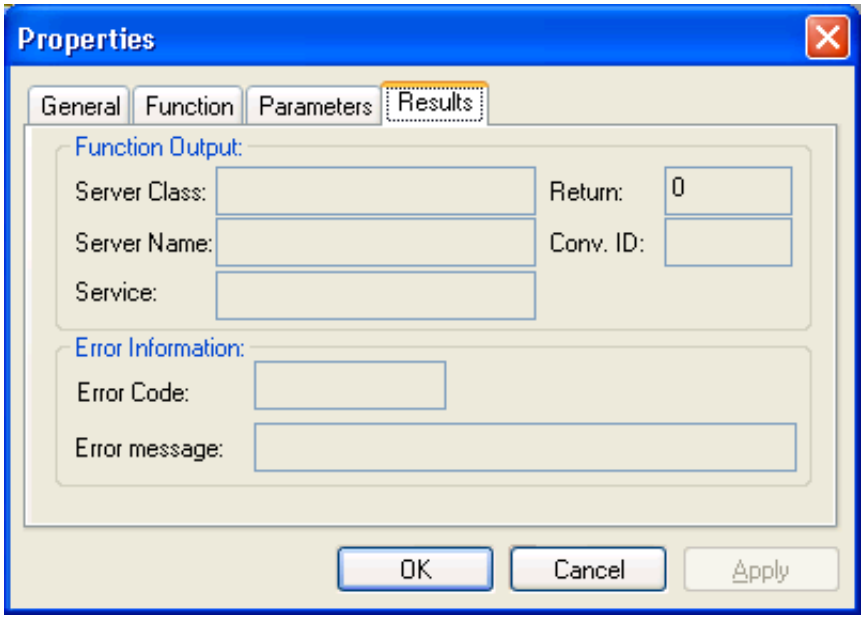
Parameters Page

With this page you can specify the Conversation ID, Broker ID, User ID, Password, Environment, Wait time, and Option.



Results Page

This page displays the results of the Broker function.


Properties
General
Function
Parameters
Results
Function Output:
Server Class:
Return:
0
Server Name:
Conv. ID:
Service:
Error Information:
Error Code:
Error message:
OK
Cancel
Apply