

## **Working with the Client Access Data Queues Control**

**by Craig Pelkie**

Although many AS/400 programmers would like to try their hand at client/server programming, it's easy to be put off by the complexity of working with the Client Access APIs. With the Client Access for Windows 95 product, IBM introduced three OLE Custom Controls (OCXs) that are remarkably easy to use. The three controls are a System listbox control, to show the names of AS/400 systems your PC is configured to work with, a Remote Command button that lets you send a command string to the AS/400, and the Data Queues textbox control, that is used to move a text string to and from the AS/400. You probably won't use these controls to develop complex client/server programs, but for some simple programs that you'd like to create, these tools may be very useful.

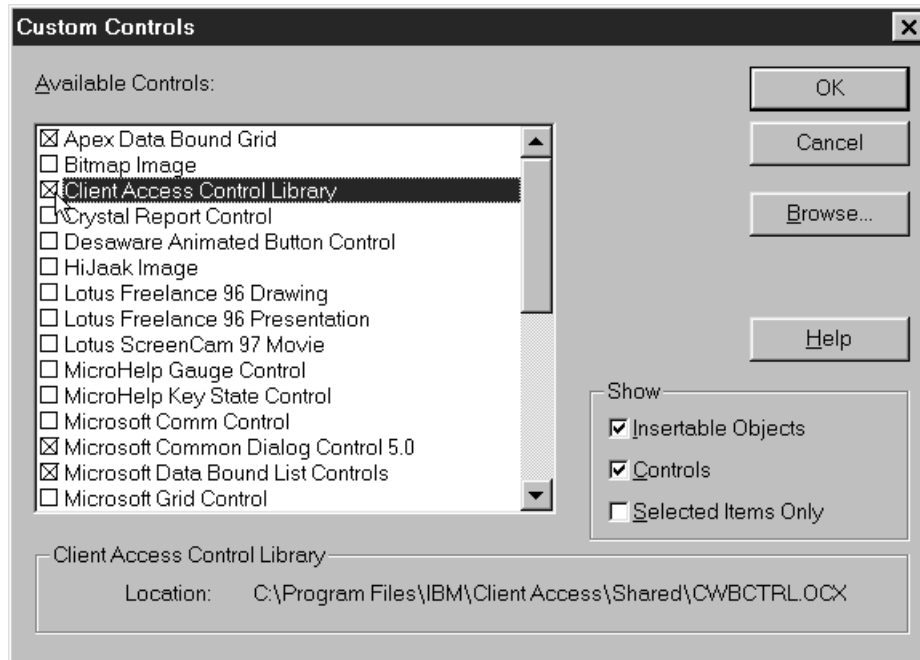
I'll describe how to use the Data Queues textbox control, since that control gives you the most power for your first applications. Using the control, you can send data from a PC program to a data queue on the AS/400. On the AS/400, you can have another program receive the entry from the data queue, perform some processing, then write a response entry to a data queue. The PC program can then retrieve entries from the response data queue, completing the client/server circle.

The examples in this article show Microsoft Visual Basic, version 4.0. You can use the Client Access OCX controls with any Windows product that supports OCXs, including Delphi and PowerBuilder.

### **Including the controls in a Visual Basic project**

The custom controls are installed on your PC when you install Client Access for Windows 95. The controls are included in the Toolkit component of the Client Access install. When you install Client Access with the Toolkit option selected, the controls are loaded into the Client Access directory and registered with Windows.

In Visual Basic, you need to include the controls in your project before you can use them in your program. Figure 1 shows the Custom Controls Selection dialog, accessed from the Tools, Custom Controls menu item. The dialog displays a checklist of available controls. Scroll through the list until you come to the Client Access Control Library entry. Check that entry to include the three tools in your project. Figure 2 shows the three additional controls included in the toolbox. The tool-tip displayed when you pass the mouse over each tool identifies the controls as `cwbSystemListBox`, `cwbDataQueueTextBox`, and `cwbRemoteCommandButton` (IBM uses the "cwb" prefix to identify their controls and API set).



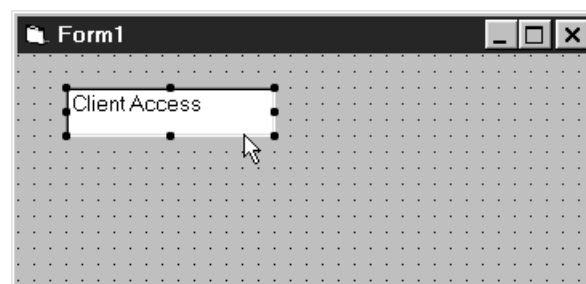
**Figure 1: The Visual Basic 4.0 Custom Controls Selection dialog**



**Figure 2: The Visual Basic Toolbox, with newly added Client Access custom controls**

### Working with the Data Queue control

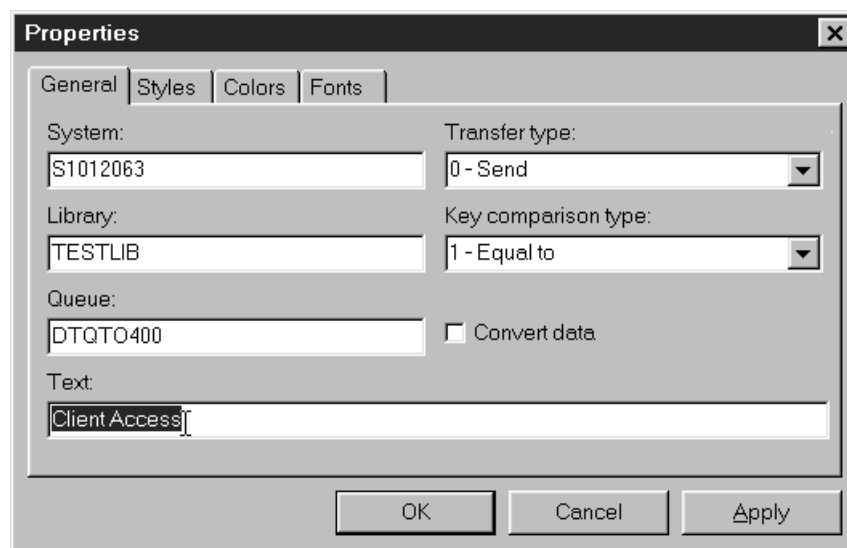
Once you have the controls included in the project, you can select them from the toolbox like any other Visual Basic tool. To work with the Data Queue control, you drag it onto the form that you want it associated with. Figure 3 shows the initial appearance of a Data Queue control when added to a form.



**Figure 3: Initial appearance of data queue control when added to a form**

The Data Queue control can be used like a textbox. That is, you can enter text into the input area of the control. The text will be sent to the AS/400. Although useful, this may be limiting for programs that you create, since you may have more than one entry field on a form that you want to send to the AS/400. An alternative to using the Data Queue control as an input field is to make the control invisible using the control's Visible property, then format a text string in your program. The text string you format can be a concatenation of many other fields. You can then move that one longer field into the text field of the Data Queue control and send that.

You can access most of the properties of the Data Queue control by right-clicking on the control, once it is on a form, then clicking the Properties item on the pop-up menu. The Properties tabbed dialog is displayed. Figure 4 shows the General tab of the Properties tabbed dialog, where you can assign the system, library, data queue name, text, and transfer type. Although you can set these properties in your code at run-time, you can also set them here if you know what the values will be as you develop the program.



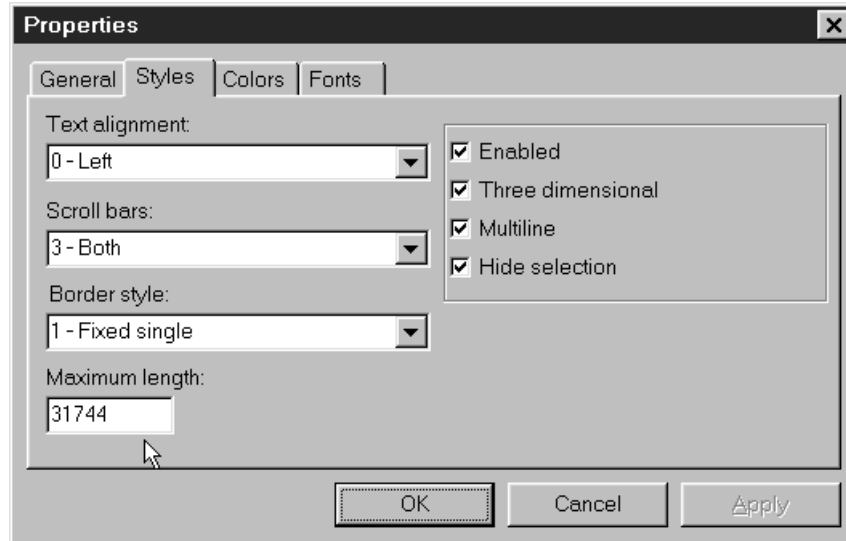
**Figure 4: The data queue control Properties tabbed dialog, General tab**

A very important property of the control is the Transfer Type. This can be set to Send or Receive. Although you can change the value of this property as your program runs, you may find it easier to add two Data Queue controls to your form, one for send and the other for receive.

If you want to work with keyed data queues, you indicate the key comparison type on the General tab. In most cases, you'll work with equal key compares.

The other tabs in the tabbed dialog, Styles, Colors, and Fonts, are concerned with the appearance of the Data Queue control when you use it as a visible textbox control. If you use the technique of making the control invisible, you won't be concerned with most of the options on those tabs.

In my testing, I found that the Maximum Length property on the Styles tab (Figure 5) must remain set to the default value of 31744. I initially set the Maximum Length to 200, which was sufficient for the amount of data I was sending and receiving. However, when using any setting other than the default I was unable to receive data from the Data Queue control. Leaving the setting at the default does not mean that you actually transmit 31744 bytes every time you send or receive. The actual number of bytes transmitted is closer to the actual entry length.

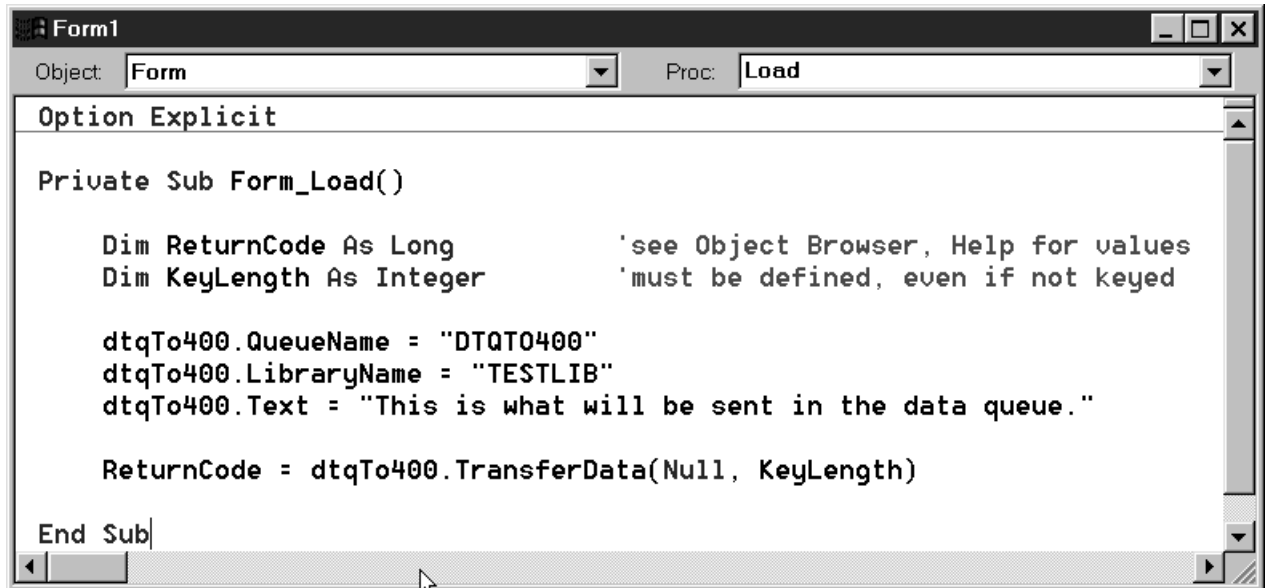


**Figure 5: The data queue control Properties tabbed dialog, Styles tab**

After you set properties in the Properties tabbed dialog, you may want to go to the Visual Basic Properties window also. You need to go to that window to set the Name property of the Data Queue control. The default name of the control in your project is `cwbDataQueueTextBox1`. As with other Visual Basic controls, the Name property is the name by which the control is known in your Visual Basic program. It is not the same as the name of the data queue that the control is sending to or receiving from; that value is set in the QueueName property.

#### **Getting data to and from the AS/400**

The Data Queue control has one method used to transfer data to or from the AS/400. The `TransferData` method performs the transfer indicated by the `TransferType` property (Send to the AS/400, Receive from the AS/400). Figure 6 shows some sample Visual Basic code used to set properties for the Data Queue control, then using the `TransferData` method to send the data to the AS/400. When you use `TransferData` to receive data from a data queue into your Visual Basic program, the value received is in the Text property of the control.



```
Form1
Object: Form Proc: Load
Option Explicit

Private Sub Form_Load()

    Dim ReturnCode As Long           'see Object Browser, Help for values
    Dim KeyLength As Integer         'must be defined, even if not keyed

    dtqTo400.QueueName = "DTQT0400"
    dtqTo400.LibraryName = "TESTLIB"
    dtqTo400.Text = "This is what will be sent in the data queue."

    ReturnCode = dtqTo400.TransferData(Null, KeyLength)

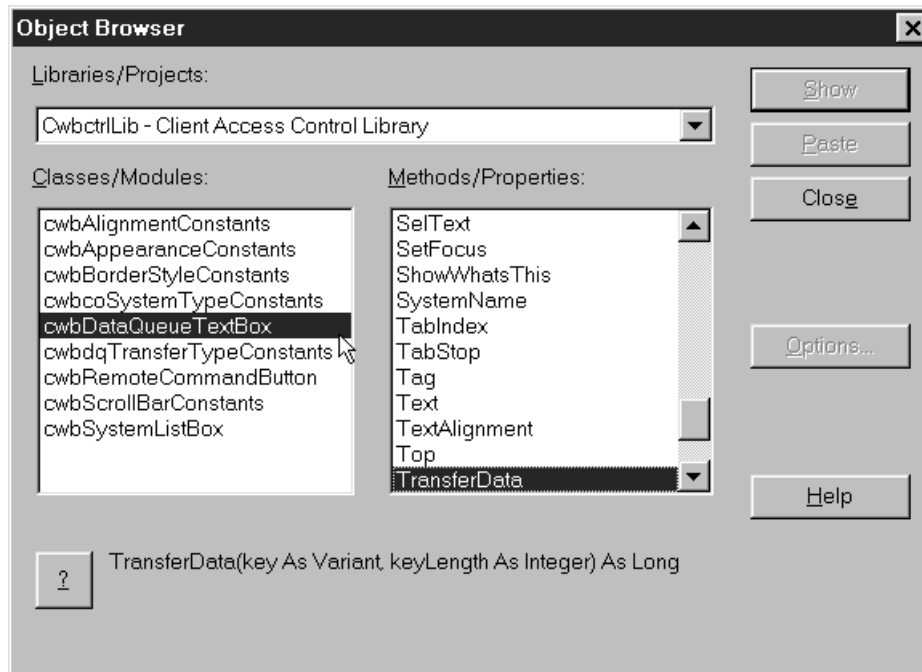
End Sub
```

Figure 6: Sample Visual Basic code for TransferData operation

Note carefully the use of the KeyLength variable on the TransferData line. In this example, I was using a non-keyed data queue, so the key length is zero. The **Client Access for Windows 95 API and Technical Reference** shows a similar example where the value 0 is used instead of the KeyLength variable. When I tried to use the numeric 0, the TransferData method ended in error. The documentation and the developer indicated to me that the parameter expects a pointer to an integer value, which is satisfied by using the KeyLength variable, but not by simply coding the actual value 0 for the second parameter.

#### Information about the Data Queue Control

In addition to the **Client Access for Windows 95 API and Technical Reference**, you'll find that most of the documentation about the control is on your PC. You can get at the documentation through the Visual Basic Object Browser. Use the View, Object Browser menu item to display the Object Browser dialog, shown in Figure 7. In the Object Browser, select the entry for CwbCtrlLib - Client Access Control Library from the list of libraries. The classes and methods/properties available in the library are displayed in the Object Browser, as shown in the figure.



**Figure 7: Visual Basic Object Browser, showing methods and properties for cwbDataQueueTextBox**

You'll want to become familiar with the location of the return codes for the TransferData method. There are many different types of errors you can make when using the Data Queue control, such as trying to refer to a non-existent data queue, AS/400 library or AS/400 system. To get to the list of return codes, click the cwbDataQueueTextBox entry in the Classes/Modules list, as shown in the figure. Then click the question mark icon in the bottom left of the Object Browser dialog. That takes you to the Help file documentation for the control. There is no specific entry for "return codes", so you'll have to select one of the other entries and work your way through to the documentation. A good selection is the entry for the GetErrorMessageText method. On the description of that method, there's an entry for Client Access Data Queues Return Codes. Following that link takes you to the list of return codes you might encounter when using the Data Queues control. Part of the list is shown in Figure 8.



**Figure 8: List of Return Codes in Object Browser Help**

### **Beware of Timeout**

There is an undocumented feature of the Data Queues control that you may have to accommodate in your program. The control automatically “times-out” after about five seconds when doing a TransferType of Receive. This is normally a good thing, because it keeps your program from locking up waiting to receive. If you send a request that causes an error (which might be waiting for a system operator response), you don’t want your program blocked waiting for the response. The timeout feature seems to be designed to handle this type of problem, but I encountered a reverse form of the problem in my testing. When testing on an old, small AS/400, my TransferData request timed out simply because it took the AS/400 program longer to respond than the timeout value allowed. This could also happen even on more capable AS/400s, especially when the system is heavily loaded.

The solution seems to be to test the return code for the timeout value (6017), and then go into a retry loop in your code. You can limit the retry loop in your code so that if in fact the AS/400 is not going to respond, your code will eventually deal with the actual timeout event.

There is no property or method associated with the Data Queue control that I could find that lets you set the value for the timeout interval or disable it.

**Worth a Try**

If you've wanted to try getting data back and forth to the AS/400 and a PC program, you should certainly consider the Data Queue control. You can create the PC program in a matter of minutes. You don't need to be concerned at all with API declarations for the Data Queues API. The beauty of OCX controls is that they insulate you from the details of direct API calls, and use the much simpler technique of setting properties and using methods of objects.

You can create simple RPG or COBOL server programs that use the QSNDDTAQ and QRCVDTAQ AS/400 APIs to send to and receive from the data queues your PC is working with. Once you have a working example, you'll find that you can quickly increase your knowledge and skill level for client/server programming jobs.