

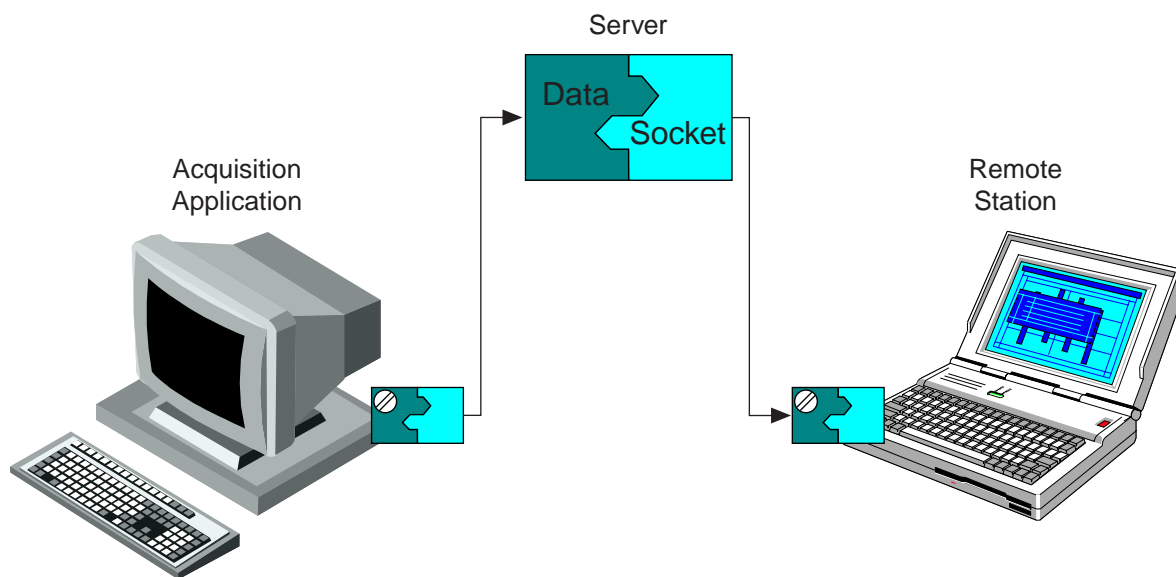
Building an Interactive Web Page with DataSocket™

Heather Edwards

Introduction

This application note explains how you can create an interactive Web page with which users can view data from a remote acquisition application without the common problems of a client-server application. You use the Component-Works DataSocket control and Microsoft Visual Basic to create a software component that you can insert in a Web page. You then use that Web page to read, write, or share data with other applications across the Internet.

Imagine a lab with 30 student workstations, a server for the lab, and a computer that acquires measurements and performs analysis on the measured data. A professor needs to acquire and distribute data to networked student workstations. The professor decides to connect the student workstations to the server so that the acquisition machine maintains only one connection to the server rather than to all 30 student workstations. The student workstations have Web pages containing DataSocket reader components that connect to the server and read the data. The following figure illustrates how DataSocket connects the acquisition machine to the student workstations through the DataSocket server.



The professor can restart the acquisition or reboot the acquisition computer without having to reconnect all 30 student workstations. The student workstations stay connected to the server and receive new measurements as soon as the acquisition restarts and sends data to the server.

This application note describes how to create the DataSocket reader component, build a Web page containing that component, and simulate the student lab scenario.

Product and company names mentioned herein are trademarks or trade names of their respective companies.

What Is DataSocket?

DataSocket is a programming tool that enables you to read, write, and share data between applications and/or different data sources and targets. DataSocket can access data in local files and data on HTTP and FTP servers. If you use general purpose file I/O functions, TCP/IP functions, and FTP/HTTP requests to transfer data, you must write separate code for each protocol. DataSocket provides a unified API for these low-level communication protocols. You can connect to different data sources without writing different code to support different data formats and protocols because the DataSocket control converts data for transfer and passes the actual values to your application.

To connect to a data source location, you need to specify a URL. Like URLs you use in a Web browser, the data source locator can point to different types of data sources depending on the prefix. The prefix is called the *URL scheme*. DataSocket recognizes several existing schemes, including `http` (hypertext transfer protocol), `ftp` (file transfer protocol), `file` (local files), and `opc` (OLE for Process Control). The DataSocket also has a new scheme, `dstp` (DataSocket transfer protocol), for sharing live data through DataSocket Servers.

The ComponentWorks DataSocket package includes three tools:

- DataSocket ActiveX Control – Component that connects applications to data sources or targets and shares data between them. Because DataSocket is an ActiveX control, you can use it to develop data-sharing applications in ActiveX containers such as Visual Basic, Visual C++, and Borland Delphi.
- DataSocket Server – Executable that communicates and exchanges data between two applications using the `dstp` protocol. In the student lab, the professor runs DataSocket Server on the lab server. When you run the DataSocket Server on your computer, you make data accessible to other DataSocket applications on the same computer or other computers connected through a TCP network, such as the Internet.
- DataSocket Server Manager – Configuration utility for the DataSocket Server through which you can specify the machines that can create items, read items, and write items.

Building an Interactive DataSocket Reader Web Page

To develop the DataSocket client Web page, you create a DataSocket reader component with which users connect to the server, automatically read and display the data, and disconnect from the server. In this example, you create that component with ActiveX controls and Visual Basic code to manipulate those controls and save this component as an ActiveX control. After making the ActiveX control, you insert it into a Web page.



Note An ActiveX control is a software component that you use to build applications. Its file extension is `.OCX`.

Overview

This example consists of four major steps:

- *Creating the DSReader Component Using ActiveX Controls in Visual Basic* – Create a component (ActiveX control) using ComponentWorks DataSocket and Visual Basic 5.0 Control Creation Edition. Later, you will insert this component on your Web page. The DSReader component reads and displays the data sent from the acquisition application.
- *Creating the HTML File* – Use the Visual Basic Application Setup Wizard to build the DSReader Web page.
- *Displaying Live Data in the DSReader Web Page* – View the DSReader Web page with Internet Explorer. Connect the DSReader Web page to the DataSocket Server and read and display wave data published by the DSWriter application.
- *Interacting with the Data on the DSReader Web Page* – Modify the DSReader component to interact with the data. Reload the component and Web page, and reconnect to the DataSocket Server to interact with wave data.

Tools

National Instruments recommends that you use the following tools to build your ActiveX control and insert it on a Web page. You can download all of these tools free from the Web. You must install ComponentWorks 2.0 or later, Visual Basic 5.0 Control Creation Edition, and Internet Explorer 4.0 to complete this example.

- ComponentWorks 2.0 DataSocket and User Interface (UI) controls and DSWriter Executable – If you have not purchased a ComponentWorks development system with DataSocket, you can download a demonstration version of the controls and this example application at www.natinst.com/cworks.
- Visual Basic 5.0 Control Creation Edition – You can download this development environment from the Microsoft Developer Network. Visit msdn.microsoft.com and search for **Visual Basic Control Creation Edition** in Additional MSDN Online Areas only. If you already have Visual Basic 4.0 or later, you can use it rather than the Control Creation Edition.
- Internet Explorer 4.0 or later and FrontPage Express – Download Internet Explorer from the Microsoft Web site (www.microsoft.com). As you install Internet Explorer, select **Full Installation** from the installation options to install FrontPage Express.

Creating the DSReader Component Using ActiveX Controls in Visual Basic

We want to design a component that connects to the DataSocket Server, reads live data from the server, and displays that data on a graph.

Opening a New Project and Loading ActiveX Controls

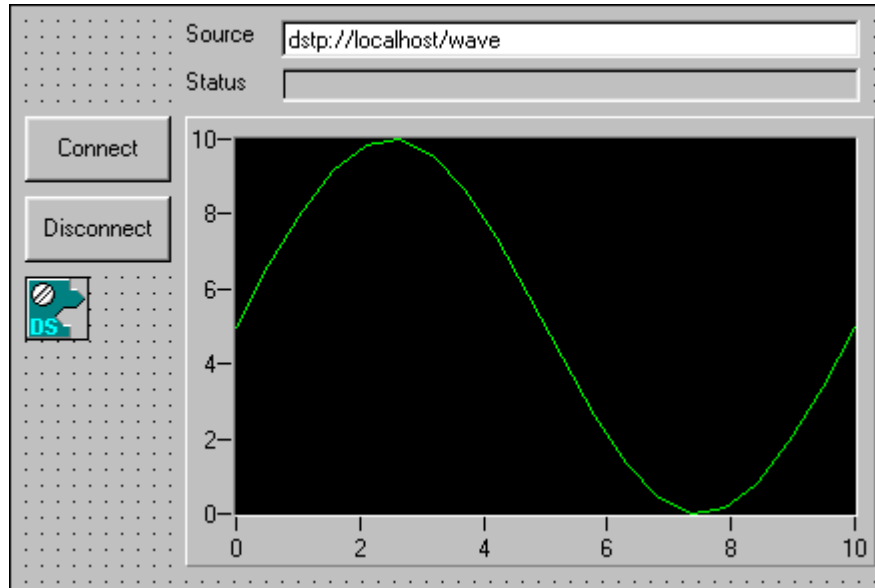
1. Launch Visual Basic 5.0 Control Creation Edition from the Windows **Start** menu.
2. In the New Project dialog, open a new ActiveX Control.
3. Right click on the toolbox and select **Components**. If the toolbox is not visible, select **Views»Toolbox**.
4. Select **National Instruments CW DataSocket** and **National Instruments CW UI**. If the ComponentWorks controls are not in the Controls list, press the **Browse** button and select `cds.ocx` and `cui.ocx` from the Windows System directory.
5. Click **OK**.









Tip You can create the DSReader component project within an executable project so that you can test the DSReader component as you develop it. After you launch Visual Basic, open a new Standard EXE and then use the **File»Add Project** option to add an ActiveX Control project. You'll see a new control in the Visual Basic Toolbox named `UserControl1`, which you'll soon be able to place on a form in the executable project and run.

Designing the Component

The following figure shows the DSReader component form. A form is the gray window or area in which you place controls and indicators to create the user interface for your application. Use the following steps to develop the DSReader interface.



Note To place controls on the control form, select the icon from the toolbox and click and drag on the form to draw the control. After you place the control, you can use your mouse to move and resize the object. To customize its appearance or behavior, modify the properties listed in the Visual Basic Properties window.

-  1. Place a CommandButton on the form and change the **Name** caption to `ConnectButton` and the **Caption** property to `Connect`.
-  2. Place a CommandButton on the form and change the **Name** caption to `DisconnectButton` and the **Caption** property to `Disconnect`.
-  3. Place a TextBox on the form for the source URL. Keep its default properties. Add a label to identify the data source.
-  4. Place a Label on the form to display the connection status. Change its **Name** caption to `StatusLbl` and **Border Style** property to **Fixed Single**. Delete the default text for the **Caption** property. Add a label to identify the Status display.
-  5. Place a CWGraph control on the form. Keep its default properties.
-  6. Place a DataSocket control on the form. The DataSocket control will not be visible at runtime.

Developing the Code

After placing controls on the form, you need to write Visual Basic code to respond to events. An event might be an action, such as a mouse movement, or a change in state, such as a completed acquisition. To develop the event procedure for an ActiveX control in Visual Basic, double click the control to open the code editor, which automatically generates a default event procedure for the control. The event procedure skeleton includes the control name, the default event, and any parameters that are passed to the event procedure.

1. In the DSReader component, you want the DataSocket control to connect to the DataSocket Server and automatically read and update the data when the user clicks the **Connect** button. Double click the **Connect** button, and add the following code:

```
Private Sub ConnectButton_Click()  
    CWDataSocket1.ConnectTo Text1.Text, cwdsReadAutoUpdate  
End Sub
```

2. When the DataSocket control is updated with new data, the graph plots it. Double click the DataSocket control, and add the following code:

```
Private Sub CWDataSocket1_OnDataUpdated(ByVal Data As CWDSLlib.CWData)  
    ' For extra error checking. Only plot the value if it's an array.  
    If IsArray(Data.Value) Then  
        CWGraph1.PlotY Data.Value  
    End If  
End Sub
```

3. To display the connection status (whether DataSocket is connected or unconnected) in the Status field, double click the DataSocket control, select OnStatusUpdated from the event list in the code window, and add the following code to the OnStatusUpdated event:

```
Private Sub CWDataSocket1_OnStatusUpdated(ByVal Status As Long, ByVal Error As  
    Long, ByVal Message As String)  
    StatusLbl.Caption = Message  
End Sub
```

4. Finally, while DataSocket is retrieving data from the data source, the user can press the **Disconnect** button to disconnect from the server and release system resources used by the DataSocket Server. Once disconnected, the DataSocket control retains the last data loaded so the user can continue to access it after the connection terminates. Double click the **Disconnect** button, and add the following code:

```
Private Sub DisconnectButton_Click()  
    CWDataSocket1.Disconnect  
End Sub
```

Saving the Application and Making the ActiveX Control

Naming the Project and Control

The file name is not the same as the control name, so you need to assign a name to the project and control through the Properties Window.

1. Open the Project Explorer, which you can access with the **View»Project Explorer** command.
2. Select Project1 in the Project Explorer and change its **Name** property in the Properties Window to **Reader**.
3. Click the form to select it and change its **Name** property in the Properties Window to **DSReader**.

Saving the Project

1. Select the **File»Save Project As** command.
2. Save the control as `DSReader.ct1`.
3. Save the project as `DSReader.vbp`.

Making the ActiveX Control

1. Select **File»Make DSReader.ocx**. Visual Basic makes the ActiveX control and saves it in the location that the project resides.
2. Select **Project»Reader Properties»Component**, and change the Version Compatibility to **Binary Compatibility**.



Note In binary compatibility mode, Visual Basic does not change the globally unique identifier (GUID) every time you make the .ocx file. The HTML page uses the GUID to refer to the control.

3. Save the project (**File»Save**).

Creating the HTML File

The Application Setup Wizard (Visual Basic 5) and the Package and Deployment Wizard (Visual Basic 6) will generate an Internet distribution package, which contains a cabinet (.cab) file and the HTML file with the DataSocket Reader component. The cabinet file contains information about the DSReader component, and it is automatically downloaded, expanded, and installed by Internet Explorer so that users on other computers can view the DSReader component with their Web browser.



Note If you change the DSReader component in Visual Basic, you have to remake the OCX and Internet distribution package (.cab and .htm files).

1. Launch either the Application Setup Wizard or the Package and Deployment Wizard to create an Internet-distributable package.
2. Complete the Wizard. As the Wizard prompts you for information, keep the following things in mind:
 - You are creating an Internet package containing an HTML file and a cabinet file, which specifies the Internet download setup.
 - You don't need to include the property page DLL because you didn't create property pages for the DSReader component.
 - Package and Deployment Wizard only: Use discretion when marking your ActiveX components safe for scripting or initialization. If you're not sure about which options to use, accept the default values.
3. Copy the HTML and cabinet files to the same directory on an HTTP server to view the Web page from other networked computers or over the Internet. Refer to the Appendix for information about accessing the HTML file with a Web browser using the HTTP protocol.



Tip Are you curious about all the files you're creating for this example? There are three files associated with the Visual Basic project: .vbp (project file), .ctl1 (DSReader component), and .vbw (workspace file). You need these files to modify your DSReader component in Visual Basic. After you modify the Visual Basic project, you need to remake the .ocx file, which saves the DSReader component as an ActiveX control that can be embedded in an HTML (.htm) file. After you make the .ocx file, you can delete previous .cab and .htm files and use either the Application Setup Wizard or the Package and Deployment Wizard to create a new Internet distribution package.

Enhancing the DSReader Web Page

Although this Web page is simple, displaying only the DSReader component, you can add text, colors, or other objects. Use FrontPage Express, a free WYSIWYG (What You See Is What You Get) HTML editor included with Internet Explorer 4.0, to create and format Web pages. You don't need to know HTML to use this editor. You can select text or paragraphs and apply formatting and alignment with toolbar buttons. You can insert custom images or clipart. You also can insert additional ActiveX controls. Use the FrontPage Express online help for information about adding these features.

Displaying Live Data in the DSReader Web Page

With the following procedure, you can simulate the student lab.

1. Launch the DataSocket Server from the Windows **Start** menu (**Programs»National Instruments ComponentWorks»DataSocket Server**).
2. Open `DSReader.htm` in Internet Explorer.
3. Press the **Connect** button on the DSReader Web page. Notice that DataSocket Server shows that one process is connected and the status on the DSReader Web page confirms that DataSocket is connected. However, no data is being plotted on the DSReader graph because there is no data being published.



Note To dynamically create the wave item on the DataSocket Server from a remote machine, open the DataSocket Server Manager and modify the **Creators** permission group to include the remote machine.

4. Launch the DSWriter executable (`\ComponentWorks\samples\Visual Basic\DataSocket\DSWriter.exe`).
5. Move the slide pointer to change the wave displayed on the graph. Because DSWriter has not connected to the DataSocket Server, DSReader is not affected.
6. Enter the URL `dstp://localhost/wave` in DSWriter and press **Connect (AutoUpdate)**. DSReader receives the wave data from the DSWriter application.
7. Move the slide in DSWriter. Notice that the DSReader wave data automatically updates to reflect changes in DSWriter.



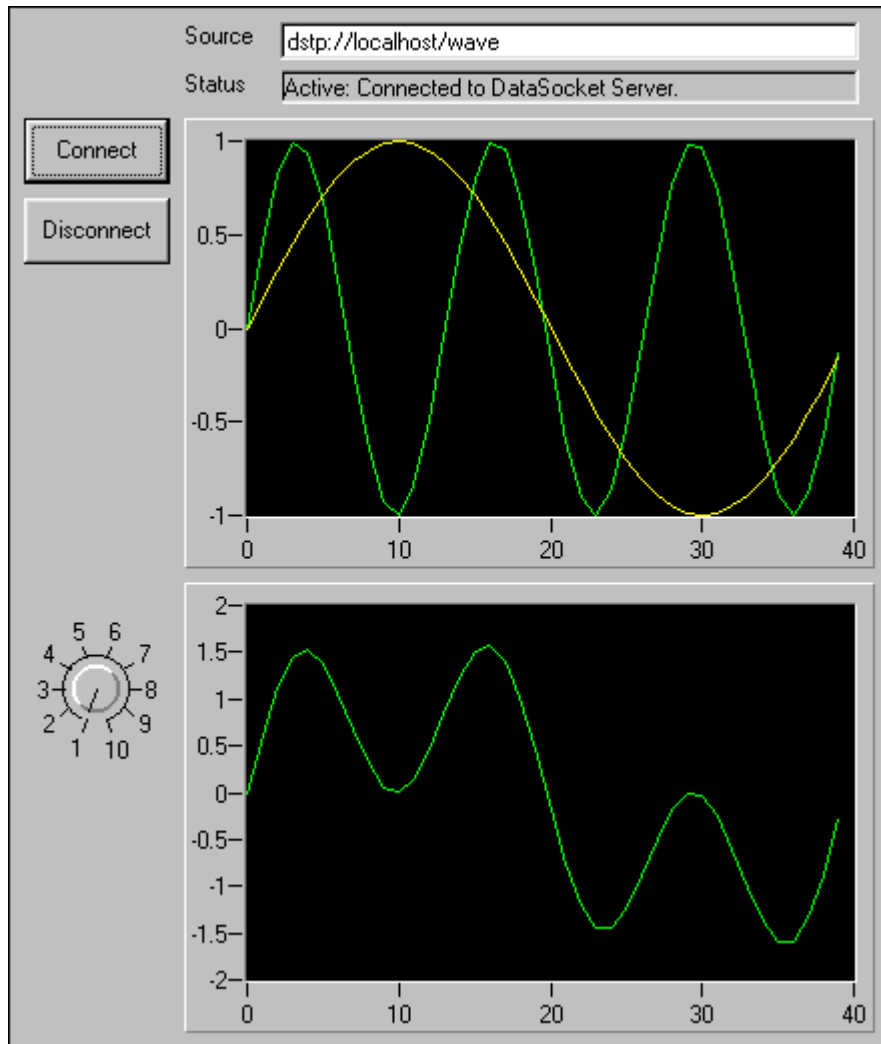
Note To use the DSReader Web page on other computers, enter the actual host name of the server, not `localhost`, which connects only to a DataSocket server on the same machine.

Interacting with the Data on the DSReader Web Page

Now that you have retrieved the wave data with the DSReader Web page, you might want to interact with that data. For example, you might need to perform a spectrum analysis. The rest of this example explains how to modify the DSReader component to make it interactive. In this case, you are going to generate a second sine wave and plot it on the first graph with the wave returned from DSWriter. You can manipulate the wave with a knob on the user interface, and as you manipulate that wave or the wave generated by the DSWriter, you are going to add the two waves together and display the result on a second graph.

Modifying the Control

Open the DSReader project in Visual Basic to modify the component. The following figure shows the final DSReader component connected to the DataSocket Server with the DSWriter wave, the wave generated, and the resultant wave.



1. Add a second plot to CWGraph1. Right click on the graph and select **Properties**. On the Plots page, click the **Add** button to add a second plot and change the line color of Plot-2 to yellow (use the color palette to the right of Line style).



2. Place a CWGraph control on the form. Keep its default properties.



3. Place a CWKnob control on the form. To modify the scale on the knob, right click on the knob, select **Properties**, and change **Minimum** to 1 on the Numeric page.

Developing the Code

1. Add the following function to your code window. This function generates a sine wave using the value of the knob, adds the generated wave to the wave returned from DSWriter, and plots the result on the second graph and the generated wave on the first graph.

```
Sub AddSinWaves()  
    Dim i  
    Dim ResultWave  
    Dim MySinWave(0 To 39)  
  
    ResultWave = CWDataSocket1.Data.Value  
    For i = 0 To 39  
        MySinWave(i) = Sin(2 * 3.14 * i * (CWKnob1.Value / 40))  
        ResultWave(i) = ResultWave(i) + MySinWave(i)  
    Next i  
    CWGraph1.Plots(2).PlotY MySinWave  
    CWGraph2.Plots(1).PlotY ResultWave  
End Sub
```

2. Call the AddSinWaves function when you receive new data from DSWriter. Add the bolded code to the following event procedure, which you created earlier in the example.

```
Private Sub CWDataSocket1_OnDataUpdated(ByVal Data As CWDSLlib.CWData)  
    ' For extra error checking. Only plot the value if it's an array.  
    If IsArray(Data.Value) Then  
        CWGraph1.PlotY Data.Value  
        AddSinWaves  
    End If  
End Sub
```

3. Call AddSinWaves when the value of the knob changes. Double click on the knob and add the following event procedure.

```
Private Sub CWKnob1_PointerValueChanged(ByVal Pointer As Long, Value As Variant)  
    AddSinWaves  
End Sub
```

Manipulating Live Data with the DSReader Web Page

1. Save the project.
2. Make the modified DSReader.ocx. You should replace the existing file.
3. Delete the existing HTML and .cab files, and rebuild them with the Application Setup Wizard.
4. View the new DSReader Web page in Internet Explorer.
5. Press the **Connect** button on the DSReader Web page. Notice that data is displayed on both graphs.
6. Turn the knob to change the number of cycles in MySinWave. The first graph displays the new value of MySinWave, and the second graph displays ResultWave.
7. Adjust the slide on DSWriter. Notice that both graphs on the DSReader Web page are updated.

Summary

With this tutorial, you have simulated the student lab scenario on a single computer. Refer to the Appendix for information about viewing the Web page from networked computers or over the Internet.

You could embellish the student lab scenario by developing a DSWriter component. With a DSWriter Web page, the professor or students could perform an acquisition from a remote site. Because the acquisition machine accommodates only one connection at a time, only one DSWriter component acquires and publishes data. The DataSocket Server can manage several data items, each one written to by a different DSWriter if your process needs them. Refer to the ComponentWorks Online Reference for more information about DataSocket Server and creating items (**Start»Programs»National Instruments ComponentWorks»ComponentWorks Reference**).

You can develop DataSocket Readers and Writers in LabVIEW and LabWindows/CVI. Visit www.natinst.com/datasocket for information about using DataSocket in different development environments.

Appendix – Accessing the DSReader Web Page with an HTTP Protocol

To view the DSReader Web page from networked computers or over the Internet, post the HTML and cabinet files on an HTTP server. The Application Setup Wizard generates the HTML file and a cabinet file. The cabinet (.cab) file contains information about all of the components included in the DSReader component, and it is automatically downloaded, expanded, and installed by Internet Explorer when referenced from an HTML page using the <CODEBASE> tag. The generated HTML includes the <CODEBASE> tag.

With its default security setting, Internet Explorer downloads only digitally signed ActiveX controls, which means that the origin of the control can be traced to a registered, commercial software developer. The DSReader component you built is unsigned. To load the DSReader Web page, you need to adjust the security settings in Internet Explorer.

1. View the security options by selecting **View»Internet Options** in Internet Explorer.
2. On the Security tab, select **Trusted site zones** in the Zone listbox and verify that the security level is **low**.



Note When you set security to low on trusted Internet zones, you are changing the security level only for the Internet sites you add. You are not changing the security level for any other site on the Internet.

3. Click on the **Add Sites** button.
4. Type in the name of the Web site, including the http://.
5. Uncheck the **Require server verification (https://) for all sites in this zone** option.
6. Click **Add**.
7. Click **OK**.

When you load the DSReader page, you will get a dialog box warning you that the ActiveX control is unsigned.

If you post the HTML file on a UNIX server, you need to make one small edit to the HTML file. Change the upper case CAB to lower case in the following line of HTML:

```
CODEBASE="reader.CAB#version=1,0,0,0"
```



341572B-01

Jun99