

DataSocket Examples

Measurement Studio delivers several examples for DataSocket. Click on the following links for example descriptions and locations:

[DSReader](#)

[DSReader With Attributes](#)

[DSServer Control](#)

[DSTransceiver](#)

[DSWeather](#)

[DSWriter](#)

[DSWriter With Attributes](#)

[Multiple OPC Items Monitor](#)

[NI Demo OPC Client](#)

[NI Fieldpoint OPC Client](#)

[OPC Quality and Timestamp Attributes](#)

[WriteToFile](#)

CWDataSocket Overview

Transfers live measurement data between applications over the Internet.

- Read and write data between different data sources and targets.
- Specify data sources and targets through a URL scheme, just like you access Web pages in a Web browser.
- Single, simple API to access OPC, HTTP, FTP, logos, and file servers.
- DataSocket Transfer Protocol (DSTP) to exchange data of any type using DataSocket servers.
- Browsing features to help you quickly locate data items on other computers and servers.

CWData

The CWData object holds a value and attributes associated with the value.

Properties

Value Specifies the value of a CWData object (number, string, date, array, or other variant type).

Methods

<u>CopyFrom</u>	Copies the value and attributes from another CWData object.
<u>DeleteAttribute</u>	Deletes existing attributes.
<u>GetAttribute</u>	Gets a CWData object for an attribute
<u>GetAttributeNames</u>	Returns the names of the attributes on the CWData object.
<u>HasAttribute</u>	Returns True if an attribute exists.
<u>Reset</u>	Clears the Value property and all attributes.
<u>SetAttribute</u>	Sets an attribute value.

See Also

[CWDataSocket.Data](#)

CWDataSocket

CWDataSocket is a component that can connect to various data sources for reading data or data targets for writing data. Sources and targets may be local (on the current machine) or on other machines in the office or over the Internet. CWDataSocket holds stores data in CWData objects, which you can access with the CWDataSocket.Data property.

Properties

- [AccessMode](#) Specifies the type of read or write connection the DataSocket makes when connecting to the data source or target.
- [ActualURL](#) Specifies the actual URL of the current source or target to which DataSocket is connected.
- [AutoConnect](#) Indicates if the DataSocket connects to a data item as soon as the program is run.
- [Data](#) Specifies the current value and attributes that the CWDataSocket has received from the data source or that have been set locally.
- [DataUpdated](#) Indicates if values or attributes on the CWDataSocket have been set since they were last read.
- [LastError](#) Specifies the last error code used in an OnStatusUpdated event.
- [LastMessage](#) Specifies the last message used in an OnStatusUpdated event.
- [Status](#) Specifies the current status of the data connection.
- [StatusUpdated](#) Indicates if the status has changed or an error has occurred.
- [URL](#) Specifies the location expressed as a URL of the data source or target to which the CWDataSocket is connecting.
-

Methods

[AboutBox](#) Displays the About Box for the control.

[Connect](#) Connects the CWDataSocket to a data source or target.

[ConnectTo](#) Connects the CWDataSocket to a data source or target.

[Disconnect](#) Disconnects the CWDataSocket from the data item to which it is currently connected.

[SelectURL](#) Enables interactive browsing and selection of data items, on DataSocket, OPC servers, Fieldpoint devices, Lookout objects, and of files so that users can easily create data URLs, rather than constructing the URLs themselves, or it displays a simple text box in which users can enter HTTP and FTP URLs.

[Update](#) Causes the CWDataSocket to read from a data source or write to a data target.

Events

[OnDataUpdated](#)

Generates when the CWDataSocket value is updated.

[OnStatusUpdated](#)

Generates when the status of the connection changes.

See Also

[CWData](#)

Example: DSReader

Reads values from different data sources using dstp, logos, opc, http, ftp, or file protocols.

Description

Through National Instruments DataSocket technology, your programs can communicate with a variety of different data sources. In this example, you can either manually type a data location for Source or select one of the sources from the listbox.

DataSocket supports the following types of connections:

Protocol	Examples
DataSocket Server ("dstp:")	dstp://localhost/wave dstp://machine/item
Standard Web Server ("http:")	http://www.ni.com/cworks/datasocket/tone.wav
Standard FTP Sites ("ftp:")	ftp://ftp.ni.com/datasocket/ping.wav The ftp site should allow anonymous connections.
Lookout/Logos objects ("lookout:" or "logos:")	lookout://localhost/testprocess/pot1.value logos://localhost/testprocess/pot1.value
Files directly accessible from your file system ("file:")	file:ping.wav file:c:/mydata/ping.wav file://machine/mydata/ping.wav
OLE for Process Control (OPC) Servers ("opc:")	opc:/National Instruments.OPCDemo/sine opc:/National Instruments.OPCDemo/sine? Accesspath=sine opc://machine/National Instruments.OPCModbus/Modbus Demo Box.4:0 opc://machine/National Instruments.OPCModbus/Modbus Demo Box.4:0? updaterate=100&deadband;=0.7

Controls, Properties, Methods, and Events

This example demonstrates the following controls, properties, methods, and events:

[CWDataSocket](#)

[ConnectTo](#), [Update](#), [Disconnect](#), [OnDataUpdated](#), [OnStatusUpdated](#)

[CWData](#)

[Value](#)

Example Location

Samples\DataSocket

See Also

[DSReader with Attributes Example](#)

[DSWriter Example](#)

Example: DSReader With Attributes

Shows how to use DataSocket attributes. Works with the DSWriter With Attributes example.

Controls, Properties, Methods, and Events

This example demonstrates the following controls, properties, methods, and events:

[CWDataSocket](#)

[ConnectTo](#), [Update](#), [Disconnect](#), [OnDataUpdated](#), [OnStatusUpdated](#)

[CWData](#)

[Value](#), [GetAttribute](#)

Example Location

Samples\DataSocket

See Also

[DSWriter With Attributes Example](#)

[DSReader Example](#)

Example: DServer Control

Programatically controls a DataSocket server through automation.

Description

If you have an application that uses the DataSocket server, you might want to launch the server, hide it, show it, or close it directly from that application. This example shows you the automation interface that you can use to programmatically control a local DataSocket server.

First, create an object that can reference a DataSocket server:

```
Dim cwdss As CWDataServer.CWDataServer
```

With the previous line of code, you have an object named `cwdss` that can hold a reference to a DataSocket server.

Now you need to reference an instance of the server. The following line of code either launches a new instance, if one doesn't already exist, or gets a reference to one running:

```
Set cwdss = New CWDataServer.CWDataServer
```

At this point, you can call the `Show`, `Hide`, and `Close` methods on the `cwdss` object.

Method Description

- | | |
|-------|--|
| Show | Shows the referenced DataSocket server. You must have a reference to a DataSocket server before calling the Show method. |
| Hide | Hides the referenced DataSocket server. You must have a reference to a DataSocket server before calling the Hide method. |
| Close | Closes the DataSocket server. The DataSocket server will not close until all objects referencing it are closed. |

Note: Close object references. For example, you can set your object, `cwdss`, to `Nothing`.

Example Location

Samples\DataSocket

Example: DTransceiver

Uses DataSocket to receive or send data items from an application.

Controls, Properties, Methods, and Events

This example demonstrates the following controls, properties, methods, and events:

[CWDataSocket](#)

[ConnectTo](#), [Update](#), [Disconnect](#), [OnDataUpdated](#), [OnStatusUpdated](#)

[CWData](#)

[Value](#)

Example Location

Samples\DataSocket

Example: DSWeather

Uses DataSocket to connect to live data over the Internet and display current weather information from National Instruments in Austin, TX.

Controls, Properties, Methods, and Events

This example demonstrates the following controls, properties, methods, and events:

[CWDataSocket](#)

[ConnectTo](#), [Update](#), [Disconnect](#), [OnDataUpdated](#), [OnStatusUpdated](#)

[CWData](#)

[Value](#)

Example Location

Samples\DSWeather

Example: DSWriter

Uses DataSocket to write values to different data targets.

Controls, Properties, Methods, and Events

This example demonstrates the following controls, properties, methods, and events:

[CWDataSocket](#)

[ConnectTo](#), [Update](#), [Disconnect](#), [OnStatusUpdated](#)

[CWData](#)

[Value](#)

Example Location

Samples\DataSocket

See Also

[DSReader Example](#)

Example: DSWriter With Attributes

Shows how to use DataSocket attributes. Works with the DSReader With Attributes example.

Controls, Properties, Methods, and Events

This example demonstrates the following controls, properties, methods, and events:

[CWDataSocket](#)

[ConnectTo](#), [Update](#), [Disconnect](#), [OnStatusUpdated](#)

[CWData](#)

[Value](#), [SetAttribute](#)

Example Location

Samples\DataSocket

See Also

[DSReader With Attributes Example](#)

[DSWriter Example](#)

Example: Multiple OPC Items Monitor

Uses control arrays to connect to multiple OPC items. This example also demonstrates browsing OPC servers with the SelectURL method.

Controls, Properties, Methods, and Events

This example demonstrates the following controls, properties, methods, and events:

[CWDataSocket](#)

[ConnectTo](#), [Update](#), [Disconnect](#), [OnDataUpdated](#), [OnStatusUpdated](#)

[CWData](#)

[Value](#), [GetAttribute](#)

Example Location

Samples\DataSocket\OPC

See Also

[NI Demo OPC Client Example](#)

Example: NI Demo OPC Client

Uses DataSocket to connect to data on OPC servers.

Description

This example connects to a single OPC item on the National Instruments Demo OPC Server. You can change the hostname, server, and item fields to work with other OPC servers installed on your computer or browse for available OPC items.

Controls, Properties, Methods, and Events

This example demonstrates the following controls, properties, methods, and events:

[CWDataSocket](#)

[ConnectTo](#), [Update](#), [Disconnect](#), [OnDataUpdated](#), [OnStatusUpdated](#)

[CWData](#)

[Value](#), [GetAttribute](#)

Example Location

Samples\DataSocket\OPC

See Also

[OPC Quality and Timestamp Attributes Example](#)

[Multiple OPC Items Monitor Example](#)

Example: NI Fieldpoint OPC Client

Demonstrates using OPC to connect to a Fieldpoint DI/DO device.

Controls, Properties, Methods, and Events

This example demonstrates the following controls, properties, methods, and events:

[CWDataSocket](#)

[ConnectTo](#), [Update](#), [Disconnect](#), [OnDataUpdated](#), [OnStatusUpdated](#)

[CWData](#)

[Value](#), [GetAttribute](#)

Example Location

Samples\DataSocket\OPC

See Also

[NI Demo OPC Client Example](#)

Example: OPC Quality and Timestamp Attributes

Connects to an OPC item and uses the Quality and Timestamp attributes.

Description

This example uses a Timer event to plot data, rather than using the OnDataUpdated event so that you can chart data constantly (with timestamps) even if the data has not changed. This is useful for correctly making a square wave chart.

Controls, Properties, Methods, and Events

This example demonstrates the following controls, properties, methods, and events:

[CWDataSocket](#)

[ConnectTo](#), [Update](#), [Disconnect](#), [OnStatusUpdated](#)

[CWData](#)

[Value](#), [GetAttribute](#)

Example Location

Samples\DataSocket\OPC

See Also

[NI Demo OPC Client Example](#)

Example: WriteToFile

Uses DataSocket to read and write to a file.

Example Location

Samples\DataSocket

Value Property

Syntax

[CWData](#).Value

Data Type

Variant

Purpose

Specifies the value of a CWData object (number, string, date, array, or other variant type).

Remarks

If the CWData object is owned by a CWDataSocket and the CWDataSocket is connected to a data target in a write access mode, setting this property triggers the CWDataSocket.Update method. If you set this property while connected in a write access mode, the CWDataSocket.DataUpdated property is set to True.

If the CWData is owned by a CWDataSocket, reading this property sets the CWDataSocket.DataUpdated property to False.

Example

```
Dim CWData1 As New CWData
Dim v As Variant
Dim vector(100) As Single
Dim matrix(100, 100) As Long
```

'CWData can hold many types of data.

```
CWData1.Value = 1
CWData1.Value = "A String"
CWData1.Value = vector
CWData1.Value = matrix
```

'CWData can be assigned to a variant

```
v = CWData1.Value
```

'Because Value is the default property you can

'omit the "Value" keyword in Visual Basic.

```
CWData1 = 10
v = CWData1
```

See Also

[CWDataSocket.Update](#)

[CWDataSocket.DataUpdated](#)

CopyFrom Method

Syntax

[CWDData](#).CopyFrom Source

Purpose

Copies the value and attributes from another CWData object.

Remarks

If the destination CWData is owned by a CWDataSocket and the CWDataSocket is connected to the data target with the `cwdsWriteAutoUpdate` access mode, copying to the CWData object triggers the `CWDataSocket.Update` method. If the source CWData is owned by a CWDataSocket, copying from it sets the `CWDataSocket.DataUpdated` property to `False`.

Parameters

Source As [CWData](#)

CWData object from which to copy data.

Example

'Copy the current value and attributes from CWDataSocket.Data.

Dim CWData1 As New CWData

CWData1.CopyFrom CWDataSocket1.Data

'Make some changes to the copy.

CWData1.Value = 1

CWData1.SetAttribute "SampleRate", 10000

'Copy data back to DataSocket.Data.

CWDataSocket1.Data.CopyFrom CWData1

See Also

[CWDataSocket.Update](#)

[CWDataSocket.DataUpdated](#)

DeleteAttribute Method

Syntax

[CWData](#).DeleteAttribute Name

Purpose

Deletes existing attributes.

Remarks

If the CWData object is owned by a CWDataSocket and the CWDataSocket is connected to a data target in a write access mode, deleting an attribute triggers the CWDataSocket.Update method. If the CWData is owned by a CWDataSocket, deleting an attribute sets the CWDataSocket.DataUpdated property to True.

Parameters

Name As [variant](#)

Attribute to delete.

Example

'Delete an attribute

```
CWDataSocket1.Data.DeleteAttribute("SampleRate")
```

See Also

[GetAttribute](#)

[HasAttribute](#)

GetAttribute Method

Syntax

[CWDData](#).**GetAttribute** (Name, Default)

Return Type

[CWData](#)

Purpose

Gets a CWData object for an attribute

Remarks

If the attribute doesn't exist, this method returns the default value that you passed in. If the `CWData` is owned by a `CWDataSocket`, getting an attribute sets the `CWDataSocket.DataUpdated` property to `False`.

Parameters

Name As [Variant](#)

Attribute to get.

Default As [Variant](#)

Default value to return if the attribute does not exist.

Example

'Get an attribute, or 0 if the attribute does not exist.

```
Text1.Text = CWDataSocket1.Data.GetAttribute("SampleRate",0)
```

See Also

[HasAttribute](#)

[GetAttributeNames](#)

[DeleteAttribute](#)

[SetAttribute](#)

GetAttributeNames Method

Syntax

[CWDData](#).GetAttributeNames ()

Return Type

Variant

Purpose

Returns the names of the attributes on the CWData object.

Example

'Print the names of the attributes.

```
v = CWDataSocket1.Data.GetAttributeNames
```

```
For i = LBound(v) To UBound(v)
```

```
    Text1.Text = Text1.Text + " " + v(i)
```

```
Next i
```

See Also

[GetAttribute](#)

[HasAttribute](#)

[DeleteAttribute](#)

[SetAttribute](#)

HasAttribute Method

Syntax

`CWData`.**HasAttribute** (Name)

Return Type

Boolean

Purpose

Returns True if an attribute exists.

Remarks

If CWData is owned by a CWDataSocket, checking for an attribute sets the CWDataSocket.

Parameters

Name As [variant](#)

Attribute name.

Example

'Check for an attribute.

```
If CWData1.HasAttribute("SampleRate") Then
```

```
    Text1.Text = "Has attribute"
```

```
End If
```

See Also

[DeleteAttribute](#)

[GetAttribute](#)

[GetAttributeNames](#)

Reset Method

Syntax

[CWData](#).Reset

Purpose

Clears the Value property and all attributes.

Remarks

If the CWData object is owned by a CWDataSocket and the CWDataSocket is connected to a data target in a write access mode, resetting the CWData object triggers the CWDataSocket.Update method and sets the CWDataSocket.DataUpdated property to True.

Example

'Clear the current value and all attributes held by CWDataSocket1.
CWDataSocket1.Data.Reset

SetAttribute Method

Syntax

[CWDData](#).**SetAttribute** Name, Value

Purpose

Sets an attribute value.

Remarks

If the CWData object is owned by a CWDataSocket and the CWDataSocket is connected to a data target in a write access mode, setting an attribute triggers the CWDataSocket.Update method and sets the CWDataSocket.DataUpdated property to True.

Parameters

Name As [Variant](#)

Attribute to set.

Value As [Variant](#)

Value for the attribute.

Example

'Set the SampleRate attribute to 1000.

```
CWDataSocket1.Data.SetAttribute "SampleRate", 1000
```


Data Property (Read Only)

Syntax

[CWDataSocket](#).Data

Data Type

[CWData](#)

Purpose

Specifies the current value and attributes that the CWDataSocket has received from the data source or that have been set locally.

Remarks

Use this property to access the CWData object that holds the current value and attributes. Then use the CWData methods and properties to access the value or specific attributes.

When you connect CWDataSocket to a data target with the cwdsWrite or cwdsWriteAutoUpdate access mode, CWDataSocket writes the current value and attributes to the target as soon as the connection is complete. When you connect DataSocket to a data source with the cwdsRead or cwdsReadAutoUpdate access mode, CWDataSocket completes the connection and starts reading new values.

Use the CWDataSocket.DataUpdated property and the OnDataUpdated event to determine if and when the value or attributes are set or changed.

When a CWDataSocket is disconnected from the previous data item, it retains the value and attributes it last had while connected.

Example

'Get the current value from a DataSocket.

```
Dim v As Variant
```

```
v = CWDataSocket1.Data.Value
```

'Value is default property in VB, so you don't have to explicitly state it:

```
v = CWDataSocket1.Data
```

'Set the value held by a DataSocket to a number

```
CWDataSocket1.Data.Value = 10
```

'Set the value held by a DataSocket to a string

```
CWDataSocket1.Data.Value = "Sample1"
```

See Also

[CWData](#)

[ConnectTo](#)

[DataUpdated](#)

[OnDataUpdated](#)

AccessMode Property

Syntax

[CWDataSocket](#).AccessMode

Data Type

[CWDSAccessModes](#)

You can use the following constants with this data type:

- `cwdsRead`—Reads once when connected. You can trigger another read by calling the `Update` method.
- `cwdsReadAutoUpdate`—Reads when connected and automatically reads again if the data at the data source is updated.
- `cwdsReadWriteAutoUpdate`—Writes the current data once connected. The data is rewritten automatically if any attribute or value is set. Reads when connected and automatically reads again if the data at the data source is updated.
- `cwdsWrite`—Writes the current data once connected. You can trigger the write again by calling the `Update` method.
- `cwdsWriteAutoUpdate`—Writes the current data once connected. The data is rewritten automatically if any attribute or value is set.

Purpose

Specifies the type of read or write connection the DataSocket makes when connecting to the data source or target.

Remarks

Use the `ConnectTo` method to set the access mode as you connect.

If you connect with the `cwdsRead` or `cwdsReadAutoUpdate` access mode, the `DataSocket` data is read-only. Attempting to set the value or attributes generates an error. If the `DataSocket` is not connected, you can set the data value and attributes regardless of the access mode.

Example

'Set the AccessMode and URL properties and connect
'to a data source

```
CWDataSocket1.AccessMode = cwdsRead  
CWDataSocket1.URL = "http://myhost/pata_path"  
CWDataSocket1.Connect
```

'The AccessMode and URL can also be set with the
'ConnectTo method.

```
CWDataSocket1.ConnectTo "http://myhost/pata_path", cwdsRead
```

See Also

[Connect](#)

[ConnectTo](#)

ActualURL Property (Read Only)

Syntax

[CWDataSocket](#).ActualURL

Data Type

String

Purpose

Specifies the actual URL of the current source or target to which DataSocket is connected.

Remarks

Once connected to a data source or target, the ActualURL property is different from the URL property if the original URL pointed to a link that redirected the DataSocket to a new URL.

If the DataSocket is not connected, the ActualURL property is an empty string.

Example

'Display actual URL in text box on user interface.

```
Text1.Text = CWDataSocket1.ActualURL
```

See Also

[URL](#)

[Connect](#)

[ConnectTo](#)

AutoConnect Property

Syntax

[CWDataSocket](#).AutoConnect

Data Type

[Boolean](#)

Purpose

Indicates if the DataSocket connects to a data item as soon as the program is run.

Remarks

Setting this property programmatically has no effect. Set the URL, Connection mode, and Auto connect properties in the property pages at design time to automatically connect CWDataSocket to the data item when the program is run or the Web page is loaded. You do not have to call Connect or ConnectTo.

Like the Connect and ConnectTo methods, this property causes the control to generate the OnStatusUpdated event when the connection is made and the OnDataUpdated event when data is updated.

See Also

[AccessMode](#)

[URL](#)

[ConnectTo](#)

DataUpdated Property (Read Only)

Syntax

[CWDataSocket](#).DataUpdated

Data Type

[Boolean](#)

Purpose

Indicates if values or attributes on the CWDataSocket have been set since they were last read.

Remarks

This property is automatically set to True when the value or attributes of the CWDataSocket data have been set and automatically reset to False after this property is queried. The OnDataUpdated event is generated when DataUpdated changes from False to True.

The DataUpdated property is most useful when using the Read mode.

This property is best used in the following way:

When the Reader application wants to get new data from the server, it calls the Update method. The Update method is asynchronous. The method returns before the new data is actually retrieved from the server. The Reader program should not read the data until the DataUpdated property has changed state. The OnDataUpdated event also is generated, and the event procedure returns the data. Use the DataUpdated property only when you do not want to write an event procedure.

If you are in ReadAutoUpdate mode, the OnDataUpdated event serves as the notification that new data is available. The DataUpdated property changes the value from True to False when the event is generated because the event returns the data. Because the event is generated more quickly than you can poll the value of the property, it is difficult to ever see the DataUpdated property set to True.

Example

'Get the value only if it has been updated.

If CWDataSocket1.DataUpdated Then

 v = CWDataSocket1.Data.Value

 'DataUpdated is now False.

End If

See Also

[OnDataUpdated](#)

LastError Property (Read Only)

Syntax

[CWDataSocket](#).LastError

Data Type

Long

Purpose

Specifies the last error code used in an OnStatusUpdated event.

Remarks

The value of `LastError` is 0 if no error occurred the last time `OnStatusUpdated` was generated.

Some common errors include errors caused by incorrect network configurations, insufficient access privileges to connect to the data source, or an incorrectly formed URL.

The value of the error code is an `HRESULT`, the ActiveX data type used for reporting errors. It might include errors detected by the `CWDataSocket` or by the operating systems networking services. If an error is encountered, the message that goes with the last error is stored in the `LastMessage` property.

To determine the task the `CWDataSocket` object was performing when the error occurred, check the `LastMessage` and `Status` properties.

Example

'Display the current status in the form

'[status:error] Message

Dim str As String

If (CWDataSocket1.StatusUpdated) Then

 str = "[" & CWDataSocket1.Status & ":" & _

 CWDataSocket1.LastError & "]" & _

 CWDataSocket1.LastMessage

 MsgBox str

End If

See Also

[LastMessage](#)

[Status](#)

[OnStatusUpdated](#)

LastMessage Property (Read Only)

Syntax

[CWDataSocket](#).LastMessage

Data Type

String

Purpose

Specifies the last message used in an OnStatusUpdated event.

Remarks

The message describes either the last error encountered or the last step taken in connecting or updating the data.

Example

'Display the current status in the form

'[status:error] Message

Dim str As String

If (CWDataSocket1.StatusUpdated) Then

 str = "[" & CWDataSocket1.Status & ":" & _

 CWDataSocket1.LastError & "]" & _

 CWDataSocket1.LastMessage

 MsgBox str

End If

See Also

[LastError](#)

[Status](#)

[OnStatusUpdated](#)

Status Property (Read Only)

Syntax

[CWDataSocket](#).**Status**

Data Type

[CWDSStatus](#)

You can use the following constants with this data type:

- `cwdsConnecting–DataSocket` is in the process of connecting to the data source or target.
- `cwdsConnectionActive–DataSocket` is in the process of transferring the data or waiting for an update.
- `cwdsConnectionError–DataSocket` encountered an error connecting to the data source or target.
- `cwdsConnectionIdle–DataSocket` has connected to the data source and transferred the data.
- `cwdsUnconnected–DataSocket` is not connected to any data source or data target.

Purpose

Specifies the current status of the data connection.

Remarks

The value of this property is the same as the last status value passed to the OnStatusUpdated event.

If an error is encountered while connecting to the source or target, the status indicates the last step attempted. The LastError and LastMessage properties describe the error.

Example

'Display the current status in the form

'[status:error] Message

Dim str As String

If (CWDataSocket1.StatusUpdated) Then

 str = "[" & CWDataSocket1.Status & ":" & _

 CWDataSocket1.LastError & "]" & _

 CWDataSocket1.LastMessage

 MsgBox str

End If

See Also

[LastError](#)

[LastMessage](#)

[StatusUpdated](#)

[OnStatusUpdated](#)

StatusUpdated Property (Read Only)

Syntax

[CWDataSocket](#).StatusUpdated

Data Type

[Boolean](#)

Purpose

Indicates if the status has changed or an error has occurred.

Remarks

The property is set to True when the OnStatusUpdated event is generated and set to False when the StatusUpdated property is queried.

Example

'Display the current status in the form

'[status:error] Message

Dim str As String

If (CWDataSocket1.StatusUpdated) Then

 str = "[" & CWDataSocket1.Status & ":" & _

 CWDataSocket1.LastError & "]" & _

 CWDataSocket1.LastMessage

 MsgBox str

End If

URL Property

Syntax

[CWDataSocket](#).URL

Data Type

String

Purpose

Specifies the location expressed as a URL of the data source or target to which the CWDataSocket is connecting.

Remarks

Use the ConnectTo method to set the URL as you connect.

The CWDataSocket can connect to different data sources or targets according the URL specified. The AccessMode property determines if the CWDataSocket is reading a data source or writing to a data target.

If the data source or target pointed to by the URL redirects the CWDataSocket to a new URL, the ActualURL property is set to the new URL.

The following types of URLs are supported:

Protocol	Examples
DataSocket Server ("dstp:")	dstp://localhost/wave dstp://machine/item
Standard Web Server ("http:")	http://www.ni.com/cworks/datasocket/tone.wav
Standard FTP Sites ("ftp:")	ftp://ftp.ni.com/datasocket/ping.wav The ftp site should allow anonymous connections.
Lookout/Logos objects ("lookout:" or "logos:")	lookout://localhost/testprocess/pot1.value logos://localhost/testprocess/pot1.value
Files directly accessible from your file system ("file:")	file:ping.wav file:c:/mydata/ping.wav file://machine/mydata/ping.wav
OLE for Process Control (OPC) Servers ("opc:")	opc:/National Instruments.OPCDemo/sine opc:/National Instruments.OPCDemo/sine? Accesspath=sine opc://machine/National Instruments.OPCModbus/Modbus Demo Box.4:0

opc://machine/National
Instruments.OPCModbus/Modbus Demo Box.4:0?
updaterate=100&deadband;=0.7

Example

'Set the AccessMode and URL properties and connect
'to a data source.

```
CWDataSocket1.AccessMode = cwdsRead
```

```
CWDataSocket1.URL = "http://myhost/pata_path"
```

```
CWDataSocket1.Connect
```

'The AccessMode and URL also can be set with ConnectTo.

```
CWDataSocket1.ConnectTo "http://myhost/pata_path", cwdsRead
```

See Also

[AccessMode](#)

[ActualURL](#)

[Connect](#)

[ConnectTo](#)

AboutBox Method

Syntax

[CWDataSocket](#).AboutBox

Purpose

Displays the About Box for the control.

Remarks

The About Box displays the version number of the control and information about contacting National Instruments. You can access the About Box from a separate tab on the property pages.

Example

'Display the DataSocket About Box.

CWDataSocket1.AboutBox

Connect Method

Syntax

[CWDataSocket](#).**Connect**

Purpose

Connects the CWDataSocket to a data source or target.

Remarks

Connect uses the current URL and AccessMode property settings. Use ConnectTo to specify the URL and AccessMode as part of the method call.

If you connect a reading client to a DataSocket item that does not exist, the DataSocket Server creates the item with a default value of 0.

Example

'Set the AccessMode and URL properties and connect
'to a data source.

```
CWDataSocket1.AccessMode = cwdsRead
```

```
CWDataSocket1.URL = "http://myhost/pata_path"
```

```
CWDataSocket1.Connect
```

'The AccessMode and URL also can be set with ConnectTo.

```
CWDataSocket1.ConnectTo "http://myhost/pata_path", cwdsRead
```

See Also

[URL](#)

[AccessMode](#)

[ConnectTo](#)

ConnectTo Method

Syntax

[CWDataSocket](#).**ConnectTo** URL, AccessMode

Purpose

Connects the CWDataSocket to a data source or target.

Remarks

The `ConnectTo` method sets the `URL` and `AccessMode` properties to the values of the `URL` and `AccessMode` parameters passed to the method.

When connecting to a data source in a read access mode, `CWDataSocket` updates the data after the connection is completed. When connecting to a data target in a write access mode, `DataSocket` writes the data as soon as the connection is made. If you connect in an automatic update mode, `CWDataSocket` automatically retrieves or writes data when new data is available.

If you connect with the `cwdsRead` or `cwdsWrite` access mode, use the `DataUpdated` property to determine if data has been updated since the last time you checked the data.

If you connect a reading client to a `DataSocket` item that does not exist, the `DataSocket Server` creates the item with a default value of 0.

The `OnDataUpdated` event is generated when data is ready to be sent (in a write access mode) or has just been received (in a read access mode). You might use the `OnDataUpdated` event to scale, display, or process new data.

Parameters

URL As [String](#)

Location expressed as a URL of the data source or target to which the CWDataSocket is connecting.

AccessMode As [CWDSAccessModes](#)

Type of read or write connection the DataSocket makes when connecting to the data source or target (cwdsRead, cwdsReadAutoUpdate, cwdsWrite, cwdsWriteAutoUpdate).

Example

'Set the AccessMode and URL properties and connect
'to a data source.

```
CWDataSocket1.AccessMode = cwdsRead
```

```
CWDataSocket1.URL = "http://myhost/pata_path"
```

```
CWDataSocket1.Connect
```

'The AccessMode and URL also can be set with ConnectTo.

```
CWDataSocket1.ConnectTo "http://myhost/pata_path", cwdsRead
```

See Also

[URL](#)

[AccessMode](#)

[Connect](#)

Disconnect Method

Syntax

[CWDataSocket](#).Disconnect

Purpose

Disconnects the CWDataSocket from the data item to which it is currently connected.

Remarks

The CWDataSocket.Data.Value and attributes remain unchanged when CWDataSocket is disconnected.

Disconnect has no effect if CWDataSocket is not connected.

Example

CWDataSocket1.Disconnect

See Also

[Connect](#)

[ConnectTo](#)

SelectURL Method

Syntax

[CWDataSocket](#).**SelectURL** ([startURL] [, Title] [, Options] [, Filters])

Return Type

Boolean

Purpose

Enables interactive browsing and selection of data items, on DataSocket, OPC servers, Fieldpoint devices, Lookout objects, and of files so that users can easily create data URLs, rather than constructing the URLs themselves, or it displays a simple text box in which users can enter HTTP and FTP URLs.

Parameters

startURL As [Variant](#)

[Optional] String containing the URL. If startURL is "opc:", "dstp:", "logos:", "file", or if you omit this parameter, this method opens a browser to enable users to interactively find and select data items. You can prompt users to create a Web or FTP location with the "http:" or "ftp:" URL schemes. You also can use an entire URL, such as "dstp://localhost/wave".

Title As [Variant](#)

[Optional] Title of the SelectURL dialog box.

Options As [Variant](#)

[Optional] Flags to pass to the dialog:

0x00002000 (CREATEPROMPT): Opens a dialog box to prompt the user for permission to create the file if that file does not already exist. If the user chooses to create the file, the dialog box closes and the function returns the specified name; otherwise, the dialog box remains open.

0x00001000 (FILEMUSTEXIST): Specifies that the user can type only names of existing files in the File Name entry field. If this flag is specified and the user enters an invalid name, the dialog box procedure displays a warning in a message box.

0x00000004 (HIDEREADONLY): Hides the Read Only check box.

0x00000008 (NOCHANGEDIR): Restores the current directory to its original value if the user changed the directory while searching for files.

0x00100000 (NODEREFERENCELINKS): Directs the dialog box to return the path and filename of the selected shortcut (.LNK) file. If this value is not given, the dialog box returns the path and filename of the file referenced by the shortcut.

0x00008000 (NOREADONLYRETURN): Specifies that the returned file does not have the Read Only check box checked and is not in a write-protected directory.

0x00010000 (NOTESTFILECREATE): Specifies that the file is not created before the dialog box is closed. This flag should be specified if the application saves the file on a create-nonmodify network share. When an application

specifies this flag, the library does not check for write protection, a full disk, an open drive door, or network protection. Applications using this flag must perform file operations carefully, because a file cannot be reopened once it is closed.

0x00000002 (OVERWRITEPROMPT): Causes the Save As dialog box to generate a message box if the selected file already exists. The user must confirm whether to overwrite the file.

0x00000800 (PATHMUSTEXIST): Specifies that the user can type only valid paths and filenames. If this flag is used and the user types an invalid path and filename in the File Name entry field, the dialog box function displays a warning in a message box.

0x00000001 (READONLY): Causes the Read Only check box to be checked initially when the dialog box is created.

0x00000020 (SAVEAS): Causes the file dialog to be a Save As file dialog, rather than an Open file dialog.

Filters As [Variant](#)

[Optional] Filter string to pass to the dialog. For example, All Files(*.*)|*.*

Example

'Browse for data items.

```
CWDataSocket1.SelectURL "opc:"
```

'Browse for files in the test directory.

```
CWDataSocket1.SelectURL "file:c:\test", "Choose File", True
```

See Also

[URL](#)

[ActualURL](#)

Update Method

Syntax

[CWDataSocket](#).Update

Purpose

Causes the CWDataSocket to read from a data source or write to a data target.

Remarks

Use the `AccessMode` property to determine if `Update` reads or writes data.

When using the `cwdsRead` or `cwdsWrite` access mode, call the `Update` method when you want a read or write to occur. If the `DataUpdated` property is `False`, it is set to `True` after the update is completed, and the `OnDataUpdated` event is generated. If you want to read or write data every time new data is available, use the `cwdsReadAutoUpdate` or `cwdsWriteAutoUpdate` access modes.

See Also

[AccessMode](#)

[DataUpdated](#)

[OnDataUpdated](#)

OnDataUpdated Event

Syntax

Sub *ControlName*_OnDataUpdated(Data As CWData)

Applies To

[CWDataSocket](#)

Purpose

Generates when the CWDataSocket value is updated.

Remarks

CWDataSocket sets the DataUpdated property to True immediately before this event is generated.

The OnDataUpdated event is generated when CWDataSocket receives new data. The event passes a reference to the DataSocket CWData object so that you can easily process the new data from the event procedure:

```
Sub CWDataSocket1_OnDataUpdated(ByVal Data As CWData)
```

```
'Plot new data
```

```
CWGraph1.PlotY Data.Value
```

```
End Sub
```

You can use the DataSocket.Data property to access the currently stored value. The Data property returns a reference to the same CWData object that is passed to the event. From the CWData object, you can get the value or attributes associated with the current data, as in the following example:

```
x = CWDataSocket1.Data.Value
```

The data value is returned as a variant (the specific type of variant depends on the source to which the DataSocket is connected).

Rather than waiting for this event, you might find it more convenient to check for new data. To determine if data has been updated since it was last read, query the value of the DataUpdated property. When a new value is loaded, DataUpdated is set to True. After the DataUpdated property is queried, DataUpdated reverts to False.

Use the DataUpdated property to check for updates along with another periodic operation, such as an operation initiated by a timer event:

```
Sub Timer1_OnTimer()
```

```
'DataUpdated is true only if the data
```

```
'has been reloaded
```

```
If DataSocket1.DataUpdated Then
```

```
' Reading the data property resets the
```

```
' DataUpdated property
```

MySub DataSocket1.Data

End If

'Other timer event code

End Sub

Parameters

Data As [CWData](#)

Data read from DataSocket.

Example

```
Private Sub CWDataSocket1_OnDataUpdated(Data as CWData)
    'Graph the new data
    CWGraph1.PlotY Data.Value
End Sub
```

See Also

[DataUpdated](#)

[CWData](#)

OnStatusUpdated Event

Syntax

Sub *ControlName*_OnStatusUpdated(Status As CWDSStatus, Error As Long, Message As String)

Applies To

[CWDataSocket](#)

Purpose

Generates when the status of the connection changes.

Remarks

This event is generated every time the connection status changes, such as when CWDataSocket connects to a data item specified by the URL property, transfers data, or encounters an error.

This event returns parameters for the status, a system error code, and a string describing the most recent progress or error. You can use these parameters to identify the cause of a problem or to report the status of a connection. The OnStatusUpdated event might be generated a number of times, depending on the data item to which you are trying to connect.

Parameters

Status As [CWDSStatus](#)

Status of the DataSocket connection.

Error As [Long](#)

Error of the DataSocket connection, if one exists.

Message As [String](#)

Descriptive message of the connection status.

Example

```
Private Sub CWDataSocket1_OnStatusUpdated(ByVal Status As Long, ByVal Er
    'Display the current status in the form
    ' [status:error] Message.
    Text1.Text = "[" & Status & ":" & Error & "]" & Message
End Sub
```


See Also

[LastError](#)

[LastMessage](#)

[Status](#)

Data Types for CWDataSocket

Visual Basic	C/C++	Description
Boolean	VARIANT_BOOL	Has the value True (-1) or False (0).
Color	OLECOLOR	A color. In many containers, colors are treated as 32-bit integers.
Double	double	64-bit floating point number.
Double Array	LPSAFEARRAY	An array of doubles.
Font	LPFONTDISP	An OLE Automation Interface to a font.
Integer	short	16-bit signed integer.
Long	long	32-bit signed integer.
LPUNKNOWN	LPUNKNOWN	
Object	LPDISPATCH	A generic OLE Automation Interface.
OLE_XPOS_PIXELS	long	
OLE_YPOS_PIXELS	long	
Picture	LPPICTUREDISP	An OLE Automation Interface to a picture or image.
Single	single	32-bit floating point number.
String	BSTR	A string.
Variant	LPVARIANT	A variant.
Void	void	Nothing.
Window	OLE_HANDLE	A handle to a window.

CWDSAccessModes Enumeration

CWDSAccessModes are the constants for the CWDataSocket.AccessMode property. These constants dictate how a CWDataSocket control transfers data. If you select an automatically updating access mode, CWDataSocket retrieves or writes data every time new data is available. If you connect with the cwdsRead or cwdsWrite access mode, use the Update method to retrieve or write new data.

You can use the following constants with this data type:

- cwdsRead–Reads once when connected. You can trigger another read by calling the Update method.
- cwdsReadAutoUpdate–Reads when connected and automatically reads again if the data at the data source is updated.
- cwdsReadWriteAutoUpdate–Writes the current data once connected. The data is rewritten automatically if any attribute or value is set. Reads when connected and automatically reads again if the data at the data source is updated.
- cwdsWrite–Writes the current data once connected. You can trigger the write again by calling the Update method.
- cwdsWriteAutoUpdate–Writes the current data once connected. The data is rewritten automatically if any attribute or value is set.

See Also

[CWDataSocket.AccessMode](#)

CWDSStatus Enumeration

CWDSStatus are the constants for the Status property on CWDataSocket.

You can use the following constants with this data type:

- `cwdsConnecting-DataSocket` is in the process of connecting to the data source or target.
- `cwdsConnectionActive-DataSocket` is in the process of transferring the data or waiting for an update.
- `cwdsConnectionError-DataSocket` encountered an error connecting to the data source or target.
- `cwdsConnectionIdle-DataSocket` has connected to the data source and transferred the data.
- `cwdsUnconnected-DataSocket` is not connected to any data source or data target.

See Also

[CWDataSocket.Status](#)

[CWDataSocket.LastError](#)

[CWDataSocket.LastMessage](#)

[CWDataSocket.StatusUpdated](#)

[CWDataSocket.OnStatusUpdated](#)