About

The dsSocket Visual Basic Custom Control was developed by Dolphin Systems Inc. to provide network communications using the Winsock standard. For additional information you may contact the developer directly at:

Dolphin Systems Inc.
13584 Kennedy Rd. N.,
Inglewood, Ontario
CANADA L0N 1K0
(905) 838-2896
(905) 838-0649 FAX

Internet: stephenc@idirect.com
Web site: www.dolphinsys.com
Compuserve: 70471,137

Accept

Description

Indicates a socket with an active Listen has received a connect request from a client.

Syntax

Sub *dsSocket*_**Accept** ([*Index* **As Integer**,] *SocketID* **As Integer**)

Remarks

In responding to this event the programmer must either create a new instance of the dsSocket control, or have created another instance at design-time, and assign the SocketID event parameter to the Socket property of the new control instance. The new control instance then receives any communication events for this connection. Upon successful completion of the setting of the Socket property the State property of the new instance of the control will be set to the value of SOCK_STATE_CONNECTED.

Example:

```
Sub dsSocket1_Accept( Index as Integer, SocketID as Integer )
        Load dsSocket1(1)
        dsSocket1(1).Socket = SocketID
End Sub
```

See Also

Action Socket

Action

Description

Specifies Connect, Listen or Close operation on a socket.

Syntax

[form.]dsSocket1.**Action** = *setting%*

Remarks

The Action properties specifies a change in state for the specified socket. The following table details valid Action property values.

Value		Action
SOCK_ACTION_CLOSE	1	Close an existing connection
SOCK_ACTION_CONNECT	2	Establish a connection
SOCK_ACTION_LISTEN	3	Listen for incoming connection
SOCK_ACTION_FWDLOOKUP	6	Resolve a hostname to an IP address
SOCK_ACTION_REVLOOKUP	7	Resolve an IP address to a hostname

Data Type

Integer

See Also

LocalPort RemoteDotAddr RemoteHost RemotePort ServiceName Methods

BindConnect

Description

Sets and returns the value of the BindConnect flag.

Syntax

[form.]dsSocket1.**BindConnect** [= *True* | *False*]

Remarks

The BindConnect affects the method of connecting to a remote computer when the Action property is set to SOCK_ACTION_CONNECT. If the RemotePort is set to a reserved port number, then the BindConnect property should be set to True. This enables the remote computer to see the port number that is in use on the local computer. This is the case when connecting to such UNIX services as rsh.

Data Type

Boolean

See Also

Action

BytesSent

Description

Returns the number of bytes transferred.

Syntax

[form.]dsSocket1.**BytesSent**

Remarks

Returns the number of bytes transferred in the last Send. If an error occurred in sending the data then the BytesSent property will return zero. Read only at run-time. Not available at design-time.

Data Type

Integer

See Also

Send

Close

Description

Indicates a socket connection has closed.

Syntax

Sub *dsSocket*_**Close** ([*Index* **As Integer**,] *ErrorCode* **As Integer**, *ErrorDesc* **As String**)

Remarks

The Close event indicates that an active connection has been closed. If the connection closed without errors then the ErrorCode will be 0 and the ErrorDesc will contain "Socket closed". If an error occurred during the closing of the socket the ErrorCode parameter provides the number of the error that occurred and the ErrorDesc provides a text description of the error.

See Also

Error Codes

Close

Description

Initiates the Close of a connected socket.

Syntax

[form.]dsSocket1.**Close**

Remarks

The Close method initiates the closing of the connected socket. The time elapsed before the socket is actually closed depends on the stack latency and the setting of the Linger and Timeout properties.

Parameters

None

See Also

Action Linger Timeout

Connect

Description

Indicates a Connect action has successfully completed.

Syntax

Sub *dsSocket*_**Connect** ([*Index* **As Integer**])

Remarks

The Connect event indicates that connection has been established in response to setting the Action property to SOCK_ACTION_CONNECT. The control is now capable of sending and receiving data with a server process.

See Also

Action

Connect

Description

Initiates the Connect of a specified control.

Syntax

[form.]dsSocket1.**Connect**

Remarks

The Connect method initiates a connection for a control. The actual connection is not complete until the Connect event is fired. If an error occurs in establishing the connection then the Exception event is fired.

Parameters

None

See Also

Action RemoteHost RemoteDotAddr RemotePort ServiceName

DataSize

Description

Sets and returns the maximum number of bytes to transfer.

Syntax

[form.]dsSocket1.**DataSize** [= *setting%*]

Remarks

Sets and returns the maximum number of bytes to be transferred from the network to the Receive event. The minimum value for DataSize is 100 and the maximum is 32767. The default value is 2048. Global memory of DataSize bytes is allocated for receiving incoming data buffers.

Data Type

Integer

See Also

Receive

Description

Description
Returns the TCP/IP protocol stack description information.

Syntax
[form.]dsSocket1.**Description**

Remarks
Returns the description string that is defined by the TCP/IP protocol stack in use on the local machine. Available at run-time only.

Data Type
String

See Also
MaxSockets MaxUDPSize

EOLChar

Description

Sets and returns the termination character value for LineMode.

Syntax

[form.]dsSocket1.**EOLChar** [= setting%]

Remarks

Sets and returns the character value for determining the end of a line when receiving data in LineMode. The value of EOLChar is set to the ASCII value of the desired terminating character (e.g.: LF = 10, CR = 13). Received data will be transferred to the control in blocks that end with the EOLchar. The character is included in the received data.

Data Type

Integer

See Also

LineMode Receive

Error Codes

The following table lists the trappable errors for the dsSocket custom control.

Error Variable	Number	Explanation
SOCK_ERR_CLOSED	20000	Connection closed
SOCK_ERR_MEMORY	20001	Memory allocation error
SOCK_ERR_ISCLOSED	20002	Socket is already closed
SOCK_ERR_ISLISTEN	20003	Socket is aready listening
SOCK_ERR_NOPORT	20004	No port number or service name specified
SOCK_ERR_ISCONNECT	20005	Socket is already connected
SOCK_ERR_ISACTIVE	20006	UDP Socket is already active
SOCK_ERR_ WINSOCKDLL	20010	Winsock DLL not found
SOCK_ERR_NOTCLOSED	20011	Socket is not closed
SOCK_ERR_INTR	21004	Interrupted system call
SOCK_ERR_BADF	21009	Bad file number
SOCK_ERR_ACCES	21013	Permission denied
SOCK_ERR_FAULT	21014	Bad address
SOCK_ERR_INVAL	21022	Invalid argument
SOCK_ERR_MFILE	21024	Too many open files
SOCK_ERR_WOULDBLOCK	21035	Operation would block
SOCK_ERR_INPROGRESS	21036	Operation now in progress
SOCK_ERR_ALREADY	21037	Operation already in progress
SOCK_ERR_NOTSOCK	21038	Socket operation on non-socket
SOCK_ERR_DESTADDRREQ	21039	Destination address required
SOCK_ERR_MSGSIZE	21040	Message too long
SOCK_ERR_PROTOTYPE	21041	Protocol wrong type for socket
SOCK_ERR_NOPROTOOPT	21042	Bad protocol option
SOCK_ERR_PROTONOSUPPORT	21043	Protocol not supported
SOCK_ERR_SOCKTNOSUPPORT	21044	Socket type not supported
SOCK_ERR_OPNOTSUPP	21045	Operation not supported on socket
SOCK_ERR_PFNOSUPPORT	21046	Protocol family not supported
SOCK_ERR_AFNOSUPPORT	21047	Address family not supported by protocol family
SOCK_ERR_ADDRINUSE	21048	Address already in use
SOCK_ERR_ADDRNOTAVAIL	21049	Can't assign requested address
SOCK_ERR_NETDOWN	21050	Network is down
SOCK_ERR_NETUNREACH	21051	Network is unreachable
SOCK_ERR_NETRESET	21052	Net dropped connection or reset
SOCK_ERR_CONNABORTED	21053	Software caused connection abort
SOCK_ERR_CONNRESET	21054	Connection reset by peer
SOCK_ERR_NOBUFS	21055	No buffer space available
SOCK_ERR_ISCONN	21056	Socket is already connected
SOCK_ERR_NOTCONN	21057	Socket is not connected
SOCK_ERR_SHUTDOWN	21058	Can't send after socket shutdown
SOCK_ERR_TOOMANYREFS	21059	Too many references, can't splice
SOCK_ERR_TIMEDOUT	21060	Connection timed out
SOCK_ERR_CONNREFUSED	21061	Connection refused
SOCK_ERR_LOOP	21062	Too many levels of symbolic links
SOCK_ERR_NAMETOOLONG	21063	File name too long
SOCK_ERR_HOSTDOWN	21064	Host is down
SOCK_ERR_HOSTUNREACH	21065	No Route to Host
SOCK_ERR_NOTEMPTY	21066	Directory not empty
SOCK_ERR_PROCLIM	21067	Too many processes
SOCK_ERR_USERS	21068	Too many users
SOCK_ERR_DQUOT	21069	Disc Quota Exceeded

SOCK_ERR_STALE	21070	Stale NFS file handle
SOCK_ERR_REMOTE	21071	Too many levels of remote in path
SOCK_ERR_SYSNOTREADY	21091	Network SubSystem is unavailable
SOCK_ERR_VERNOTSUPPORTED	21092	WINSOCK DLL Version out of range
SOCK_ERR_NOTINITIALISED	21093	Successful WSASTARTUP not yet performed
SOCK_ERR_HOST_NOT_FOUND	22001	Host not found
SOCK_ERR_TRY_AGAIN	22002	Non-Authoritative Host not found
SOCK_ERR_NO_RECOVERY	22003	Non-Recoverable errors: FORMERR, REFUSED, NOTIMP
SOCK_ERR_NO_DATA	22004	Valid name, no data record of requested type

Events

All of the events for this control are listed in the following table. Events that apply only to this control, or that require special consideration when used with it, are marked with an asterisk (*). For documentation of the remaining events, see Appendix A, "Standard Properties, Events, and Methods," in the Custom Control Reference.

*Accept	*Connect	*Listen	*SendReady
*Close	*Exception	*Receive	

Exception

Description

Indicates an error condition has occurred.

Syntax

Sub *dsSocket*_**Exception** ([*Index* **As Integer**,] *ErrorCode* **As Integer**, *ErrorDesc* **As String**)

Remarks

The ErrorCode parameter provides the number of the error that occurred and the ErrorDesc provides a text description of the error. Only asynchronous errors are provided with this event. Errors occuring during the setting of properties are returned using the standard Visual Basic Err and Error variables.

See Also

Error Codes

FwdLookup

Description

Resolves a host name to an IP address.

Syntax

[form.]dsSocket1.**FwdLookup**

Remarks

The FwdLookup method uses name resolution to convert a host name to its equivalent IP address. The host name to be resolved is taken from the RemoteHost property and the resulting address, if successful, is placed in the RemoteDotAddr property. If many connections will be repeatedly made to the same remote host, then resolving the address before connecting can improve performance. In this way the host name is only resolved once and the connections will be completed using the IP address.

Parameters

None

See Also

Action RemoteHost RemotDotAddr

LineMode

Description

Sets and returns the value of the LineMode flag.

Syntax

[form.]dsSocket1.**LineMode** [= *True* | *False*]

Remarks

The LineMode property determines the method of receiving data. When the LineMode is set True the data will be transferred in blocks that are terminated with the EOLChar value. Receive events will occur once for each line of data received on the network. If the LineMode is set True, and no EOLChar has been found in the data stream, the Receive event will be fired when the received data reaches DataSize bytes. If the connections is closed and data remains in the incoming buffer the Receive event will be fired with the remaining data regardless of whether it is terminated with EOLChar. When the LineMode is set False the data will be transferred as received.

Data Type

Boolean

See Also

EOLChar Receive

Linger

Description

Sets and returns the value of the Linger flag.

Syntax

[form.]dsSocket1.**Linger** [= *True* | *False*]

Remarks

The Linger flag determines the type of close utilized when the socket connection is closed. It is used in conjunction with the Timeout property. If the Linger property is set to True and the Timeout property is greater than 0 seconds, then setting the Action property to SOCK_ACTION_CLOSE will block until the socket is closed or the Timeout expires. The default value is True.

Linger flag	Timeout	Type of close	Wait for close?
True	0	Hard	No
True	>0	Graceful	Yes
False	<any>	Graceful	No

Data Type

Boolean

See Also

Action Timeout

Listen

Description

Indicates that a Listen has been established.

Syntax

Sub *dsSocket*_**Listen** ([*Index* **As Integer**])

Remarks

The Listen event indicates that a Listen has been established in response to setting the Action property to SOCK_ACTION_LISTEN. The control is now capable of receiving incoming connections from a client process.

See Also

Action LocalPort

Listen

Description

Initiates an active Listen on the specified control.

Syntax

[form.]dsSocket1.**Listen**

Remarks

The Listen method opens a port to listen for incoming connections. An incoming connection is signified by the firing of the Accept event. A socket that is listening for connection cannot be used to transfer data without first closing the Listen.

Parameters

None

See Also

Action LocalDotaddr LocalPort ServiceName

LocalDotAddr

Description

Sets and returns the IP address for accepting incoming connections.

Syntax

[form.]dsSocket1.**LocalDotAddr** [= setting$]

Remarks

Sets and returns the IP address to be used in accepting incoming network connections. The property value is in Internet dot notation (e.g.: 124.35.20.12). The LocalDotAddr can be set to "0.0.0.0" to accept any incoming connections on the LocalPort or ServiceName. The default value for LocalDotAddr is as defined in the network configuration parameters. Setting LocalDotAddr to an empty string will reset the property to the default value. Available at run-time only.

Data Type

String

See Also

LocalName

LocalName

Description

Sets and returns the network name of the local computer.

Syntax

[form.]dsSocket1.**LocalName** [= setting$]

Remarks

Sets and returns the network name of the local computer. The default value for LocalName is as defined in the network configuration parameters. Setting LocalName to an empty string will reset the property to the default value. Available at run-time only.

Data Type

String

See Also

LocalDotAddr

LocalPort

Description

Sets and returns the port number used on the local computer.

Syntax

[form.]dsSocket1.**LocalPort** [= *setting%*]

Remarks

Sets and returns the network port number that will be used for incoming connections. When the dsSocket control is in Listen mode a client application can connect to the computer by specifying this port number as its RemotePort. If the LocalPort property is set to zero then the ServiceName property will be used to resolve a port number for network Listen's. If the ServiceName property is also empty then Winsock will use a unique port number in the range 1024 to 5000. The assigned port number will be contained in the LocalPort property after a Listen action completes.

Data Type

Integer

See Also

Action RemotePort ServiceName

dsSocket Winsock Custom Control

Properties Events Methods Error Codes About

Description

The dsSocket OLE or VBX custom control provides support for TCP/IP network communications using Visual Basic. The control utilizes the Windows Sockets version 1.1 (Winsock) standard. This help file covers both the VBX and OCX version of dsSocket. Certain differences exist between the VBX and OCX version of the product.

Object Type

WinSock

Remarks

The Windows socket control provides the capability of communicating as either a client process or a server process depending on the specified action for that particular socket. Properties are utilized to setup the communication parameters and initiate the connection. While connected, events are provided that inform the programmer of changes in the connection status as well as data availability.

The standard Visual Basic variables Err and Error are used to return error status from property modification. The Exception event provides information regarding asynchronous errors.

A client connection can be established with only a few lines of code as shown below:

```
Sub Command1_Click()
        dsSocket1.RemoteHost = "Washington"
        dsSocket1.RemotePort = 7
        dsSocket1.Action = 1
End Sub

Sub dsSocket1_SendReady()
        dsSocket1.Send = "Testing 1... 2... 3..."
End Sub
```

In this sample the Receive event would be fired and the ReceiveData would contain an echo of the data placed in the Send property.

Distribution

Registered product owners are licensed to distribute the DSSOCK.VBX, DSSOCK16.OCX or DSSOCK32.OCX file with their applications. Under no circumstances are you permitted to distribute the DSSOCK.LIC file, this help file or any other files provided with the licensed distribution of the dsSocket product.

MaxSockets

Description

Returns the maximum number of sockets supported by the TCP/IP protocol stack.

Syntax

[form.]dsSocket1.**MaxSockets**

Remarks

Returns the maximum number of sockets supported by the TCP/IP protocol stack. This is a theoretical limit and is affected by various sytem configuration options including available memory and resources. Available at run-time only.

Data Type

Long

See Also

Description MaxUDPSize

MaxUDPSize

Description

Returns the maximum size of UDP packets supported by the TCP/IP protocol stack.

Syntax

[form.]dsSocket1.**MaxUDPSize**

Remarks

Returns the maximum size of a UDP network packet as defined by the TCP/IP protocol stack in use on the local computer. Available at run-time only.

Data Type

Integer

See Also

Description MaxSockets

Methods

All of the methods for this control are listed in the following table. Methods that apply only to this control, or that require special consideration when used with it, are marked with an asterisk (*). For documentation of the remaining methods, see Appendix A, "Standard Properties, Events, and Methods," in the Custom Control Reference.

*Close *FwdLookup *Connect *Listen *RevLookup

Overview

Description

The dsSocket custom control provides support for TCP/IP network communications using Visual Basic. The control utilizes the Windows Sockets (Winsock) standard.

Object Type

WinSock

Remarks

The dsSocket Windows socket control provides the capability of communicationg as either a client process or a server process depending on the specified action for that particular socket. Properties are utilized to setup the communication parameters and initiate the connection. While connected events are provided that inform the programmer of changes in the connection status as well as data availability.

The standard Visual Basic variables Err and Error are used to return error status from property modification. The Exception event provides information regarding asynchronous errors.

A client connection can be established with only a few lines of code as shown below:

```
dsSocket1.RemoteHost = "Washington"
dsSocket1.RemotePort = 7  ' port number of echo service
dsSocket1.Action = SOCK_ACTION_CONNECT
dsSocket1.Send = "Testing 1... 2... 3..."
```

In this sample the Receive event would be fired and the ReceiveData would contain an echo of the data placed in the Send property.

Distribution

Registered product owners are licensed to distribute the DSSOCK.VBX file with their applications. Under no circumstance are you permitted to distribute the DSSOCK.LIC file.

Properties

All of the properties for this control are listed in the following table. Properties that apply only to this control, or that require special consideration when used with it, are marked with an asterisk (*). For documentation of the remaining properties, see Appendix A, "Standard Properties, Events, and Methods," in the Custom Control Reference.

About	*EOLChar	*LocalPort	*Send
*Action	Index	*MaxSockets	*ServiceName
*BindConnect	Left	*MaxUDPSize	*Socket
*BytesSent	*LineMode	Name	*SocketType
*DataSize	*Linger	*RemoteDotAddr	*State
*Description	*LocalDotAddr	*RemoteHost	Tag
Enabled	*LocalName	*RemotePort	*Timeout
Top			

Receive

Description

Indicates that data has been received on a connection.

Syntax

Sub *dsSocket*_**Receive** ([*Index* **As Integer**,] *ReceiveData* **As String**)

Remarks

The Receive event occurs whenever a connected process sends data. The data is contained in the ReceiveData parameter. The length of the data string will not exceed the value specified in the DataSize property. The network is not required to transfer data in the same buffer sizes that they were sent in from the connected process. Partial or multiple buffers can arrive in the ReceiveData parameter. The normal method for processing the Receive event is to add the data to a predefined string and process the data when the required amount has been received.

Example:

```
Global szData   As String
Sub Receive( Index as Integer, ReceiveData as String )
        szData = szData + ReceiveData
        If ( Len(szData) > 100 ) Then
                Call ProcessData( Left( szData, 100 ) )
                szData = Mid( szData, 101 )
        End If
End Sub
```

See Also

DataSize Send

RemoteDotAddr

Description

Sets and returns the IP address of server for establishing a connection.

Syntax

[form.]dsSocket1.**RemoteDotAddr** [= *setting$*]

Remarks

The RemoteDotAddr property accepts a string in standard IP address dot notation (e.g.: 120.20.24.12) that will be used when making a connection. If an IP address is provided the control will not perform a name lookup when connecting to a server. If name resolution services are not available for a wide-area network use the RemoteDotAddr to provide the connection information. On a local-area network you can use either the RemoteHost or RemoteDotAddr property. The IP address must contain four numbers seperated by periods.

Data Type

String

See Also

Action RemoteHost

RemoteHost

Description

Sets and returns the name of the remote computer.

Syntax

[form.]dsSocket1.**RemoteHost** [= *setting$*]

Remarks

The RemoteHost property may be set the network name of the remote computer to connect to. If the RemoteDotAddr property is empty the dsSocket control will use the RemoteHost property to determine the network address for out-going connections. The RemoteHost name must either be contained in the HOSTS file or a name resoultion service (e.g.: DNS) must be available to resolve the network address of the remote computer.

Data Type

String

See Also

Action RemoteDotAddr

RemotePort

Description

Sets and returns the port number to connect to.

Syntax

[form.]dsSocket1.**RemotePort** [= *setting%*]

Remarks

Sets and returns the network port number that will be utilized for establishing connections. When setting the Action property to SOCK_ACTION_CONNECT, the RemotePort number must equate to a port number with a Listen active on the remote computer. If the RemotePort property is set to zero the ServiceName property will be used to resolve a port number for connections.

Data Type

Integer

See Also

Action RemoteHost ServiceName

RevLookup

Description

Resolves an IP address to a host name.

Syntax

[form.]dsSocket1.**RevLookup**

Remarks

The RevLookup method uses name and address resolution to convert an IP address to its equivalent host name . The IP address to be resolved is taken from the RemoteDotAddr property and the resulting host name , if successful, is placed in the RemoteHost property. This method can be used to determine the name of a client connecting to a server application.

Parameters

None

See Also

Action RemoteHost RemotDotAddr

Send

Description

Sets the data for sending on the network.

Syntax

[form.]dsSocket1.**Send** = *setting$*

Remarks

Setting the Send property to a string will cause the data to be sent over the network. To send data the socket must be in a connected state. The maximum string size for the Send property is 32767 characters, a limit of the WinSock 1.1 specification. Data blocks larger than 32767 must be broken down for network transfers. The Winsock network support will attempt to send the data immediately. If this is not possible then the error SOCK_ERR_WOULDBLOCK will be returned and the SendReady event will occur when the network is again ready for data transfer. You can use this event to set the data into the Send property again.

```
Example:    dsSocket1.Send = "This is my information."
            ' this will send a string, the other end of the connection will get a Receive event
            ' with the string passed in the ReceiveData parameter
```

Data Type

String

See Also

Receive SendReady

SendReady

Description

Indicates the network is ready to transfer data.

Syntax

Sub *dsSocket*_**SendReady** ([*Index* **As Integer**])

Remarks

The SendReady event occurs when the network sub-system changes from a not-ready state to a ready state. When sending data, if the network is not ready the error SOCK_ERR_WOULDBLOCK is returned. When the network is again ready the SendReady event will occur and the programmer can then resend the failed data transfer. The normal use of this event is to set a global flag that indicates the network is ready. When the SOCK_ERR_WOULDBLOCK occurs the flag is cleared. Before sending data check the flag and if it is set the network can accept data.

See Also

Send

ServiceName

Description

Sets and returns the name of the network service to use.

Syntax

[form.]dsSocket1.**ServiceName** [= *setting$*]

Remarks

Sets and returns the name of the network service that will be utilized for incoming Listen's and outgoing Connect's. This name is converted to a port number on the local computer by accessing the services data file or a service name resolution process. The ServiceName may be used in place of specifying the RemotePort or LocalPort properties. This property setting will be ignored if the RemotePort (connect) or LocalPort (listen) properties are non-zero.

Data Type

String

See Also

Action LocalPort RemotePort

Socket

Description

Sets or returns the socket number to be used for communication.

Syntax

[form.]dsSocket1.**Socket** [= *setting%*]

Remarks

The Socket property is set in response to an Accept event. It must be set to the value of the SocketID parameter provided in the Accept event. The Socket property will be associated with a new instance of the control that is created when handling the Accept event.

Example:

```
Sub dsSocket1_Accept( Index As Integer, SocketID As Integer )
        ' create a new control to handle the incoming connection
        Load dsSocket(1)
        ' set it to the specified socket number
        dsSocket1.Socket = SocketID
        ' dsSocket1(1) will now be used to send and receive data
End Sub
```

Data Type

Integer

See Also

Accept

SocketType

Description

Sets and returns the mode of data transfer.

Syntax

[form.]dsSocket1.**SocketType** [= *setting%*]

Setting

The SocketType property settings are:

0	TCP streams
1	UDP datagrams

Remarks

Sets and returns the method used for transferring data on the network. This property is an enumerated list which can be set to 0 ("TCP streams") or 1 ("UDP datagrams"). The transfer method must be set before setting the Action property, or using the Connect method, and cannot be changed until the socket connection is closed. When using TCP streams a point to point private connection is established to the remote host. This is a reliable link with error checking and data recovery. When using UDP datagrams a socket is created but there is no connection established. This is considered an unreliable link and there is no error checking or data recovery. The size of a data packet is limited by the value returned from MaxUdpSize. Using UDP datagrams provides a faster method of sending data where a connection would normally be closed after each data transmission.

Data Type

Integer (Enumerated)

See Also

Action Connect

State

Description

Sets and returns the current state of a socket control.

Syntax

[form.]dsSocket1.**State** [= *setting%*]

Remarks

The State property will reflect the known condition of the control. If an Exception has occurred the State property will not change to reflect the error condition. The programmer should determine if the error has disconnected the socket. The property will be correctly set after an Action has been specified.

Possible values for the State property are:

1	SOCK_STATE_CLOSED
2	SOCK_STATE_CONNECTED
3	SOCK_STATE_LISTENING
4	SOCK_STATE_CONNECTING
5	SOCK_STATE_ERROR
6	SOCK_STATE_CLOSING
7	SOCK_STATE_UNKNOWN
8	SOCK_STATE_BUSY
9	SOCK_STATE_UDPACTIVATING
10	SOCK_STATE_UDPACTIVE

Data Type

Integer

See Also

Action Exception

Timeout

Description

Sets or returns the timeout for socket closing.

Syntax

[form.]dsSocket1.**Timeout** [= *setting%*]

Remarks

The Timeout property is utilized in conjuction with the Linger property to determine the type of close used when a socket connection is closed. The value is specified in seconds. If the Linger property is set to True then setting the Action property to SOCK_ACTION_CLOSE will block until the socket is closed or the Timeout expires. The default value is 10 seconds.

Linger flag	Timeout	Type of close	Wait for close?
True	0	Hard	No
True	>0	Graceful	Yes
False	<any>	Graceful	No

Data Type

Integer

See Also

Action Linger