VBSock (version 1.1)

This is the help for the alpha release of the VBSock.VBX

It does not elaborate on the functions of creating a TCP/IP application but only specifies what that various properties are and what they represent. It is assumed that the user will have a working knowledge of the current Winsock 1.1 standards. (Or at least have a copy to look at).

Overview:

This all started when we needed to write a TCP/IP application. We began parts of the project in C++ so we wrote a DLL that took care of the TCP/IP functions. We later switched to Visual Basic for development. We decided to use the same DLL and then just elaborated on that DLL making into a VBX.

Making the VBX was an exercise to see if we could do it.

The VBX has been tested with various stacks including Trumpet, WFWG TCP/IP, Netmanage Chameleon, and Distinct. It worked equally well with all mentioned stacks.

The VBX was primarily created to be used for TCP but we decided to include UDP abilities as well. It has most of the features listed in the Winsock 1.1 standards and can be expanded to met those needs and any future implementations of the standard.

This is an alpha release of the VBX. It might have some bugs that we havent encountered in out applications and can not be held liable for problems you might incur in the use of this product. The cost of this release is FREE. The only thing we ask is a byline mention within any product you write and distribute. We want to find out if the world really wants the product first. Depending on the response we will upgrade the product per any opinions and suggests we get from users. At that time we will charge a modest fee to cover our costs. This amount has not been set but I estimate it will be around $25. It also will not have any royalties attached.

If you register this copy by simply sending an email note indicating that you are using the product we will inform you of any updates to the product.

Properties:

ErrorFlag
hWnd
Height
HostAddress
Index
Left
Name
OpenFlag
Parent
Port
Protocol
SelectAction
SinFamily
SocketNumber
Tag
Top
Width

Name Property

Applies To

VBServer, VBClient

Description

Specifies the name used in code to identify a form, control, or data access object. Not available at run time.

Index Property

Applies To

VBServer, VBClient

Description

Specifies the number that uniquely identifies a control in a control array. Available only if the control is part of a control array; read-only at run time.

Parent Property

Applies To

VBServer, VBClient

Description

Specifies the form on which a control is located; not available at design time and read-only at run time.

Tag Property

Applies To

VBServer, VBClient

Description

Stores any extra data needed for your program. Unlike other properties, the value of the Tag property is not used by Visual Basic; you can use this property to identify specific objects.

Left Property

Applies To

VBServer, VBClient

Description

Left determines the distance between the internal left edge of an object and the left edge of its container.

Top Property

Applies To

VBServer, VBClient

Description

Top determines the distance between the internal top edge of an object and the top edge of its container.

Width Property

Applies To

VBServer, VBClient

Description

Determine the dimensions of an object.

Height Property

Applies To

VBServer, VBClient

Description

Determine the dimensions of an object.

hWnd Property

Applies To

VBServer, VBClient

Description

Specifies a handle to a form or control. Not available at design time; read-only at run time.

SocketNumber Property

Applies To

VBServer, VBClient

Description

Specifies the socket handle of the control; Not available at design time; read-only at run time.

.

Protocol Property

Applies To

VBServer, VBClient

Description

Specifies which protocol to use when opening the socket. Currently only TCP and UDP are supported in the Winsock standard, but all other protocols have been included for upward compatibility.

Port Property

Applies To

VBServer, VBClient

Description

Specifies which port to use during the socket open call.

SelectAction Property

Applies To

VBServer, VBClient

Description

Specifies the actions that WSASelect should monitor. This property is important to the Message Event.

OpenFlag Property

Applies To

VBServer, VBClient

Description

Set this property to TRUE to open the socket and FALSE to close it.; Not available at design time; read-only at run time.

ErrorFlag Property

Applies To

VBServer, VBClient

Description

This property returns the last Winsock error that occurred; Not available at design time; read-only at run time.

SinFamily Property

Applies To

VBServer, VBClient

Description

This Property is used during the open, this property sets up the Sin Family according to the Winsock standard.

HostAddress Property

Applies To

VBClient

Description

This property sets the address that the client should use to attach to the host.

Events:

There is only the one event:

Message

Applies To

VBServer, VBClient

Description

This event occurs in response to the action the *Select* command is monitoring - the SelectAction property sets this up. Common events that are monitored are *Read, Write, Close etc.*

Procedures

We designed this VBX to use several procedure calls. We could have embedded them into properties but this gave us more flexibility to use the VBX as a DLL and deal with all of the controls features manually without needing VBX capabilities. The following procedures are just wrappers to the actual calls that they mimick in Winsock.DLL. You can read a detailed account of them in the Winsock standards. We placed them hear as wrappers so that the program only needs to deal with one file the VBSock.VBX.

GetSelectEventSocket
RecvSocket
acceptSocket
SendSocket
RecFrom SendTo

GetSelectEventSocket

Following is the actual C++ code that is performed from this call

RetVal = WSAGETSELECTEVENT (PParam);

RecvSocket

Declare Function RecvSocket Lib "vbsock.vbx" (ByVal PSocket As Integer, ByVal PBuf As String, ByVal PLen As Integer, ByVal PFlags As Integer) As Integer

(What follows was extracted from the Winsock 1.1 specs)

int PASCAL FAR recv (int s, char FAR * buf, int len, int flags);

s	A descriptor identifying a connected socket.
buf	A buffer for the incoming data.
len	The length of buf.
flags	Specifies the way in which the call is made.
Remarks	This function is used on connected datagram or stream sockets specified by the s parameter and is used to read incoming data.

acceptSocket

Declare Function acceptSocket Lib "vbsock.vbx" (ByVal PSocket As Integer) As Integer

(What follows was extracted from the Winsock 1.1 specs)

SOCKET PASCAL FAR accept (SOCKET s, struct sockaddr FAR * addr, int FAR * addrlen);

s	A descriptor identifying a socket which is listening for connections after a listen().
addr	An optional pointer to a buffer which receives the address of the connecting entity, as known to the communications layer. The exact format of the addr argument is determined by the address family established when the socket was created.
addrlen	A optional pointer to an integer which contains the length of the address addr.
Remarks	This routine extracts the first connection on the queue of pending connections on s, creates a new socket with the same properties as s and returns a handle to the new socket. If no pending connections are present on the queue, and the socket is not marked as non-blocking, accept() blocks the caller until a connection is present. If the socket is marked non-blocking and no pending connections are present on the queue, accept() returns an error as described below. The accepted socket may not be used to accept more connections. The original socket remains open.

SendSocket

Declare Function SendSocket Lib "vbsock.vbx" (ByVal s As Integer, ByVal buf As String, ByVal llen As Integer, ByVal flags As Integer) As Integer

(What follows was extracted from the Winsock 1.1 specs)

int PASCAL FAR send (SOCKET s, const char FAR * buf, int len, int flags);

s	A descriptor identifying a connected socket.
buf	A buffer containing the data to be transmitted.
len	The length of the data in buf.
flags	Specifies the way in which the call is made.

Remarks send() is used on connected datagram or stream sockets and is used to write outgoing data on a socket. For datagram sockets, care must be taken not to exceed the maximum IP packet size of the underlying subnets, which is given by the iMaxUdpDg element in the WSAData structure returned by WSAStartup(). If the data is too long to pass atomically through the underlying protocol the error WSAEMSGSIZE is returned, and no data is transmitted.

RecvFrom SentTo

The follow routines are available in the VBSock.VBX files but have never been tested by use. If you would like to try them give it a go. They are declared in the VBX as follows. If you design your Delcare to this you should be OK.

```
int _export PASCAL FAR RecvFromSocket (SOCKET PSocket, char FAR * PBuf, int PLen, int PFlags,
                                       struct sockaddr FAR *PAddress, int FAR * PAddressLen)

int _export PASCAL FAR SendToSocket(SOCKET PSocket, const char FAR * PBuf, int PLen, int PFlags,
                                    const struct sockaddr FAR *PAddress, int PAddressLen)
```

Chat

This is a simple example program that uses the features of the VBSock.VBX

It acts as both a server and client application. You are free to use and adapt this program as you see fit.

Contact the authors

Send any questions, registration, or feedback to:

Miracle Concepts, Inc.
miraclec@epix.net