Getting Started with the CITCP Control

Related Topics

This topic introduces the TCP/IP protocol and the Crescent Internet ToolPak TCP control (CITCP). You can access the following information:

- What TCP/IP is
- What the CITCP Control is
- What kinds of applications you can build
- Client Applications
- Server Applications
- UDP Applications

The programming tasks topic identifies each task that CITCP supports and list the properties to set and the methods to call, then it illustrates the task with a code sample.

What is TCP/IP?

Related Topics

TCP/IP stands for Transmission Control Protocol/Internet Protocol. TCP/IP is the network protocol that enables computers to communicate using the Internet. It is a stream-oriented protocol in which the host simply creates, fills, and sends buffers (or packets) to a specified address until the transaction is complete. It is part of the seven-layer Open Systems Interconnect (OSI) networking model.

What is the CITCP Control?

Related Topics

The Crescent Internet ToolPak TCP (CITCP control) lets Visual Basic programmers build customized TCP/IP client, server, and User Datagram Protocol (UDP) applications.

CITCP has a simple programming interface in which you set property values and call methods to perform the desired tasks. The methods (which generally do not take arguments) take the necessary data directly from the property sheet, and either submit the request to the server (for client applications), or act as a server by listening on a port for connection requests. For client applications, when the server returns its response, the control parses the data and updates the appropriate properties. You can simply read the data from a property value or a ListBox object.

Client Applications

Related Topics

Client applications include applications that perform database, or lookup, tasks such as:

- Retrieving a host name given a host address (bGetHostByName)
- Retrieving a port number given a service (bGetServiceByName)

Because CITCP implements these tasks as methods, these applications can be quite simple to develop.

The lookup tasks are implemented as blocking methods. A *blocking method* prevents (or blocks) your application from executing any other code until the function completes. The CITCP control provides these blocking methods (identified by the lowercase b) for client lookup tasks; bGetHostByAddress, bGetHostByName, bGetHostName, bGetServiceByName, and bGetServiceByPort.

Unlike the other Internet ToolPak controls, the client applications do not always have to connect to a TCP/IP host before it can retrieve data. The blocking methods, for example, access data from the local workstations hosts file. You can perform these tasks with the blocking methods

Server Applications

Related Topics

Server applications are more flexible but can be more complicated to develop. CITCP includes a method that enables a local workstation to act as a server (ListenForConnection method) of any type. You can customize the service (ServiceName property) that the workstation provides as well as the port (Port property) on which it is provided. However, you must code the part of the application that provides the actual service.

Act as a TCP/IP server.

UDP Applications

Related Topics

CITCP also lets you build applications that use the User Datagram Protocol (UDP) layer of the TCP/IP model. UDP is a connectionless protocol that does not guarantee that a packet it sends will be delivered, or if it is delivered, that it will be delivered in the order in which it was sent. CITCP provides support for both client applications (SendDatagram method) and server applications (ListenForDatagram method). The programming tasks include:

Act as a UDP server.

Act as a UDP client.

Get a Host Name When You Have The Host Address

Related Topics

1.Set this property:
HostAddress property

2. Call the bGetHostByAddress method

The code looks like this:

```
Form_Load
      CITCP1.HostAddress = 198.137.64.1
      CITCP1.bGetHostByAddress

Property_Changed()
      Print CITCP1.HostName
```

Once the PropertyChanged event fires, you can read the contents of the HostName property.

NOTE The other blocking methods can be programmed the same way.

Connect to a TCP/IP Host

Related Topics

To build other client-type applications, you must connect to a TCP/IP host. The connection includes:

1. Setting the HostName or HostAddress property and the Port property.

2. Calling the ConnectToHost method.

The code looks like this:

```
CITCP1.HostName = TCPHostName
CITCP1.PORT = 119
nResult = CITCP1.ConnectToHost
```

You are connected when the Connection event fires. When the connection attempt succeeds, *nResult* is an integer that represents the socket number opened by the ConnectToHost method. When the connection attempt fails, *nResult* is 0 and the WSAError event fires.

Once connected, you can send and receive data with the CITCP control.

Act as a TCP Server

Related Topics

1. Set this property:
   Port property

2. Call the Call the ListenForConnection method.

The code looks like this:

```
Form_Load
      CITCP1.Port = 119
      CITCP1. ListenForConnection

PacketReceived
      Code to respond to connection request (for example the SEND
      method)
```

Act as a UDP Server

Related Topics

1. Set this property:
   Port property

2. Call the ListenForDatagram method.

The code looks like this:

```
Form_Load
   CITCP1.Port = 119
   CITCP1. ListenForDatagram

PacketReceived
   Code to respond to datagram (for example the SendDatagram method)
```

Act as a UDP Client

Related Topics

1. Call the SendDatagram method.

The code looks like this:

```
Form_Load
      CITCP1. SendDatagram
```

The Crescent Internet ToolPak TCP/IP Control

Related Topics

Control File

CITCP.OCX

Object Type

CITCP

Purpose

The Crescent Internet ToolPak TCP/IP Conrol (CITCP.OCX) provides support for general purpose TCP/IP messaging and TCP/IP database functions.

Properties

About
Container
Drag
DragIcon
DragMode
Height
HostAddressc
HostName c
Index
Left
Move
Name
Object
Parent
Port c
ServiceName c
ShowWhatThis
Socket c
Tag
Top
Visible
WhatsThisHelpID
Width
ZOrder

Events

BlockingFunctionCompletedc
Connectionc
ConnectionClosedc
DragDrop
DragOver
PacketReceived c
PacketSentc
PropertyChangedc
WSAErrorc

Methods

bGetHostByAddressc
bGetHostByNamec
bGetHostNamec
bGetServiceByNamec
bGetServiceByPortc
CloseSocketc
ConnectToHostc
ListenForConnectionc
ListenForDatagramc
Sendc
SendDatagramc

c A custom or modified property, method, or event.

HostAddress Property

Applies To

CITCP

Purpose

The HostAddress property sets or returns the IP address of the TCP/IP host.

Syntax

```
[Form.]CITCP.HostAddress[ = String$]
```

Data Type

String

Usage

Read/Write at design time.

Comments

The HostAddress identifies the host to the Internet. HostAddress consists of the network number and the local host number.

An example of an IP address is:

```
198.137.64.1
```

NOTE You must provide either the HostName or HostAddress property to connect to a TCP/IP host. When you provide both, CITCP uses the HostName.

See Also

ConnectToHost, HostName

HostName Property

Applies To

CITCP

Purpose

The HostName property sets or returns the DNS name of the TCP/IP host.

Syntax

[*Form.***]** *CITCP*.HostName**[** = *String$***]**

Data Type

String

Usage

Read/Write at design time and runtime.

Comments

The HostName is the name that identifies a TCP/IP host to the Internet. Depending on your connection to the Internet, the HostName might contain simply a machine name like `crescentserver`, or it can be a fully qualified Internet name that follows this format: `machine.organizationname.com.` Contact your site administrator to determine the correct format to supply.

Set HostAddress or HostName before calling the ConnectToHost method.

NOTE You must provide either the HostName or HostAddress property to connect to a TCP/IP host. When you provide both, CITCP uses the HostName.

See Also

ConnectToHost, HostAddress

Port Property

Applies To

CITCP

Purpose

The Port property sets the TCP/IP port number.

Syntax

```
[Form.] CITCP.Port[ = Integer%]
```

Data Type

Integer

Usage

Read/Write at design time and runtime.

Comments

The port number is assigned to the service that the port provides. The following table lists the default port assignments for the services used by the other Crescent Internet ToolPak controls.

Control Name	TCP/IP Service Name	Port Assignment
CIMAIL	SMTP	25
CIMAIL	POP3	110
CINEWS	NNTP	119
CIHTTP	HTTP	80
CIFTP	FTP	21

The Port property is used with the the bGetServiceByPort and the bGetPortByService methods. The bGetServiceByPort method requires that you provide a port number and then populates the Service property with the name of the service that the port provides. The bGetPortByService method requires that you provide a service name and then populates the Port property with the port number that provides that service.

See Also

bGetServiceByName, bGetServiceByPort, BlockingFunctionCompleted, PropertyChanged

ServiceName Property

Applies To

CITCP

Purpose

The ServiceName property sets the name of the TCP/IP service provided by the port specified by the Port property.

Syntax

```
[Form.] CITCP.ServiceName[ = String$]
```

Data Type

String

Usage

Read/Write at design time and runtime.

Comments

This table lists the services and ports used by the other Crescent Internet ToolPak controls.

Control Name	TCP/IP Service Name	Port Assignment
CIMAIL	SMTP	25
CIMAIL	POP3	110
CINEWS	NNTP	119
CIHTTP	HTTP	80
CIFTP	FTP	21

The Service property is used with the the bGetServiceByPort and the bGetPortByService methods. The bGetServiceByPort method requires that you provide a port number and then populates the Service property with the name of the service that the port provides. The bGetPortByService method requires that you provide a service name and then populates the Port property with the port number that provides that service.

See Also

bGetServiceByName, bGetServiceByPort, BlockingFunctionCompleted, PropertyChanged

Socket Property

Applies To

CITCP

Purpose

The Socket property sets or returns the socket opened by the ConnectToHost method.

Syntax

```
[Form.]CITCP.Socket[ = Integer%]
```

Data Type

Integer

Usage

Read/Write at design time and runtime.

Comments

ConnectToHost

bGetHostByAddress Method

Applies To

CITCP

Purpose

The bGetHostByAddress method is a blocking function that obtains a host name for a given host address.

Syntax

```
CITCP.bGetHostByAddress
```

Data Type

Integer

Comments

Set the HostAddress property before invoking bGetHostByAddress. When bGetHostByAddress completes, the following occurs:

- The BlockingFunctionCompleted event fires.
- The bGetHostByAddress populates the HostName property with the DNS host name for the given address.
- The PropertyChanged event fires, when the HostName property gets updated.

The methods that perform blocking functions (bGetHostByAddress, bGetHostByName, bGetHostName, bGetServiceByPort, bGetPortByService) do not require connection to a TCP/IP host. They simply query the local workstations hosts file for the requested data.

See Also

BlockingFunctionCompleted, HostAddress, HostName, PropertyChanged

bGetHostByName Method

Applies To

CITCP

Purpose

The bGetHostByName method is a blocking function that obtains the host address for a given host name.

Syntax

CITCP.bGetHostByName

Data Type

Integer

Comments

Set the HostName property before invoking bGetHostByName. When bGetHostByName completes, the following occurs:

- The BlockingFunctionCompleted event fires
- The bGetHostByName populates the HostName property with the DNS host name for the given address.
- The PropertyChanged event fires, when the HostName property gets updated.

The methods that perform blocking functions (bGetHostByAddress, bGetHostByName, bGetHostName, bGetServiceByPort, bGetPortByService) do not require connection to a TCP/IP host. They simply query the local workstations hosts file for the requested data.

See Also

bGetHostByAddress, BlockingFunctionCompleted, HostAddress, HostName, PropertyChanged

bGetHostName Method

Applies To

CITCP

Purpose

The bGetHostName method is a blocking function that obtains the host name of the local workstation.

Syntax

CITCP.bGetHostName

Data Type

Integer

Comments

The bGetHostName method retrieves the local workstations host name from the host file. When bGetHostName completes, the following occurs:

- The BlockingFunctionCompleted event fires
- The bGetHostName populates the HostName property.
- The PropertyChanged event then fires, when the HostName property is populated.

The methods that perform blocking functions (bGetHostByAddress, bGetHostByName, bGetHostName, bGetServiceByPort, bGetPortByService) do not require connection to a TCP/IP host. They simply query the local workstations hosts file for the requested data.

See Also

bGetHostByAddress, BlockingFunctionCompleted, HostAddress, HostName, PropertyChanged

bGetServiceByName Method

Applies To

CITCP

Purpose

The bGetServiceByName method is a blocking function that obtains the port number from the service name.

Syntax

```
CITCP.bGetServiceByName
```

Data Type

Integer

Comments

Set the HostName or HostAddress and the ServiceName properties before invoking bGetServiceByName. When bGetServiceByName completes, the following occurs:

- The BlockingFunctionCompleted event fires.
- The bGetServiceByName populates the Port property with the port number that the host uses for the named service.
- The PropertyChanged event fires, when the Port property gets updated by bGetServiceByName.

The methods that perform blocking functions (bGetHostByAddress, bGetHostByName, bGetHostName, bGetServiceByPort, bGetPortByService) do not require connection to a TCP/IP host. They simply query the local workstations hosts file for the requested data.

See Also

bGetServiceByPort, BlockingFunctionCompleted, HostAddress, HostName, Port, PropertyChanged, ServiceName

bGetServiceByPort Method

Applies To

CITCP

Purpose

The bGetServiceByPort method is a blocking function that obtains the service name from the port number.

Syntax

```
CITCP.bGetServiceByPort
```

Data Type

Integer

Comments

Set the HostName or HostAddress and the Port properties before invoking bGetServiceByPort. When bGetServiceByPort completes, the following occurs:

- The BlockingFunctionCompleted event fires.
- bGetServiceByPort populates the ServiceName property with the name of the service that the host associates with the given port number.
- The PropertyChanged event fires, when the Port property gets updated by bGetServiceByPort.

The methods that perform blocking functions (bGetHostByAddress, bGetHostByName, bGetHostName, bGetServiceByPort, bGetPortByService) do not require connection to a TCP/IP host. They simply query the local workstations hosts file for the requested data.

See Also

bGetServiceByName, BlockingFunctionCompleted, HostAddress, HostName, Port, PropertyChanged, ServiceName

CloseSocket Method

Applies To

CITCP

Purpose

The CloseSocket method lets you close the current socket.

Syntax

```
nResult = CITCP.CloseSocket
```

Data Type

Integer

Comments

Use the CloseSocket method when you are finished with a particular socket.

ConnectToHost Method

Applies To

CITCP

Purpose

The ConnectToHost method establishes a connection to a TCP/IP host.

Syntax

```
nResult = CITCP.ConnectToHost
```

Data Type

Integer

Comments

Set the HostName **or** HostAddress property and the Port property before calling ConnectToHost.

When the connection attempt succeeds, *nResult* is an integer that represents the socket number opened by the ConnectToHost method. In addition, when the ConnectToHost method succeeds, the Connection event fires.When the connection attempt fails, *nResult* is 0 and the WSAError event fires.

You do not have to be connected to a TCP/IP host to use any of the blocking functions to obtain information about port numbers, service names, host addresses or host names, or any of the connectionless methods (SendDatagram and ListenForDatagram).

See Also

HostAddress, HostName, ListenForDatagram, Port, SendDatagram, WSAError

ListenForConnection Method

Applies To

CITCP

Purpose

The ListenForConnection method listens on a port for connection.

Syntax

```
nResult = CITCP.ListenForConnection
```

Data Type

Integer

Comments

The workstation listens on the port specified by the Port property. The ListenForConnection method enables a local workstation to act as a TCP/IP host. It can perform any valid TCP/IP service once acting as a TCP/IP host.

When the method succeeds, *nResult* is an integer that represents the socket number opened by the ListenForConnection method. In addition, when the ListenForConnection method succeeds, the Connection event fires.When the connection attempt fails, *nResult* is 0 and the WSAError event fires.

See Also

Port

ListenForDatagram Method

Applies To

CITCP

Purpose

The ListenForDatagram method listens on a port for a datagram.

Syntax

```
nResult = CITCP.ListenForDatagram
```

Data Type

Integer

Comments

The workstation listens on the port specified by the DataPort property.This method enables a local workstation to act as a TCP/IP host machine.

When the method succeeds, *nResult* is an integer that represents the socket number opened by the ListenForDatagram method.

Call the ListenForDatagram method if you need to receive and process datagrams.

See Also

Port

Send Method

Applies To

CITCP

Purpose

The Send method sends a packet to a specified host address and port number.

Syntax

```
CITCP.Send
```

Data Type

Integer

Comments

Set the HostAddress and Port properties before calling the Send method. For client applications, you must have called the ConnectToHost method before calling Send. For server applications, you must have called the ListenForConnection method before calling Send.

When the Send method succeeds, the Connection event fires. When the connection attempt fails, the WSAError event fires.

See Also

HostAddress, Port

SendDatagram Method

Applies To

CITCP

Purpose

The SendDatagram method sends a connectionless packet.

Syntax

```
CITCP.SendDatagram
```

Data Type

Integer

Comments

Set the HostAddress and Port properties before calling the SendDatagram method.

When the SendDatagram method succeeds, the Connection event fires. When the connection attempt fails the WSAError event fires.

See Also

HostAddress, Port

BlockingFunctionCompleted Event

Applies To

CITCP

Purpose

The BlockingFunctionCompleted event fires when a blocking function has finished operation.

Syntax

```
Sub CITCP_BlockingFunctionCompleted()
```

Comments

A blocking function suspends your application until Winsock completes the operation that you requested. For example, if you invoke the bGetHostByAddress method which obtains the host address for a given host name, your application is suspended until the host address is returned.

The CITCP blocking functions include: bGetHostByAddress, bGetHostByName, bGetHostName,bGetServiceByName and bGetServiceByPort.

See Also

bGetHostByAddress, bGetHostByName, bGetHostName, bGetServiceByName, bGetServiceByPort

Connection Event

Applies To

CITCP

Purpose

The Connection event fires when a connection occurs.

Syntax

```
Sub CITCP_Connection()
```

Comments

If the current application is a client application, the Connection event fires in when the application successfully connects to a TCP/IP host (initiated by the ConnectToHost method). If the current application is a server application, the Connection event fires when a client successfully connects to it. An application acts as a server when the ListenForConnection method is called.

See Also

ConnectToHost, ListenForConnection

ConnectionClosed Event

Applies To

CITCP

Purpose

The ConnectionClosed event fires when a connection to a TCP/IP host or a TCP/IP client terminates.

Syntax

```
Sub CITCP_Connection()
```

Comments

Because the CITCP control can act as either a host or a client, the ConnectionClosed event fires when a connection to a TCP/IP host or a connection fromn a TCP/IP client terminates.

PacketReceived Event

Applies To

CITCP

Purpose

The PacketReceived event fires when the CITCP control receives a packet from the connected host.

Syntax

```
Sub CITCP_PacketReceived(ByVal Packet As String, ByVal bytes_in As Integer)
```

Comments

The *Packet* is the actual data received from the server. Its contents will vary depending on the method used to obtain the packet. The *bytes_in* value is the number of bytes in the current packet (which may be different than the total number of bytes received by any given operation).

You might receive any number of packets from the server in response to any single request for data because the protocol determines the packet size.

See Also

bGetHostByAddress, bGetHostByName, bGetHostName, bGetServiceByName, bGetServiceByPort, PacketSent

PacketSent Event

Applies To

CITCP

Purpose

The PacketSent event fires when the CITCP control sends a TCP/IP packet to a TCP/IP host.

Syntax

```
Sub CITCP_PacketSent(ByVal bytes_out As Integer)
```

Comments

The value of *bytes_out* is the number of bytes in the current packet. The PacketSent event fires in response to the Send and SendDatagram methods.

See Also

PacketReceived, Send, SendDatagram

PropertyChanged Event

Applies To

CITCP

Purpose

The PropertyChanged event fires when a property value has been changed by a blocking function.

Syntax

```
Sub CITCP_PropertyChanged()
```

Comments

Each of the blocking methods (bGetHostByAddress, bGetHostByName, bGetHostName,bGetServiceByName, bGetServiceByPort) updates one or more properties. When any of these methods completes, the BlockingFunction event fires first, followed by PropertyChanged.

See Also

bGetHostByAddress, bGetHostByName, bGetHostName, bGetServiceByName, bGetServiceByPort

WSAError Event

Applies To

CITCP

Purpose

The WSAError event fires when the CITCP control receives a Winsock error.

Syntax

```
Sub CITCP_WSAError(ByVal error_number As Integer)
```

Comments

Use the WSAError event to monitor the Winsock activity. The *error_number* identifies the Winsock error that occurred. These errors are enumerated in the CITPAK.BAS file.

See Also

CITPAK.BAS

General Tab

Applies To

CITCP

This custom tab lets you set these custom properties:

TCP/IP Configuration:

HostName

HostAddress

Port

ServiceName