Delete Event

{ewc HLP95EN.DLL,DYNALINK,"See Also":"vbevtDeleteEventC"} {ewc HLP95EN.DLL,DYNALINK,"Example":"vbevtDeleteEventX":1} {ewc HLP95EN.DLL,DYNALINK,"Applies To":"vbevtDeleteEventA"} {ewc HLP95EN.DLL,DYNALINK,"Specifics":"vbevtDeleteEventS"}

This event occurs after the successful completion of a Delete request.

Syntax

Development Tool	Syntax
Microsoft Access and Visual Basic	*object*_**Delete**
Visual FoxPro	PROCEDURE *Object*.Delete
Visual C++	**void** *dialogclass*::OnDelete*Control*( );

The *object* placeholder represents an object expression that evaluates to an object in the Applies To list.

Arguments

None.

Last Event

{ewc HLP95EN.DLL,DYNALINK,"See Also":"vbevtLastEventC"} {ewc HLP95EN.DLL,DYNALINK,"Example":"vbevtLastEventX":1} {ewc HLP95EN.DLL,DYNALINK,"Applies To":"vbevtLastEventA"} {ewc HLP95EN.DLL,DYNALINK,"Specifics":"vbevtLastEventS"}

This event occurs after the successful completion of a Last request. The event passes the number of the last message accessed by the client.

Syntax

Development Tool	Syntax
Microsoft Access and Visual Basic	*object*_**Last** *(***ByVal** *Number* **As Long***)*
Visual FoxPro	PROCEDURE *Object*.Last LPARAMETERS *nMessageNumber*
Microsoft Visual C++	**void** *dialogclass*::OnLast*Control*(**long** *Number*);

The *object* placeholder represents an object expression that evaluates to an object in the Applies To list.

Arguments

Development Tool	Argument	Description
Microsoft Access, Visual Basic, and Visual C++	*Number*	Number of the last message accessed by the client.
Visual FoxPro	*nMessageNumber*	Number of the last message accessed by the client.

Data Type

Long.

MessageSize Event

{ewc HLP95EN.DLL,DYNALINK,"See Also":"vbevtMessageSizeEventC"} {ewc HLP95EN.DLL,DYNALINK,"Example":"vbevtMessageSizeEventX":1} {ewc HLP95EN.DLL,DYNALINK,"Applies To":"vbevtMessageSizeEventA"} {ewc HLP95EN.DLL,DYNALINK,"Specifics":"vbevtMessageSizeEventS"}

This event occurs after successful completion of a MessageSize request.

Syntax

Development Tool	Syntax
Microsoft Access and Visual Basic	*object*_**MessageSize** *(***ByVal** *msgSize* **As Long***)*
Visual FoxPro	PROCEDURE *Object*.MessageSize LPARAMETERS *nMessageSize*
Visual C++	**void** *dialogclass*::OnMessageSize*Control*(**long** *msgSize*);

The *object* placeholder represents an object expression that evaluates to an object in the Applies To list.

Arguments

Development Tool	Argument	Description
Microsoft Access, Visual Basic, and Visual C++	*msgSize*	Size of the message requested.
Visual FoxPro	*nMessageSize*	Size of the message requested.

Data Type

Long.

Reset Event

{ewc HLP95EN.DLL,DYNALINK,"See Also":"vbevtResetEventC"} {ewc HLP95EN.DLL,DYNALINK,"Example":"vbevtResetEventX":1} {ewc HLP95EN.DLL,DYNALINK,"Applies To":"vbevtResetEventA"} {ewc HLP95EN.DLL,DYNALINK,"Specifics":"vbevtResetEventS"}

This event occurs after the successful completion of a Reset request.

Syntax

Development Tool	Syntax
Microsoft Access and Visual Basic	*object*_**Reset ( )**
Visual FoxPro	PROCEDURE *Object*.Reset
Visual C++	**void** *dialogclass*::OnReset*Control*( );

The *object* placeholder represents an object expression that evaluates to an object in the Applies To list.

Arguments

None.

Delete Method

{ewc HLP95EN.DLL,DYNALINK,"See Also":"vbmthDeleteMethodC"} {ewc HLP95EN.DLL,DYNALINK,"Example":"vbmthDeleteMethodX":1} {ewc HLP95EN.DLL,DYNALINK,"Applies To":"vbmthDeleteMethodA"} {ewc HLP95EN.DLL,DYNALINK,"Specifics":"vbmthDeleteMethodS"}

Initiates a Delete request. If successful, a Delete event occurs, otherwise an Error event occurs.

Return Value

Void

Syntax

Development Tool	Syntax
Microsoft Access and Visual Basic	*object*.**Delete** *msgNumber*
Visual FoxPro	*Object.*Delete(*nMessageNumber)*
Visual C++	**void Delete**(**short** *MessageNumber*);

The *object* placeholder represents an object expression that evaluates to an object in the Applies To list.

Arguments

Development Tool	Argument	Description
Microsoft Access and Visual Basic	*msgNumber*	Number of message to be deleted.
Visual FoxPro	*nMessageNumber*	Number of message to be deleted.
Visual C++	*MessageNumber*	Number of message to be deleted.

Data Type

Integer

Last Method

{ewc HLP95EN.DLL,DYNALINK,"See Also":"vbmthLastMethodC"} {ewc HLP95EN.DLL,DYNALINK,"Example":"vbmthLastMethodX":1} {ewc HLP95EN.DLL,DYNALINK,"Applies To":"vbmthLastMethodA"} {ewc HLP95EN.DLL,DYNALINK,"Specifics":"vbmthLastMethodS"}

Initiates a LAST request. If successful, the Last event occurs, otherwise the Error event occurs.

Return Value

Void

Syntax

Development Tool	Syntax
Microsoft Access and Visual Basic	*object*.**Last**
Visual FoxPro	PROCEDURE *Object.*Last( )
Visual C++	**void Last**( );

The *object* placeholder represents an object expression that evaluates to an object in the Applies To list.

Arguments

None.

MessageSize Method

{ewc HLP95EN.DLL,DYNALINK,"See Also":"vbmthMessageSizeMethodC"} {ewc HLP95EN.DLL,DYNALINK,"Example":"vbmthMessageSizeMethodX":1} {ewc HLP95EN.DLL,DYNALINK,"Applies To":"vbmthMessageSizeMethodA"} {ewc HLP95EN.DLL,DYNALINK,"Specifics":"vbmthMessageSizeMethodS"}

Initiates a request to retrieve the message size. If successful, a MessageSize event occurs, otherwise the Error event occurs

Return Value

Void

Syntax

Development Tool	Syntax
Microsoft Access and Visual Basic	*object*.**MessageSize** *msgNumber*
Visual FoxPro	*Object.*MessageSize(*nMessageNumber*)
Visual C++	**void MessageSize**(**short** *MessageNumber*);

The *object* placeholder represents an object expression that evaluates to an object in the Applies To list.

Arguments

Development Tool	Argument	Description
Microsoft Access and Visual Basic	*msgNumber*	Message number.
Visual FoxPro	*nMessageNumber*	Message number.
Visual C++	*MessageNumber*	Message number.

Data Type

Integer

Reset Method

{ewc HLP95EN.DLL,DYNALINK,"See Also":"vbmthResetMethodC"} {ewc HLP95EN.DLL,DYNALINK,"Example":"vbmthResetMethodX":1} {ewc HLP95EN.DLL,DYNALINK,"Applies To":"vbmthResetMethodA"} {ewc HLP95EN.DLL,DYNALINK,"Specifics":"vbmthResetMethodS"}

Initiates a RSET request. Any messages marked as deleted will be unmarked. If successful, a corresponding Reset event occurs, otherwise an Error event occurs.

Return Value

Void

Syntax

Development Tool	Syntax
Microsoft Access and Visual Basic	*object*.**Reset**
Visual FoxPro	*Object.*Reset( )
Visual C++	**void Reset**( );

The *object* placeholder represents an object expression that evaluates to an object in the Applies To list.

Arguments

None.

RetrieveMessage Method

{ewc HLP95EN.DLL,DYNALINK,"See Also":"vbmthRetrieveMessageMethodC"} {ewc HLP95EN.DLL,DYNALINK,"Example":"vbmthRetrieveMessageMethodX":1} {ewc HLP95EN.DLL,DYNALINK,"Applies To":"vbmthRetrieveMessageMethodA"} {ewc HLP95EN.DLL,DYNALINK,"Specifics":"vbmthRetrieveMessageMethodS"}

Initiates a RetrieveMessage request for the message specified in msgNumber. The retrieval will be streamed into the **DocOutput** object passed.

Return Value

Void

Syntax

Development Tool	Syntax
Microsoft Access and Visual Basic	*object*.**RetrieveMessage** *msgNumber* [*DocOutput*]
Visual FoxPro	*Object*.RetrieveMessage(*nMessageNumber* [, *nDocOutput*])
Visual C++	**void RetrieveMessage**(**short** *MessageNumber*);

The *object* placeholder represents an object expression that evaluates to an object in the Applies To list.

Arguments

Development Tool	Argument	Description
Microsoft Access and Visual Basic	*msgNumber*	Specifies number of message to be retrieved.
	DocOuput	Optional. **DocOuput** object to be used for streaming.
Visual FoxPro	*nMessageNumber*	Specifies number of message to be retrieved.
	nDocOutput	Optional. **DocOutput** object to be used for streaming.
Visual C++	*MessageNumber*	Specifies number of message to be retrieved.

Remarks

DocOutput event can be used to retrieve the data.

TopMessage Method

{ewc HLP95EN.DLL,DYNALINK,"See Also":"vbmthTopMessageMethodC"} {ewc HLP95EN.DLL,DYNALINK,"Example":"vbmthTopMessageMethodX":1} {ewc HLP95EN.DLL,DYNALINK,"Applies To":"vbmthTopMessageMethodA"} {ewc HLP95EN.DLL,DYNALINK,"Specifics":"vbmthTopMessageMethodS"}

Initiates a Top of Message request for the message specified in *msgNumber*. The retrieval will be streamed into the **DocOutput** object being passed. If no **DocOutput** object is specified, the POP control uses the **DocOutput** property of the control.

Syntax

Development Tool	Syntax
Microsoft Access and Visual Basic	*object*.**TopMessage** *msgNumber* [*DocOutput*]
Visual FoxPro	*Object*.TopMessage(*nMessageNumber* [, *nDocOutput*])
Visual C++	**void TopMessage**(**short** *MessageNumber*);

The *object* placeholder represents an object expression that evaluates to an object in the Applies To list.

Arguments

Development Tool	Argument	Description
Microsoft Access and Visual Basic	*msgNumber*	Specifies number of message to be retrieved.
	DocOutput	Optional. Object to be used for data streaming.
Visual FoxPro	*nMessageNumber*	Specifies number of message to be retrieved.
	nDocOutput	Optional. DocOuput to be used for streaming.
Visual C++	*MessageNumber*	Specifies number of message to be retrieved.

Remarks

DocOutput event can be used to retrieve the data.

POP Client Control

{ewc HLP95EN.DLL,DYNALINK,"See Also":"vbobjPopClientOLEControlC"} {ewc HLP95EN.DLL,DYNALINK,"Example":"vbobjPopClientOLEControlX":1} {ewc HLP95EN.DLL,DYNALINK,"Properties":"vbobjPopClientOLEControlP"} {ewc HLP95EN.DLL,DYNALINK,"Methods":"vbobjPopClientOLEControlM"} {ewc HLP95EN.DLL,DYNALINK,"Events":"vbobjPopClientOLEControlE"} {ewc HLP95EN.DLL,DYNALINK,"Specifics":"vbobjPopClientOLEControlS"}

The **POP** Client control implements the POP3 Protocol Client as specified by RFC 1081, *Post Office Protocol*.

Remarks

The **POP** control provides access to Internet mail servers using the POP3 protocol. It can be used by Internet mail developers or system integrators. The major advantage of this control is its ability to retrieve mail from UNIX or other servers supporting the POP3 protocol.

The main features of the **POP** control allow you to

- Connect to a server.
- Send authentication information (user and password) to the server.
- Retrieve user mailbox information, such as the number of messages waiting to be retrieved.
- Retrieve messages from the server.
- Delete messages from the server.

POP3 Commands

The following table summarizes POP3 commands. All the commands are implemented in the control, although some of them are abstracted at a higher level (e.g. USER + PASS = Authorization).

Command	Usage	When Valid
DELE msg	required	TRANSACTION state
LAST	required	TRANSACTION state
LIST [msg]	required	TRANSACTION state
NOOP	required	TRANSACTION state
PASS string	required	AUTHORIZATION state
QUIT	required	AUTHORIZATION and UPDATE state
RETR msg	required	TRANSACTION state
RPOP user	optional	AUTHORIZATION state; not supported in current release
RSET	required	TRANSACTION state
STAT	required	TRANSACTION state
TOP msg n	optional	TRANSACTION state; supported if it is supported by the server.
USER name	required	AUTHORIZATION state

Each of the POP3 commands can return either:

- +OK

- -ERR

The reply given by the POP3 server to any command is significant only up to "+OK" and "-ERR". The client can ignore any text occurring after this reply. The only exception is the STAT command.

MessageCount Property

{ewc HLP95EN.DLL,DYNALINK,"See Also":"vbproMessageCountPropertyC"} {ewc HLP95EN.DLL,DYNALINK,"Example":"vbproMessageCountPropertyX":1} {ewc HLP95EN.DLL,DYNALINK,"Applies To":"vbproMessageCountPropertyA"} {ewc HLP95EN.DLL,DYNALINK,"Specifics":"vbproMessageCountPropertyS"}

Returns the number of messages in the mailbox. The property is unavailable until authentication has been sucessfully performed. Read-only at run time, and unavailable at design time.

Syntax

Development Tool	Syntax
Microsoft Access and Visual Basic	*object*.**MessageCount**
Visual FoxPro	*[Form.]Object.*MessageCount
Visual C++	**short GetMessageCount**( );

The *object* placeholder represents an object expression that evaluates to an object in the Applies To list.

Data Type

Integer

TopLines Property

{ewc HLP95EN.DLL,DYNALINK,"See Also":"vbproTopLinesPropertyC"} {ewc HLP95EN.DLL,DYNALINK,"Example":"vbproTopLinesPropertyX":1} {ewc HLP95EN.DLL,DYNALINK,"Applies To":"vbproTopLinesPropertyA"} {ewc HLP95EN.DLL,DYNALINK,"Specifics":"vbproTopLinesPropertyS"}

Returns or sets the number of lines to be retrieved in a top request.

Syntax

Development Tool	Syntax
Microsoft Access and Visual Basic	*object*.**TopLines** [= *Integer*]
Visual FoxPro	*[Form.]Object.*TopLines[= *nNumberOfLines*]
Visual C++	**long GetTopLines**( ); **void SetTopLines**(**long** *nNewValue*);

The *object* placeholder represents an object expression that evaluates to an object in the Applies To list.

Arguments

Development Tool	Argument	Description
Microsoft Access and Visual Basic	*Integer*	Specifies number of lines to be retrieved.
Visual FoxPro	*nNumberOfLines*	Specifies number of lines to be retrieved.
Visual C++	*nNewValue*	Specifies number of lines to be retrieved.

Data Type

Long.

TopSupported Property

{ewc HLP95EN.DLL,DYNALINK,"See Also":"vbproTopSupportedPropertyC"} {ewc HLP95EN.DLL,DYNALINK,"Example":"vbproTopSupportedPropertyX":1} {ewc HLP95EN.DLL,DYNALINK,"Applies To":"vbproTopSupportedPropertyA"} {ewc HLP95EN.DLL,DYNALINK,"Specifics":"vbproTopSupportedPropertyS"}

Returns a value that determines if the TOP command is supported on a server. The property is only available after a connection to the server has been established. It is set to **True** if the particular server supports the TOP command. Read-only at run time, and unavailable at design time.

Syntax

Development Tool	Syntax
Microsoft Access and Visual Basic	*object*.**TopSupported**
Visual FoxPro	*[Form.]Object.*TopSupported
Visual C++	**BOOL GetTopSupported**( );

The *object* placeholder represents an object expression that evaluates to an object in the Applies To list.

Settings

The settings for the **TopSupported** property are:

Setting	Description
True	The server supports the TOP command.
False	The server does not support the TOP command.

Data Type

Boolean.

SMTP Client Control

{ewc HLP95EN.DLL,DYNALINK,"See Also":"vbobjSMTPClientControlC"} {ewc HLP95EN.DLL,DYNALINK,"Example":"vbobjSMTPClientControlX":1} {ewc HLP95EN.DLL,DYNALINK,"Properties":"vbobjSMTPClientControlP"} {ewc HLP95EN.DLL,DYNALINK,"Methods":"vbobjSMTPClientControlM"} {ewc HLP95EN.DLL,DYNALINK,"Events":"vbobjSMTPClientControlE"} {ewc HLP95EN.DLL,DYNALINK,"Specifics":"vbobjSMTPClientControlS"}

The SMTP control implements the basic client SMTP Protocol as specified by RFC821, *Simple Mail Transfer Protocol* which is used to send Internet mail messages to SMTP servers. Refer to RFC821 for the SMTP command syntax and valid states and reply codes

Remarks

The **SMTP** Client control can be used to develop Microsoft Access, Visual Basic, and Visual FoxPro applications that communicate with SMTP servers to send mail messages. It provides a reusable component that gives applications access to SMTP mail servers and mail posting capabilities.

The **SMTP** control supports a high level interface that incorporates all SMTP commands used in sending out a mail message. Using this interface, a mail message can be sent with a single call.

About Box

Displays information about this control.

Using the SMTP and POP3 Controls

{ewc HLP95EN.DLL,DYNALINK,"See Also":"vbmscSMTPPOP3OverviewC"}

The **SMTP** and the **POP3** Client control can be used in tandem to create applications that can send and receive electronic mail from the Internet.

- The **SMTP** control is used to send messages using the Simple Mail Transport Protocol.
- The **POP** control implements the Post Office Protocol (version 3) to receive mail from the Internet.

The controls are discussed in further detail below.

The POP3 Control

The **POP** control allows users to log on to a mail server and retrieve messages stored there.

Possible Uses

The **POP** control can be used in the following scenarios:

- To receive daily email reports that can then be compiled into a weekly report.
- To create an application that logs on at regular intervals to retrieve messages.

Scenario: Retrieve a Message from a Remote Server

This sample session demonstrates how to use the **POP** ActiveX™ Control. The application retrieves a message for user `johng` whose password is `secret` on the Mail server.

To achieve this, the following steps are required:

1. Connect to a remote host using the **Connect** method.
2. Use the **Authenticate** method to activate retrieval privileges.
3. Invoke the **RetrieveMessage** method to get a particular message.
4. Use the DocOutput event to stream the message into a container.

Connect to a Remote Host Using the Connect Method.

The **Connect** method is invoked to connect with the mail server. The name of the remote machine is contained in the text box called "txtRemoteHost"

```
Private Sub cmdConnect_Click()
   Popct1.RemoteHost = txtRemoteHost
   Popct1.Connect
End Sub
```

Use the Authenticate Method to Activate Retrieval Privileges.

Before any mail can be retrieved, the user must be authenticated with a UserID and Password. To accomplish this, invoke the **Authenticate** method after the connection is established and send the information to the server. The code below assumes there are two **TextBox** controls, one called "UserID," and a second named "Password."

```
Private Sub Authenticate_Click()
   ' authenticate the user
   ' UserId  and Password properties are used
   Popct1.UserId = UserId.Text
   Popct1.Password = Password.Text
   Popct1.Authenticate
End Sub
```

Invoke the RetrieveMessage Method to Get a Particular Message.

By now the total number of messages residing on the server for this user is known and can be retrieved.

```
Private Sub cmdRetrieve_Click()
   dim strMessage as String
   strMessage = ""
   Popct1.MessageRetrieve Val(txtMsgNumber.Text)
End Sub
```

Stream the Message into a Text Box with the DocOutput Event

In response to the **MessageRetrieve** method, the DocOutput event occurs. The code below then invokes the **GetData** method of the **DocOutput** object to put the data into a variable of type variant. The data is then streamed into a **TextBox** control named "txtMessage".

```
Private Sub POPCT1_DocOutput(DocOutput DocOutput)
   Dim vtData as Variant ' Variant for data.

   ' Retrieve the data from the buffer.
   POPCT1.GetData vtData
   ' Stream the data into the txtMessage container.
   txtMessage = txtMessage & vtData & vbCrLf
End Sub
```

The SMTP Client Control

The **SMTP** control can be used to send mail from any application.

Possible Uses

The **SMTP** control can be used in the following scenarios:

- To create an application that compiles data from various sources before sending out a report.
- To send mail that includes data from an FTP server.

Scenario: Sending Mail to a Server

In this common scenario, the **SMTP** Client Control is used to send a mail message to a server. The steps to accomplish this are as follows:

1. Connect to a mail server using the **Connect** method.
2. Authenticate the user using the **Authenticate** method.
3. Create a collection of **DocHeader** objects, with each **DocHeader** object containing specific information such as the subject of the message.
4. Send the message by invoking the **SendDoc** method.

Connect and Authenticate

As with the **POP3** control, you must use the **Connect** method and the **Authenticate** method before beginning a transaction. The code for this is found in the scenario for the **POP3** control.

Create a Collection of DocHeader Objects

The most complex step is creating a collection of **DocHeader** objects. Each **DocHeader** object contains a significant piece of data that the mail server requires. The code below first creates an object variable for the **DocHeaders** collection, then creates an instance of the **DocHeaders** collection. The code then uses the **Add** method of the **DocHeaders** collection to add several **DocHeader** objects to the collection. The **Add** method requires two arguments which are set to the

Name and the **Value** property.

```
Sub CreateCollection()
   Dim docX as DocHeaders ' DocHeaders variable.
   Set docX = New DocHeaders ' Create an instance.

   ' Add DocHeader objects using the Add method.

   With docX
      .Add "From", "joe@xyz.com"   ' From:joe@xyz.com
      .Add "To", "adam@abc.com"    ' To:adam@abc.com
      .Add "CC", "sue@acme.com"    ' CC:sue@acme.com
      .Add "Subject", "New address"  ' Subject: New
                                     ' address
   End With

End Sub
```

Send the Message by Invoking the SendDoc Method

The final step to sending a message is to invoke the **SendDoc** method. The method requires three arguments, the URL (where the data is sent), the **DocHeaders** collection, and the data to be sent. The code below takes a reference to the **DocHeaders** collection created in the previous code and passes it as an argument to the **SendDoc** method.

```
Sub SendMessage(docX as DocHeaders)
   Dim strMessage as String ' Message variable.

   ' Use the RemoteHost property to specify
   ' the mail server's address.
   Smtpct1.RemoteHost = "mail"

   mailText = "New address: joe@newcom.com"

   ' Invoke the SendDoc method.
   SMTP1.SendDoc , docX, strMessage

End Sub
```