NMHttp unit

The NMHTTP unit contains the TNMHTTP component, and it's related objects and types.

Components
TNMHTTP

Objects
THeaderInfo

Types
CmdType
TResultEvent

THeaderInfo object

See also Properties

Unit
NMHttp

Description
The **THeaderInfo** object provides an easy to use interface for adding the most common HTTP header items. These items are optional, but are useful for a variety of purposes.

See also

<<< See also of THeaderInfo object >>>

Properties

- Run-time only
- Key properties
  - Cookie
  - LocalMailAddress
  - LocalProgram
  - Password
  - Referer
  - UserId

Cookie property

Applies to
THeaderInfo object

Declaration
```
property Cookie: string;
```

Description
The **Cookie** property contains the outgoing cookie that will be sent to the remote host when making an HTTP request. This is useful for returning a cookie previously sent by the same host.

LocalMailAddress property

Applies to
THeaderInfo object

Declaration
```
property LocalMailAddress: string;
```

Description
The **LocalMailAddress** property specifies the E-Mail address of the local user when making an HTTP request.

LocalProgram property

Applies to
THeaderInfo object

Declaration
```
property LocalProgram: string;
```

Description
The **LocalProgram** property specifies the name of the local application sending the HTTP request.

Password property

Applies to
THeaderInfo object

Declaration
```
property Password: string;
```

Description
The **Password** property specifies a password to use if Basic Authentication is required to access a document on the world wide web. The password provided must match with the User ID presented in the **UserId** property.

Referer property

Applies to
THeaderInfo object

Declaration
```
property Referer: string;
```

Description
The **Referer** property specifies the address of the site that referred the client to the requested document. An application of this property would be from someone clicking a link on a page, that page would be the referer to the page requested by the link.

UserId property

Applies to
THeaderInfo object

Declaration
```
property UserId: string;
```

Description
The **UserId** property specifies a User ID to use if Basic Authentication is required to access a document on the world wide web. The User ID provided must match with the password presented in the **Password** property.

TNMHTTP component

Heirarchy Properties Methods Events Tasks

Unit

NMHttp

Description

The TNMHTTP Component is used for conducting HTTP transfers across the World Wide Web. The TNMHTTP component is HTTP 1.1 compliant.

TNMHTTP Properties
TNMHTTP
Legend

In TNMHTTP
Body
CookieIn
Header
HeaderInfo
InputFileMode
OutputFileMode
SendHeader

Derived from TPowersock
About
BeenCanceled
BeenTimedOut
BytesRecvd
BytesSent
BytesTotal
Connected
Handle
Host
LastErrorNo
LocalIP
Port
Proxy
ProxyPort
RemoteIP
ReplyNumber
ReportLevel
Status
TimeOut
TransactionReply
WSAInfo

Derived from TComponent
ComObject
ComponentCount

- ComponentIndex
- Components
- ComponentState
- ComponentStyle
- DesignInfo
- Owner
- Tag

VCLComObject

TNMHTTP Methods
TNMHTTP
Legend

In TNMHTTP

Delete
Get
Head
Options
Post
Put
Trace

Derived from TPowersock

Abort
Accept
Cancel
CaptureFile
CaptureStream
CaptureString
CertifyConnect
Connect
Create
Destroy
Disconnect
FilterHeader
GetLocalAddress
GetPortstring
Listen
read
ReadLn
RequestCloseSocket
SendBuffer
SendFile
SendStream
Transaction
write
writeln

Derived from TComponent

DestroyComponents
Destroying
FindComponent
FreeNotification
FreeOnRelease
GetParentComponent
HasParent
InsertComponent
RemoveComponent
SafeCallException

Derived from TPersistent

Assign
GetNamePath

Derived from TObject

ClassInfo
ClassName
ClassNameIs
ClassParent
ClassType
CleanupInstance
DefaultHandler
Dispatch
FieldAddress
Free
FreeInstance
GetInterface
GetInterfaceEntry
GetInterfaceTable
InheritsFrom
InitInstance
InstanceSize
MethodAddress
MethodName
NewInstance

TNMHTTP Events
TNMHTTP
Legend

In TNMHTTP

- OnAboutToSend
- OnAuthenticationNeeded
- OnFailure
- OnRedirect
- OnSuccess

Derived from TPowersock

- OnAccept
- OnConnect
  - OnConnectionFailed
- OnConnectionRequired
- OnDisconnect
- OnError
- OnHostResolved
- OnInvalidHost
- OnPacketRecvd
- OnPacketSent
- OnRead
- OnStatus

About the TNMHTTP component

TNMHTTP reference

Purpose

The TNMHTTP component is used for conducting HTTP transfers across the internet or an intranet.

Tasks

Because of the nature of URLs, the **Host** and **Port** properties do not need to be set under normal circumstances, since the URL contains the host and port.

Using the **Get** method, documents can be retrieved from a World Wide Web server.

Using the **Post** method, data can be posted to a document on a World Wide Web server.

Using the **Put** method, a document can be created on a World Wide Web server.

Using the **Delete** method, a document can be removed from a World Wide Web server.

Using the **Trace** method, you can view the data and header that was sent to the remote host.

The **Options** method will retrieve the commands supported by the server and document specified.

Body property

See also Example

Applies to
TNMHTTP component

Declaration
```
property Body: string;
```

Description
The **Body** property is used in one of two ways:

If the **InputFileMode** property is TRUE:
If the InputFileMode property is TRUE, the **Body** property must be set to a file path and name to store the retrieved document in. After a call to one of the methods that retrieves documents from the internet using HTTP, the file specified by this property will contain the body of the retrieved document. This method is useful when retrieving binary documents, such as .WAV files, images, etc.

If the **InputFileMode** property is FALSE:
If the InputFileMode property is FALSE, the **Body** property will contain the body of a document retrieved using one of the methods that retrieves documents using HTTP. This method is useful when retrieving HTML or other text documents from the web for parsing.

Scope: Published
Accessability: Runtime, designtime

See also

Get method
Header property
InputFileMode property
OnFailure event
OnRedirect event
OnSuccess event
Trace method

Example

To recreate this example, you will need to create a new blank Delphi application.

Place 4 TMemos, a TEdit, a TOpenDialog, a TNMHTTP, and 7 TButtons on the form.

Component Descriptions

Memo1: Header Display
Memo2: Body Display
Memo3: Status Display
Memo4: Cookie Display
Edit1: URL input
Button1: HTTP Get
Button2: HTTP Head
Button3: HTTP Options
Button4: HTTP Trace
Button5: HTTP Put
Button6: HTTP Post
Button7: HTTP Delete

Insert the following code into Button1's OnClick event:

```
procedure TForm1.Button1Click(Sender: TObject);
begin
  NMHTTP1.Get(Edit1.Text);
end;
```

When Button1 is clicked, the **Get** method is used to retrieved the document at the address specified by Edit1. Both the document body and the document header are retrieved.

Insert the following code into Button2's OnClick event:

```
procedure TForm1.Button2Click(Sender: TObject);
begin
  NMHTTP1.Head(Edit1.Text);
end;
```

When Button2 is clicked, the **Head** method is used to retrieve the header of the document at the address specified by Edit1.

Insert the following code into Button3's OnClick event:

```
procedure TForm1.Button3Click(Sender: TObject);
begin
  NMHTTP1.Options(Edit1.Text);
end;
```

When Button3 is clicked, the HTTP options for the document at the address in Edit1 are retrieved using the **Options** method. **Note:** Not all servers support the **Options** method.

Insert the following code into Button4's OnClick event:

```
procedure TForm1.Button4Click(Sender: TObject);
var
  S: String;
begin
  if InputQuery('Trace Data Required', 'Input data to send as trace', S) then
    NMHTTP1.Trace(Edit1.Text, S);
end;
```

When Button4 is clicked, the **InputQuery** function is used to get data from the user to use as trace data. If the user inputs data and clicks Ok, the **Trace** method sends the data to the server as trace data.

Insert the following code into Button5's OnClick event:

```
procedure TForm1.Button5Click(Sender: TObject);
begin
  if OpenDialog1.Execute then
  begin
    NMHTTP1.OutputFileMode := TRUE;
    NMHTTP1.Put(Edit1.Text, OpenDialog1.FileName);
    NMHTTP1.OutputFileMode := FALSE;
  end;
end;
```

When Button5 is clicked, OpenDialog1 prompts for a file. If a file is selected, the **OutputFileMode** property is set to TRUE, so that the data to be put will be read from the file specified. The **Put** method is used to store the file at the address specified by Edit1. When the file is put, the OutputFileMode property is returned to FALSE.

Insert the following code into Button6's OnClick event:

```
procedure TForm1.Button6Click(Sender: TObject);
var
  S: String;
begin
  if InputQuery('Post Data Required', 'Input data to Post', S) then
    NMHTTP1.Post(Edit1.Text, S);
end;
```

When Button6 is clicked, the **InputQuery** function is used to retrieve the data to be posted. If the Ok button is clicked, the data that was input is posted using the **Post** method to the document specified by the address in Edit1.

Insert the following code into Button7's OnClick event:

```
procedure TForm1.Button7Click(Sender: TObject);
begin
  NMHTTP1.Delete(Edit1.Text);
end;
```

When Button7 is clicked, the **Delete** method attempts an HTTP Delete of the document specified by the address in Edit1.

Insert the following code into NMHTTP1's OnAuthenticationNeeded event:

```
procedure TForm1.NMHTTP1AuthenticationNeeded(Sender: TObject);
var
  AnID,
  APass: String;
begin
  InputQuery('Authentication required', 'Enter a user ID', AnID);
  InputQuery('Authentication required', 'Enter a password', APass);
  NMHTTP1.HeaderInfo.UserId := AnID;
  NMHTTP1.HeaderInfo.Password := APass;
  ShowMessage('Authentication information in place, please retry the previous command');
end;
```

If basic authentication is used to access the document specified by the address in Edit1, the **OnAuthenticationNeeded** event is called. In this example, the **InputQuery** function is used to retrieve a user ID and password. These values are then stored in the **UserId** and **Password** properties of the **HeaderInfo** property (See the **THeaderInfo** reference). A message is shown to the user asking them to attempt the HTTP transaction again once the password and user id are in place.

Insert the following code into NMHTTP1's OnFailure event:

```
procedure TForm1.NMHTTP1Failure(Cmd: CmdType);
begin
  Memo1.Text := NMHTTP1.Header;
  Memo2.Text := NMHTTP1.Body;
  case Cmd of
    CmdGET: Memo3.Lines.Add('HTTP GET Failed');
    CmdPOST: Memo3.Lines.Add('HTTP Post Failed');
    CmdHEAD: Memo3.Lines.Add('HTTP HEAD Failed');
    CmdOPTIONS: Memo3.Lines.Add('HTTP OPTIONS Failed');
    CmdTrace: Memo3.Lines.Add('HTTP TRACE Failed');
    CmdPut: Memo3.Lines.Add('HTTP PUT Failed');
    CmdDelete: Memo3.Lines.Add('HTTP Delete Failed');
  end;
end;
```

When an HTTP command fails, the **OnFailure** event is called. In this case, the header for the returned error is displayed in Memo1, and the body is displayed in Memo2. Memo3 is updated by checking which command failed using the **Cmd** parameter, and adding a specific fail message for each of the supported commands.

Insert the following code into NMHTTP1's OnRedirect event:

```
procedure TForm1.NMHTTP1Redirect(var Handled: Boolean);
begin
  If MessageDlg('This site is redirecting you to another site. Allow redirect?', mtConfirmation, [mbYes,
mbNo], 0) = mrNo then
    Handled := TRUE;
end;
```

If the document specified by the address in Edit1 redirects the client to another site for it's content, the **OnRedirect** event is called. Using the **MessageDlg** function, the user is asked to allow the redirect. If the user selects No, the **Handled** parameter is set to TRUE, which prevents the redirect (default action) from

taking place. If the user selects Yes, the default action is taken by the component, and the redirected document is loaded.

Insert the following code into NMHTTP1's OnSuccess event:

```
procedure TForm1.NMHTTP1Success(Cmd: CmdType);
begin
  if NMHTTP1.CookieIn <> '' then
    Memo4.Text := NMHTTP1.CookieIn;
  Case Cmd of
    CmdGET:
      begin
        Memo1.Text := NMHTTP1.Header;
        Memo2.Text := NMHTTP1.Body;
        Memo3.Lines.Add('HTTP GET Successful');
      end;
    CmdPOST:
      begin
        Memo1.Text := NMHTTP1.Header;
        Memo2.Text := NMHTTP1.Body;
        Memo3.Lines.Add('HTTP POST Successful');
      end;
    CmdHEAD:
      begin
        Memo1.Text := NMHTTP1.Header;
        Memo2.Text := NMHTTP1.Body;
        Memo3.Lines.Add('HTTP HEAD Successful');
      end;
    CmdOPTIONS:
      begin
        Memo1.Text := NMHTTP1.Header;
        Memo2.Text := NMHTTP1.Body;
        Memo3.Lines.Add('HTTP OPTIONS Successful');
      end;
    CmdTrace:
      begin
        Memo1.Text := NMHTTP1.Header;
        Memo2.Text := NMHTTP1.Body;
        Memo3.Lines.Add('HTTP TRACE Successful');
      end;
    CmdPut:
      begin
        Memo1.Text := NMHTTP1.Header;
        Memo2.Text := NMHTTP1.Body;
        Memo3.Lines.Add('HTTP PUT Successful');
      end;
    CmdDelete:
      begin
        Memo1.Text := NMHTTP1.Header;
        Memo2.Text := NMHTTP1.Body;
        Memo3.Lines.Add('HTTP DELETE Successful');
      end;
  end;
end;
```

When an HTTP command succeeds, the **OnSuccess** event is called, signifying the success. In this example, if a cookie is returned from the remote host in the **CookieIn** property, it is displayed in Memo4. The header and body returned from the server in the **Header** and **Body** properties, respectively. The header is displayed in Memo1, and the body is displayed in Memo2. Memo3 acts as a status screen, displaying that the command specified by the **Cmd** parameter was successful.

CookieIn property

See also Example

Applies to
TNMHTTP component

Declaration
```
property CookieIn: string;
```

Description
The **CookieIn** property contains a cookie sent from the remote host if one has been sent.

Scope: Runtime, Read-Only

Note:
The **CookieIn** property is set only after a document has been retrieved from the network.

See also

Body property
Get method
Header property

Header property

See also Example

Applies to
TNMHTTP component

Declaration
```
property Header: string;
```

Description
The **Header** property is used in one of two ways:

If the **InputFileMode** property is TRUE:
If the InputFileMode property is TRUE, the **Header** property must be set to a file path and name to store the retrieved documents Header in. After a call to one of the methods that retrieves documents from the internet using HTTP, the file specified by this property will contain the header of the retrieved document.

If the **InputFileMode** property is FALSE:
If the InputFileMode property is FALSE, the **Header** property will contain the header of a document retrieved using one of the methods that retrieves documents using HTTP.

Scope: Published
Accessability: Runtime, designtime

See also

Body property
CookieIn property
Get method
Head method
InputFileMode property
Options method
OutputFileMode property
Trace method

HeaderInfo property

See also Example1 Example2

Applies to
TNMHTTP component

Declaration
```
property HeaderInfo: THeaderInfo;
```

Description
The **HeaderInfo** property specifies header information that is sent to the HTTP server when making a request for a document.

Scope: Published
Accessability: Runtime, Designtime

Note:
The header sent to the remote host when requesting a document using HTTP can be modified before it is sent to include additional fields by modifying the **SendHeader** property within the **OnAboutToSend** event.
For details on the properties within the **HeaderInfo** property, see the reference for the **THeaderInfo** Object.

See also

Get method
Head method
OnAboutToSend event
Options method
Put method
Post method
Trace method
SendHeader property

InputFileMode property

See also Example

Applies to
TNMHTTP component

Declaration
```
property InputFileMode: boolean;
```

Description
The **InputFileMode** property specifies how to handle incoming documents and their headers.

If the InputFileMode property is **TRUE**, documents and their headers are saved to files as specified by the **Header** and **Body** properties.
If the InputFileMode property is **FALSE**, documents are stored in the **Body** property itself, and their headers are stored in the **Header** property.

See also

Body property
Header property
OutputFileMode property

Example

To recreate this example, you will need to create a new blank Delphi application.

Place 7 TEdits, 3 TMemos, a TButton, and a TNMHTTP on the form.

Component Descriptions:

Edit1: URL input
Edit2: Outgoing Cookie input
Edit3: Local E-Mail Address input
Edit4: Local Program name input
Edit5: UserID input
Edit6: Password input
Edit7: Referer input
Memo1: Header display
Memo2: Body display
Memo3: Outgoing header (SendHeader) display
Button1: Perform HTTP Get

Insert the following code into Button1's OnClick event:

```
procedure TForm1.Button1Click(Sender: TObject);
begin
  with NMHTTP1.HeaderInfo do
  begin
    LocalMailAddress := Edit3.Text;
    LocalProgram := Edit4.Text;
    Cookie := Edit2.Text;
    Referer := Edit7.Text;
    UserId := Edit5.Text;
    Password := Edit6.Text;
  end;
  NMHTTP1.InputFileMode := TRUE;
  NMHTTP1.Header := '.\head.txt';
  NMHTTP1.Body := '.\Body.txt';
  NMHTTP1.Get(Edit1.Text);
end;
```

When Button1 is clicked, the **HeaderInfo** property's properties are filled with the values entered in the edit boxes on the form. The **InputFile** mode property is set to TRUE, so document retrieved from the remote host will be stored in files. The **Header** property is set to the file head.txt in the current directory. The **Body** property is set to the file Body.txt in the current directory. Finally, the document specified in the address of Edit1 is retrieved using the **Get** method.

Insert the following code into NMHTTP1's OnAboutToSend event:

```
procedure TForm1.NMHTTP1AboutToSend(Sender: TObject);
begin
  NMHTTP1.SendHeader.Values['If-Modified-Since'] := 'Wed, 09 Sep 1998 10:49:15 GMT';
  Memo3.Text := NMHTTP1.SendHeader.Text;
end;
```

Directly before a request is made to the remote host, the **OnAboutToSend** event is called prior to the

request header being sent. The **SendHeader** property represents the outgoing header. In this example, the header item If-Modified-Since is added to the outgoing header. This header item will cause the server to send the requested document only if it has been modified since the date provided. The outgoing header is then simply displayed in Memo3.

Insert the following code into NMHTTP1's OnSuccess event:

```
procedure TForm1.NMHTTP1Success(Cmd: CmdType);
begin
  if Cmd = CmdGET then
  begin
    Memo1.Lines.LoadFromFile('.\head.txt');
    Memo2.Lines.LoadFromFile('.\Body.txt');
  end;
end;
```

When an HTTP Get succeeds, the **OnSuccess** event is called. In this case, the header of the document is loaded into Memo1 from the file head.txt, and the body is loaded into Memo2 from the file Body.txt. The document is loaded from files because the **InputFileMode** property was set to TRUE prior to the call to the **Get** method.

OutputFileMode property

See also Example

Applies to
TNMHTTP component

Declaration
```
property OutputFileMode: boolean;
```

Description
The **OutputFileMode** property specifies how to handle outgoing data used in the following methods: **Post**, **Put**, and **Trace**.

If the OutputFileMode property is **TRUE**, data for these methods will be retrieved from a local file.
If the OutputFileMode property is **FALSE**, data for these methods is passed as a parameter.

See also

InputFileMode property
Post method
Put method
Trace method

SendHeader property

See also Example

Applies to

TNMHTTP component

Declaration

```
property SendHeader: TExStringList;
```

Description

The **SendHeader** property contains the header information that is sent to the remote host when requesting a document using HTTP.

Scope: Public
Accessability: Runtime only

Note:
The **SendHeader** property can be modified in the **OnAboutToSend** event to add custom header information.

See also

HeaderInfo property
OnAboutToSend event

Delete method

See also Example

Applies to

TNMHTTP component

Declaration

```
procedure Delete(URL: string); virtual;
```

Description

The **Delete** method will delete the document specified by the **URL** parameter.

Parameters:

The **URL** parameter specifies the document to delete.
The **URL** parameter must be a fully qualified URL, including the http:// portion, in order for this method to function correctly.

Scope: Public

Notes:

You must have access to modify (write) files at the location you wish to delete the file from.
If the method succeeds, the **OnSuccess** event will be called, passing CmdDELETE as the **Cmd** parameter.
If the method fails, the **OnFailure** event will be called, passing CmdDELETE as the **Cmd** parameter.
The **Body** property will contain either an error statement if the document was not deleted, or a confirmation if the document was deleted.

See also

Header property
HeaderInfo property
OnAboutToSend event
OnFailure event
OnSuccess event
Put method
SendHeader property

Get method

See also Example

Applies to
TNMHTTP component

Declaration
```
procedure Get(URL: string); virtual;
```

Description
The **Get** method retrieves the document specified by the **URL** parameter.

If **InputFileMode** is TRUE, the document retrieved by the Get method is stored in the file that is specified by the **Body** property, and the header is stored in the file that is specified by the **Header** property.

If **InputFileMode** is FALSE, the document retrieved by the Get method is stored in the **Body** property as a string. The header is stored in the **Header** property as a string.

Parameters:
The **URL** parameter specifies the document to retrieve.

Note:
If the operation is successful, the **OnSuccess** event is called, passing CmdGET as the **Cmd** parameter.
If the operation fails, the **OnFailure** event is called, passing CmdGET as the **Cmd** parameter.

See also

Body property
Head method
Header property
HeaderInfo property
OnAboutToSend event
OnFailure event
OnSuccess event
Post method
SendHeader property

Head method

See also Example

Applies to
TNMHTTP component

Declaration
```
procedure Head(URL: string); virtual;
```

Description
The **Head** method retrieved the header for the document specified by the **URL** parameter.

If **InputFileMode** is TRUE, the header retrieved is stored in the file specified by the **Header** property.
If **InputFileMode** is FALSE, the header is stored in the **Header** property as a string.

Parameters:
The **URL** parameter specifies the document to retrieve the header for.

Note:
If the operation succeeds, the **OnSuccess** event is called, passing CmdHEAD as the **Cmd** parameter.
If the operation fails, the **OnFailure** event is called, passing CmdHEAD as the **Cmd** parameter.

See also

Get method
Header property
HeaderInfo property
OnAboutToSend event
OnFailure event
OnSuccess event
SendHeader property

Options method

See also Example

Applies to
TNMHTTP component

Declaration
```
procedure Options(URL: string); virtual;
```

Description
The **Options** method returns the commands supported by the HTTP server and the actual document being queried.

Parameters:
The **URL** parameter specifies the location to retrieve options for.

Notes:
The supported commands are specified in the **Header** property. The **Public** header item specifies the commands supported by the HTTP server. The **Allow** header item specifies the commands that are supported by the document specified in the **URL** parameter.

If this command is executed successfully, the **OnSuccess** event is called passing CmdOPTIONS as the **Cmd** parameter.
If this command fails to execute successfully, the **OnFailure** event is called, passing CmdOPTIONS as the **Cmd** parameter.

See also

Body property
Head method
Header property
HeaderInfo property
OnAboutToSend event
OnFailure event
OnSuccess event
SendHeader property

Post method

See also Example

Applies to

TNMHTTP component

Declaration

```
procedure Post(URL, PostData: string); virtual;
```

Description

The **Post** method posts the information specified by the **PostData** parameter to the document specified by the **URL** parameter.

Parameters:

The **URL** parameter specifies the document to post the data to.
If the **OutputFileMode** property is TRUE, the **PostData** parameter specifies a file path and name that contains the data to be posted to the document specified by the **URL** parameter.
If the **OutputFileMode** property is FALSE, the **PostData** parameter must contain the data that is to be posted to the document specified by the **URL** parameter.

Note:

If this method succeeds, the **OnSuccess** event is called passing CmdPOST as the **Cmd** parameter.
If this method fails, the **OnFailure** event is called passing CmdPOST as the **Cmd** parameter.

See also

Body property
Head method
Header property
HeaderInfo property
OnAboutToSend event
OnFailure event
OnSuccess event
SendHeader property

Put method

See also Example

Applies to

TNMHTTP component

Declaration

```
procedure Put(URL, PutData: string); virtual;
```

Description

The **Put** method is used to create documents on an HTTP server.

Parameters:

The **URL** parameter specifies the document to create on the remote HTTP server.
If the **OutputFileMode** property is TRUE, the **PutData** parameter specifies a file to read the data out of to create the document on the remote host.
If the **OutputFileMode** property is FALSE, the **PutData** parameter contains the data that will be put into the newly created document on the remote host.

Notes:

For this method to work correctly, you must have rights to write on the host you are creating the document on.
If this method executes successfully, the **OnSuccess** event is called, passing CmdPUT as the **Cmd** parameter.
If this method fails to execute successfully, the **OnFailure** event is called, passing CmdPUT as the **Cmd** parameter.

See also

Body property
Head method
Header property
HeaderInfo property
OnAboutToSend event
OnFailure event
OnSuccess event
SendHeader property

Trace method

See also Example

Applies to

TNMHTTP component

Declaration

```
procedure Trace(URL, TraceData: string); virtual;
```

Description

The **Trace** method returns the request header sent by the client as the body of a document in addition to returning the data sent in the **TraceData** parameter. It is used mostly in debugging and troubleshooting troubles with an HTTP host.

Parameters:

The **URL** parameter specifies the document that will be "requested".
If the **OutputFileMode** property is TRUE, the **TraceData** parameter specifies a file to use as the data to send.
If the **OutputFileMode** property is FALSE, the **TraceData** parameter contains the data that will be sent to the remote host as a string.

Note:

If this method succeeds, the **OnSuccess** event is called, passing CmdTRACE as the **Cmd** parameter.
If this method fails, the **OnFailure** event is called, passing CmdTRACE as the **Cmd** parameter.

See also

Body property
Head method
Header property
HeaderInfo property
OnAboutToSend event
OnFailure event
OnSuccess event
Post method
SendHeader property

OnAboutToSend event

See also Example

Applies to
TNMHTTP component

Declaration
```
property OnAboutToSend: TNotifyEvent;
```

Description
The **OnAboutToSend** event is called before a request is sent to the remote host. This is the perfect opportunity to modify the request header, contained in the **SendHeader** property, to include any custom header fields.

Note:
This event is called by any of the methods that retrieve data from, or send data to, the remote host.

See also

HeaderInfo property
SendHeader property

OnAuthenticationNeeded event

See also Example

Applies to

TNMHTTP component

Declaration

```
property OnAuthenticationNeeded: TNotifyEvent;
```

Description

The **OnAuthenticationNeeded** event is called when a document trying to be accessed requires basic authentication to be accessed.

Note:

Basic Authentication can be accomplished by setting the **Password** and **UserID** properties of the **HeaderInfo** property.

See also

HeaderInfo property
OnFailure event
OnSuccess event

OnFailure event

See also Example

Applies to
TNMHTTP component

Declaration
```
property OnFailure: TResultEvent;
```

Description
The **OnFailure** event is called when an operation fails.

Event Parameters:
The **Cmd** parameter specifies the command that failed. The following values are possible for the Cmd parameter:

CmdGET
CmdOPTIONS
CmdHEAD
CmdPOST
CmdPUT
CmdDELETE

See also

Delete method
Get method
Head method
OnSuccess event
Options method
Post method
Put method
Trace method

OnRedirect event

See also Example

Applies to

TNMHTTP component

Declaration

```
property OnRedirect: THandlerEvent;
```

Description

The **OnRedirect** event is called when a document redirects clients to another document, either on the same host or a different host.

Event Parameters:

If the **handled** parameter is set to TRUE, the component does not automate the redirect.
If the **handled** parameter is set to FALSE (the default), the document the client is redirected to is retrieved.

See also

Body property
Header property
OnSuccess event

OnSuccess event

See also Example

Applies to
TNMHTTP component

Declaration
```
property OnSuccess: TResultEvent;
```

Description
The **OnSuccess** event is called when an operation completes successfully.

Event Parameters:
The **Cmd** parameter specifies the command that completed successfully. The following values are possible for the Cmd parameter:

CmdGET
CmdOPTIONS
CmdHEAD
CmdPOST
CmdPUT
CmdDELETE

See also

Delete method
Get method
Head method
OnFailure event
Options method
Post method
Put method
Trace method

CmdType type

Unit
NMHttp

Declaration
```
type
  CmdType = (CmdGET, CmdOPTIONS, CmdHEAD, CmdPOST, CmdPUT, CmdPATCH, CmdCOPY,
CmdMOVE, CmdDELETE, CmdLINK, CmdUNLINK, CmdTRACE, CmdWRAPPED);
```

Description
The **CmdType** type is used when it is required that the type of HTTP command that was executed be known.

TResultEvent type

Unit
NMHttp

Declaration
```
type
  TResultEvent = procedure(Cmd: CmdType) of object;
```

Description
The TResultEvent event type is used when an event needs to specify the HTTP command that the event is being executed for. See the **CmdType** type for information on the **Cmd** parameter.

Legend

- Run-time only
- Read-Only
- Published
- Protected
- Key item

Heirarchy

TObject
|
TPersistent
|
TComponent
|
TPowersock