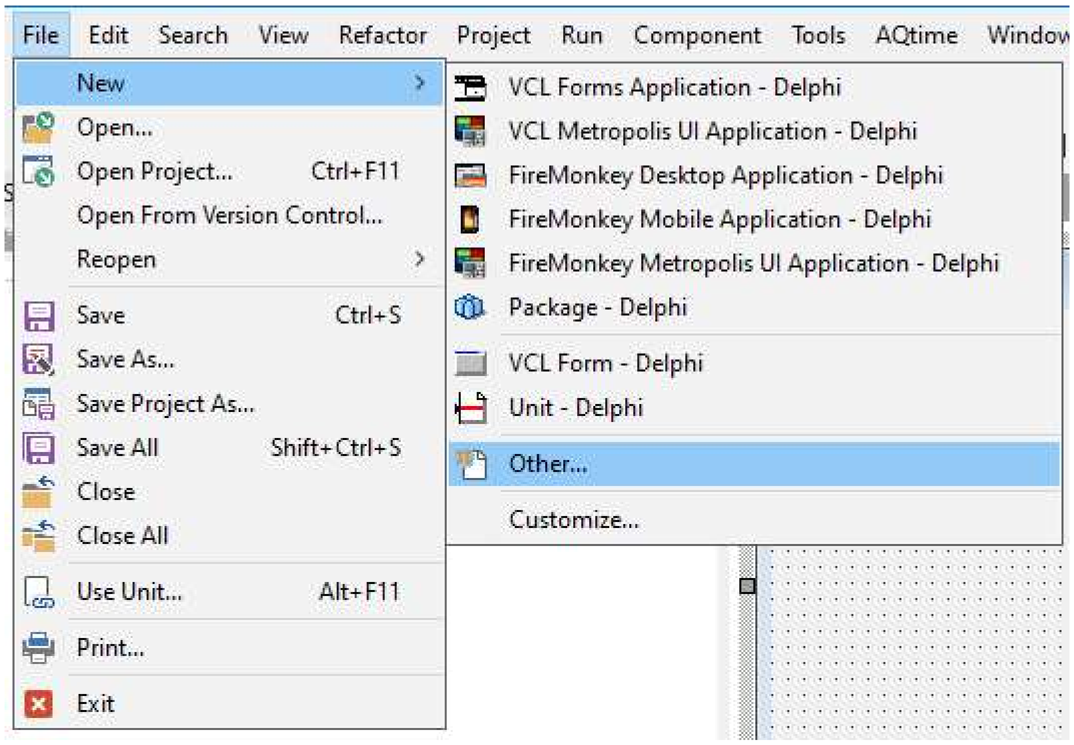
Elevate Web Builder 3 Modules Manual

Table Of Contents

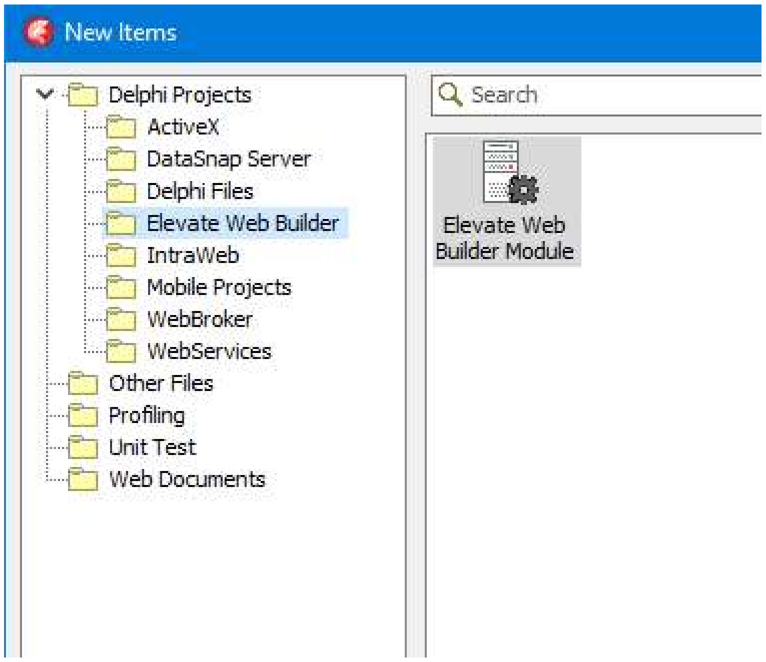
Chapter 1
Getting Started

1.1 Creating a Native Server Module

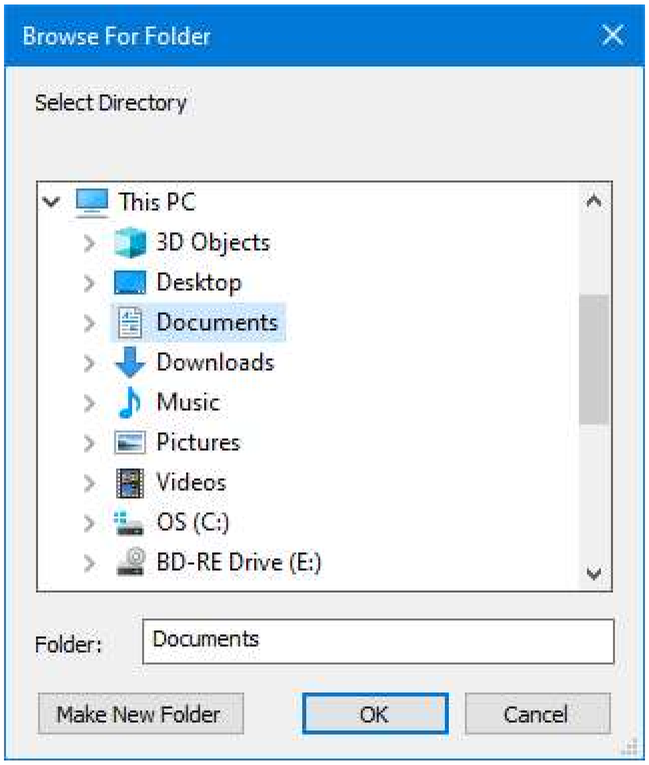
1. In the RAD Studio IDE, select the **File/New/Other** menu option from the main menu.



2. The **New Items** dialog will then appear. Select the **Elevate Web Builder** option folder from the list on the left-hand side of the dialog. Click on the **Elevate Web Builder Module** and then click on the OK button.



3. The **Browse for Folder** dialog will then appear. Select the desired target folder for the new module project and click on the OK button.



4. The new module project will be created in the desired target folder and opened as the active project in the RAD Studio IDE.

Please see the Handling Server Requests topic for more information on adding the appropriate code for handling incoming requests to the native server module.

1.2 Handling Server Requests

Note
Before reading the following material, please be sure that you understand how the Elevate Web Builder Web Server and web server requests work by reading the **Using the Web Server** section of the Elevate Web Builder manual that is included with the Elevate Web Builder product installation.

Incoming Server Requests

When an incoming request is routed to a particular native server module and the library code is executed, an execution environment is obtained. These execution environments are cached for each server module in order to minimize module startup times.

The native server module memory is initialized once when the library is loaded and finalized when the library is unloaded. If any global initialization is required for a database engine or other support code, it should be placed in the initialization section of a source unit in the Delphi native server module project. Similarly, any global teardown code should be place in the finalization section of a source unit in the Delphi native server module project.

A global TEWBModule component instance is created for every incoming request to the web server. The incoming web server TEWBWebServerRequest instance is then routed to the TEWBModule instance's OnExecute event handler, if one is defined.

Warning
Each module instance is executed in a separate thread, so you must make sure that all code included in an OnExecute event handler is completely thread-safe. Also, all native server modules are loaded into the web server process, so a fatal error such as a memory overwrite due to improper threading code in a native server module could cause the web server to fail.

Determining if a Request Is Secure

You can determine if a request is secure (using HTTPS) by examining the RequestSecure property of the incoming request.

Determining the Request Type

You can determine the request type by examining the RequestMethod property of the incoming request. HEAD requests are normally associated with static resources such as files, but they can be sent to modules when the content type of the resource being handled by the module is unknown to the client application.

Reading URL Parameters

The URL and any parameters included in the URL are specified in the request's RequestURL and RequestURLParams properties, respectively. The RequestURLParams property, however, is in its raw form. An easier way to examine the URL parameters would be to use the RequestParameters property. The RequestParameters property is a list of key-value pairs that represent all URL parameters.

Reading Request Cookies

Cookies are simple textual data that is persisted in the client across connections to the web server. Any cookies that apply to the request URL will automatically be sent by the client and will be available in the TEWBWebServerRequest RequestCookies property. The RequestCookies property is a list of key-value pairs that represent all cookies included with the request. See the **Sending a Response** section below for information on how to set cookies in responses.

Reading Request Content

Certain types of requests like HTTP POST requests are normally accompanied by content that is submitted as part of the request. The RequestContentType, RequestContentLength, RequestContent, and RequestContentStream properties of the request contain the content type, content length, and actual content.

HTML Form Submittals

When an HTML form is submitted to the URL for the module, the form data may be sent over using one of two formats. Depending upon value of the RequestContentType property, the web server will automatically perform some pre-processing of the incoming conent in the following ways:

- **application/x-www-form-urlencoded**

  This is the default encoding for form submittals, and common for form submittals that do not include file uploads. The web server will pre-parse all textual form variables and make them accessible via the RequestFormValues property as a list of key-value pairs

- **multipart/form-data**

  If a form submittal includes one or more file uploads, then the browser will use a special multi-part content encoding. The web server will still pre-parse all textual form variables and will also include any file content in the RequestContentParts property, with each file in its own TEWBHTTPContentPart instance. Each TEWBHTTPContentPart instance includes properties that can be used to determine how to deal with each content part:

Property	Description
ContentTransferEncoding	This property identifies the encoding of the file content contained in the Content and ContentStream properties. This is essential in knowing how to deal with the content.
ContentFileName	This property identifies the name of the file as provided by the client.

Sending a Response

The web server doesn't automatically send a response to an HTTP request once the OnExecute event handler terminates, so it is important that you send a response from within the OnExecute event handler. You can use one of several methods to send a response to the client's request:

Method	Description

SendContentHeader	Sends a response for a HEAD request.
SendCustomContentHeader	Sends a custom content response for a HEAD request.
SendContent	Sends a UTF-8-encoded text response with a **Content-Type** header of "text/html; charset=utf-8" along with an optional status code and message.
SendCustomContent	Sends a UTF-8-encoded text response with a custom content type, encoding, and disposition.
SendRedirect	Sends a redirect response to a new URL along with an optional UTF-8-encoded text response with a **Content-Type** header of "text/html; charset=utf-8". By default, the redirect HTTP status code is 302, but you can specify a different status code.
SendError	Sends a status code and message along with an optional UTF-8-encoded text response with a **Content-Type** header of "text/plain; charset=utf-8".
SendContentStream	Sends a UTF-8-encoded text stream response with a **Content-Type** header of "text/html; charset=utf-8" along with an optional status code and message.
SendCustomContentStream	Sends a binary stream response with a custom content type, encoding, and disposition.

Warning
If none of the above methods are called by the native server module and execution of the module terminates, the web server will not automatically send a response. This can cause the client application to hang and, eventually, time out waiting on a response that will never arrive.

Setting Cookies

As indicated above, any cookies that are stored on the client and are applicable to the request URL can be found in the RequestCookies property. You can add or modify these cookies using the TEWBWebServerRequest ResponseCookies or ResponseSessionCookies properties.

Note
You must set any cookies **before** calling any of the TEWBWebServerRequest Send* methods detailed above, or the cookies will not be set properly on the client.

1.3 Native Database Modules

Native database modules are just like any other native server module, but are coded to handle database access API requests using the TEWBDatabaseAdapter and TEWBDataSetAdapter components. Specifically, any incoming request can be passed directly to the TEWBDatabaseAdapter HandleRequest method and the database adapter component will completely handle the database request, calling the applicable TEWBDatabaseAdapter and TEWBDataSetAdapter methods and events as necessary to complete the request and send back a response.

Note
If you use a native server module for servicing database access requests, you will need to make sure to change the active TServerSession instance's DatabasesResource property so that it refers to the native server module instead of the standard database access API resource path. The active TServerSession instance is available via the TDatabase ActiveServerSession property.

Please see the Web Server Database Access topic in the **Using the Web Server** section in the Elevate Web Builder manual for more information on the database access API.

Database Module Example

The Elevate Web Builder Modules installation includes an example project that shows how to use a database module to accept row inserts from a contact form application that uses the standard TDatabase and TDataSet components for client browser applications. This project is installed into the \examples\contactform subdirectory under the main installation directory of the Elevate Web Builder Modules installation. Please see the comments in the project source for more information on how the module handles incoming database access requests.

Chapter 2
Component Reference

2.1 TEWBBlobStream Component

Unit: ewbdataset

Inherits From TStream

Use the TEWBBlobStream object to access or modify the contents of a BLOB or CLOB column in a dataset using a stream interface. A BLOB column is represented by the TBlobField object, and a CLOB column is represented by a TMemoField object. TBlobField and TMemoField objects use streams to implement many of their data access properties and methods via the standard CreateBlobStream method that is implemented by the Elevate Web Builder TEWBDataSet component.

To use a TEWBBlobStream object, create an instance of TEWBBlobStream class, use the methods of the TEWBBlobStream object to read or write the data, and then free the object. Do not use the same instance of a TEWBBlobStream object to access data from more than one row. Instead, create a new TEWBBlobStream object every time you need to read or write to a BLOB or CLOB column for a row.

Note
For proper results when updating a BLOB or CLOB column using a TEWBBlobStream object, you must create the TEWBBlobStream object after calling the Append/Insert or Edit method for the dataset containing the BLOB or CLOB column. Also, you must free the TEWBBlobStream object before calling the Post method to post the changes to the dataset. Finally, be sure to use the proper open mode when creating a TEWBBlobStream object for updating (either bmReadWrite or bmWrite).

Properties	Methods	Events
	Create	
	Read	
	Seek	
	Truncate	
	Write	

TEWBBlobStream.Create Method

```
constructor Create(Field: TBlobField; Mode: TBlobStreamMode)
```

Call the Create constructor to create an instance of the TEWBBlobStream class.

TEWBBlobStream.Read Method

```
function Read(var Buffer; Count: Integer): Integer
```

Read transfers up to Count bytes from the BLOB or CLOB column into Buffer, starting in the current position, and then advances the current position by the number of bytes actually transferred. Read returns the number of bytes actually transferred (which may be less than the number requested in Count). Buffer must have at least Count bytes allocated to hold the data that was read from the column.

All the other reading methods of a TEWBBlobStream object (ReadBuffer, ReadComponent) call Read to do their actual reading.

Note
Do not call Read when the TEWBBlobStream object was created in bmWrite mode. Also, please remember that if you are using a stream on a CLOB column, then the number of Unicode characters in the CLOB column will not be equal to the number of bytes in the stream.

TEWBBlobStream.Seek Method

```
function Seek(Offset: Integer; Origin: Word): Integer

function Seek(const Offset: Int64; Origin: TSeekOrigin): Int64
```

Use Seek to move the current position within the BLOB or CLOB column by the indicated offset. Seek allows an application to read from or write to a particular location within the BLOB or CLOB column.

The Origin parameter indicates how to interpret the Offset parameter. Origin should be one of the following values:

Origin	Description
soFromBeginning	Offset is from the beginning of the BLOB or CLOB column. Seek moves to the position Offset. Offset must be >= 0.
soFromCurrent	Offset is from the current position in the BLOB or CLOB column. Seek moves to Position + Offset.
soFromEnd	Offset is from the end of the BLOB or CLOB column. Offset must be <= 0 to indicate a number of bytes before the end of the BLOB or CLOB.

Seek returns the new value of the Position property, the new current position in the BLOB or CLOB column.

Note
Please remember that if you are using a stream on a CLOB column, then the number of characters in the CLOB column will not be equal to the number of bytes in the stream.

TEWBBlobStream.Truncate Method

```
procedure Truncate
```

Use Truncate to limit the size of the BLOB or CLOB column. Calling Truncate when the current position is 0 will clear the contents of the BLOB or CLOB column.

Note
Do not call Truncate when the TEWBBlobStream was created in bmRead mode. Please remember that if you are using a stream on a CLOB column, then the number of characters in the CLOB column will not be equal to the number of bytes in the stream.

TEWBBlobStream.Write Method

```
function Write(const Buffer; Count: Integer): Integer
```

Use Write to write Count bytes to the BLOB or CLOB column, starting at the current position.

All the other data-writing methods of a TEWBBlobStream object (WriteBuffer, WriteComponent) call Write to do their actual writing.

Note
Do not call Write when the TEWBBlobStream object was created in bmRead mode. Also, please remember that if you are using a stream on a CLOB column, then the number of characters in the CLOB column will not be equal to the number of bytes in the stream.

2.2 TEWBDatabase Component

Unit: ewbdataset

Inherits From TComponent

The TEWBDatabase component represents a database container for datasets in an application and provides properties and methods for loading/saving datasets and using transactions to modify the datasets. An instance of the TEWBDatabase component called **DefaultEWBDatabase** is automatically created at application startup as the default, global TEWBDatabase instance.

Properties	Methods	Events
ActiveServerSession	CancelPendingRequests	AfterCommit
AutoTransactions	Commit	AfterRollback
BaseURL	Create	BeforeCommit
DatabaseName	GetDataSetByName	BeforeRollback
DataSetCount	LoadColumns	OnCommitError
DataSets	LoadParameters	OnCreate
InTransaction	LoadRows	OnDestroy
NumPendingRequests	RetryPendingRequests	OnRollbackError
Params	Rollback	
ServerSession	StartTransaction	
Timeout		
TransactionLevel		

TEWBDatabase.ActiveServerSession Property

```
property ActiveServerSession: TEWBServerSession
```

Indicates the TEWBServerSession instance being used by the database for authentication. If no session has been manually assigned to the ServerSession property, then this property will reference the global **DefaultEWBSession** instance that is automatically created at application startup.

The database uses the active server session's UserName, Password, and BaseURL properties along with the Authenticate method to authenticate against the web server before executing any database API requests. The database then uses the BaseURL and DatabasesResource properties for constructing database API requests.

TEWBDatabase.AutoTransactions Property

```
property AutoTransactions: Boolean
```

Specifies whether transactions will be automatically started and committed or rolled back as rows are inserted, updated, and deleted in the datasets owned by the database. The default value is True.

Note
Automatic transactions are implicitly nested. If you insert a row in one dataset, and then edit a row in another dataset, the TransactionLevel property will be 1.

TEWBDatabase.BaseURL Property

```
property BaseURL: String
```

Indicates the base URL used for all database API requests. This URL is a combination of:

- The ActiveServerSession BaseURL property
- The ActiveServerSession DatabasesResource property
- The DatabaseName property

This property is used by datasets to build the proper URLs for loading rows and BLOB data from the web server.

TEWBDatabase.DatabaseName Property

```
property DatabaseName: String
```

Specifies the name of the database to use when interacting with the database API.

TEWBDatabase.DataSetCount Property

```
property DataSetCount: Integer
```

Indicates the number of datasets associated with this database.

TEWBDatabase.DataSets Property

```
property DataSets[Index: Integer]: TEWBDataSet
```

Accesses all datasets associated with this database by index.

TEWBDatabase.InTransaction Property

```
property InTransaction: Boolean
```

Indicates whether a transaction is active or not.

TEWBDatabase.NumPendingRequests Property

```
property NumPendingRequests: Integer
```

Indicates the number of pending database load/commit web server requests. If any errors were encountered during a dataset columns/parameters/rows load or database transaction commit operation, then the web server request used with the operation will remain as a pending request. You can use the RetryPendingRequests to retry all pending requests, or the CancelPendingRequests to cancel all pending requests.

TEWBDatabase.Params Property

```
property Params: TStrings
```

Specifies any custom database-specific parameters to use with any dataset commands for the datasets in the database. These parameters are included with any relevant database API requests and are useful for specifying additional parameters for the web server to use when executing dataset commands.

TEWBDatabase.ServerSession Property

```
property ServerSession: TEWBServerSession
```

Specifies the TEWBServerSession instance to use with the database. The default value is nil.

The ActiveServerSession property refers to the server session that is actually being used by the database for authentication and database API requests. If the ServerSession property is nil, then the ActiveServerSession property will reference the global **DefaultEWBSession** instance that is automatically created at application startup.

TEWBDatabase.Timeout Property

```
property Timeout: Integer
```

Specifies how long any database request should wait, in seconds, for a successful connection to the server before returning an error. The default value is 0, which means to wait a browser-defined number of seconds.

TEWBDatabase.TransactionLevel Property

```
property TransactionLevel: Integer
```

Indicates the current transaction level. A value of -1 means that no transaction is active, while a value of 0 or higher indicates that a transaction is active up to N levels of nesting.

TEWBDatabase.CancelPendingRequests Method

```
procedure CancelPendingRequests
```

Use this method to cancel any pending dataset columns/parameters/rows load or transaction commit requests. You can determine if there are any pending requests by examining the NumPendingRequests property.

TEWBDatabase.Commit Method

```
procedure Commit
```

Use this method to commit the active transaction.

TEWBDatabase.Create Method

```
constructor Create(AOwner: TComponent)
```

Use this method to create a new instance of the TEWBDatabase class.

TEWBDatabase.GetDataSetByName Method

```
function GetDataSetByName(const Name: String): TEWBDataSet
```

Use this method to retrieve a dataset associated with this database by its name.

TEWBDatabase.LoadColumns Method

```
procedure LoadColumns(DataSet: TEWBDataSet)
```

Use this method to load the columns for the specified dataset from the web server. This method will replace all existing columns in the specified dataset with the current columns for the web server dataset.

TEWBDatabase.LoadParameters Method

```
procedure LoadParameters(DataSet: TEWBDataSet)
```

Use this method to load the parameters for the specified dataset from the web server. This method will replace all existing parameters in the specified dataset with the current parameters for the web server dataset.

TEWBDatabase.LoadRows Method

```
procedure LoadRows(DataSet: TEWBDataSet; Append: Boolean=False)
```

Use this method to load the rows for the specified dataset from the web server. Specify True for the Append parameter to have the rows appended to the dataset. The default behavior is to replace all rows in the dataset with the rows returned by the web server.

TEWBDatabase.RetryPendingRequests Method

```
procedure RetryPendingRequests
```

Use this method to retry any pending dataset columns/parameters/rows load or transaction commit requests. You can determine if there are any pending requests by examining the NumPendingRequests property.

TEWBDatabase.Rollback Method

```
procedure Rollback
```

Use this method to roll back the active transaction.

TEWBDatabase.StartTransaction Method

```
procedure StartTransaction
```

Use this method to start a new transaction.

TEWBDatabase.AfterCommit Event

```
property AfterCommit: TNotifyEvent
```

This event is triggered after the Commit method completes.

TEWBDatabase.AfterRollback Event

```
property AfterRollback: TNotifyEvent
```

This event is triggered after the Rollback method completes.

TEWBDatabase.BeforeCommit Event

```
property BeforeCommit: TEWBDatabaseEvent
```

This event is triggered before the Commit method starts. Return False from the event handler to prevent the commit from occurring.

TEWBDatabase.BeforeRollback Event

```
property BeforeRollback: TEWBDatabaseEvent
```

This event is triggered before the Rollback method starts. Return False from the event handler to prevent the rollback from occurring.

TEWBDatabase.OnCommitError Event

```
property OnCommitError: TEWBDatabaseErrorEvent
```

This event is triggered whenever an error occurs during the execution of the Commit method.

Note
If an event handler is not assigned to this event, then the error will be raised as an exception.

TEWBDatabase.OnCreate Event

```
property OnCreate: TNotifyEvent
```

This event is triggered after the database is created and initialized.

TEWBDatabase.OnDestroy Event

```
property OnDestroy: TNotifyEvent
```

This event is triggered before the database is destroyed. Use this event to dispose of any instances or resources that may have been allocated in the OnCreate event handler.

TEWBDatabase.OnRollbackError Event

```
property OnRollbackError: TEWBDatabaseErrorEvent
```

This event is triggered whenever an error occurs during the execution of the Rollback method.

Note
If an event handler is not assigned to this event, then the error will be raised as an exception.

2.3 TEWBDatabaseAdapter Component

Unit: ewbdatasetadapter

Inherits From TComponent

The TEWBDatabaseAdapter component is used in conjunction with various TEWBDataSetAdapter components to implement a custom database access API in a native server module. A custom database access API will often deal with data from a data source that isn't supported by the web server's built-in database access or needs to perform special processing for database requests.

The TEWBDatabaseAdapter component implements the following functionality:

- Retrieving the list of datasets for the custom database access API.
- Retrieving TEWBDataSetAadapter instances in order to work with one or more datasets.
- Starting, committing, and rolling back transactions.

You can use the HandleRequest method to automatically handle the incoming TEWBWebServerRequest web server request instance for the native server module. The request will be handled according to the database access API for the web server. Please see the **Web Server Database Access API** topic under the **Using the Web Server** section in the Elevate Web Builder manual for more information on how the API requests should be structured.

Properties	Methods	Events
InTransaction	CommitTransaction	OnCommit
	Create	OnGetDataSetAdapter
	GetDataSetAdapter	OnGetDataSets
	GetDataSets	OnRollback
	HandleRequest	OnStartTransaction

TEWBDatabaseAdapter.InTransaction Property

```
property InTransaction: Boolean
```

Specifies whether the database adapter is currently processing a transaction commit request. This is useful when you wish to use different datasets, or dataset adapters, with transactions than with requests for column definitions, rows, or BLOB columns. You can examine this property in an OnGetDataSetAdapter event handler in order to perform this type of conditional processing.

TEWBDatabaseAdapter.CommitTransaction Method

```
function CommitTransaction(const DatabaseName: String; const
      Operations: String): String
```

Use this method to commit a transaction.

This method will automatically call the GetDataSetAdapter method as necessary to get access to the dataset adapters required to perform the insert, update, and delete dataset command executions for the transaction. If a dataset adapter cannot be accessed for a particular dataset, an exception will be raised and the transaction commit will fail.

Please see the **Web Server Database Access API** topic under the **Using the Web Server** section in the Elevate Web Builder manual for more information on the structure of the JSON used for transaction commits.

Note
This method will be automatically called when the TEWBDatabaseAdapter HandleRequest method is called for automatic database access API request handling, so you normally will not need to call this method directly.

TEWBDatabaseAdapter.Create Method

```
constructor Create(AOwner: TComponent)
```

Call the constructor to create an instance of the TEWBDatabaseAdapter component.

TEWBDatabaseAdapter.GetDataSetAdapter Method

```
function GetDataSetAdapter(const DataSetName: String):
      TEWBDataSetAdapter
```

Use this method to trigger the OnGetDataSetAdapter event and retrieve the dataset adapter instance for the specified dataset name.

Note
This method will be automatically called when the TEWBDatabaseAdapter HandleRequest method is called for automatic database access API request handling, so you normally will not need to call this method directly.

TEWBDatabaseAdapter.GetDataSets Method

```
procedure GetDataSets(DataSets: TStrings)
```

Use this method to trigger the OnGetDataSets event and retrieve a list of the available datasets.

Note
This method will be automatically called when the TEWBDatabaseAdapter HandleRequest method is called for automatic database access API request handling, so you normally will not need to call this method directly.

TEWBDatabaseAdapter.HandleRequest Method

```
procedure HandleRequest(Request: TEWBWebServerRequest; const
      DatabaseName: String='')
```

Use this method to automatically handle the incoming TEWBWebServerRequest web server request instance for the native server module.

This method will automatically call the GetDataSets, GetDataSetAdapter, and CommitTransaction methods as necessary.

The request will be handled according to the database access API for the web server. Please see the **Web Server Database Access API** topic under the **Using the Web Server** section in the Elevate Web Builder manual for more information on how the API requests should be structured.

TEWBDatabaseAdapter.OnCommit Event

```
property OnCommit: TNotifyEvent
```

This event is triggered when the database adapter needs to commit a transaction. The developer can define an event handler for this event in order to handle committing a transaction using a database-specific component.

TEWBDatabaseAdapter.OnGetDataSetAdapter Event

```
property OnGetDataSetAdapter: TEWBGetDataSetAdapterEvent
```

This event is triggered when the GetDataSetAdapter method is called.

Warning
If the Adapter variable parameter is not assigned a value, an exception will occur when database access API operations try to use the dataset adapter.

TEWBDatabaseAdapter.OnGetDataSets Event

```
property OnGetDataSets: TEWBGetDataSetsEvent
```

This event is triggered when a datasets list request is handled by the database adapter. Usage of this event is completely optional, but the event is useful for allowing client applications to enumerate the available datasets.

TEWBDatabaseAdapter.OnRollback Event

```
property OnRollback: TNotifyEvent
```

This event is triggered when the database adapter needs to roll back a transaction. The developer can define an event handler for this event in order to handle rolling back a transaction using a database-specific component.

TEWBDatabaseAdapter.OnStartTransaction Event

```
property OnStartTransaction: TNotifyEvent
```

This event is triggered when the database adapter needs to start a transaction. The developer can define an event handler for this event in order to handle starting a transaction using a database-specific component.

2.4 TEWBDataSet Component

Unit: ewbdataset

Inherits From TDataSet

The TEWBDataSet component is a TDataSet-descendant component that represents a dataset and includes functionality for opening, loading, navigating, searching, sorting, updating, and closing datasets. By default, every dataset that is created is automatically added to the DataSets property for the default TEWBDatabase instance that can be referenced using the global DefaultEWBDatabase variable. You can associate a dataset with a different database by assigning the new database instance reference to the Database property.

Properties	Methods	Events
ActiveDatabase	ClearSort	AfterLoad
BaseRowsURL	GetColumns	AfterLoadColumns
BookmarkSize	GetRows	AfterLoadParameters
CopyOnAppend	LoadColumns	BeforeLoad
Database	LoadParameters	BeforeLoadColumns
DataSetName	LoadRows	BeforeLoadParameters
DescSortedFields	Sort	OnLoadColumnsError
IncludeInTransaction		OnLoadError
LocalizeDateTimeColumns		OnLoadParametersError
Params		
ReadOnly		
SortedFields		
SortOptions		

TEWBDataSet.ActiveDatabase Property

```
property ActiveDatabase: TEWBDatabase
```

Indicates the TEWBDatabase instance that the dataset is associated with. By default, all TEWBDataSet components are associated with the default TEWBDatabase instance that can be referenced using the global **DefaultEWBDatabase** variable.

TEWBDataSet.BaseRowsURL Property

```
property BaseRowsURL: String
```

Indicates the base URL that will be used for retrieving the rows for the dataset. This URL is a combination of:

- The Database BaseURL property
- The DataSetName property
- The hard-coded "data" resource

This property is used by bound controls to build the proper URLs for loading BLOB data from the web server.

TEWBDataSet.BookmarkSize Property

```
property BookmarkSize: Integer
```

TEWBDataSet.CopyOnAppend Property

```
property CopyOnAppend: Boolean
```

Specifies whether the current or last row's contents should be copied automatically to any newly inserted or appended rows.

Note
Using the Append method will cause the last row to be copied, not the current row. If you wish to copy the current row's contents then you should use the Insert method.

TEWBDataSet.Database Property

```
property Database: TEWBDatabase
```

Specifies the TEWBDatabase instance that the dataset is associated with. By default, every dataset that is created is automatically associated with the default TEWBDatabase instance that can be referenced using the global DefaultEWBDatabase variable.

TEWBDataSet.DataSetName Property

```
property DataSetName: String
```

Specifies the name of the dataset to use when interacting with the database API.

TEWBDataSet.DescSortedFields Property

```
property DescSortedFields: String
```

Indicates which fields that are present in the SortedFields property are sorted in a descending, instead of ascending, direction. Multiple fields are delimited with a semicolon (;).

A dataset can be sorted by calling the Sort method with the appropriate parameters.

Note
If the SortedFields property is blank, then there is no active sort on the dataset.

TEWBDataSet.IncludeInTransaction Property

```
property IncludeInTransaction: Boolean
```

Specifies whether the dataset should be included in any transactions started using the TEWBDatabase StartTransaction method, or indirectly by setting the TEWBDatabase AutoTransactions property to True.

TEWBDataSet.LocalizeDateTimeColumns Property

```
property LocalizeDateTimeColumns: Boolean
```

Specifies whether the dataset will localize date-time (TDateTime) column values when assigning them to/from the dataset. The default value is False.

TEWBDataSet.Params Property

```
property Params: TStrings
```

Specifies any custom dataset-specific parameters to use with any dataset commands for this dataset. These parameters are included with any relevant database API requests and are useful for specifying additional parameters for the web server to use when executing dataset commands.

TEWBDataSet.ReadOnly Property

```
property ReadOnly: Boolean
```

Specifies whether the dataset should be editable or not.

TEWBDataSet.SortedFields Property

```
property SortedFields: String
```

Indicates which fields, if any, are part of the active sort. Multiple fields are delimited with a semicolon (;). If this property is blank, then there is no active sort on the dataset. Any fields that are sorted in a descending, instead of ascending, order are indicated in the DescSortedFields property.

A dataset can be sorted by calling the Sort method with the appropriate parameters.

TEWBDataSet.SortOptions Property

```
property SortOptions: TEWBSortOptions
```

Indicates the sort options for the active sort.

A dataset can be sorted by calling the Sort method with the appropriate parameters.

Note
If the SortedFields property is blank, then there is no active sort on the dataset.

TEWBDataSet.ClearSort Method

```
procedure ClearSort
```

Use this method to clear any active sort and revert the dataset back to the implicit row ordering, which is the order in which the rows were added to the dataset.

TEWBDataSet.GetColumns Method

```
function GetColumns: String
```

Use this method to retrieve the defined Fields for the dataset as a JSON string.

Please see the Web Server Database Access API in Elevate Web Builder manual for more information on the format of the JSON string returned from this method.

TEWBDataSet.GetRows Method

```
function GetRows: String
```

Use this method to retrieve the rows in the dataset as a JSON string.

Please see the Web Server Database Access API in Elevate Web Builder manual for more information on the format of the JSON string returned from this method.

TEWBDataSet.LoadColumns Method

```
procedure LoadColumns(const ColumnData: String)
```

Use this method to load the columns for a dataset from a JSON string parameter.

Please see the Web Server Administration API - Databases topic for more information on how the JSON string should be formatted.

Note
The dataset must be closed or an exception will be raised when this method is called.

TEWBDataSet.LoadParameters Method

```
procedure LoadParameters(const ParameterData: String)
```

Use this method to load the Params property from a JSON string parameter.

Please see the Web Server Administration API - Databases topic for more information on how the JSON string should be formatted.

TEWBDataSet.LoadRows Method

```
procedure LoadRows(const RowData: String; Append: Boolean=False)
```

Use this method to load the rows for a dataset from a JSON string parameter. The Append parameter determines if the rows should be appended to the existing dataset rows, or if the dataset should be emptied before the rows are loaded.

Please see the Web Server Administration API - Databases topic for more information on how the JSON string should be formatted.

Note
The dataset must be open or an exception will be raised when this method is called.

TEWBDataSet.Sort Method

```
procedure Sort(const SortFields: String; const DescSortFields:
      String; Options: TEWBSortOptions)
```

Use this method to sort the dataset.

The SortFields parameter is a semicolon-delimited (;) list of all fields that should be involved in the sort. By default, all sorted fields are sorted in ascending order.

The DescSortFields parameter is a semicolon-delimited (;) list of all fields in the SortFields parameter that should be sorted in descending order instead of ascending order.

The Options parameter specifies any sort options for the sort. Currently, there is only one option that allows you to specify that any string fields involved in the sort should be sorted in a case-insensitive manner.

Note
The Sort method only needs to be called once. After that point, any row operations will automatically maintain the active sort.

You can use the ClearSort method to remove the active sort.

TEWBDataSet.AfterLoad Event

```
property AfterLoad: TNotifyEvent
```

This event is triggered after the LoadRows method completes.

TEWBDataSet.AfterLoadColumns Event

```
property AfterLoadColumns: TNotifyEvent
```

This event is triggered after the LoadColumns method completes.

TEWBDataSet.AfterLoadParameters Event

```
property AfterLoadParameters: TNotifyEvent
```

This event is triggered after the LoadParameters method completes.

TEWBDataSet.BeforeLoad Event

```
property BeforeLoad: TEWBDataSetEvent
```

This event is triggered before the LoadRows method starts. Return False from the event handler to prevent the rows load operation from occurring.

TEWBDataSet.BeforeLoadColumns Event

```
property BeforeLoadColumns: TEWBDataSetEvent
```

This event is triggered before the LoadColumns method starts. Return False from the event handler to prevent the columns load operation from occurring.

TEWBDataSet.BeforeLoadParameters Event

```
property BeforeLoadParameters: TEWBDataSetEvent
```

This event is triggered before the LoadParameters method starts. Return False from the event handler to prevent the parameters load operation from occurring.

TEWBDataSet.OnLoadColumnsError Event

```
property OnLoadColumnsError: TEWBDataSetErrorEvent
```

This event is triggered whenever an error occurs during the execution of the LoadColumns method.

Note
If an event handler is not assigned to this event, then the error will be raised as an exception.

TEWBDataSet.OnLoadError Event

```
property OnLoadError: TEWBDataSetErrorEvent
```

This event is triggered whenever an error occurs during the execution of the LoadRows method.

Note
If an event handler is not assigned to this event, then the error will be raised as an exception.

TEWBDataSet.OnLoadParametersError Event

```
property OnLoadParametersError: TEWBDataSetErrorEvent
```

This event is triggered whenever an error occurs during the execution of the LoadParameters method.

Note
If an event handler is not assigned to this event, then the error will be raised as an exception.

2.5 TEWBDataSetAdapter Component

Unit: ewbdatasetadapter

Inherits From TComponent

The TEWBDataSetAdapter component is used in conjunction with the TEWBDatabaseAdapter component to implement a custom database access API in a native server module. A custom database access API will often deal with data from a data source that isn't supported by the web server's built-in database access or needs to perform special processing for database requests.

The TEWBDataSetAdapter component implements the following functionality:

- Preparing and executing dataset commands for select operations for retrieving dataset rows.
- Preparing and executing dataset commands for column select operations for retrieving dataset BLOB columns.
- Preparing and executing dataset commands for insert, update, and delete operations during transaction commits.

Properties	Methods	Events
LocalizeDateTimeColumns	Create	OnExecuteDelete
	GetColumnsJSON	OnExecuteInsert
	GetRowColumn	OnExecuteSelect
	GetRowsJSON	OnExecuteSelectColumn
		OnExecuteUpdate
		OnGetDeleteParams
		OnGetInsertParams
		OnGetSelectColumnParams
		OnGetSelectParams
		OnGetUpdateParams

TEWBDataSetAdapter.LocalizeDateTimeColumns Property

```
property LocalizeDateTimeColumns: Boolean
```

Specifies whether the adapter will localize date/time column values in the dataset. The default value is False.

TEWBDataSetAdapter.Create Method

```
constructor Create(AOwner: TComponent)
```

Use this method to create an instance of the TEWBDataSetAdapter component.

TEWBDataSetAdapter.GetColumnsJSON Method

```
function GetColumnsJSON(Params: TParams; const DatabaseName:
      String=''): String

function GetColumnsJSON(Request: TEWBWebServerRequest; const
      DatabaseName: String=''): String
```

Use this method to retrieve the column definitions for the dataset as a JSON string.

When this method is called, the dataset adapter will:

- Trigger the OnGetSelectParams event handler to get a list of parameters for the **Select** dataset command.

  Note
  This is actually not necessary for retrieving column definitions, but is called as part of the **Select** dataset command execution

- Trigger the OnExecuteSelect event handler to execute the **Select** dataset command. The TDataSet descendant class instance that is used for building the column definitions is returned in this event handler.

There are two versions of this method: one for Elevate Web Builder server module applications and one for non-Elevate Web Builder server module applications such as ISAPI modules.

- For Elevate Web Builder server module applications, the Request parameter indicates the incoming web server request and is used to retrieve any incoming parameters that should be used when executing the **Select** dataset command, and the DatabaseName parameter is a pass-through parameter that is passed to any event handlers in order to indicate that the operation is occurring for a specific database.

- For non-Elevate Web Builder server module applications, the Params parameter indicates the incoming parameters that should be used when executing the **Select** dataset command, and the DatabaseName parameter is a pass-through parameter that is passed to any event handlers in order to indicate that the operation is occurring for a specific database.

Please see the **Web Server Database Access API** topic under the **Using the Web Server** section in the Elevate Web Builder manual for more information on the structure of the JSON used for dataset column definitions.

Note
This method will be automatically called when the TEWBDatabaseAdapter HandleRequest method is called for automatic database access API request handling, so you normally will not need to call this method directly if you are using the HandleRequest method.

TEWBDataSetAdapter.GetRowColumn Method

```
function GetRowColumn(const ColumnName: String; Params: TParams;
      var ContentType: String; const DatabaseName: String=''):
      AnsiString

function GetRowColumn(Request: TEWBWebServerRequest; const
      ColumnName: String; var ContentType: String; const DatabaseName:
      String=''): AnsiString
```

Use this method to retrieve the a BLOB column for the dataset as a raw string (byte per character with no translation).

When this method is called, the dataset adapter will:

- Trigger the OnGetSelectColumnParams event handler to get a list of parameters for the **Select<Column>** dataset command (where <Column> is the ColumnName parameter being passed into this method).

- Trigger the OnExecuteSelectColumn event handler to execute the **Select<Column>** dataset command. The TDataSet instance that is used for retrieving the BLOB column is returned in this event handler.

There are two versions of this method: one for Elevate Web Builder server module applications and one for non-Elevate Web Builder server module applications such as ISAPI modules.

- For Elevate Web Builder server module applications, the Request parameter indicates the incoming web server request, and is used to retrieve any incoming parameters that should be used when executing the **Select** dataset command, the ColumnName parameter is the name of the BLOB column to retrieve, and the DatabaseName parameter is a pass-through parameter that is passed to any event handlers in order to indicate that the operation is occurring for a specific database.

- For non-Elevate Web Builder server module applications, the Params parameter indicates the incoming parameters that should be used when executing the **Select** dataset command, the ColumnName parameter is the name of the BLOB column to retrieve, and the DatabaseName parameter is a pass-through parameter that is passed to any event handlers in order to indicate that the operation is occurring for a specific database.

If the dataset contains a column called **<Column>_ContentType**, then the value of that column will be returned in the variable ContentType parameter and should indicate the MIME type of the BLOB column for the row that matches the incoming request parameters.

Please see the **Web Server Database Access API** topic under the **Using the Web Server** section in the Elevate Web Builder manual for more information on how BLOB column access is performed.

Note
This method will be automatically called when the TEWBDatabaseAdapter HandleRequest method is called for automatic database access API request handling, so you normally will not need to call this method directly if you are using the HandleRequest method.

TEWBDataSetAdapter.GetRowsJSON Method

```
function GetRowsJSON(Params: TParams; const DatabaseName:
      String=''): String

function GetRowsJSON(Request: TEWBWebServerRequest; const
      DatabaseName: String=''): String
```

Use this method to retrieve the rows for the dataset as a JSON string.

When this method is called, the dataset adapter will:

- Trigger the OnGetSelectParams event handler to get a list of parameters for the **Select** dataset command.

- Trigger the OnExecuteSelect event handler to execute the **Select** dataset command. The TDataSet descendant class instance that is used for building the rows is returned in this event handler.

There are two versions of this method: one for Elevate Web Builder server module applications and one for non-Elevate Web Builder server module applications such as ISAPI modules.

- For Elevate Web Builder server module applications, the Request parameter indicates the incoming web server request and is used to retrieve any incoming parameters that should be used when executing the **Select** dataset command, and the DatabaseName parameter is a pass-through parameter that is passed to any event handlers in order to indicate that the operation is occurring for a specific database.

- For non-Elevate Web Builder server module applications, the Params parameter indicates the incoming parameters that should be used when executing the **Select** dataset command, and the DatabaseName parameter is a pass-through parameter that is passed to any event handlers in order to indicate that the operation is occurring for a specific database.

Please see the **Web Server Database Access API** topic under the **Using the Web Server** section in the Elevate Web Builder manual for more information on the structure of the JSON used for dataset rows.

Note
This method will be automatically called when the TEWBDatabaseAdapter HandleRequest method is called for automatic database access API request handling, so you normally will not need to call this method directly if you are using the HandleRequest method.

TEWBDataSetAdapter.OnExecuteDelete Event

```
property OnExecuteDelete: TEWBExecuteDataSetCommandEvent
```

This event is triggered when a **Delete** dataset command is being executed as part of a transaction commit. The Params property indicates the incoming parameters for the command execution. The RowsAffected variable parameter should be populated with the number of rows that were deleted during the command execution, and the DataSet variable parameter can be ignored for **Delete** dataset commands.

TEWBDataSetAdapter.OnExecuteInsert Event

```
property OnExecuteInsert: TEWBExecuteDataSetCommandEvent
```

This event is triggered when a **Insert** dataset command is being executed as part of a transaction commit. The Params property indicates the incoming parameters for the command execution. The RowsAffected variable parameter should be populated with the number of rows that were inserted during the command execution, and the DataSet variable parameter can be ignored for **Insert** dataset commands.

TEWBDataSetAdapter.OnExecuteSelect Event

```
property OnExecuteSelect: TEWBExecuteDataSetCommandEvent
```

This event is triggered when a **Select** dataset command is being executed while retrieving the column definitions for a dataset or retrieving the rows for a dataset. The Params property indicates the incoming parameters for the command execution. The RowsAffected variable parameter should be populated with the number of rows that were selected during the command execution, and the DataSet variable parameter should be populated with the TDataSet descendant class instance that will be used for returning the rows.

TEWBDataSetAdapter.OnExecuteSelectColumn Event

```
property OnExecuteSelectColumn: TEWBExecuteDataSetColumnCommandEvent
```

This event is triggered when a **Select<Column>** dataset command is being executed while retrieving a BLOB column for a dataset. The Params property indicates the incoming parameters for the command execution. The RowsAffected variable parameter should be populated with the number of rows that were selected during the command execution, and the DataSet variable parameter should be populated with the TDataSet descendant class instance that will be used for returning the BLOB column and, optionally, its MIME type.

TEWBDataSetAdapter.OnExecuteUpdate Event

```
property OnExecuteUpdate: TEWBExecuteDataSetCommandEvent
```

This event is triggered when a **Update** dataset command is being executed as part of a transaction commit. The Params property indicates the incoming parameters for the command execution. The RowsAffected variable parameter should be populated with the number of rows that were updated during the command execution, and the DataSet variable parameter can be ignored for **Update** dataset commands.

TEWBDataSetAdapter.OnGetDeleteParams Event

```
property OnGetDeleteParams: TEWBGetDataSetCommandParamsEvent
```

This event is triggered when a **Delete** dataset command is being prepared as part of a transaction commit. The Params parameter should be populated with the list of parameters that are required for the command execution.

TEWBDataSetAdapter.OnGetInsertParams Event

```
property OnGetInsertParams: TEWBGetDataSetCommandParamsEvent
```

This event is triggered when a **Insert** dataset command is being prepared as part of a transaction commit. The Params parameter should be populated with the list of parameters that are required for the command execution.

TEWBDataSetAdapter.OnGetSelectColumnParams Event

```
property OnGetSelectColumnParams: TEWBGetDataSetColumnCommandParamsEvent
```

This event is triggered when a **Select<Column>** dataset command is being prepared while retrieving a BLOB column for a dataset. The Params parameter should be populated with the list of parameters that are required for the command execution.

TEWBDataSetAdapter.OnGetSelectParams Event

```
property OnGetSelectParams: TEWBGetDataSetCommandParamsEvent
```

This event is triggered when a **Select** dataset command is being prepared while retrieving the column definitions for a dataset or retrieving the rows for a dataset. The Params parameter should be populated with the list of parameters that are required for the command execution.

TEWBDataSetAdapter.OnGetUpdateParams Event

```
property OnGetUpdateParams: TEWBGetDataSetCommandParamsEvent
```

This event is triggered when a **Update** dataset command is being prepared as part of a transaction commit. The Params parameter should be populated with the list of parameters that are required for the command execution.

2.6 TEWBHTTPContentPart Component

Unit: ewbhttpcommon

Inherits From TObject

The TEWBHTTPContentPart class is used by the TEWBHTTPContentParts class to represent one part of the multi-part content in a web server request.

Properties	Methods	Events
Content	Create	
ContentCharSet	SaveToStream	
ContentDisposition		
ContentFileName		
ContentName		
ContentStream		
ContentTransferEncoding		
ContentType		
Headers		

TEWBHTTPContentPart.Content Property

```
property Content: String
```

Contains any textual content included with the content part. If the ContentType property is set to one of the following:

MIME Type	Description
text/plain	Plain text
text/html	HTML
text/xml	XML
application/xml	XML
application/json	JSON
application/javascript	JavaScript
application/pdf	PDF document

then the part content will be available in this property. If the ContentType property contains any other value, then the part content will need to be accessed using the ContentStream property.

Note
If the character set is set as part of the ContentType property using the **charset** attribute, then it must be set to "utf-8". If the character set is set to a different value, then the part content will need to be accessed using the ContentStream property.

TEWBHTTPContentPart.ContentCharSet Property

```
property ContentCharSet: String
```

Indicates the **charset** attribute included with the **Content-Type** content part header.

TEWBHTTPContentPart.ContentDisposition Property

```
property ContentDisposition: String
```

Indicates the value of the **Content-Disposition** content part header.

TEWBHTTPContentPart.ContentFileName Property

```
property ContentFileName: String
```

Indicates the **filename** attribute included with the **Content-Disposition** content part header.

TEWBHTTPContentPart.ContentName Property

```
property ContentName: String
```

Indicates the **name** attribute included with the **Content-Disposition** content part header.

TEWBHTTPContentPart.ContentStream Property

```
property ContentStream: TEWBStream
```

Makes any non-textual content included with the content part accessible via a TStream instance. Please see the Content property for more information on what constitutes textual content in the content part.

TEWBHTTPContentPart.ContentTransferEncoding Property

```
property ContentTransferEncoding: String
```

Indicates the value of the **Content-Transfer-Encoding** content part header.

TEWBHTTPContentPart.ContentType Property

```
property ContentType: String
```

Indicates the value of the **Content-Type** content part header.

TEWBHTTPContentPart.Headers Property

```
property Headers: TEWBHTTPContentPartHeaders
```

Accesses all headers included with the content part.

TEWBHTTPContentPart.Create Method

```
constructor Create(Owner: TEWBHTTPContentParts)
```

Use this method to create a new TEWBHTTPContentPart instance.

Note
Because the headers are automatically created and included as part of the multi-part content for the incoming web server request, you will most likely never need to call this method.

TEWBHTTPContentPart.SaveToStream Method

```
procedure SaveToStream(Stream: TStream)
```

Use this method to save any content included with the content part to a TStream instance.

2.7 TEWBHTTPContentPartHeaders Component

Unit: ewbhttpcommon

Inherits From TObject

The TEWBHTTPContentPartHeaders class is used by the TEWBHTTPContentPart class for representing the headers for one part of the multi-part content in a web server request.

Properties	Methods	Events
Count	Clear	
Names	Create	
Text		
ValueFromIndex		
Values		

TEWBHTTPContentPartHeaders.Count Property

```
property Count: Integer
```

Indicates the total number of content part headers.

TEWBHTTPContentPartHeaders.Names Property

```
property Names[Index: Integer]: String
```

Indicates the name of the content part header at the specified index.

TEWBHTTPContentPartHeaders.Text Property

```
property Text: String
```

Gets or sets the content part headers as a set of name-value pairs (<Name>: <Value>) with CRLF delimiters between each name-value pair.

TEWBHTTPContentPartHeaders.ValueFromIndex Property

```
property ValueFromIndex[Index: Integer]: String
```

Gets or sets the content part header value at the specified index.

TEWBHTTPContentPartHeaders.Values Property

```
property Values[const Name: String]: String
```

Gets or sets the content part header value with the specified name.

TEWBHTTPContentPartHeaders.Clear Method

```
procedure Clear
```

Use this method to delete all content part headers.

TEWBHTTPContentPartHeaders.Create Method

```
constructor Create
```

Use this method to create a new instance of the TEWBHTTPContentPartHeaders class.

Note
Because the headers are automatically created and included as part of the multi-part content for the incoming web server request, you will most likely never need to call this method.

2.8 TEWBHTTPContentParts Component

Unit: ewbhttpcommon

Inherits From TObject

The TEWBHTTPContentParts class is used by the TEWBWebServerRequest class to represent the multi-part content for an incoming web server request.

Properties	Methods	Events
Count	Add	
Items	Clear	
	Create	
	Delete	

TEWBHTTPContentParts.Count Property

```
property Count: Integer
```

Indicates the total number of content parts.

TEWBHTTPContentParts.Items Property

```
property Items[Index: Integer]: TEWBHTTPContentPart
```

Accesses the content part at the specified index.

TEWBHTTPContentParts.Add Method

```
function Add: TEWBHTTPContentPart
```

Use this method to add a new content part.

TEWBHTTPContentParts.Clear Method

```
procedure Clear
```

Use this method to delete all content parts.

TEWBHTTPContentParts.Create Method

```
constructor Create
```

Use this method to create a new instance of the TEWBHTTPContentParts class.

Note
Because the content parts are automatically created and included as part of the incoming web server request, you will most likely never need to call this method.

TEWBHTTPContentParts.Delete Method

```
procedure Delete(Index: Integer)
```

Use this method to delete the content part at the specified index.

2.9 TEWBHTTPCookie Component

Unit: ewbhttpcommon

Inherits From TObject

The TEWBHTTPCookie class is used with the TEWBHTTPCookies class to represent an HTTP cookie in the set of HTTP cookies included with an incoming web server request, as well as the HTTP cookies that are set in response to the web server request.

HTTP cookies are text items set by server applications and/or native server modules and returned to the browser, where they are cached according to the THTTPCookie Expires and MaxAge properties. In addition, the cookies are sent back to the same server applications and/or native server modules with any subsequent web server requests. Which cookies are sent with which server requests is controlled by the THTTPCookie Domain, Path, and Secure properties.

Properties	Methods	Events
CreatedOn	Create	
Domain		
Expires		
HTTPOnly		
MaxAge		
Name		
Path		
Secure		
Text		
Value		

TEWBHTTPCookie.CreatedOn Property

```
property CreatedOn: TEWBDateTime
```

Indicates the date/time when the cookie was originally created.

TEWBHTTPCookie.Domain Property

```
property Domain: String
```

Specifies the domain that the cookie should be used with. If the cookie domain does not match the domain of the current web server request, then it will not be sent by the browser to the web server as part of the request. The default value is blank ('').

TEWBHTTPCookie.Expires Property

```
property Expires: TEWBDateTime
```

Specifies the date/time when the cookie is set to expire. The default value is blank (0), which indicates that the cookie is a per-session cookie.

This property is the alternate of the MaxAge property. Whereas this property allows you to set the actual expiration date/time for the cookie, the MaxAge property allows you to specify how long you want the cookie to be available before it expires.

TEWBHTTPCookie.HTTPOnly Property

```
property HTTPOnly: Boolean
```

Specifies that the cookie should only be visible to the browser and not visible to any JavaScript code executing in the browser. The default value is False.

TEWBHTTPCookie.MaxAge Property

```
property MaxAge: Int64
```

Specifies how long, in seconds, the cookie should be available before it expires. The default value is 0, which indicates that the cookie is a per-session cookie.

This property is the alternate of the Expires property. Whereas this property allows you to set how long you want the cookie to be available before it expires, the Expires property allows you to set the the actual expiration date/time for the cookie.

TEWBHTTPCookie.Name Property

```
property Name: String
```

Indicates the name of the cookie.

TEWBHTTPCookie.Path Property

```
property Path: String
```

Specifies the path that the cookie should be used with. If the cookie path is not contained within the path of the current web server request, then it will not be sent by the browser to the web server as part of the request. The default value is blank ('').

TEWBHTTPCookie.Secure Property

```
property Secure: Boolean
```

Specifies that the cookie should only be used with secure HTTPS requests. If the current web server request was not a secure HTTPS request, then the cookie will not be sent by the browser to the web server as part of the request. The default value is blank ('').

TEWBHTTPCookie.Text Property

```
property Text: String
```

Gets or sets the cookie using its raw HTTP **Set-Cookie** response header format.

TEWBHTTPCookie.Value Property

```
property Value: String
```

Specifies the cookie value. The default value is blank ('').

TEWBHTTPCookie.Create Method

```
constructor Create(const AName: String)
```

Use this method to create a new TEWBHTTPCookie instance with the specified name.

2.10 TEWBHTTPCookies Component

Unit: ewbhttpcommon

Inherits From TObject

The TEWBHTTPCookies class is by the TEWBWebServerRequest class to represent the set of HTTP cookies included with an incoming web server request, as well as the HTTP cookies that are set in response to the web server request. When a request is sent to the web server, any included cookies will be available in the TEWBWebServerRequest RequestCookies property.

HTTP cookies are text items set by server applications and/or native server modules and returned to the browser, where they are cached according to the TEWBHTTPCookie Expires and MaxAge properties. In addition, the cookies are sent back to the same server applications and/or native server modules with any subsequent web server requests. Which cookies are sent with which server requests is controlled by the TEWBHTTPCookie Domain, Path, and Secure properties.

Properties	Methods	Events
Cookie	Add	
Count	Assign	
	Clear	
	Create	
	Exists	
	GetNames	
	Remove	

TEWBHTTPCookies.Cookie Property

```
property Cookie[const Name: String]: TEWBHTTPCookie
```

Retrieves the cookie that matches the specified name, or nil if no such cookie exists.

TEWBHTTPCookies.Count Property

```
property Count: Integer
```

Indicates the total number of cookies.

TEWBHTTPCookies.Add Method

```
function Add(const Name: String): TEWBHTTPCookie
```

Use this method to add a new cookie with the specified name to the set of cookies.

TEWBHTTPCookies.Assign Method

```
procedure Assign(Cookies: TEWBHTTPCookies)
```

Use this method to assign the specified cookies. All existing cookies will be deleted and overwritten with the source cookies.

TEWBHTTPCookies.Clear Method

```
procedure Clear
```

Use this method to delete all cookies in the set.

TEWBHTTPCookies.Create Method

```
constructor Create
```

Use this method to create a new instance of the TEWBHTTPCookies class.

Note
Because the cookies are automatically created and included as part of the incoming web server request, you will most likely never need to call this method.

TEWBHTTPCookies.Exists Method

```
function Exists(const Name: String): Boolean
```

Use this method to determine if a cookie with the specified name exists in the set of cookies.

TEWBHTTPCookies.GetNames Method

```
function GetNames: TEWBStringArray
```

Use this method to get an array of strings that contains the names of all of the cookies in the set.

TEWBHTTPCookies.Remove Method

```
procedure Remove(const Name: String)
```

Use this method to remove the cookie with the specified name from the set of cookies.

2.11 TEWBHTTPFormValues Component

Unit: ewbhttpcommon

Inherits From TObject

The TEWBHTTPFormValues class is used by the TEWBWebServerRequest class for representing any HTML form values included with an incoming web server request. When an HTML form is submitted to the web server, any form values will be available in the TEWBWebServerRequest RequestFormValues property.

Note
Any file uploads included as part of a multi-part HTML form submittal request will be available in the RequestContentParts property.

Properties	Methods	Events
Count	Clear	
EncodedText	Create	
Names		
Text		
ValueFromIndex		
Values		

TEWBHTTPFormValues.Count Property

```
property Count: Integer
```

Indicates the total number of HTML form values.

TEWBHTTPFormValues.EncodedText Property

```
property EncodedText: AnsiString
```

TEWBHTTPFormValues.Names Property

```
property Names[Index: Integer]: String
```

Indicates the name of the HTML form value at the specified index.

TEWBHTTPFormValues.Text Property

```
property Text: String
```

Gets or sets the HTML form values as a set of name-value pairs (<Name>=<Value>) with CRLF delimiters between each name-value pair.

TEWBHTTPFormValues.ValueFromIndex Property

```
property ValueFromIndex[Index: Integer]: String
```

Gets or sets the HTML form value at the specified index.

TEWBHTTPFormValues.Values Property

```
property Values[const Name: String]: String
```

Gets or sets the HTML form value with the specified name.

TEWBHTTPFormValues.Clear Method

```
procedure Clear
```

Use this method to delete all HTML form values.

TEWBHTTPFormValues.Create Method

```
constructor Create
```

Use this method to create a new instance of the TEWBHTTPFormValues class.

Note
Because the HTML form values are automatically created and included as part of the incoming web server request, you will most likely never need to call this method.

2.12 TEWBHTTPHeaders Component

Unit: ewbhttpcommon

Inherits From TObject

The TEWBHTTPHeaders class is used by the TEWBWebServerRequest class for representing the request/response headers for an incoming web server request. When a request is sent to the web server, any HTTP request headers will be available in the TEWBWebServerRequest RequestHeaders property, and the server application can set any HTTP response headers using the TEWBWebServerRequest ResponseHeaders property.

Properties	Methods	Events
Count	Assign	
Names	Clear	
Text	Create	
ValueFromIndex		
Values		

TEWBHTTPHeaders.Count Property

```
property Count: Integer
```

Indicates the total number of headers.

TEWBHTTPHeaders.Names Property

```
property Names[Index: Integer]: String
```

Indicates the name of the header at the specified index.

TEWBHTTPHeaders.Text Property

```
property Text: String
```

Gets or sets the headers as a set of name-value pairs (<Name>: <Value>) with CRLF delimiters between each name-value pair.

TEWBHTTPHeaders.ValueFromIndex Property

```
property ValueFromIndex[Index: Integer]: String
```

Gets or sets the header value at the specified index.

TEWBHTTPHeaders.Values Property

```
property Values[const Name: String]: String
```

Gets or sets the header value with the specified name.

TEWBHTTPHeaders.Assign Method

```
procedure Assign(Strings: TStrings)

procedure Assign(Headers: TEWBHTTPHeaders)
```

Use this method to assign the headers contained within another TEWBHTTPHeaders instance to the current instance.

TEWBHTTPHeaders.Clear Method

```
procedure Clear
```

Use this method to delete all headers.

TEWBHTTPHeaders.Create Method

```
constructor Create
```

Use this method to create a new instance of the TEWBHTTPHeaders class.

Note
Because the headers are automatically created and included as part of the incoming web server request, you will most likely never need to call this method.

2.13 TEWBHTTPParameters Component

Unit: ewbhttpcommon

Inherits From TObject

The TEWBHTTPParameters class is used by the TEWBWebServerRequest class for representing the URL parameters for an incoming web server request. When a request is sent to the web server, any URL parameters in the HTTP request will be available in the TEWBWebServerRequest RequestParameters property.

Properties	Methods	Events
Count	Assign	
EncodedText	Clear	
Names	Create	
Text		
ValueFromIndex		
Values		

TEWBHTTPParameters.Count Property

```
property Count: Integer
```

Indicates the total number of parameters.

TEWBHTTPParameters.EncodedText Property

```
property EncodedText: String
```

Gets or sets the parameters as a set of name-value pairs (<Name>=<Value>) in URL format:

```
?<Name>=<Value>[&<Name>=<Value>[&<Name>=<Value>]]
```

Note
The value returned by this property is encoded and is usable as part of a valid URL.

TEWBHTTPParameters.Names Property

```
property Names[Index: Integer]: String
```

Indicates the name of the parameter at the specified index.

TEWBHTTPParameters.Text Property

```
property Text: String
```

Gets or sets the parameters as a set of name-value pairs (<Name>=<Value>) in URL format:

```
?<Name>=<Value>[&<Name>=<Value>[&<Name>=<Value>]]
```

Note
The value returned by this property is not encoded and usable as part of a valid URL. Please see the TEWBHTTPParameters EncodedText property for the encoded version of this property.

TEWBHTTPParameters.ValueFromIndex Property

```
property ValueFromIndex[Index: Integer]: String
```

Gets or sets the parameter value at the specified index.

TEWBHTTPParameters.Values Property

```
property Values[const Name: String]: String
```

Gets or sets the parameter value with the specified name.

TEWBHTTPParameters.Assign Method

```
procedure Assign(Parameters: TEWBHTTPParameters)
```

Use this method to assign the specified parameters. All existing parameters will be deleted and overwritten with the source parameters.

TEWBHTTPParameters.Clear Method

```
procedure Clear
```

Use this method to delete all parameters.

TEWBHTTPParameters.Create Method

```
constructor Create
```

Use this method to create a new instance of the TEWBHTTPParameters class.

Note
Because the parameters are automatically created and included as part of the incoming web server request, you will most likely never need to call this method.

2.14 TEWBHTTPPathComponents Component

Unit: ewbhttpcommon

Inherits From TObject

The TEWBHTTPPathComponents class is used by the TEWBWebServerRequest class for representing the URL path components for an incoming web server request. When a request is sent to the web server, any URL path components in the HTTP request will be available in the TEWBWebServerRequest RequestPathComponents property.

Properties	Methods	Events
Components	Assign	
Count	Clear	
Delimiter	Create	
Text		

TEWBHTTPPathComponents.Components Property

```
property Components[Index: Integer]: String
```

Gets the path component at the specified index.

TEWBHTTPPathComponents.Count Property

```
property Count: Integer
```

Indicates the total number of path components.

TEWBHTTPPathComponents.Delimiter Property

```
property Delimiter: String
```

Specifies the delimiter to use with the Text property. The default value is "/".

TEWBHTTPPathComponents.Text Property

```
property Text: String
```

Gets or sets the path components as a set of values delimited by the Delimiter property.

TEWBHTTPPathComponents.Assign Method

```
procedure Assign(PathComponents: TEWBHTTPPathComponents)
```

Use this method to assign all of the specified path components. All path components will be deleted and overwritten with the source path components.

TEWBHTTPPathComponents.Clear Method

```
procedure Clear
```

Use this method to delete all path components.

TEWBHTTPPathComponents.Create Method

```
constructor Create
```

Use this method to create a new instance of the TEWBHTTPPathComponents class.

Note
Because the path components are automatically created and included as part of the incoming web server request, you will most likely never need to call this method.

2.15 TEWBModule Component

Unit: ewbhttpmodule

Inherits From TDataModule

The TEWBModule component represents an instance of a native server module in the web server. The web server automatically creates a new instance of the TEWBModule component whenever an incoming web server request is routed to the native server module and fires the OnExecute event in order to allow the component to handle the request.

Warning
All code in any OnExecute event handler must be completely thread-safe. Also, all native server modules in the web server run in-process, so a fatal error such as a memory overwrite due to improper threading code could cause the server to fail.

Properties	Methods	Events
	Create	OnExecute

TEWBModule.Create Method

```
constructor Create(AOwner: TComponent)
```

Use this method to create a new instance of the TEWBModule class. The AOwner parameter indicates the component instance that will own and manage the module instance.

TEWBModule.OnExecute Event

```
property OnExecute: TEWBModuleExecuteEvent
```

This event fires when a new request is received by the web server and is routed to the native server module.

2.16 TEWBMultiPartServerRequestContent Component

Unit: ewbsrvrequest

Inherits From TObject

The TEWBMultipartServerRequestContent class is used by the TEWBServerRequest component to represent multi-part content in a web server request.

Properties	Methods	Events
Content	Add	
Count	Assign	
	Clear	
	Create	
	Delete	

TEWBMultiPartServerRequestContent.Content Property

```
property Content[Index: Integer]: TEWBServerRequestContent
```

Accesses the content part at the specified index.

TEWBMultiPartServerRequestContent.Count Property

```
property Count: Integer
```

Indicates the total number of content parts.

TEWBMultiPartServerRequestContent.Add Method

```
function Add: TEWBServerRequestContent
```

Use this method to add a new content part.

TEWBMultiPartServerRequestContent.Assign Method

```
procedure Assign(AContent: TEWBMultiPartServerRequestContent)
```

Use this method to assign the contents of another TEWBMultipartServerRequestContent instance to the current instance.

TEWBMultiPartServerRequestContent.Clear Method

```
procedure Clear
```

Use this method to delete all content parts.

TEWBMultiPartServerRequestContent.Create Method

```
constructor Create
```

Use this method to create a new instance of the TEWBMultipartServerRequestContent class.

Note
Because the multi-part content is automatically created and included as part of the TEWBServerRequest component, you will most likely never need to call this method.

TEWBMultiPartServerRequestContent.Delete Method

```
procedure Delete(Index: Integer)
```

Use this method to delete the content part at the specified index.

2.17 TEWBServerRequest Component

Unit: ewbsrvrequest

Inherits From TComponent

The TEWBServerRequest component represents a dynamic HTTP request to a web server.

Properties	Methods	Events
CompleteURL	Cancel	OnCancel
ConnectTimeout	Create	OnComplete
Content	Execute	OnError
ContentStream	Reset	OnProgress
Cookies		OnStart
Data		OnTimeout
Headers		
Method		
MethodLiteral		
MultiPartContent		
Params		
Queue		
RawContent		
RequestType		
ResponseContent		
ResponseContentStream		
ResponseContentType		
ResponseHeaders		
ResponseType		
StatusCode		
StatusText		
Timeout		
URL		

TEWBServerRequest.CompleteURL Property

```
property CompleteURL: String
```

Indicates the complete URL that will be used with the web server request.

TEWBServerRequest.ConnectTimeout Property

```
property ConnectTimeout: Integer
```

Specifies the connection timeout for the web server request. A value of 0 specifies that the server request should wait indefinitely. The default value is 30 seconds.

TEWBServerRequest.Content Property

```
property Content: String
```

Specifies the content to send with the web server request.

TEWBServerRequest.ContentStream Property

```
property ContentStream: TStream
```

Specifies a reference to a TStream descendant class instance to send as the content with the web server request.

Warning
This is just a reference to the TStream descendant class instance, so please make sure that you do not free this class instance before the TEWBServerRequest Execute method completes.

TEWBServerRequest.Cookies Property

```
property Cookies: TEWBHTTPCookies
```

Specifies the cookies to send with the web server request. These cookies are also updated with any cookies returned when the request is handled by the web server and a response has been received.

TEWBServerRequest.Data Property

```
property Data: TObject
```

Specifies a object reference that you wish to associate with the current web server request.

Note
The web server request does not manage the lifetime of the underlying object, and will not free it when the web server request has been freed.

TEWBServerRequest.Headers Property

```
property Headers: TEWBHTTPHeaders
```

Specifies any additional headers to be sent with the web server request.

Note
All necessary headers will be automatically populated by the web browser, so only use this property for specifying headers that are necessary for the web server request, such as a **Content-Type** header when sending content to the web server in the Content property.

TEWBServerRequest.Method Property

```
property Method: TEWBHTTPMethod
```

Specifies the HTTP method to use for the web server request.

TEWBServerRequest.MethodLiteral Property

```
property MethodLiteral: String
```

Indicates the actual HTTP method literal specified by the Method property.

TEWBServerRequest.MultiPartContent Property

```
property MultiPartContent: TEWBMultiPartServerRequestContent
```

Specifies the multi-part content to send with the web server request.

TEWBServerRequest.Params Property

```
property Params: TEWBHTTPParameters
```

Specifies the URL parameters for the web server request as a set of name/value pairs of strings.

TEWBServerRequest.Queue Property

```
property Queue: TEWBServerRequestQueue
```

If the current server request instance was created by a TEWBServerRequestQueue component, then this property will contain a reference to the instance of the queue component that created the request.

TEWBServerRequest.RawContent Property

```
property RawContent: AnsiString
```

TEWBServerRequest.RequestType Property

```
property RequestType: TEWBServerRequestType
```

Specifies whether the web server request will be executed synchronously (the default) or asynchronously.

Note
It is not recommended that you execute web server requests asynchronously in a native web server module or any other type of environment where one expects the Execute method to wait until the request is executed before returning.

TEWBServerRequest.ResponseContent Property

```
property ResponseContent: String
```

Indicates any content that was returned from the web server in the response to the server request.

Note
This property will only be populated once the OnComplete event has been triggered.

TEWBServerRequest.ResponseContentStream Property

```
property ResponseContentStream: TStream
```

Contains any content included with the response to the server request as a TStream descendant class instance when the ResponseContentType is set to srStream. The use of a TStream interface makes it easier to work with response content that is binary and not textual.

TEWBServerRequest.ResponseContentType Property

```
property ResponseContentType: String
```

Indicates the value of the **Content-Type** HTTP header, if present, that was returned from the web server in the response to the server request. If no **Content-Type** header was returned, then this property will return an empty string ('').

TEWBServerRequest.ResponseHeaders Property

```
property ResponseHeaders: TEWBHTTPHeaders
```

Indicates any headers that were returned from the web server in the response to the server request.

Note
This property will only be populated once the OnComplete event has been triggered.

TEWBServerRequest.ResponseType Property

```
property ResponseType: TEWBHTTPResponseType
```

Specifies how any content included with the response to the server request should be represented. The default value is srText.

TEWBServerRequest.StatusCode Property

```
property StatusCode: Integer
```

Indicates the status code returned by the web server in response to the server request.

Note
This property will only be populated once the OnComplete event has been triggered.

TEWBServerRequest.StatusText Property

```
property StatusText: String
```

Indicates the status message returned by the web server in response to the server request.

Note
This property will only be populated once the OnComplete event has been triggered.

TEWBServerRequest.Timeout Property

```
property Timeout: Integer
```

Specifies how long the server request should wait, in seconds, for a successful connection to the server before returning an error. A value of 0 specifies that the server request should wait indefinitely. The default value is 30 seconds.

TEWBServerRequest.URL Property

```
property URL: String
```

Specifies the URL of the resource that the server request wishes to get data from, or send data to.

TEWBServerRequest.Cancel Method

```
procedure Cancel(KeepExecuting: Boolean=True)
```

Use this method to abort a pending web server request.

TEWBServerRequest.Create Method

```
constructor Create

constructor Create(AOwner: TComponent)
```

Use this method to create a new instance of the TEWBServerRequest class.

TEWBServerRequest.Execute Method

```
procedure Execute
```

Use this method to execute a web server request after specifying the Method and URL properties, at a minimum.

TEWBServerRequest.Reset Method

```
procedure Reset
```

Use this method to reset all server request properties back to their default values. This is useful for situations where a server request instance is being reused multiple times and you need to be sure that the server request is initialized back to its default state.

TEWBServerRequest.OnCancel Event

```
property OnCancel: TEWBServerRequestEvent
```

This event is triggered when the server request has been cancelled using the Cancel method.

Note
This event is triggered before the OnComplete event.

TEWBServerRequest.OnComplete Event

```
property OnComplete: TEWBServerRequestEvent
```

This event is triggered when the server request is complete. Use the StatusCode property to determine the status code returned by the web server and, subsequently, whether the request was successful or not.

TEWBServerRequest.OnError Event

```
property OnError: TEWBServerRequestErrorEvent
```

This event is triggered when the server request has encountered an error.

Note
This event is triggered before the OnComplete event.

TEWBServerRequest.OnProgress Event

```
property OnProgress: TEWBServerRequestProgressEvent
```

This event is triggered as the server request is executed and represents the current progress of the request.

TEWBServerRequest.OnStart Event

```
property OnStart: TEWBServerRequestEvent
```

This event is triggered when the server request is started. A server request is started when the Execute method is called.

TEWBServerRequest.OnTimeout Event

```
property OnTimeout: TEWBServerRequestEvent
```

This event is triggered when the server request has timed out.

Note
This event is triggered before the OnComplete event.

2.18 TEWBServerRequestContent Component

Unit: ewbsrvrequest

Inherits From TObject

The TEWBServerRequestContent class is used by the TEWBMultipartServerRequestContent class to represent one part of the multi-part content for a web server request.

Properties	Methods	Events
Content	Create	
ContentStream		
Headers		
RawContent		

TEWBServerRequestContent.Content Property

```
property Content: String
```

Available In: Server Applications

Specifies the textual content to include as the content part. This property and the ContentStream property are mutually-exclusive. If both properties are assigned a value, then the ContentStream property will take precedence.

Note
If this property is assigned a value and the **Content-Type** header is not specified using the Headers property, it will be automatically set to "application/json; charset=utf-8" when the server request is executed.

TEWBServerRequestContent.ContentStream Property

```
property ContentStream: TStream
```

Available In: Server Applications

Specifies a TStream descendant class instance that contains the binary content to include as the content part. This property and the Content property are mutually-exclusive. If both properties are assigned a value, then the ContentStream property will take precedence.

Note
If this property is assigned a value and the **Content-Type** header is not specified using the Headers property, it will be automatically set to "application/octet-stream" when the server request is executed.

Warning
This is just a reference to the TStream descendant class instance, so please make sure that you do not free this class instance before the TServerRequest Execute method is called.

TEWBServerRequestContent.Headers Property

```
property Headers: TEWBHTTPHeaders
```

Available In: Server Applications

Provides access to the headers for the content part.

TEWBServerRequestContent.RawContent Property

```
property RawContent: AnsiString
```

TEWBServerRequestContent.Create Method

```
constructor Create
```

Available In: Server Applications

Use this method to create a new instance of the TServerRequestContent class.

2.19 TEWBServerRequestQueue Component

Unit: ewbsrvrequest

Inherits From TComponent

The TEWBServerRequestQueue component represents a queue of TEWBServerRequest instances.

Properties	Methods	Events
NumPendingRequests	AddRequest	
	CancelAllRequests	
	CancelRequest	
	Create	
	ExecuteRequests	
	GetNewRequest	
	NextPendingRequest	

TEWBServerRequestQueue.NumPendingRequests Property

```
property NumPendingRequests: Integer
```

Available In: Client Applications

Indicates the number of requests currently awaiting execution.

Note
The value returned by this property does **not** include the currently-executing request, if one exists. If a currently-executing request fails for any reason, then the value returned by this property **will** include the failed request when this property is referenced from an OnComplete event handler.

TEWBServerRequestQueue.AddRequest Method

```
procedure AddRequest(Value: TEWBServerRequest)
```

Available In: Client Applications

Use this method to add a new request to the end of the queue of web server requests. When using this method, always use the GetNewRequest to obtain a new request that can be modified before calling this method.

TEWBServerRequestQueue.CancelAllRequests Method

```
procedure CancelAllRequests
```

Available In: Client Applications

Use this method to cancel all pending web server requests in the queue.

TEWBServerRequestQueue.CancelRequest Method

```
procedure CancelRequest(KeepExecuting: Boolean=True)
```

Available In: Client Applications

Use this method to cancel the currently executing web server request in the queue. After the current request is cancelled, the next request in the queue will be executed.

TEWBServerRequestQueue.Create Method

```
constructor Create

constructor Create(AOwner: TComponent)
```

Use this method to create a new instance of the TEWBServerRequestQueue class.

TEWBServerRequestQueue.ExecuteRequests Method

```
procedure ExecuteRequests
```

Available In: Client Applications

Use this method to execute any pending web server requests in the queue.

TEWBServerRequestQueue.GetNewRequest Method

```
function GetNewRequest: TEWBServerRequest
```

Available In: Client Applications

Use this method to allocate a new web server request to use with the queue. After modifying the request as required, call the AddRequest method to add the request to the queue.

TEWBServerRequestQueue.NextPendingRequest Method

```
function NextPendingRequest: TEWBServerRequest
```

Available In: Client Applications

Use this method to get access to the next pending web server request in the queue.

2.20 TEWBServerSession Component

Unit: ewbsession

Inherits From TEWBNotificationComponent

The TEWBServerSession component represents a web server session and is used for authentication and retrieving basic information about the authenticated user after authentication has taken place. In addition, the TDatabase component uses a TEWBServerSession component instance for performing authentication as necessary when making database API requests. The server session is associated with the TEWBDatabase component using its ServerSession property.

Note
Eventually this component will provide the classes and methods for remotely administering a web server instance, but currently does not provide such functionality. At this time, the only way to remotely administer a web server is by using the IDE.

Properties	Methods	Events
AdministrationResource	Authenticate	OnAuthenticate
ApplicationsResource	CancelRequest	OnComplete
Authenticated	Create	OnProgress
Authenticating	Deauthenticate	
AuthenticationResource	HasPrivilege	
BaseURL	HasRole	
DatabasesResource	RefreshEffectiveAccess	
DebuggerResource		
EffectivePrivileges		
EffectiveRoles		
FullName		
ID		
KeepAliveResource		
ModulesResource		
Password		
RequestCookies		
RequestExecuting		
RequestStatusCode		
RequestStatusText		
UserName		

TEWBServerSession.AdministrationResource Property

```
property AdministrationResource: String
```

Specifies the resource name that should be used with any administration API requests.

TEWBServerSession.ApplicationsResource Property

```
property ApplicationsResource: String
```

Specifies the resource name that should be used with any server application API requests.

TEWBServerSession.Authenticated Property

```
property Authenticated: Boolean
```

Indicates whether the current session is authenticated (from the perspective of the client application).

TEWBServerSession.Authenticating Property

```
property Authenticating: Boolean
```

Indicates whether the current session is in the process of authenticating.

TEWBServerSession.AuthenticationResource Property

```
property AuthenticationResource: String
```

Specifies the resource name that should be used with any authentication API requests.

TEWBServerSession.BaseURL Property

```
property BaseURL: String
```

Specifies the base URL for the target web server for any server requests made by the server session. In addition, this base URL is used by the TEWBDatabase component to create the proper URLs for making database API requests. The server session is associated with the TEWBDatabase component using its ServerSession property.

TEWBServerSession.DatabasesResource Property

```
property DatabasesResource: String
```

Specifies the resource name that should be used with any database API requests.

TEWBServerSession.DebuggerResource Property

```
property DebuggerResource: String
```

Specifies the resource name that should be used with any debugger API requests.

TEWBServerSession.EffectivePrivileges Property

```
property EffectivePrivileges: TStringList
```

Contains a list of the effective privileges for the user when the Authenticated property is True.

TEWBServerSession.EffectiveRoles Property

```
property EffectiveRoles: TStringList
```

Contains a list of the effective roles for the user when the Authenticated property is True.

TEWBServerSession.FullName Property

```
property FullName: String
```

Indicates the full name for the user when the Authenticated property is True.

TEWBServerSession.ID Property

```
property ID: String
```

Indicates the ID assigned to the session on the web server when the Authenticated property is True.

TEWBServerSession.KeepAliveResource Property

```
property KeepAliveResource: String
```

Specifies the resource name that should be used with any keep-alive requests.

TEWBServerSession.ModulesResource Property

```
property ModulesResource: String
```

Specifies the resource name that should be used with any native server module API requests.

TEWBServerSession.Password Property

```
property Password: String
```

Specifies the password for the user specified in the UserName property.

TEWBServerSession.RequestCookies Property

```
property RequestCookies: TEWBHTTPCookies
```

Provides access to the cookies that were returned with the last server request executed using the server session.

TEWBServerSession.RequestExecuting Property

```
property RequestExecuting: Boolean
```

Indicates whether a server request is currently being executed using the server session.

TEWBServerSession.RequestStatusCode Property

```
property RequestStatusCode: Integer
```

Indicates the status code of the last server request executed using the server session.

TEWBServerSession.RequestStatusText Property

```
property RequestStatusText: String
```

Indicates the status text of the last server request executed using the server session.

TEWBServerSession.UserName Property

```
property UserName: String
```

Specifies the user name to use with the server session.

TEWBServerSession.Authenticate Method

```
procedure Authenticate
```

Use this method to authenticate the server session using the UserName and Password properties.

TEWBServerSession.CancelRequest Method

```
procedure CancelRequest
```

Use this method to cancel any web server request that is being currently being executed to satisfy a server session method such as the Authenticate method.

TEWBServerSession.Create Method

```
constructor Create(AOwner: TComponent)
```

Use this method to create a new instance of the TEWBServerSession class.

TEWBServerSession.Deauthenticate Method

```
procedure Deauthenticate
```

Use this method to deauthenticate the server session.

TEWBServerSession.HasPrivilege Method

```
function HasPrivilege(const AName: String): Boolean
```

Use this method to determine if the authenticated user for the server session has been granted the specified privilege (indirectly through the roles that the user has been granted).

TEWBServerSession.HasRole Method

```
function HasRole(const AName: String): Boolean
```

Use this method to determine if the authenticated user for the server session has been granted the specified role.

TEWBServerSession.RefreshEffectiveAccess Method

```
procedure RefreshEffectiveAccess
```

User this method to refresh the EffectivePrivileges and EffectiveRoles properties from the web server. This is useful for situations where the user security has been updated and the server session needs to retrieve the new effective privileges and roles for the authenticated user.

TEWBServerSession.OnAuthenticate Event

```
property OnAuthenticate: TEWBServerSessionAuthenticateEvent
```

This event is triggered before a server session attempts to perform authentication. You can use an event handler attached to this event to update the UserName and/or Password properties before the authentication occurs.

Set the Continue parameter to False to prevent the authentication from proceeding.

Note
If the Continue parameter is set to False, an exception will be raised indicating that the authentication was cancelled.

TEWBServerSession.OnComplete Event

```
property OnComplete: TEWBServerSessionEvent
```

This event is triggered when a server session request is complete. Use the RequestStatusCode property to determine the status code returned by the web server and, subsequently, whether the session request was successful or not.

TEWBServerSession.OnProgress Event

```
property OnProgress: TEWBServerSessionProgressEvent
```

This event is triggered as a server session request is executed and represents the current progress of the request.

2.21 TEWBWebServerRequest Component

Unit: ewbhttpmodule

Inherits From TObject

The TEWBWebServerRequest class represents an incoming web server request in native server modules and is passed as a parameter to the TEWBModule OnExecute event.

Properties	Methods	Events
RequestClientAddress	CheckMethod	
RequestContent	ComputeHash	
RequestContentLength	Create	
RequestContentParts	DateTimeToGMTStr	
RequestContentStream	DateTimeToMSecs	
RequestContentType	GMTDateTimeToLocal	
RequestCookies	LocalDateTimeToGMT	
RequestFormValues	MSecsToDateTime	
RequestHeaders	ParseHeaderAttribute	
RequestHost	SendContent	
RequestMethod	SendContentHeader	
RequestMethodName	SendContentStream	
RequestParameters	SendCustomContent	
RequestPathComponents	SendCustomContentHeader	
RequestSecure	SendCustomContentStream	
RequestURL	SendError	
RequestURLParams	SendRawCustomContent	
RequestURLPath	SendRedirect	
ResponseCookies	SetResponseNoCacheHeader	
ResponseHeaders		
ResponseSessionCookies		

TEWBWebServerRequest.RequestClientAddress Property

```
property RequestClientAddress: String
```

Indicates the public IP address of the client making the request to the web server.

TEWBWebServerRequest.RequestContent Property

```
property RequestContent: String
```

Contains any textual content included with the web server request. If the RequestContentType property is set to one of the following:

MIME Type	Description
text/plain	Plain text
text/html	HTML
text/xml	XML
application/xml	XML
application/json	JSON
application/javascript	JavaScript
application/pdf	PDF document

then the included content will be available in this property. If the RequestContentType property contains any other value, then the included content will need to be accessed using the RequestContentStream property.

Note
If the character set is set as part of the RequestContentType property using the **charset** attribute, then it must be set to "utf-8". If the character set is set to a different value, then the included content will need to be accessed using the RequestContentStream property.

TEWBWebServerRequest.RequestContentLength Property

```
property RequestContentLength: Int64
```

Indicates the size, in bytes, of any content included with the web server request. This value is the integer value specified in the **Content-Length** header in the RequestHeaders property.

TEWBWebServerRequest.RequestContentParts Property

```
property RequestContentParts: TEWBHTTPContentParts
```

If the web server request includes multi-part content, then this property will contain the various parts that make up the content.

TEWBWebServerRequest.RequestContentStream Property

```
property RequestContentStream: TStream
```

Makes any non-textual content included with the web server request accessible via a TStream instance. Please see the RequestContent property for more information on what constitutes textual content in the web server request.

TEWBWebServerRequest.RequestContentType Property

```
property RequestContentType: String
```

Indicates the content type of any content included with the web server request. This value is the string value specified in the **Content-Type** header in the RequestHeaders property.

TEWBWebServerRequest.RequestCookies Property

```
property RequestCookies: TEWBHTTPCookies
```

Contains any HTTP cookies included with the web server request.

TEWBWebServerRequest.RequestFormValues Property

```
property RequestFormValues: TEWBHTTPFormValues
```

If the web server request is an HTML form submittal, this property will contain the submitted form values.

Note
Any file uploads included as part of a multi-part HTML form submittal request will be available in the RequestContentParts property.

TEWBWebServerRequest.RequestHeaders Property

```
property RequestHeaders: TEWBHTTPHeaders
```

Contains any HTTP headers included with the web server request.

TEWBWebServerRequest.RequestHost Property

```
property RequestHost: String
```

Indicates the target host for the web server request. This value is the string value specified in the **Host** header in the RequestHeaders property.

TEWBWebServerRequest.RequestMethod Property

```
property RequestMethod: TEWBWebServerRequestMethod
```

Indicates the HTTP method for the web server request.

TEWBWebServerRequest.RequestMethodName Property

```
property RequestMethodName: String
```

Indicates the textual version of the HTTP method for the web server request.

TEWBWebServerRequest.RequestParameters Property

```
property RequestParameters: TEWBHTTPParameters
```

Contains any URL parameters included with the web server request.

TEWBWebServerRequest.RequestPathComponents Property

```
property RequestPathComponents: TEWBHTTPPathComponents
```

Contains all path components contained in the RequestURLPath property.

TEWBWebServerRequest.RequestSecure Property

```
property RequestSecure: Boolean
```

Indicates whether the web server request as a secure HTTPs or insecure HTTP request.

TEWBWebServerRequest.RequestURL Property

```
property RequestURL: String
```

Contains the full URL for the web server request, including any URL parameters.

TEWBWebServerRequest.RequestURLParams Property

```
property RequestURLParams: String
```

Contains any URL parameters included as part of the RequestURL property.

TEWBWebServerRequest.RequestURLPath Property

```
property RequestURLPath: String
```

Contains the URL path included as part of the RequestURL property.

Note
This property only contains the path and does not contain the origin (protocol, host, port) or any parameters.

TEWBWebServerRequest.ResponseCookies Property

```
property ResponseCookies: TEWBHTTPCookies
```

Specifies any cookies included with the response to the web server request.

TEWBWebServerRequest.ResponseHeaders Property

```
property ResponseHeaders: TEWBHTTPHeaders
```

Specifies any headers to be included with the response to the web server request.

TEWBWebServerRequest.ResponseSessionCookies Property

```
property ResponseSessionCookies: TEWBHTTPCookies
```

Specifies any session-only cookies to be included with the response to the web server request.

Note
Please use this property instead of the ResponseCookies property when returning session-only cookies. The web server request automatically sets the TEWBHTTPCookie Expires property to a default value of 1 year from the current date/time for all cookies contained in the ResponseCookies property.

TEWBWebServerRequest.CheckMethod Method

```
function CheckMethod(AllowedMethods:
      TEWBWebServerRequestMethods): Boolean
```

Use this method to check to see if the web server request's HTTP method is in the specified set of HTTP methods. If the HTTP method is not in the specified set, then a "Method not allowed" 405 HTTP error status is immediately sent as the response to the web server request.

TEWBWebServerRequest.ComputeHash Method

```
function ComputeHash(AHashType: TEWBHashType; const Value:
      AnsiString): String
```

Use this method to compute a cryptographic hash for the string input parameter. The hash type parameter determines the type of computed hash that is returned, and the return value is a hexadecimal string containing the computed hash.

TEWBWebServerRequest.Create Method

```
constructor Create
```

Use this method to create a new instance of the TEWBWebServerRequest class.

TEWBWebServerRequest.DateTimeToGMTStr Method

```
function DateTimeToGMTStr(Value: TDateTime): String
```

Converts a TDateTime value into a GMT date/time string that adheres to RFC1123 and has the format:

```
dow, day month year hours:mins:secs GMT
```

TEWBWebServerRequest.DateTimeToMSecs Method

```
function DateTimeToMSecs(Value: TDateTime): Int64
```

Converts a TDateTime value into the number of milliseconds since midnight, January 1, 1970.

Note
The result is the native date/time representation used by JavaScript and Elevate Web Builder client applications.

TEWBWebServerRequest.GMTDateTimeToLocal Method

```
function GMTDateTimeToLocal(Value: TDateTime; IncludeDST:
      Boolean=True): TDateTime
```

Converts a GMT (UTC) TDateTime value into its local equivalent, optionally accounting for Daylight Savings Time.

TEWBWebServerRequest.LocalDateTimeToGMT Method

```
function LocalDateTimeToGMT(Value: TDateTime; IncludeDST:
      Boolean=True): TDateTime
```

Converts a local TDateTime value into its GMT (UTC) equivalent, optionally accounting for Daylight Savings Time.

TEWBWebServerRequest.MSecsToDateTime Method

```
function MSecsToDateTime(Value: Int64): TDateTime
```

Converts the number of milliseconds since midnight, January 1, 1970 into a TDateTime value.

Note
The input is the native date/time representation used by JavaScript and Elevate Web Builder client applications.

TEWBWebServerRequest.ParseHeaderAttribute Method

```
function ParseHeaderAttribute(const Attribute: String; const
      Header: String): String
```

Use this method to parse a specific HTTP header attribute from the passed header value.

HTTP headers use the following format:

```
<Header Name>: <Header Value> [; <Attribute=<Attribute Value>...]
```

TEWBWebServerRequest.SendContent Method

```
procedure SendContent(const Content: String; StatusCode:
      Integer=HTTP_OK; const StatusMessage: String='')
```

Sends a UTF-8-encoded text response with a **Content-Type** header of "text/html; charset=utf-8" along with an optional status code and message.

TEWBWebServerRequest.SendContentHeader Method

```
procedure SendContentHeader(StatusCode: Integer=HTTP_OK; const
      StatusMessage: String='')
```

Sends a response for a HEAD request.

TEWBWebServerRequest.SendContentStream Method

```
procedure SendContentStream(const ContentStream: TStream;
      StatusCode: Integer=HTTP_OK; const StatusMessage: String='')
```

Sends a UTF-8-encoded text stream response with a **Content-Type** header of "text/html; charset=utf-8" along with an optional status code and message.

TEWBWebServerRequest.SendCustomContent Method

```
procedure SendCustomContent(const Content: String; const
      ContentType: String; const ContentEncoding: String=''; const
      ContentDisposition: String='')
```

Sends a UTF-8-encoded text response with a custom content type, encoding, and disposition.

TEWBWebServerRequest.SendCustomContentHeader Method

```
procedure SendCustomContentHeader(const ContentType: String;
      const ContentEncoding: String=''; const ContentDisposition:
      String='')
```

Sends a custom content response for a HEAD request.

TEWBWebServerRequest.SendCustomContentStream Method

```
procedure SendCustomContentStream(const ContentStream: TStream;
      const ContentType: String; const ContentEncoding: String='';
      const ContentDisposition: String='')
```

Sends a binary stream response with a custom content type, encoding, and disposition.

TEWBWebServerRequest.SendError Method

```
procedure SendError(StatusCode: Integer; const StatusMessage:
      String; const Content: String='')
```

Sends a status code and message along with an optional UTF-8-encoded text response with a **Content-Type** header of "text/plain; charset=utf-8".

TEWBWebServerRequest.SendRawCustomContent Method

```
procedure SendRawCustomContent(const Content: AnsiString; const
      ContentType: String; const ContentEncoding: String=''; const
      ContentDisposition: String='')
```

Sends a raw (AnsiString) text response with a custom content type, encoding, and disposition.

TEWBWebServerRequest.SendRedirect Method

```
procedure SendRedirect(const NewLocationURL: String; const
      Content: String=''; StatusCode: Integer=HTTP_FOUND; const
      StatusMessage: String='Redirecting')
```

Sends a redirect response to the client browser using the passed new location, content, HTTP status code and message. The Unicode content is automatically sent in UTF-8 format.

Please see the following link for a complete list of HTTP status/error codes:

Status Code and Reason Phrase

Note
Only one response can be sent per request. Attempting to send more than one response will cause errors in the client browser.

TEWBWebServerRequest.SetResponseNoCacheHeader Method

```
procedure SetResponseNoCacheHeader
```

Use this method to set the **Cache-Control** response header to "no-cache" in the ResponseHeaders property.

Chapter 3
Type Reference

3.1 TEWBDatabaseErrorEvent Type

Unit: ewbdataset

```
TEWBDatabaseErrorEvent = procedure (Sender: TObject; const
      ErrorMsg: String) of object
```

The TEWBDatabaseErrorEvent type is used by the TEWBDatabase OnCommitError and OnRollbackError events.

The Sender parameter is the TEWBDatabase instance that triggered the event and the ErrorMsg parameter is the complete error message.

3.2 TEWBDatabaseEvent Type

Unit: ewbdataset

```
TEWBDatabaseEvent = function (Sender: TObject): Boolean of
      object
```

The TEWBDatabaseEvent type is used by the TEWBDatabase BeforeCommit and BeforeRollback events.

The Sender parameter is the TEWBDatabase instance that triggered the event. Return True from the event to allow the applicable database functionality to continue, or False to prevent the functionality from occurring.

3.3 TEWBDataSetErrorEvent Type

Unit: ewbdataset

```
TEWBDataSetErrorEvent = procedure (Sender: TObject; const
      ErrorMsg: String) of object
```

The TEWBDataSetErrorEvent type is used by the TEWBDataSet OnLoadError event.

The Sender parameter is the TEWBDataSet instance that triggered the event and the ErrorMsg parameter is the specific network error message.

3.4 TEWBDataSetEvent Type

Unit: ewbdataset

```
TEWBDataSetEvent = function (Sender: TObject): Boolean of object
```

The TEWBDataSetEvent type is used by the TEWBDataSet BeforeOpen, BeforeClose, BeforeScroll, BeforeInsert, BeforeUpdate, BeforeSave, BeforeCancel, BeforeDelete, and BeforeLoad events.

The Sender parameter is the TEWBDataSet instance that triggered the event. Return True from the event to allow the applicable dataset functionality to continue, or False to prevent the functionality from occurring.

3.5 TEWBExecuteDataSetColumnCommandEvent Type

Unit: ewbdatasetadapter

```
TEWBExecuteDataSetColumnCommandEvent = procedure(Adapter:
      TEWBDataSetAdapter; const DatabaseName: String; const
      ColumnName: String; Params: TParams; var RowsAffected: Integer;
      var DataSet: TDataSet) of object;
```

The TEWBExecuteDataSetColumnCommandEvent type is used by the TEWBDataSetAdapter OnExecuteSelectColumn event to execute a **Select<Column>** dataset command when retrieving BLOB columns.

The DatabaseName parameter is the name of the database (optional), the ColumnName parameter is the name of the BLOB column that needs to be retrieved, and the Params parameter is the list of parameters (and values) to use for executing the dataset command.

Both the RowsAffected and DataSet variable parameters must be populated by any event handler for this event type.

3.6 TEWBExecuteDataSetCommandEvent Type

Unit: ewbdatasetadapter

```
TEWBExecuteDataSetCommandEvent = procedure(Adapter:
      TEWBDataSetAdapter; const DatabaseName: String; Params: TParams;
      var RowsAffected: Integer; var DataSet: TDataSet) of object
```

The TEWBExecuteDataSetCommandEvent type is used by the TEWBDataSetAdapter OnExecuteSelect event to execute a **Select** dataset command when retrieving column definitions and rows. It is also used by the TEWBDatabaseAdapter OnExecuteInsert, OnExecuteUpdate, and OnExecuteDelete events to execute **Insert**, **Update**, and **Delete** dataset commands when committing transactions.

The DatabaseName parameter is the name of the database (optional) and the Params parameter is the list of parameters (and values) to use for executing the dataset command.

The RowsAffected variable parameter must be populated by any event handler for this event type, but the DataSet variable parameter only needs to be populated with OnExecuteSelect event handlers.

3.7 TEWBGetDataSetAdapterEvent Type

Unit: ewbdatasetadapter

```
TEWBGetDataSetAdapterEvent = procedure(const DataSetName: String;
       var Adapter: TEWBDataSetAdapter) of object
```

The TEWBGetDataSetAdapterEvent type is used by the TEWBDatabaseAdapter OnGetDataSetAdapter event.

The DataSetName parameter specifies the name of the dataset that requires an adapter, and the Adapter variable parameter should be assigned a valid TEWBDataSetAdapter instance.

3.8 TEWBGetDataSetColumnCommandParamsEvent Type

Unit: ewbdatasetadapter

```
TEWBGetDataSetColumnCommandParamsEvent = procedure(Adapter:
      TEWBDataSetAdapter; const DatabaseName: String; const
      ColumnName: String; Params: TParams) of object
```

The TEWBGetDataSetColumnCommandParamsEvent type is used by the TEWBDataSetAdapter OnGetSelectColumnParams event to populate the parameters for a **Select<Column>** dataset command when retrieving BLOB columns.

The DatabaseName parameter is the name of the database (optional), the ColumnName parameter is the name of the BLOB column that needs to be retrieved, and the Params parameter is the list of parameters that needs to be populated for the dataset command.

3.9 TEWBGetDataSetCommandParamsEvent Type

Unit: ewbdatasetadapter

```
TEWBGetDataSetCommandParamsEvent = procedure(Adapter:
      TEWBDataSetAdapter; const DatabaseName: String; Params: TParams)
      of object
```

The TEWBGetDataSetCommandParamsEvent type is used by the TEWBDataSetAdapter OnGetSelectParams event to populate the parameters for a **Select** dataset command when retrieving column definitions and rows. It is also used by the TEWBDatabaseAdapter OnGetInsertParams, OnGetUpdateParams, and OnGetDeleteParams events to execute **Insert**, **Update**, and **Delete** dataset commands when committing transactions.

The DatabaseName parameter is the name of the database (optional) and the Params parameter is the list of parameters that needs to be populated for the dataset command.

3.10 TEWBGetDataSetsEvent Type

Unit: ewbdatasetadapter

```
TEWBGetDataSetsEvent = procedure(DataSets: TStrings) of object
```

This type is used by the TEWBDatabaseAdapter OnGetDataSets event.

The DataSets parameter specifies a TStrings instance to use for populating the list of available datasets.

3.11 TEWBHashType Type

Unit: ewbhttpmodule

```
TEWBHashType = (htMD5,htSHA1,htSHA256,htSHA512)
```

This type is used to represent the type of hash being computed with the TEWBWebServerRequest ComputeHash method.

Element	Description
htMD5	Specifies that the computed hash will be a 128-bit MD-5 (Message Digest) hash. **Warning** MD-5 hashes are cryptographic hashes, but are **not** secure and should only be used for purposes such as computing checksums for data.
htSHA1	Specifies that the computed hash will be a 160-bit SHA-1 (Secure Hash Algorithm) hash. **Warning** SHA-1 hashes are cryptographic hashes, but are **not** secure and should only be used for purposes such as computing checksums for data.
htSHA256	Specifies that the computed hash will be a 256-bit SHA-2 (Secure Hash Algorithm) hash.
htSHA512	Specifies that the computed hash will be a 512-bit SHA-2 (Secure Hash Algorithm) hash.

3.12 TEWBModuleExecuteEvent Type

Unit: ewbhttpmodule

```
TEWBModuleExecuteEvent = procedure (Request:
      TEWBWebServerRequest) of object
```

This type is used by the TEWBModule OnExecute event.

The Request parameter specifies the TEWBWebServerRequest instance that is being handled by the module.

3.13 TEWBServerRequestErrorEvent Type

Unit: ewbsrvrequest

```
TEWBServerRequestErrorEvent = procedure (Request:
      TEWBServerRequest; const ErrorMsg: String) of object
```

The TEWBServerRequestErrorEvent event type is used by the TEWBServerRequest OnError event to indicate when a server request could not be completed due to a network error.

The Request parameter indicates the server request that triggered the event and the ErrorMsg indicates the specific network error message.

3.14 TEWBServerRequestEvent Type

Unit: ewbsrvrequest

```
TEWBServerRequestEvent = procedure (Request: TEWBServerRequest)
      of object
```

The TEWBServerRequestEvent event type is used by the TEWBServerRequest OnComplete event to indicate when a server request is complete.

The Request parameter indicates the server request that triggered the event, and the StatusCode property of the server request can be examined to determine if the request completed successfully.

3.15 TEWBServerRequestProgressEvent Type

Unit: ewbsrvrequest

```
TEWBServerRequestProgressEvent = procedure (Request:
      TEWBServerRequest; const Current: Int64; const Total: Int64) of
      object
```

The TEWBServerRequestProgressEvent event type is used by the TEWBServerRequest OnProgress event to indicate the progress of a server request as it is executing.

The Request parameter indicates the server request that triggered the event, the Current parameter indicates the current number of bytes that have been received, and the Total parameter indicates the total number of bytes that will be received when the request completes successfully.

3.16 TEWBServerRequestType Type

Unit: ewbsrvrequest

```
TEWBServerRequestType = (rtSynchronous,rtAsynchronous)
```

The TEWBServerRequestType type is used by the TEWBServerRequest RequestType property to indicate whether the server request should execute synchronously (the default), or asynchronously.

If the server request executes synchronously, then the Execute method will wait for the request to completely execute before returning.

If the server request executes asynchronously, then the Execute method will not wait for the request to completely execute before returning.

Element	Description
rtAsynchronous	The server request should execute asynchronously.
rtSynchronous	The server request should execute synchronously.

3.17 TEWBServerSessionAuthenticateEvent Type

Unit: ewbsession

```
TEWBServerSessionAuthenticateEvent = procedure (Session:
      TEWBServerSession; var Continue: Boolean) of object
```

The TEWBServerSessionAuthenticateEvent event type is used by the TEWBServerSession OnAuthenticate event to allow the developer to provide custom authentication credentials and/or specify if authentication should proceed.

The Session parameter indicates the server session that triggered the event and the Continue parameter determines if the authentication should proceed. If the Continue property is set to False, then authentication will fail with an exception indicating that authentication was cancelled.

3.18 TEWBServerSessionEvent Type

Unit: ewbsession

```
TEWBServerSessionEvent = procedure (Session: TEWBServerSession)
      of object
```

The TEWBServerSessionEvent event type is used by the TEWBServerSession OnComplete event to indicate when a given session request, such as the request initiated by the TEWBServerSession Authenticate method, has completed.

The Session parameter indicates the server session that triggered the event.

3.19 TEWBServerSessionProgressEvent Type

Unit: ewbsession

```
TEWBServerSessionProgressEvent = procedure (Session:
      TEWBServerSession; const Current: Int64; const Total: Int64; var
      Aborted: Boolean) of object
```

The TEWBServerSessionProgressEvent event type is used by the TEWBServerSession OnProgress event to indicate the progress of a given session request, such as the request initiated by the TEWBServerSession Authenticate method, while it is executing. It also provides the developer the opportunity to abort the execution of the request.

The Session parameter indicates the server session that triggered the event, the Current parameter indicates the current number of bytes that have been received, and the Total parameter indicates the total number of bytes that will be received when the request completes successfully.

The Aborted variable parameter can be set to True to indicate that the request should be cancelled.

3.20 TEWBSortOption Type

Unit: ewbdataset

```
TEWBSortOption = (soCaseInsensitive)
```

This type is used to represent a sort option for the TEWBDataSet Sort method.

Element	Description
soCaseInsensitive	Any sort comparisons of string columns should be done in a case-insensitive manner.

3.21 TEWBSortOptions Type

Unit: ewbdataset

```
TEWBSortOptions = set of TEWBSortOption
```

This set type is used to represent the sort options for the TEWBDataSet Sort method.

3.22 TEWBWebServerRequestMethod Type

Unit: ewbhttpmodule

```
TEWBWebServerRequestMethod = (rmUnknown,rmGet,rmPost,rmHead,
      rmPut,rmDelete,rmPatch)
```

This type is used to represent the type of request in the TEWBWebServerRequest RequestMethod property.

Element	Description
rmDelete	The request is an HTTP DELETE request.
rmGet	The request is an HTTP GET request.
rmHead	The request is an HTTP HEAD request.
rmPatch	The request is an HTTP PATCH request.
rmPost	The request is an HTTP POST request.
rmPut	The request is an HTTP PUT request.
rmUnknown	The request is an unknown HTTP request (not used for native server modules - an error will occur before the native server module is called if the request type is unknown).

3.23 TEWBWebServerRequestMethods Type

Unit: ewbhttpmodule

```
TEWBWebServerRequestMethods = set of TEWBWebServerRequestMethod
```

This type is used to represent a set of request types and is used with the TEWBWebServerRequest CheckMethod method.