AMLayout Object Reference Guide

NetIQ AppManager

Version 6.0

Legal Notice

Contents

AMLayout Object reference

Chapter 1

AMLayout object reference

About AMLayout

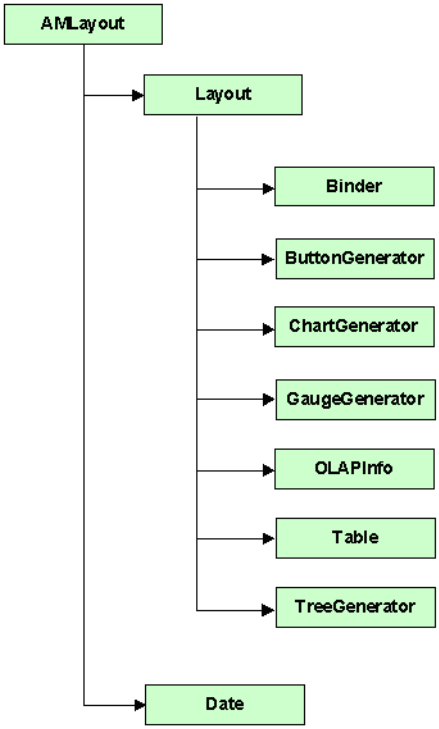
The AMLayout object is an OLE automation-based API for producing HTML pages. The HTML pages depend on style sheets for formatting.

AMLayout can render OLAP ADOMD cellsets into multi-dimensional tables.

AMLayout is used by the reporting engine, the charting tool, and the AppManager Web Operator Console. It offers a number of visualization components to aid in reporting design.

AMLayout object model

The following diagram illustrates the AMLayout object model:



Chapter 2

Object reference table

The following table summarizes the objects provided by AMLayout.

Object	Description
Layout object	The Layout object is the root of the layout hierarchy. It controls where a document is built and where the build directories are found.
Date object	The Date object is used to convert Universally Coordinated Time (UTC) to local time, and vise versa.
Binder object	The Binder object is used to build an XML tree. The XML tree can be passed to the TreeGenerator object to draw an HTML tree.
ButtonGenerator object	The ButtonGenerator object is used to draw a button on a document.
ChartGenerator object	The ChartGenerator object is used to render charts on the document.
GaugeGenerator object	The GaugeGenerator object is used to draw gauges and meters on the document.
OLAPInfo object	The OLAPInfo object is used to generate an MDX query against a cube, and render the results of the query to an OLAP table based on ADOMD cellsets.

Object	Description
Table object	The Table object is used to insert a table in a document. In addition to drawing the table, this object provides methods to parse XML data and render it in tabular form.
TreeGenerator object	The TreeGenerator object is used to draw an HTML tree based on the structure of the XML binder. It provides the methods of parsing an XML binder and then attaches an HTML tree.

Chapter 3

Layout object

The Layout object is the root of the layout hierarchy. It controls where a document is built and where the build directories are found. This object includes properties and methods for adding images and text, attaching style sheets and other external files, and for returning references to all other sub-objects (for example, Table, Binder, and ChartGenerator).

When you create an HTML-based report using this object, some of the pre-defined constants are created and written at the beginning of the report. For example, the header template is loaded from the file `rptTemplate.txt` (located in `..\reports\build_files`), and the text for the header is determined by values set by the Layout object.

The Layout object creates entries in the registry under `HKEY_LOCAL_MACHINE\SOFTWARE\NetIQ\Common\AMREPORT.`

These entries include:

- **`BaseOutputPath`** specifies the directory in which all reports are generated
- **`BuildFilesPath`** specifies the directory where all the common graphic and text files used to build reports are located
- **`HeaderTemplate`** specifies which header template is used for reports
- **`Footer`** specifies which footer template is used for reports
- **`SharedFilesPath`** specifies the path to all shared files used by a report (for example, cascading style sheets)
- Font name and size, text color and alignment

This section covers the following topics:

- Layout properties
- Layout methods

Layout properties

The following properties are available for the Layout object.

Property	Description
AppendMode	Sets or returns a flag that determines whether a report can be appended.
Author	Sets or returns the Author property of the report, a metadata property used by the binder when grouping and displaying reports.
Binder	Returns a reference to the Binder object.
Buffer	Returns the actual contents of the report.
BufferedMode	Sets or returns a value that causes the Layout object to cache the output in a buffer.
BuildFilesDirectory	Sets or returns the directory that contains the files used during report generation.
ButtonGenerator	Returns a reference to the ButtonGenerator object.
Category	Sets or returns the Category property of the report, a metadata property used by the binder when grouping and displaying reports.
CenterDocument	Sets or returns a flag that indicates whether the entire document should be centered.
ChartGenerator	Returns a reference to the ChartGenerator object.
Company	Sets or return the Company property of the report, a metadata property used by the binder when grouping and displaying reports.
Component	Sets or returns the Component property of the report, a metadata property used by the binder when grouping and displaying reports.
Custom1	Sets or returns the Custom1 property of the report, a metadata property used by the binder when grouping and displaying reports.

Property	Description
Custom2	Sets or returns the Custom2 property of the report, a metadata property used by the binder when grouping and displaying reports.
DataSource	Sets or returns the DataSource property of the report, a metadata property used by the binder when grouping and displaying reports.
DefaultSytleSheet	Sets or returns the name of the default style sheet.
Description	Sets or returns the Description property of the report, a metadata property used by the binder when grouping and displaying reports.
DocumentName	Sets or returns the name of the document.
EmbedStyleSheet	Sets or returns a flag that determines whether the style sheet is embedded in the HTML report.
EndDate	Sets or returns the EndDate property of the report, a metadata property used by the binder when grouping and displaying reports.
ExpireInXDays	Sets or returns the ExpireInXDays property of the report. This property specifies when a report should be automatically deleted.
GaugeGenerator	Returns a reference to the GaugeGenerator object.
GenerateEndTags	Sets or returns a flag that indicates whether the necessary HTML tags will be written to signal the end of the page.
GenerateHeaderTemplate	Sets or returns a flag that indicates whether an HTML header will be attached to the report.
GenerateIndex	Sets or returns a flag that indicates whether a report index file will be written.
HasData	Sets or returns the HasData property of the report, a metadata property used by the binder when grouping and displaying reports.
HeaderTemplate	Sets or returns the HeaderTemplate property of the report. This is the property that starts a document.
KeyWords	Sets or returns the KeyWords property of the report, a metadata property used by the binder when grouping and displaying reports.
OLAPInfo	Returns a reference to the OLAP object.

Property	Description
OutputDirectory	Sets or returns a string value that indicates the directory where all output is sent.
RootRegistryKey	Sets or returns the root registry key. Layout object uses the root registry key to find its settings.
SharedFilesDirectory	Sets or returns the directory that contains files that are shared by reports at view time (for example, style sheets and shared graphics).
StartDate	Sets or returns the StartDate property of the report, a metadata property used by the binder when grouping and displaying reports.
Subject	Sets or returns the Subject property of the report, a metadata property used by the binder when grouping and displaying reports.
Table	Returns a reference to the Table object.
Title	Sets or returns the Title property of the report, a metadata property used by the binder when grouping and displaying reports.
TreeGenerator	Returns a reference to the TreeGenerator object.

AppendMode

Sets or returns a flag that determines whether a report can be appended.

Example

```
Property AppendMode As Long
```

Parameters

- Long bAppendMode

Return values

- Long

Remarks

Sample code

```
Dim layoutObj
Set layoutobj = CreateObject(“AMLAYOUT.Layout”)
‘### Specify where to get/put files
layoutObj.OutputDirectory = “d:\reports\Layout”

layoutobj.AppendMode = TRUE
layoutobj.DocumentName = “AppendMode.html”
layoutobj.BeginReport

layoutobj.AddHeader “AppendMode”,“”
layoutobj.WriteLine “<br><p>Test Report for _
     AMLayout::AppendMOde”, TRUE

layoutobj.AddFooter “”
layoutobj.EndReport
```

Author

Sets or returns the Author property of the report, a metadata property used by the binder when grouping and displaying reports.

Example

```
Property Author As String
```

Sample code

```
Dim layoutObj
Set layoutobj = CreateObject(“AMLAYOUT.Layout”)
‘### Specify where to get/put files
layoutObj.OutputDirectory = “d:\reports\Layout”

layoutobj.Title = “Test Report for Property Layout::Author”
layoutobj.Company = “”
layoutobj.Author = “Kiran Jain”
layoutobj.Keywords = “LAYOUT”
layoutobj.Category = “AMLayout”
layoutobj.Description = “”

layoutobj.DocumentName = “Author.html”
layoutobj.BeginReport “”
```

```
layoutobj.AddHeader “Test Report for Property Layout::Author”
layoutobj.CenterDocument = FALSE
layoutobj.AddVerticalSpace = “style=’height:10pt”

layoutobj.AddFooter “”
layoutobj.EndReport
```

Parameters

- String strAuthor

Return values

- String

Remarks

Binder

Returns a reference to the Binder object.

Example

```
Property Binder As Object
```

Parameters

- Object pVal

Return values

- Object

Remarks

Buffer

Returns the actual contents of the report. This property only returns data if the BufferedMode property is set to TRUE.

Example

```
Property Buffer As String
```

Parameters

- String strBuffer

Return values

- String

Remarks

BufferedMode

Sets or returns a value that causes the Layout object to cache the output in a buffer.

Example

```
Property BufferedMode As Long
```

Parameters

- Long bBufferedMode

Return values

- Long

Remarks

BuildFilesDirectory

Sets or returns the directory that contains the files used during report generation. A build directory contains the core graphics and header templates that might be needed for a report.

Example

```
Property BuildFilesDirectory As String
```

Parameters

- String strBuildFilesDirectory

Return values

- String

Remarks

ButtonGenerator

Returns a reference to the ButtonGenerator object.

Example

```
Property ButtonGenerator As Ojbect
```

Parameters

Return values

- Object

Remarks

Category

Sets or returns the Category property of the report, a metadata property used by the binder when grouping and displaying reports.

Example

```
Property Category As String
```

Parameters

- String strCategory

Return values

- String

Remarks

See Author sample code.

CenterDocument

Sets or returns a flag that indicates whether the entire document should be centered.

Example

```
Property CenterDocument As Long
```

Parameters

- Long bCenterDocument

Return values

- Long

Remarks

See Author sample code.

ChartGenerator

Returns a reference to the ChartGenerator object.

Example

```
Property ChartGenerator As Object
```

Parameters

Return values

- Object

Remarks

Company

Sets or returns the Company property of the report, a metadata property used by the binder when grouping and displaying reports.

Example

```
Property Company As String
```

Parameters

- String strCompany

Return values

- String

Remarks

See Author sample code.

Component

Sets or returns the Component property of the report, a metadata property used by the binder when grouping and displaying reports.

Example

```
Property Component As String
```

Parameters

- String strComponent

Return values

- String

Remarks

Custom1

Sets or returns the Custom1 property of the report, a metadata property used by the binder when grouping and displaying reports.

Example

```
Property Custom1 As String
```

Parameters

- String strCustom1

Return values

- String

Remarks

Custom2

Sets or returns the Custom2 property of the report, a metadata property used by the binder when grouping and displaying reports.

Example

```
Property Custom2 As String
```

Parameters

- String strCustom2

Return values

- String

Remarks

DataSource

Sets or returns the DataSource property of the report, a metadata property used by the binder when grouping and displaying reports.

Example

```
Property DataSource As String
```

Parameters

- String strDataSource

Return values

- String

Remarks

DefaultSytleSheet

Sets or returns the name of the default style sheet. The style sheet must be located in the `Shared_Files` folder.

Example

```
Property DefaultStyleSheet As String
```

Parameters

- String strDefaultStyleSheet

Return values

- String

Remarks

Description

Sets or returns the Description property of the report, a metadata property used by the binder when grouping and displaying reports.

Example

```
Property Description As String
```

Parameters

- String strDescription

Return values

- String

Remarks

DocumentName

Sets or returns the name of the document.

Example

```
Property DocumentName As String
```

Parameters

- String strDocumentName

Return values

- String

Remarks

See Author example

EmbedStyleSheet

Sets or returns a flag that determines whether the style sheet is embedded in the HTML report.

Example

```
Property EmbeddedStyleSheet As Long
```

Parameters

- Long bEmbeddedStyleSheet

Return values

- Long

Remarks

EndDate

Sets or returns the EndDate property of the report, a metadata property used by the binder when grouping and displaying reports.

Example

```
Property EndDate As String
```

Parameters

- String strEndDate

Return values

- String

Remarks

ExpireInXDays

Sets or returns the ExpireInXDays property of the report. This property specifies when a report should be automatically deleted.

Example

```
Property ExpireInXDays As Integer
```

Parameters

- int nExpireInXDays

Return values

- int

Remarks

GaugeGenerator

Returns a reference to the GaugeGenerator object.

Example

```
Property GaugeGenerator As Object
```

Parameters

Return values

- Object

Remarks

GenerateEndTags

Sets or returns a flag that indicates whether the necessary HTML tags will be written to signal the end of the page.

Example

```
Property GenerateEndTags As Long
```

Parameters

- Long bGenerateEndTags

Return values

- Long

Remarks

GenerateHeaderTemplate

Sets or returns a flag that indicates whether an HTML header will be attached to the report.

Example

```
Property GenerateHeaderTemplate As Long
```

Parameters

- Long bGenerateHeaderTemplate

Return values

- Long

Remarks

GenerateIndex

Sets or returns a flag that indicates whether a report index file will be written.

Example

```
Property GenerateIndex As Long
```

Parameters

- Long bGenerateIndex

Return values

- Long

Remarks

HasData

Sets or returns the HasData property of the report, a metadata property used by the binder when grouping and displaying reports.

Example

```
Property HasData As Long
```

Parameters

- Long bHasData

Return values

- Long

Remarks

HeaderTemplate

Sets or returns the HeaderTemplate property of the report. This is the property that starts a document.

Example

```
Property HeaderTemplate As String
```

Parameters

- String strHeaderTemplate

Return values

- String

Remarks

KeyWords

Sets or returns the KeyWords property of the report, a metadata property used by the binder when grouping and displaying reports.

Example

```
Property KeyWords As String
```

Parameters

- String strKeyWords

Return values

- String

Remarks

OLAPInfo

Returns a reference to the OLAP object.

Example

```
Property OLAPInfo As Object
```

Parameters

Return values

- IDipatch

Remarks

OutputDirectory

Sets or returns a string value that indicates the directory where all output is sent.

Example

```
Property OutputDirectory As String
```

Parameters

- String strOutputDirectory

Return values

- String

Remarks

RootRegistryKey

Sets or returns the root registry key. Layout object uses the root registry key to find its settings.

Example

```
Property RootRegistryKey As String
```

Parameters

- String strRootRegistryKey

Return values

- String

Remarks

SharedFilesDirectory

Sets or returns the directory that contains files that are shared by reports at view time (for example, style sheets and shared graphics).

Example

```
Property SharedFilesDirectory As String
```

Parameters

- String strSharedFilesDirectory

Return values

- String

Remarks

StartDate

Sets or returns the StartDate property of the report, a metadata property used by the binder when grouping and displaying reports.

Example

```
Property StartDate As String
```

Parameters

- String strStartDate

Return values

- String

Remarks

Subject

Sets or returns the Subject property of the report, a metadata property used by the binder when grouping and displaying reports.

Example

```
Property Subject As String
```

Parameters

- String strSubject

Return values

- String

Remarks

Table

Returns a reference to the Table object.

Example

```
Property Table As Object
```

Parameters

Return values

- Object

Remarks

Title

Sets or returns the Title property of the report, a metadata property used by the binder when grouping and displaying reports.

Example

```
Property Title As String
```

Parameters

- String strTitle

Return values

- String

Remarks

TreeGenerator

Returns a reference to the TreeGenerator object.

Example

```
Property TreeGenerator As Object
```

Parameters

Return values

- Object

Remarks

Layout methods

The following methods are available for the AMLayout object.

Methods	Descriptions
AddFooter	Adds a graphical footer to the current location in the report.
AddHeader	Adds a graphical header to the current location in the report. The graphic is dynamically generated based on the parameter.
AddImage	Attaches an image to the current document.
AddSubHeader	Adds a graphic to the current location in the report to separate sections of the report. The graphic is dynamically generated based on the parameter.
AddText	Adds text to the current cell in a table.
AddVerticalSpace	Adds a vertical space to the current document. The vertical space is specified in the custom style sheet.
BeginHyperLink	Starts an HTML anchor section. Any elements between the start and end act as links.
BeginReport	Opens the report file.
CleanUpDirectory	Erases specific file types from the target directory.
CloseReport	Closes the report file.
CreateDirectory	Creates a new directory. This call supports creating multiple levels of directories.
DeriveRelativePath	Creates a relative path given two absolute paths.
EndHyperLink	Ends an HTML anchor section.
GetFolderName	Returns the name of the folder based on the folder type (Unique, Unique with prefix, or Specific).
GetSetting	Returns the value of a given setting. The settings for the Layout object are stored in the registry.
GetUniqueToken	Returns a unique string value. The unique string is a Globally Unique Identifier (GUID).
IncludeFile	Specifies the script library files that are to be included.
InsertFile	Inserts a file at the current location within the document.

Methods	Descriptions
InsertPageBreak	Inserts a CSS page break into the current document. The page break is only visible when the document is printed.
LogMessage	Appends a message to the report logfile. Logs are automatically created in the same directory as the report.
QualityBuildFile	Converts a file name into a fully qualified absolute path based on the specified build directory.
QualityOutputFile	Converts a file name into a fully qualified absolute path based on the specified output directory.
QualitySharedFile	Converts a file name into a fully qualified absolute path based on the specified shared directory.
WriteIndexFile	Writes a report index.
WriteLine	Writes a line of text in the document.

AddFooter

Adds a graphical footer to the current location in the report.

Example

```
Sub AddFooter(strFooterFileName As String)
```

Parameters

- String strFooterFileName

Return values

- void

Remarks

Sample code

```
Dim layoutObj
Set layoutobj = CreateObject(“AMLAYOUT.Layout”)

‘###Specify to get/put files
layoutObj.OutputDirectory = “d:\reports\Layout”

layoutObj.DocumentName = “AddFooter.html”
```

```
layoutObj.CenterDocument = TRUE
layoutObj.BeginReport “”

layoutObj.AddHeader “AddFooter Report”, “”
layoutObj.AddVerticalSpace “10pt”

layoutObj.AddFooter “Footer.txt”
layoutObj.EndReport
```

AddHeader

Adds a graphical header to the current location in the report. The graphic is dynamically generated based on the parameter.

Example

```
Sub AddHeader(strHeaderText As String, strStyle As String)
```

Parameters

- String strHeaderText
- String strStyle

Return values

- void

Remarks

See AddFooter sample code.

AddImage

Attaches an image to the current document.

Example

```
Sub AddImage(strImagePath As String, strStyle As String,
width As Long, height As Long, strHyperLink As String)
```

Parameters

- String strImagePath
- String strStyle
- int nWidth
- int nHeight
- String strHyperlink

Return values

- void

Remarks

Sample code

```
Dim layoutObj
Set layoutobj = CreateObject(“AMLAYOUT.Layout”)

‘###Specify to get/put files
layoutObj.OutputDirectory = “d:\reports\Layout”

layoutObj.DocumentName = “AddImage.html”

layoutObj.CenterDocument = TRUE
layoutObj.BeginReport “”

layoutObj.AddVerticalSpace “style=’height:10pt”
layoutObj.AddImage “D:\NetIQ\AppManager\Web\Shared_Files _
      \netiqlogo.gif”, “style=vertical-align:top”, _
      86,45, “”

layoutObj.EndReport
```

AddSubHeader

Adds a graphic to the current location in the report to separate sections of the report. The graphic is dynamically generated based on the parameter.

Example

```
Sub AddSubHeader(strHeaderText As String, strStyle As String)
```

Parameters

- String strHeaderText
- String strStyle

Return values

- void

Remarks

Sample code

```
Dim layoutObj
Set layoutobj = CreateObject(“AMLAYOUT.Layout”)

‘###Specify to get/put files
layoutObj.OutputDirectory = “d:\reports\Layout”

layoutObj.DocumentName = “AddSubHeader.html”

layoutObj.CenterDocument = TRUE
layoutObj.BeginReport “”

layoutObj.AddHeader “Layout AddSubHeader Method Report”,“”
layoutObj.AddVerticalSpapce “20pt”
layoutObj.AddSubHeader “AddSubHeader Method”,“”

layoutObj.EndReport
```

AddText

Adds text to the current cell in a table.

Example

```
Sub AddText(strText As String, strStyle As String)
```

Parameters

- String strText

- String strStyle

Return values

- void

Remarks

Sample code

```
Dim layoutObj
Set layoutobj = CreateObject(“AMLAYOUT.Layout”)

‘###Specify to get/put files
layoutObj.OutputDirectory = “d:\reports\Layout”

layoutObj.DocumentName = “AddText.html”

layoutObj.CenterDocument = FALSE
layoutObj.BeginReport “”

layoutObj.AddVerticalSpace “10pt”
layoutObj.AddSubHeader “These Are Header Styles”,“”

layoutObj.AddText “Header 1 example”, “class=’Header1”
layoutObj.AddText “Header 2 example”, “class=’Header2”
layoutObj.AddText “Header 3 example”, “class=’Header3”

layoutObj.AddFooter “”

layoutObj.EndReport
```

AddVerticalSpace

Adds a vertical space to the current document. The vertical space is specified in the custom style sheet.

Example

```
Sub AddVerticalSpace(strHeight As String)
```

Parameters

- String strHeight

Return values

- void

Remarks

See the AddText code sample.

BeginHyperLink

Starts an HTML anchor section. Any elements between the start and end act as links.

Example

```
Sub BeginHyperLink(strHyperLink As String, strStyle As
String)
```

Parameters

- String strHyperLink
- String strStyle

Return values

- void

Remarks

Sample code

```
Dim layoutObj
Set layoutobj = CreateObject("AMLAYOUT.Layout")

'###Specify to get/put files
layoutObj.OutputDirectory = "d:\reports\Layout"

'###Specify the file to be created - should only be 8.3 name
layoutObj.DocumentName = "BeginHyperLink.html"

layoutObj.CenterDocument = FALSE
layoutObj.BeginReport ""

layoutObj.AddHeader "Begin HyperLink Report",""
```

```
layoutObj.AddVerticalSpace “10pt”

layoutObj.BeginHyperLink “AddFooter”&“.html”, “”
layoutObj.AddText “AddFooter example”, “”
layoutObj.EndHyperLink

layoutObj.EndReport
```

BeginReport

Opens the report file.

Example

```
Sub BeginReport(strStyle As String)
```

Parameters

- String strStyle

Return values

- Long

Remarks

Sample code

```
Dim layoutObj
Set layoutobj = CreateObject(“AMLAYOUT.Layout”)

‘###Specify to get/put files
layoutObj.OutputDirectory = “d:\reports\Layout”

‘###Specify the file to be created - should only be 8.3 name
layoutObj.DocumentName = “BeginReport.html”

background = “D:\reports\kj\backgrounds\Paradise.jpg”
layoutObj.BeginReport “bgcolor=white background=” & _
      background

layoutObj.EndReport
```

CleanUpDirectory

Erases specific file types from the target directory. The file types are:

- GIF
- HTM
- HTML
- JPG
- JPEG
- LOG
- PNG
- XML

Example

```
Sub CleanUpDirectory(strDirectory As String)
```

Parameters

- String strDirectory

Return values

- Long

Remarks

Sample code

```
Dim layoutObj
Set layoutobj = CreateObject("AMLAYOUT.Layout")

dim strOutputDir
strOutputDir = "D:\Reports\Temp"
layoutObj.CreateDirectory strOutputDir
layoutObj.CleanUpDirectory strOutputDir
```

CloseReport

Closes the report file.

Example

```
Sub CloseReport()
```

Parameters

Return values

- void

Remarks

CreateDirectory

Creates a new directory. This call supports creating multiple levels of directories.

Example

```
Sub CreateDirectory(strDirectory As String)
```

Parameters

- String strDirectory

Return values

- Long

Remarks

See CleanUpDirectory sample code.

DeriveRelativePath

Creates a relative path given two absolute paths.

Example

```
Sub DeriveRelativePath(strFrom As String, strTo As String) As
String
```

Parameters

- String strFrom

- String strTo

Return values

- String

Remarks

EndHyperLink

Ends an HTML anchor section.

Example

```
Sub EndHyperLink()
```

Parameters

Return values

- void

Remarks

See BeginHyperLink sample code.

GetFolderName

Returns the name of the folder based on the folder type (Unique, Unique with prefix, or Specific).

Example

```
Sub GetFolderName(nFolder Type As Integer, strFolder As
String, strPrefix As String) As String
```

Parameters

- integer nFolder Type
- String strFolder
- String strPrefix

Return values

- String

Remarks

Sample code

```
Dim layoutObj
set layoutObj = CreateObject(“AMLAYOUT.Layout”)

const FOLDERTYPE_UNIQUEWPPREFIX= 1
Dim filename
fileName = “GetFolderName.html”

Dim folder, prefix, folder Type

folder = “folder1”
prefix = “prefix_”
folder Type = 1

WScript.Echo “Filename=” & filename
WScript.Echo layoutObj.GetFolderName(folder Type, folder,
prefix)
```

GetSetting

Returns the value of a given setting. The settings for the Layout object are stored in the registry.

Example

```
Sub GetSetting(strKey As String) As String
```

Parameters

- String strKey

Return values

- String

Remarks

GetUniqueToken

Returns a unique string value. The unique string is a Globally Unique Identifier (GUID).

Example

```
Sub GetUniqueToken() As String
```

Parameters

Return values

- String

Remarks

IncludeFile

Specifies the script library files that are to be included.

Example

```
Sub IncludeFile(strIncludeFile As String)
```

Parameters

- String strIncludeFile

Return values

- void

Remarks

Sample code

```
Dim layoutObj
set layoutObj = CreateObject("AMLAYOUT.Layout")

'###Specify where to get/put files
layoutObj.OutputDirectory = "d:\reports\Layout"
```

```
'###Specify the file to be created- should only be 8.3 name
layoutObj.DocumentName= "IncludeFile.html"

layoutObj.IncludeFile "D:\NetIQ\AppManager\Web\Build_Files _
      \UpdateGraphics.inc."

background = "D:\reports\kj\backgrounds\Paradise.jpg"
layoutObj.BeginReport "bgcolor=white background=" _
      & background

layoutObj.EndReport
```

InsertFile

Inserts a file at the current location within the document.

Example

```
Sub InsertFile(strFile As String)
```

Parameters

- String strFile

Return values

- void

Remarks

InsertPageBreak

Inserts a CSS page break into the current document. The page break is only visible when the document is printed.

Example

```
Sub InsertPageBreak()
```

Parameters

Return values

- void

Remarks

Sample code

```
Dim layoutObj
set layoutObj = CreateObject(“AMLAYOUT.Layout”)

‘###Specify where to get/put files
layoutObj.OutputDirectory = “d:\reports\Layout”

layoutObj.DocumentName= “InsertPageBreak.html”

layoutObj.CenterDocument = FALSE
laoutOjb.BeginReport “”

layoutObj.AddVerticalSpace “10pt”
layoutObj.AddSubHeader “Insert PageBreak Example”,“”

layoutObj.InsertPageBreak
layoutObj.AddText “Header 1 example”, “class=’Header1”
layoutObj.InsertPageBreak
layoutObj.AddText “Header 2 example”, “class=’Header2”
layoutObj.InsertPageBreak
layoutObj.AddText “Header 3 example”, “class=’Header3”

layoutObj.AddFooter “”

layoutObj.EndReport
```

LogMessage

Appends a message to the report logfile. Logs are automatically created in the same directory as the report.

Example

```
Sub LogMessage(strLogMessage As String)
```

Parameters

- String strLogMessage

Return values

- void

Remarks

Sample code

```
Dim layoutObj
set layoutObj = CreateObject(“AMLAYOUT.Layout”)

dim strOutputDir
strOutputDir = “d:\reports\temp”
layoutObj.CreateDirectory strOutputDir
layoutObj.CleanupDirectory strOutputDir
layoutObj.OutputDirectory = strOutputDir

layoutObj.LogMessage “this is before the begin report”
layoutObj.DocumentName = “LogMessage.html”

layoutObj.BeginReport “”
layoutObj.AddHeader “Log Message Report”,“”

layoutObj.CenterDocument = FALSE
layoutObj.AddVerticalSpace “10pt”

layoutObj.LogMessage “this is after the begin report”
layoutObj.EndReport
layoutObj.LogMessage “this is after the end report”
```

QualityBuildFile

Converts a file name into a fully qualified absolute path based on the specified build directory.

Example

```
Sub QualityBuildFile(strFile As String) As String
```

Parameters

- String strFile

Return values

- String

Remarks

QualityOutputFile

Converts a file name into a fully qualified absolute path based on the specified output directory.

Example

```
Sub QualityOutputFile(strFile As String) As String
```

Parameters

- String strFile

Return values

- String

Remarks

QualitySharedFile

Converts a file name into a fully qualified absolute path based on the specified shared directory.

Example

```
Sub QualitySharedFile(strFile As String) As String
```

Parameters

- String strFile

Return values

- String

Remarks

WriteIndexFile

Writes a report index.

Example

```
Sub WriteIndexFile()
```

Parameters

Return values

- void

Remarks

WriteLine

Writes a line of text in the document.

Example

```
Sub WriteLine(strLine As String, bCR As Long)
```

Parameters

- String strLine
- Long bCR

Return values

- void

Remarks

Sample code

```
Dim layoutObj
```

```
set layoutObj = CreateObject("AMLAYOUT.Layout")

'###Specify where to get/put the files
layoutObj.OutputDirectory = "d:\reports\Layout"

layoutObj.DocumentName = "WriteLine.html"
layoutObj.BeginReport
layoutObj.AppendMode = FALSE

layoutObj.AddHeader "Write Line",""
layoutObj.WriteLine "<b><p>Test Report for _
      AMLayout:WriteLine", TRUE

layoutObj.AddFooter
layoutObj.EndReport
```

Chapter 4

Binder object

The Binder object is used to build an XML tree. The XML tree can be passed to the TreeGenerator object to draw an HTML tree. The Binder object can generate the XML tree two ways:

- By hand
- By searching the reports in the Report folder

This section covers the following topics:

- Binder properties
- Binder methods

Binder properties

The following properties are available for the Binder object.

Property	Description
BinderOutputFile	Sets or returns an output file that stores the XML version of the binder (if GenerateBinderOutputFile is set to TRUE).
DefaultFolderIcon	Sets or returns a default icon for the folder in the tree.
DefaultItemIcon	Sets or returns a default icon for child item under the folder in the tree.
DOSBaseMapping	Sets or returns the DOS-based path names to be translated.
FindExpiredReports	Sets or returns a flag to find all reports whose expiration dates have passed.
GenerateBinderOutputFile	Sets or returns a flag to create an output file to show the XML part of the binder.
GroupingType	Sets or returns the type for grouping reports (for example, according to author or datasource).
ImagePath	Sets or returns the path to find the specified images (usually specified as a relative path).

Property	Description
OnlyAddReportsWithData	Sets or returns a flag that adds only reports with data to the tree.
RootImage	Sets or returns the image of the root item in the report tree.
RootText	Sets or returns the text of the root item in the report tree.
SearchPath	Sets or returns the path to use to search for reports.
TargetFrame	Sets or returns a target frame window for the binder.
URLBaseMapping	Sets or returns the URL-based path names to be translated.

BinderOutputFile

Sets or returns an output file that stores the XML version of the binder (if GenerateBinderOutputFile is set to TRUE).

Example

```
Property BinderOutputFile As String
```

Parameters

- String BinderOutputFile

Return values

- String

Remarks

See the GetXMLBinder code sample.

DefaultFolderIcon

Sets or returns a default icon for the folder in the tree.

Example

```
Property DefaultFolderIcon As String
```

Parameters

- String DefaultFolderIcon

Return values

- String

Remarks

See the GetXMLBinder code sample.

DefaultItemIcon

Sets or returns a default icon for child item under the folder in the tree.

Example

```
Property DefaultItemIcon As String
```

Parameters

- String DefaultItemIcon

Return values

- String

Remarks

See the GetXMLBinder code sample.

DOSBaseMapping

Sets or returns the DOS-based path names to be translated.

Example

```
Property DOSBaseMapping As String
```

Parameters

- String DOSBaseMapping

Return values

- String

Remarks

See the GetXMLBinder code sample.

FindExpiredReports

Sets or returns a flag to find all reports whose expiration dates have passed.

Example

```
Property FindExpiredReports As Long
```

Parameters

- long FindExpiredReports

Return values

- long

Remarks

See the GetXMLBinder code sample.

GenerateBinderOutputFile

Sets or returns a flag to create an output file to show the XML part of the binder.

Example

```
Property FindGenerateBinderOutputFile As Long
```

Parameters

- long GenerateBinderOutputFile

Return values

- long

Remarks

See the GetXMLBinder code sample.

GroupingType

Sets or returns the type for grouping reports (for example, according to author or datasource).

Example

```
Property GroupingType As Integer
```

Parameters

- integer GroupingType

Return values

- integer

Remarks

See the GetXMLBinder code sample.

ImagePath

Sets or returns the path to find the specified images (usually specified as a relative path).

Example

```
Property ImagePath As String
```

Parameters

- String ImagePath

Return values

- String

Remarks

See the GetXMLBinder code sample.

OnlyAddReportsWithData

Sets or returns a flag that adds only reports with data to the tree.

Example

```
Property OnlyAddReportsWithData As Long
```

Parameters

- Long OnlyAddReportsWithData

Return values

- Long

Remarks

See the GetXMLBinder code sample.

RootImage

Sets or returns the image of the root item in the report tree.

Example

```
Property RootImage As String
```

Parameters

- String RootImage

Return values

- String

Remarks

See the GetXMLBinder code sample.

RootText

Sets or returns the text of the root item in the report tree.

Example

```
Property RootText As String
```

Parameters

- String RootText

Return values

- String

Remarks

See the GetXMLBinder code sample.

SearchPath

Sets or returns the path to use to search for reports.

Example

```
Property SearchPath As String
```

Parameters

- String SearchPath

Return values

- String

Remarks

See the GetXMLBinder code sample.

TargetFrame

Sets or returns a target frame window for the binder.

Example

```
Property TargetFrame As String
```

Parameters

- String TargetFrame

Return values

- String

Remarks

See the GetXMLBinder code sample.

URLBaseMapping

Sets or returns the URL-based path names to be translated.

Example

```
Property URLBaseMapping As String
```

Parameters

- String URLBaseMapping

Return values

- String

Remarks

See the GetXMLBinder code sample.

Binder methods

The following methods are available for the Binder object.

Methods	Descriptions
BuildBinder	Builds the binder shown in the report tree.
ConvertPathToURL	Maps a file path to a hyperlink for a specific HTML file.
CreateFolder	Creates a folder.
CreateFolder2	Creates a folder specifying a custom image.
CreateItem	Creates an item under a specified folder.
CreateItem2	Creates an item under a specified folder. Can also specify an image.
GetRootObject	Returns a reference to the top-level node of the XML document.
GetXMLBinder	Creates and XMLDoc object from the binder, and can be saved as a file. An HTML tree can be generated from this XML binder.
Initialize	Initializes a Layout object.
Save	Saves the XML structure to a file on disk.
SetDefaultBinder	Sets a default binder attribute in the XML. This function is no longer necessary.
SetDefaultItemURL	Sets a default URL for an item in the XML. This function is no longer necessary.

BuildBinder

Builds the binder shown in the report tree.

Example

```
Sub BuildBinder()
```

Parameters

Return values

- void

Remarks

See the GetXMLBinder code sample.

ConvertPathToURL

Maps a file path to a hyperlink for a specific HTML file.

Example

```
Sub ConvertPathToURL(strFullPath As String) As String
```

Parameters

- String strFullPath

Return values

- String

Remarks

See the GetXMLBinder code sample.

CreateFolder

Creates a folder.

Example

```
Sub CreateFolder(parentNode As Object,strFolderName As
String,strURL As String) As Object
```

Parameters

- Object parentNode
- String strFolderName
- String strURL

Return values

- Object

Remarks

CreateFolder2

Creates a folder specifying a custom image.

Example

```
Sub CreateFolder2(parentNode As Object,strFolderName As
String,strImage As String,strURL As String) As Object
```

Parameters

- Object parentNode
- String strFolderName
- String strImage
- String strURL

Return values

- Object

Remarks

CreateItem

Creates an item under a specified folder.

Example

```
Sub CreateItem(parentNode As Object,strItemName As
String,strURL As String,nItemType As Integer) As Object
```

Parameters

- Object parentNode
- String strItemName
- String strURL
- Integer nItemType

Return values

- Object

Remarks

CreateItem2

Creates an item under a specified folder. Can also specify an image.

Example

```
Sub CreateItem2(parentNode As Object,strItemName As
String,strURL As String,strImage As String) As Object
```

Parameters

- Object parentNode
- String strItemName
- String strURL
- String strImage

Return values

- Object

Remarks

GetRootObject

Returns a reference to the top-level node of the XML document.

Example

```
Sub GetRootObject() As Object
```

Parameters

Return values

- Object

Remarks

GetXMLBinder

Creates and XMLDoc object from the binder, and can be saved as a file. An HTML tree can be generated from this XML binder.

Example

```
Sub GetXMLBinder() As Object
```

Parameters

Return values

- Object

Remarks

Sample code

```
Dim layoutObj
Set layoutObj = CreateObject("AMLAYOUT.Layout")

' ### Specify where to get/put the files
strOutDir = "d:\reports\Binder"
layoutObj.CreateDirectory strOutDir
layoutObj.CleanupDirectory strOutDir
layoutObj.OutputDirectory = strOutDir

layoutObj.DocumentName = "Binder.html"
layoutObj.GenerateEndTags = true
layoutObj.EmbedStyleSheet = true
layoutObj.GenerateEndTags = false

' ### Set the default Style Sheet
layoutObj.DefaultStyleSheet =
"D:\NetIQ\ReportCenter\web\Shared_Files\tree.css"

' ### Report GroupBY
Const GROUPBY_NONE= 0
Const GROUPBY_CATEGORY= 1
Const GROUPBY_CREATEDDATE_YEAR= 2
Const GROUPBY_CREATEDDATE_YEARTHENMONTH= 3
Const GROUPBY_DATASOURCE= 4
```

```
Const GROUPBY_COMPONENT= 5
Const GROUPBY_CUSTOM1= 6
Const GROUPBY_CUSTOM2= 7
Const GROUPBY_COMPANY= 8
Const GROUPBY_CREATEDDATE_YEARTHENMONTHTHENDAY= 9


' ### Dynamically build the binder
Set BinderObj = layoutObj.Binder

BinderObj.FindExpiredReports = false
BinderObj.OnlyAddReportsWithData = true
BinderObj.GenerateBinderOutputFile = true
BinderObj.BinderOutputFile =
"d:\reports\Binder\BinderOutputFile.txt"

BinderObj.DefaultFolderIcon =
"D:\NetIQ\ReportCenter\web\Images\tree\foldericon.gif"
BinderObj.DefaultItemIcon =
"D:\NetIQ\ReportCenter\web\Images\tree\htmlicon.gif"
BinderObj.RootImage =
"D:\NetIQ\ReportCenter\web\Images\tree\topopen.gif"
BinderObj.TargetFrame = "main"
BinderObj.ImagePath =
"D:\NetIQ\ReportCenter\web\Images\tree\"
BinderObj.GroupingType = 4
BinderObj.RootText = "Reports"

' ### Specifies where to search for reports
BinderObj.SearchPath = "D:\NetIQ\ReportCenter\web\Report"

' ### Information required to translate pathnames
BinderObj.DOSBaseMapping =
"D:\NetIQ\ReportCenter\web\Report"
BinderObj.URLBaseMapping = "http://kiranj/ReportCenter/
Report"

' ### URL mapping function
Dim strConvertedPath
strConvertedPath =
BinderObj.ConvertPathToURL("D:\NetIQ\ReportCenter\web\Report
\ComputerMembership\default.htm")

' ### Dynamically build the binder
BinderObj.BuildBinder
```

```
Set xmlDOC = BinderObj.GetXMLBinder()
xmlDOC.Save "d:\reports\Binder\BinderTest.xml"

' ### Create the HTML tree (based on the XML binder)
layoutobj.BeginReport ""
Set TreeObject = layoutObj.TreeGenerator
TreeObject.ParseXMLBinder(xmlDOC)
TreeObject.Attach
layoutObj.EndReport

Set xmlDOC = Nothing
```

Initialize

Initializes a Layout object.

Example

```
Sub Initialize(layoutObj As Object)
```

Parameters

- Object layoutObj

Return values

- Object

Remarks

Save

Saves the XML structure to a file on disk.

Example

```
Sub Save(strFilename As String)
```

Parameters

- String strFilename

Return values

- void

Remarks

SetDefaultBinder

Sets a default binder attribute in the XML. This function is no longer necessary.

Example

```
Sub SetDefaultBinder(strURL As String)
```

Parameters

- String strURL

Return values

- void

Remarks

SetDefaultItemURL

Sets a default URL for an item in the XML. This function is no longer necessary.

Example

```
Sub SetDefaultItemURL(strURL As String)
```

Parameters

- String strURL

Return values

- void

Remarks

Chapter 5

Date object

The Date object is used to convert Universally Coordinated Time (UTC) to local time, and vise versa.

This section covers the following topics:

- Date methods

Date methods

The following methods are available for the Date object.

Methods	Descriptions
DateToUTC	Converts local time to UTC.
UTCToDate	Converts UTC time to local time.

DateToUTC

Converts local time to UTC.

Example

```
Sub DateToUTC(dtValue As Date) As Long
```

Parameters

- Date dtValue

Return values

- long

Remarks

Sample code

```
Dim layoutObj
Set layoutobj = CreateObject(“AMLAYOUT.Layout”)

‘###Specify where to get/put files
strOutDir = “d:\reports\table”
layoutObj.CreateDirectory strOutDir
layoutObj.CleanupDirectory strOutDir
layoutObj.OutputDirectory = strOutDir

layoutObj.DocumentName = “Date.html”
layoutObj.BeginReport “”
layoutObj.AppendMode = FALSE

Dim dateObj, dtemp, dt, tm, UDate, UTC, dUTC
Set dateObj = CreateObject(“AMLayout.Date”)

dUTC = 993247564
dtemp = dateObj.UTCToDate(dUTC)
dt = FormatDateTime(dtemp, vbShortDate)
tm = FormatDateTime(dtemp, vbShortTime)

Udate = dt + “” + tm
Dim MyVar, sDate, sUTC
MyVar = Now
UTC = dateObj.DateToUtc(MyVar)

layoutObj.AddHeader “Example of Date Object Methods”,“”

layoutObj.WriteLine“<b><p>The UTCToDate of” & “ ‘” & _
      dUTC & “‘ ” & “is” & “<b>” & UTC & UDate & “<b>”, TRUE
layoutObj.WriteLine“<b><p>The DateToUTC of” & Now & “is” _
      & “ ‘” & “<b>” & UTC & “</b>” & “‘ ”, TRUE
layoutObj.EndReport
```

UTCToDate

Converts UTC time to local time.

Example

```
Sub UTCToDate(dtValue As 8 byte real) As Date
```

Parameters

- double utcValue

Return values

- Date

Remarks

See the DateToUTC sample code.

Chapter 6

ChartGenerator object

The ChartGenerator object is used to render charts on the document.

This section covers the following topics:

- ChartGenerator properties
- ChartGenerator methods

ChartGenerator properties

The following properties are available for the ChartGenerator object.

Property	Description
ChartTitle	Sets or returns the title of the chart.
ColorScheme	Sets or returns the color scheme of the chart.
DataLabelAngle	Sets or returns the angle (relative to the chart) of data labels.
DecimalPrecision	Sets or returns the number of decimal places to display (for example, 3 would display 0.000).
DemoMode	Sets or returns the Boolean value to display demo data in the chart.
ViewMode	Sets or returns the mode of current view of the chart control.
YAxisAutoRange	Sets or returns the Boolean value that enables the chart to set Y-axis maximum and minimum ranges automatically, based on the values of data points in the chart.
YAxisMaxRange	Sets or returns the maximum range for the Y-axis scale that a chart can plot. Any data points having higher values than YaxisMaxRange are clipped.

ChartTitle

Sets or returns the title of the chart.

Example

```
Property ChartTitle As String
```

Parameters

- String ChartTitle

Return values

- String

Remarks

ColorScheme

Sets or returns the color scheme of the chart.

Example

```
Property ColorScheme As String
```

Parameters

- String ColorScheme

Return values

- String

Remarks

DataLabelAngle

Sets or returns the angle (relative to the chart) of data labels.

Example

```
Property DataLabelAngle As Integer
```

Parameters

- Integer DataLabelAngle

Return values

- Integer

Remarks

DecimalPrecision

Sets or returns the number of decimal places to display (for example, 3 would display 0.000).

Example

```
Property DecimalPrecision As Integer
```

Parameters

- Integer DecimalPrecision

Return values

- Integer

Remarks

DemoMode

Sets or returns the Boolean value to display demo data in the chart.

Example

```
Property DemoMode As Long
```

Parameters

- Long DemoMode

Return values

- Long

Remarks

ViewMode

Sets or returns the mode of current view of the chart control. Possible values are:

- 0 = Perspective (approximately 35mm projection)
- 1 = Orthographic (parallel projection)

Example

```
Property ViewMode As Integer
```

Parameters

- Integer ViewMode

Return values

- Integer

Remarks

YAxisAutoRange

Sets or returns the Boolean value that enables the chart to set Y-axis maximum and minimum ranges automatically, based on the values of data points in the chart.

Example

```
Property YAxisAutoRange As Long
```

Parameters

- Long YAxisAutoRange

Return values

- Long

Remarks

YAxisMaxRange

Sets or returns the maximum range for the Y-axis scale that a chart can plot. Any data points having higher values than YaxisMaxRange are clipped.

Example

```
Property YAxisMaxRange As 8 byte real
```

Parameters

- Double YAxisMaxRange

Return values

- Double

Remarks

ChartGenerator methods

The following methods are available for the ChartGenerator object.

Methods	Descriptions
AddChartHighThreshold	Adds a high threshold to the chart, with the specified value and color.
AddData	Adds data dValue to a series specified in nSeriesIndex.
AddSeriesHighThreshold	Adds a high threshold to the series, with the specified value and color.
AttachAllImages	Saves large amounts of data as separate charts.
AttachImage	Saves one image of the chart to disk, and attaches it to the HTML document.
DeleteAllSeries	Deletes all series in the chart.
DeleteChartHighThresholds	Deletes all of the high thresholds from a chart.
DeleteSeries	Deletes a specified series from a chart.

Methods	Descriptions
EnableScrolling	Enables/disables scrolling in the chart pane. Scrolling is useful for large amounts of data (hundreds or thousands of points).
EstablishSeries	Add a series to the chart.
EstablishSeriesADO_DSN	Add a series of time-based data to the chart.
EstablishSeriesADO_RS	Add a series of time-based data to the chart.
GetBottomYAxisLabel	Sub GetBottomYAxisLabel() As String
GetChartAggregation	Returns a string representing the aggregation interval.
GetChartStyle	Returns the style of a chart.
GetSavedImageHeight	Returns the height (in pixels) of the last image saved using SaveImage, SaveAllImages, AttachImage, or AttachAllImages.
GetSavedImageWidth	Returns the width (in pixels) of the last image saved using SaveImage, SaveAllImages, AttachImage, or AttachAllImages.
GetSeriesCount	Gets the number of series in a chart.
GetTopYAxisLabel	Gets the string label attached to the top of Y-axis in a chart.
GetYAxisLabel	Gets the string label attached to the Y-axis.
Initialize	Initializes a Chart object.
RenderOLAP	Returns OLAP charts in an HTML file.
RenderScene	Instructs OpenGL to render the chart.
SaveAllImages	Saves large amounts of data as separate charts.
SaveImage	Saves the chart as an image file.
SetBottomYAxisLabel	Sets the label at the bottom of the Y-axis with the parameter.
SetChartAggregation	Sets the aggregation interval for the chart.
SetChartStyle	Sets the style of a chart.
SetChartStyleArea	Sets chart's style to Area.

Methods	Descriptions
SetChartStyleAreaStacked	Sets chart's style to AreaStacked.
SetChartStyleBar	Sets chart's style to Bar.
SetChartStyleCylinder	Sets chart's style to Cylinder.
SetChartStylePie	Sets chart's style to Pie.
SetChartStylePoint	Sets chart's style to Point.
SetChartStyleRibbon	Sets chart's style to Ribbon.
SetChartTransparency	Enables/disables the transparency of all series in the chart. When enabled, each series is drawn with a translucent quality.
SetDataAtDATE	Given a DATE, value of datapoint, and an index to a series, this method adds a data point to the series at the specified DATE (date and time).
SetDataLabel	Inserts a data label in the label array. Data labels appear along the X-Axis when using a chart with non-time-based data (zero interval).
SetDataAtTime	Given UTC time, value of datapoint, and an index to a series, this method adds the datapoint to a series.
SetDataAtTimeEx	Given date, value of datapoint, and an index to a series, this method creates a time stamp and calls SetDataAtTime to add the datapoint to a series.
SetRenderSize	Sets the rendering size (in pixels) of the control. Output is generated in the given dimensions. Usually used in conjunction with the SaveImage method, primarily with AMLayout and AMChartLite, and rarely with AMChart.
SetRotation	Sets the camera angle.
SetSeriesStyle	Sets the style of a given series.
SetTopYAxisLabel	Sets the label attached to the top of the Y-axis.
SetYAxisLabel	Sets the label for the Y-axis.
SortSeriesByLegend	Sorts all series in the chart from front to back (either ascending or descending) based upon the text of the legend.
SortSeriesByValue	Sorts all series in the chart from front to back (either ascending or descending) based upon the most recent value in each series.

Methods	Descriptions
Uninitialize	Uninitializes a Chart object.
UpdateChart	Updates the entire chart (rebuild all series) and render the scene.
ZoomIn	Zooms in on the chart.
ZoomNormal	Sets the zoom to the default.
ZoomOut	Zooms out from the chart.

AddChartHighThreshold

Adds a high threshold to the chart, with the specified value and color.

Example

```
Sub AddChartHighThreshold(bShowThresholdLevel As
Long,dblThresholdValue As 8 byte real,nRed As Integer,nGreen
As Integer,nBlue As Integer)
```

Parameters

- Long bShowThresholdLevel
- Double dblThresholdValue
- Integer nRed
- Integer nGreen
- Integer nBlue

Return values

- void

Remarks

AddData

Adds data dValue to a series specified in nSeriesIndex. Set bRebuildSeries TRUE to rebuild OpenGL display lists. AddData is

used for data that is not time-based, i.e. interval of zero. See the EstablishSeries method.

Example

```
Sub AddData(nSeriesIndex As Integer,dValue As 8 byte
real,bRebuildSeries As Long)
```

Parameters

- Integer nSeriesIndex
- Double dValue
- Long bRebuildSeries

Return values

- void

Remarks

AddSeriesHighThreshold

Adds a high threshold to the series, with the specified value and color.

Example

```
Sub AddSeriesHighThreshold(nSeries As
Integer,bShowThresholdLevel As Long,dblThresholdValue As 8
byte real,nRed As Integer,nGreen As Integer,nBlue As Integer)
```

Parameters

- Integer nSeries
- Long bShowThresholdLevel
- Double dblThresholdValue
- Integer nRed
- Integer nGreen
- Integer nBlue

Return values

- void

Remarks

AttachAllImages

Saves large amounts of data as separate charts. It pages through the data in the chart and saves each page as a unique chart image. Each image is saved with an automatic numbering method starting with 00000, and each image is automatically attached (included in the HTML document).

Example

```
Sub AttachAllImages(strBaseImage As String,strStyle As
String,strHyperLink As String,bLegend As Long,bBorder As
Long) As Long
```

Parameters

- String strBaseImage
- String strStyle
- String strHyperLink
- Long bLegend
- Long bBorder

Return values

- Long

Remarks

AttachImage

Saves one image of the chart to disk, and attaches it to the HTML document.

Example

```
Sub AttachImage(strImage As String,strStyle As
String,strHyperLink As String,bLegend As Long,bBorder As
Long)
```

Parameters

- String strImage
- String strStyle
- String strHyperLink
- Long bLegend
- Long bBorder

Return values

- void

Remarks

DeleteAllSeries

Deletes all series in the chart.

Example

```
Sub DeleteAllSeries()
```

Parameters

Return values

- void

Remarks

DeleteChartHighThresholds

Deletes all of the high thresholds from a chart.

Example

```
Sub DeleteChartHighThresholds()
```

Parameters

Return values

- void

Remarks

DeleteSeries

Deletes a specified series from a chart.

Example

```
Sub DeleteSeries(nSeriesIndex As Integer)
```

Parameters

- Integer nSeriesIndex

Return values

- void

Remarks

EnableScrolling

Enables/disables scrolling in the chart pane. Scrolling is useful for large amounts of data (hundreds or thousands of points). When scrolling is disabled, the data is all fit within the chart window (sometimes referred to as "Fit Data To Window").

Example

```
Sub EnableScrolling(bEnable As Long)
```

Parameters

- Long bEnable

Return values

- void

Remarks

EstablishSeries

Add a series to the chart. The legend for the series is specified in strName. The expected time interval (in seconds) between each point should be specified in lInterval.

Two options are possible:

- Zero Interval: Your chart will not have a date/time associated with each point. You need to add your points to this series using the AddData method. Finally, after (or before) adding your series and data to the chart, you need to use SetDataLabel to specify your X-Axis labels. The chart will not draw any data until the data labels have been specified.
- Nonzero Interval: Your chart will have a data/time associated with each point. For example: Your application gathers a new data point every 5 minutes and records the time at which the point was taken. You would specify an lInterval of 300 seconds. Each series can have a different interval (as long as the interval is > 0). The chart will examine the intervals of all series and calculate a common interval so that many series with different intervals can be plotted correctly in the same chart. You would use SetDataAtTime or SetDataAtDATE to add your points at a specified time.

Example

```
Sub EstablishSeries(strName As String,lInterval As Long) As
Integer
```

Parameters

- String strName
- Long lInterval

Return values

- Integer

Remarks

EstablishSeriesADO_DSN

Add a series of time-based data to the chart. The legend for the series is specified in strName. A SQL statement is specified in strSQL. The DSN is specified in strDSN. The expected time interval (in seconds) between each point should be specified in lInterval. The mininmum interval is 1 second.

For example, your application gathers a new data point every 5 minutes and records the time at which the point was taken. You would specify an lInterval of 300 seconds. Each series can have a different interval (as long as the interval is > 0). The chart will examine the intervals of all series and calculate a common interval so that many series with different intervals can be plotted correctly in the same chart.

There is no need to call SetDataAtTime or SetDataAtDATE because the chart will bind to the ADO recordset and fetch the data points automatically as needed. The database table in which your data exists should have time and value columns. The time column should contain a date/time value in UTC (long) format.

Your SELECT statement must specify the time first, then the value, and the order of the data must be descending from newest to oldest. You must use the "ORDER BY x DESC" clause.

Example

```
Sub EstablishSeriesADO_DSN(strName As String,strSQL As
String,strDSN As String,lInterval As Long) As Integer
```

Parameters

- String strName
- String strSQL

- String strDSN
- Long lInterval

Return values

- Integer

Remarks

EstablishSeriesADO_RS

Add a series of time-based data to the chart. The legend for the series is specified in strName. A SQL statement is specified in strSQL. The DSN is specified in strDSN. The expected time interval (in seconds) between each point should be specified in lInterval. The mininmum interval is 1 second.

For example, your application gathers a new data point every 5 minutes and records the time at which the point was taken. You would specify an lInterval of 300 seconds. Each series can have a different interval (as long as the interval is > 0). The chart will examine the intervals of all series and calculate a common interval so that many series with different intervals can be plotted correctly in the same chart.

There is no need to call SetDataAtTime or SetDataAtDATE because the chart will bind to the ADO recordset and fetch the data points automatically as needed. The database table in which your data exists should have time and value columns. The time column should contain a date/time value in UTC (long) format.

Your SELECT statement must specify the time first, then the value, and the order of the data must be descending from newest to oldest. You must use the "ORDER BY x DESC" clause.

Example

```
Sub EstablishSeriesADO_RS(strLegend As String,lInterval As
Long,pADORecordset As Object) As Integer
```

Parameters

- String strLegend
- Long lInterval
- Object pADORecordset

Return values

- Integer

Remarks

GetBottomYAxisLabel

Returns a string label at the bottom of the Y-axis in a chart.

Example

```
Sub GetBottomYAxisLabel() As String
```

Parameters

Return values

- String

Remarks

GetChartAggregation

Returns a string representing the aggregation interval. Possible values are:

- None
- 5 Minutes
- 10 Minutes
- 15 Minutes
- 30 Minutes
- 1 Hour

- 2 Hours
- 4 Hours
- 8 Hours
- 12 Hours
- 1 Day
- 1 Week

Example

```
Sub GetChartAggregation() As String
```

Parameters

Return values

- String

Remarks

GetChartStyle

Returns the style of a chart. Possible styles are:

- -1 = Mixed (read-only, indicates all series are not the same style)
- 0 = Area
- 1 = Ribbon
- 2 = Line
- 3 = Bar
- 4 = Cylinder
- 5 = Area Stacked
- 6 = Pie

Example

```
Sub GetChartStyle() As Integer
```

Parameters

Return values

- Integer

Remarks

GetSavedImageHeight

Returns the height (in pixels) of the last image saved using SaveImage, SaveAllImages, AttachImage, or AttachAllImages.

Example

```
Sub GetSavedImageHeight() As signed machine int
```

Parameters

Return values

- Integer

Remarks

GetSavedImageWidth

Returns the width (in pixels) of the last image saved using SaveImage, SaveAllImages, AttachImage, or AttachAllImages.

Example

```
Sub GetSavedImageWidth() As signed machine int
```

Parameters

Return values

- Integer

Remarks

GetSeriesCount

Gets the number of series in a chart.

Example

```
Sub GetSeriesCount() As Integer
```

Parameters

Return values

- Integer

Remarks

GetTopYAxisLabel

Gets the string label attached to the top of Y-axis in a chart.

Example

```
Sub GetTopYAxisLabel() As String
```

Parameters

Return values

- String

Remarks

GetYAxisLabel

Gets the string label attached to the Y-axis.

Example

```
Sub GetYAxisLabel() As String
```

Parameters

Return values

- String

Remarks

Initialize

Initializes a Chart object.

Example

```
Sub Initialize(pDispatch As Object)
```

Parameters

- Object pDispatch

Return values

- void

Remarks

RebuildScene

Rebuilds the chart based on current information, but does not display any changes. Sometimes used in conjunction with RenderScene.

Example

```
Sub RebuildScene()
```

Parameters

Return values

- void

Remarks

RenderOLAP

Returns OLAP charts in an HTML file.

Example

```
Sub RenderOLAP(cellset As Object,strStyle As
String,strHyperLink As String,bLegend As Long,bBorder As
Long,bTransparency As Long,bShowThresholdLevel As
Long,dblThresholdValue As 8 byte real,nRed As Integer,nGreen
As Integer,nBlue As Integer)
```

Parameters

- Object cellset
- String strStyle
- String strHyperLink
- Long bLegend
- Long bBorder
- Long bTransparency
- Long bShowThresholdLevel
- Double dblThresholdValue
- Integer nRed
- Integer nGreen
- Integer nBlue

Return values

- void

Remarks

RenderScene

Instructs OpenGL to render the chart.

Sometimes used in conjunction with UpdateChart prior to calling RenderScene.

Example

```
Sub RenderScene()
```

Parameters

Return values

- void

Remarks

SaveAllImages

Saves large amounts of data as separate charts. It pages through the data in the chart and saves each page as a unique chart image. Each image is saved with an automatic numbering method starting with 00000. Note that each image is simply saved to disk, not automatically attached to the HTML document as with AttachAllImages.

Example

```
Sub SaveAllImages(strBaseImage As String,bLegend As
Long,bBorder As Long) As Long
```

Parameters

- String strBaseImage
- Long bLegend
- Long bBorder

Return values

- Long

Remarks

SaveImage

Saves the chart as an image file. The only supported format is aPortable Network Graphics (PNG) file.

Example

```
Sub SaveImage(strImage As String,bLegend As Long,bBorder As
Long)
```

Parameters

- String strImage
- Long bLegend
- Long bBorder

Return values

- void

Remarks

SetBottomYAxisLabel

Sets the label at the bottom of the Y-axis with the parameter.

Example

```
Sub SetBottomYAxisLabel(strLabel As String)
```

Parameters

- String strLabel

Return values

- void

Remarks

SetChartAggregation

Sets the aggregation interval for the chart.

Example

```
Sub SetChartAggregation(strAggregation As String)
```

Parameters

- String strAggregation

Return values

- void

Remarks

SetChartStyle

Sets the style of a chart. Possible styles are:

- -1 = Mixed (read-only, indicates all series are not the same style)
- 0 = Area
- 1 = Ribbon
- 2 = Line
- 3 = Bar
- 4 = Cylinder
- 5 = Area Stacked
- 6 = Pie

Example

```
Sub SetChartStyle(newVal As Integer)
```

Parameters

- Integer newVal

Return values

- void

Remarks

SetChartStyleArea

Sets chart's style to Area.

Example

```
Sub SetChartStyleArea()
```

Parameters

Return values

- void

Remarks

SetChartStyleAreaStacked

Sets chart's style to AreaStacked.

Example

```
Sub SetChartStyleAreaStacked()
```

Parameters

Return values

- void

Remarks

SetChartStyleBar

Sets chart's style to Bar.

Example

```
Sub SetChartStyleBar()
```

Parameters

Return values

- void

Remarks

SetChartStyleCylinder

Sets chart's style to Cylinder.

Example

```
Sub SetChartStyleCylinder()
```

Parameters

Return values

- void

Remarks

SetChartStylePie

Sets chart's style to Pie.

Example

```
Sub SetChartStylePie()
```

Parameters

Return values

- void

Remarks

SetChartStylePoint

Sets chart's style to Point.

Example

```
Sub SetChartStylePoint()
```

Parameters

Return values

- void

Remarks

SetChartStyleRibbon

Sets chart's style to Ribbon.

Example

```
Sub SetChartStyleRibbon()
```

Parameters

Return values

- void

Remarks

SetChartTransparency

Enables/disables the transparency of all series in the chart. When enabled, each series is drawn with a translucent quality.

Example

```
Sub SetChartTransparency(bTransparent As Long)
```

Parameters

- Long bTransparent

Return values

- void

Remarks

SetDataAtDATE

Given a DATE, value of datapoint, and an index to a series, this method adds a data point to the series at the specified DATE (date and time).

Example

```
Sub SetDataAtDATE(nSeriesIndex As signed machine int,dblValue
As 8 byte real,dtTimeStamp As Date,bRebuildSeries As Long) As
Long
```

Parameters

- Integer nSeriesIndex
- Double dblValue
- Date dtTimeStamp
- Long bRebuildSeries

Return values

- Long

Remarks

SetDataLabel

Inserts a data label in the label array. Data labels appear along the X-Axis when using a chart with non-time-based data (zero interval). See EstablishSeries for more information.

Example

```
Sub SetDataLabel(lDataIndex As Long,strLabel As String) As
Long
```

Parameters

- Long lDataIndex
- String strLabel

Return values

- Long

Remarks

SetDataAtTime

Given UTC time, value of datapoint, and an index to a series, this method adds the datapoint to a series.

Example

```
Sub SetDateAtTime(nSeriesIndex As Integer,dblValue As 8 byte
real,lTimeStamp As Long,bRebuildSeries As Long) As Long
```

Parameters

- Integer nSeriesIndex
- Double dblValue
- Long lTimeStamp
- Long bRebuildSeries

Return values

- Long

Remarks

SetDataAtTimeEx

Given date, value of datapoint, and an index to a series, this method creates a time stamp and calls SetDataAtTime to add the datapoint to a series.

Example

```
Sub SetDateAtTimeEx(nSeriesIndex As Integer,dblValue As 8
byte real,lYear As Long,nMonth As Integer,nDay As
Integer,nHour As Integer,nMinute As Integer,nSecond As
Integer,bRebuildSeries As Long) As Long
```

Parameters

- Integer nSeriesIndex
- Double dblValue
- Long lYear
- Integer nMonth
- Integer nDay
- Integer nHour
- Integer nMinute
- Integer nSecond
- Long bRebuildSeries

Return values

- Long

Remarks

SetRenderSize

Sets the rendering size (in pixels) of the control. Output is generated in the given dimensions. Usually used in conjunction with the

SaveImage method, primarily with AMLayout and AMChartLite, and rarely with AMChart.

Example

```
Sub SetRenderSize(lXSize As Long,lYSize As Long)
```

Parameters

- Long lXSize
- Long lYSize

Return values

- void

Remarks

SetRotation

Sets the camera angle.

Example

```
Sub SetRotation(nXDegrees As Integer,nYDegrees As Integer)
```

Parameters

- Integer nXDegrees
- Integer nYDegrees

Return values

- void

Remarks

SetSeriesStyle

Sets the style of a given series. Possible styles are:

- 0 = Area

- 1 = Ribbon
- 2 = Line
- 3 = Bar
- 4 = Cylinder
- 5 = Area Stacked
- 6 = Pie

Example

```
Sub SetSeriesStyle(nSeries As signed machine int,nStyle As
signed machine int)
```

Parameters

- Integer nSeries
- Integer nStyle

Return values

- void

Remarks

SetTopYAxisLabel

Sets the label attached to the top of the Y-axis.

Example

```
Sub SetTopYAxisLabel(strLabel As String)
```

Parameters

- String strLabel

Return values

- void

Remarks

SetYAxisLabel

Sets the label for the Y-axis.

Example

```
Sub SetYAxisLabel(strLabel As String)
```

Parameters

- String strLabel

Return values

- void

Remarks

SortSeriesByLegend

Sorts all series in the chart from front to back (either ascending or descending) based upon the text of the legend.

Example

```
Sub SortSeriesByLegend(bEnable As Long,bAscending As Long)
```

Parameters

- Long bEnable
- Long bAscending

Return values

- void

Remarks

SortSeriesByValue

Sorts all series in the chart from front to back (either ascending or descending) based upon the most recent value in each series.

Example

```
Sub SortSeriesByValue(bEnable As Long,bAscending As Long)
```

Parameters

- Long bEnable
- Long bAscending

Return values

- void

Remarks

Uninitialize

Uninitializes a Chart object.

Example

```
Sub Uninitialize()
```

Parameters

Return values

- void

Remarks

UpdateChart

Updates the entire chart (rebuild all series) and render the scene.

Example

```
Sub UpdateChart()
```

Parameters

Return values

- void

Remarks

ZoomIn

Zooms in on the chart.

Example

```
Sub ZoomIn()
```

Parameters

Return values

- void

Remarks

ZoomNormal

Sets the zoom to the default.

Example

```
Sub ZoomNormal()
```

Parameters

Return values

- void

Remarks

ZoomOut

Zooms out from the chart.

Example

```
Sub ZoomOut()
```

Parameters

Return values

- void

Remarks

Chapter 7

ButtonGenerator object

The ButtonGenerator object is used to draw a button on a document. It provides various methods to set the font, text width and height, and the image used for the button.

This section covers the following topics:

- ButtonGenerator properties
- ButtonGenerator methods

ButtonGenerator properties

The following properties are available for the ButtonGenerator object.

Property	Description
AutoWidth	Auto-sizes the button to fit the text.
FontName	Specifies the name of the font to use for the button text.
FontSize	Specifies the size of the font to use for the button text.
Height	Sets or returns the height of the button.
Text	Sets or returns the text displayed on the button.
Width	Sets or returns the width of the button.

AutoWidth

Auto-sizes the button to fit the text.

Example

```
Property AutoWidth As Long
```

Parameters

- Long bAutoWidth

Return values

- Long

Remarks

See AttachImage sample code.

FontName

Specifies the name of the font to use for the button text.

Example

```
Property FontName As String
```

Parameters

- String strText

Return values

- String

Remarks

See AttachImage sample code.

FontSize

Specifies the size of the font to use for the button text.

Example

```
Property FontSize As Long
```

Parameters

- Long lSize

Return values

- Long

Remarks

See AttachImage sample code.

Height

Sets or returns the height of the button.

Example

```
Property Height As Long
```

Parameters

- Long lHeight

Return values

- Long

Remarks

See AttachImage sample code.

Text

Sets or returns the text displayed on the button.

Example

```
Property Text As String
```

Parameters

- String strText

Return values

- String

Remarks

See AttachImage sample code.

Width

Sets or returns the width of the button.

Example

```
Property Width As Long
```

Parameters

- Long lWidth

Return values

- Long

Remarks

See AttachImage sample code.

ButtonGenerator methods

The following methods are available for the ButtonGenerator object.

Methods	Descriptions
AttachImage	Saves a PNG image of the button to disk and attaches it to the document.
Initialize	Initializes a button object.
SaveImage	Saves a PNG image of the button to disk.
SetClickImage	Sets the button image after the button is clicked.
SetHoverImage	Sets the button image when the pointer is over the button.
SetNormalImage	Sets the button image when the button is drawn.
SetTextAlignCenter	Centers the text.
SetTextAlignLeft	Left-justifies text.
SetTextAlignRight	Right-justifies text.

AttachImage

Saves a PNG image of the button to disk and attaches it to the document.

Example

```
Sub AttachImage(strNormal As String, strHover As String,
strClickImage As String, strStyle As String, strHyperLink As
String)
```

Parameters

- String strNormal
- String strHover
- String strClickImage
- String strStyle
- String strHyperLink

Return values

- void

Remarks

Sample code

```
Dim layoutObj
Set layoutobj = CreateObject("AMLAYOUT.Layout")

'###Specify where to get/put files
strOutDir = "d:\reports\Button"
layoutObj.CreateDirectory strOutDir
layoutObj.CleanupDirectory strOutDir
layoutObj.OutputDirectory = strOutDir

layoutObj.DocumentName = "ButtonGenerator.html"
layoutObj.CenterDocument = TRUE
layoutObj.BeginReport ""
layoutObj.AddHeader "Test Report",""
layoutObj.AddVerticalSpace "10pt"

'###Create Button object
```

```
Set ButtonObj = layoutObj.ButtonGenerator
ButtonObj.AutoWidth = TRUE
ButtonObj.Width = 340
ButtonObj.FontSize = 18
ButtonObj.FontName = “Arial”
ButtonObj.SetTextAlignCenter

‘###Orange Button
ButtonObj.SetNormalImage “OButtonL.png”, “OButtonC.png”, _
      “OButtonR.png”
ButtonObj.SetHoverImage “OButtonL2.png”, “OButtonC2.png”, _
      “OButtonR2.png”
ButtonObj.SetClickImage “GButtonL.png”, “GButtonC.png”, _
      “GButtonR.png”
ButtonObj.Text = “Text On A Button”
ButtonObj.AttachImage “OrangeButton.png”, _
      “OrangeButton2.png”, “”,“”,“”
   layoutObj.AddFooter “Footer.txt”
   layoutObj.EndReport
```

Initialize

Initializes a button object.

Example

```
Sub Initialize(pDispatch As Object)
```

Parameters

- Object layoutObj

Return values

- void

Remarks

SaveImage

Saves a PNG image of the button to disk.

Example

```
Sub SaveImage(strNormalImage As String, strHoverImage As
String, strClickImage As String)
```

Parameters

- String strNormalImage
- String strHoverImage
- String strClickImage

Return values

- void

Remarks

SetClickImage

Sets the button image after the button is clicked.

Example

```
Sub SetClickImage(strLeft As String, strCenter As String,
strRight As String)
```

Parameters

- String strLeft
- String strCenter
- String strRight

Return values

- void

Remarks

SetHoverImage

Sets the button image when the pointer is over the button.

Example

```
Sub SetHoverImage(strLeft As String, strCenter As String,
strRight As String)
```

Parameters

- String strLeft
- String strCenter
- String strRight

Return values

- void

Remarks

SetNormalImage

Sets the button image when the button is drawn.

Example

```
Sub SetNormalImage(strLeft As String, strCenter As String,
strRight As String)
```

Parameters

- String strLeft
- String strCenter
- String strRight

Return values

- void

Remarks

SetTextAlignCenter

Centers the text.

Example

```
Sub SetTextAlignCenter()
```

Parameters

Return values

- void

Remarks

SetTextAlignLeft

Left-justifies text.

Example

```
Sub SetTextAlignLeft()
```

Parameters

Return values

- void

Remarks

SetTextAlignRight

Right-justifies text.

Example

```
Sub SetTextAlignRight()
```

Parameters

Return values

- void

Remarks

Chapter 8

GaugeGenerator object

The GaugeGenerator object is used to draw gauges and meters on the document.

This section covers the following topics:

- GaugeGenerator properties
- GaugeGenerator methods

GaugeGenerator properties

The following properties are available for the GaugeGenerator object.

Property	Description
DecimalPrecision	Sets or returns the number of decimal places to display (for example, 3 would display 0.000).
DialFontSize	Sets or returns the font size of the numerals on the dial.
DialSizePercent	Sets or returns the dial size percent (from 0.0 to 200.0).
DialSweepAngle	Sets or returns the dial sweep angle (from 0 to 360 degrees).
DialTicks	Sets or returns the number of ticks on the dial (for example, 10).
EnableWarningGradient	Sets or returns a flag that will enable the warning gradient.
Height	Sets or returns the height or length of the gauge.
MaxValue	Sets or returns the maximum value indicated by the needle on the dial (for example, 100).
MinValue	Sets or returns the minimum value indicated by the needle on the dial (for example, 0).
NeedleThickness	Sets or returns the thickness of the needle.

Property	Description
NeedleXPos	Sets or returns the horizontal offset from the center of the gauge where the needle base is drawn.
NeedleYPos	Sets or returns the vertical offset from the center of the gauge where the needle base is drawn.
Value	Sets or returns the value to which the needle points.
ValueFontSize	Sets or returns the font size for drawing the value.
ValueSuffix	Sets or returns the units to append when displaying a value (for example, %, KB, or MB).
ValueXPos	Sets or returns the horizontal offset from the center of the gauge where the value is drawn.
ValueYPos	Sets or returns the vertical offset from the center of the gauge where the value is drawn.
WarningColorThickness	Sets or returns the thickness of the warning color band.
Width	Sets or returns the width of the gauge.

DecimalPrecision

Sets or returns the number of decimal places to display (for example, 3 would display 0.000).

Example

```
Property DecimalPrecision As Integer
```

Parameters

- Integer DecimalPrecision

Return values

- Integer

Remarks

See the AttachImage code sample.

DialFontSize

Sets or returns the font size of the numerals on the dial.

Example

```
Property DialFontSize As Long
```

Parameters

- Long DialFontSize

Return values

- Long

Remarks

See the AttachImage code sample.

DialSizePercent

Sets or returns the dial size percent (from 0.0 to 200.0).

Example

```
Property DialSizePercent As 8 byte real
```

Parameters

- Double MaxValue

Return values

- Double

Remarks

See the AttachImage code sample.

DialSweepAngle

Sets or returns the dial sweep angle (from 0 to 360 degrees).

Example

```
Property DialSweepAngle As 8 byte real
```

Parameters

- Double MaxValue

Return values

- Double

Remarks

See the AttachImage code sample.

DialTicks

Sets or returns the number of ticks on the dial (for example, 10).

Example

```
Property DialTicks As Long
```

Parameters

- Long DialTicks

Return values

- Long

Remarks

See the AttachImage code sample.

EnableWarningGradient

Sets or returns a flag that will enable the warning gradient.

Example

```
Property EnableWarningGradient As Long
```

Parameters

- Long EnableWarningGradient

Return values

- Long

Remarks

See the AttachImage code sample.

Height

Sets or returns the height or length of the gauge.

Example

```
Property Height As Long
```

Parameters

- Long Height

Return values

- Long

Remarks

See the AttachImage code sample.

MaxValue

Sets or returns the maximum value indicated by the needle on the dial (for example, 100).

Example

```
Property MaxValue As 8 byte real
```

Parameters

- Double MaxValue

Return values

- Double

Remarks

See the AttachImage code sample.

MinValue

Sets or returns the minimum value indicated by the needle on the dial (for example, 0).

Example

```
Property MinValue As 8 byte real
```

Parameters

- Double MinValue

Return values

- Double

Remarks

See the AttachImage code sample.

NeedleThickness

Sets or returns the thickness of the needle.

Example

```
Property NeedleThickness As Long
```

Parameters

- Long NeedleThickness

Return values

- Long

Remarks

See the AttachImage code sample.

NeedleXPos

Sets or returns the horizontal offset from the center of the gauge where the needle base is drawn.

Example

```
Property NeedleXPos As Long
```

Parameters

- Long NeedleXPos

Return values

- Long

Remarks

See the AttachImage code sample.

NeedleYPos

Sets or returns the vertical offset from the center of the gauge where the needle base is drawn.

Example

```
Property NeedleYPos As Long
```

Parameters

- Long NeedleYPos

Return values

- Long

Remarks

See the AttachImage code sample.

Value

Sets or returns the value to which the needle points.

Example

```
Property Value As 8 byte real
```

Parameters

- Double Value

Return values

- Double

Remarks

See the AttachImage code sample.

ValueFontSize

Sets or returns the font size for drawing the value.

Example

```
Property ValueFontSize As Long
```

Parameters

- Long ValueFontSize

Return values

- Long

Remarks

See the AttachImage code sample.

ValueSuffix

Sets or returns the units to append when displaying a value (for example, %, KB, or MB).

Example

```
Property ValueSuffix As String
```

Parameters

- String ValueSuffix

Return values

- String

Remarks

See the AttachImage code sample.

ValueXPos

Sets or returns the horizontal offset from the center of the gauge where the value is drawn.

Example

```
Property ValueXPos As Long
```

Parameters

- Long ValueXPos

Return values

- Long

Remarks

See the AttachImage code sample.

ValueYPos

Sets or returns the vertical offset from the center of the gauge where the value is drawn.

Example

```
Property ValueYPos As Long
```

Parameters

- Long ValueYPos

Return values

- Long

Remarks

See the AttachImage code sample.

WarningColorThickness

Sets or returns the thickness of the warning color band.

Example

```
Property WarningColorThickness As Long
```

Parameters

- Long WarningColorThickness

Return values

- Long

Remarks

See the AttachImage code sample.

Width

Sets or returns the width of the gauge.

Example

```
Property width As Long
```

Parameters

- Long Width

Return values

- Long

Remarks

See the AttachImage code sample.

GaugeGenerator methods

The following methods are available for the GaugeGenerator object.

Methods	Descriptions
AddWarningColor	Adds a warning color to the dial, using the `R,G,B,Min,Max` format.
AttachImage	Saves a PNG image of the gauge to the specified file and attaches it to the document.
Initialize	Initializes a Layout object.
RemoveWarningColors	Removes all warning colors from the gauge.
SaveImage	Saves a PNG image of the gauge to the specified file.
SetBackgroundImage	Sets the background image. The dial is drawn after the background image.
SetDialColor	Sets the RGB color for the values on the dial.
SetForegroundImage	Sets the foreground image (with RGB transparency). This image is drawn after the dial.

Methods	Descriptions
SetNeedleColor	Sets the RGB color of the needle.
SetValueColor	Sets the RGB color of the value.
UpdateGauge	

AddWarningColor

Adds a warning color to the dial, using the `R,G,B,Min,Max` format.

Example

```
Sub AddWarningColor(nRed As Integer,nGreen As Integer,nBlue
As Integer,dMinVal As 8 byte real,dMaxVal As 8 byte real)
```

Parameters

- Integer nRed
- Integer nGreen
- Integer nBlue
- Double dMinVal
- Double dMaxVal

Return values

- void

Remarks

See the AttachImage code sample.

AttachImage

Saves a PNG image of the gauge to the specified file and attaches it to the document.

Example

```
Sub AttachImage(strImage As String,strStyle As String,
strHyperLink As String)
```

Parameters

- String strImage
- String strStyle
- String strHyperLink

Return values

- void

Remarks

Sample code

```
Dim layoutobj
Set layoutobj = CreateObject("AMLAYOUT.Layout")

dim strOutputDir
strOutputDir = "D:\reports\Gauge"
layoutObj.CreateDirectory strOutputDir
layoutObj.CleanupDirectory strOutputDir
layoutObj.OutputDirectory = strOutputDir
layoutObj.EmbedStyleSheet = true

' ### Set the document properties
layoutObj.Title= "Sample Gauge"
layoutObj.Author = "Kiran Jain"
layoutobj.DocumentName = "GauageGenerator.html"
layoutobj.BeginReport ""
layoutobj.AddHeader "Gauge Examples",""
layoutobj.CenterDocument = FALSE
layoutobj.AddVerticalSpace "10pt'"

Dim GaugeObj
Set GaugeObj = layoutObj.GaugeGenerator

Dim bSuccess
layoutobj.AddVerticalSpace "10pt"
layoutObj.AddText "Gauge 1", "class='Header3'"
'***********************************
' Create a Large Gauge
'***********************************
' Set the Background Image
bSuccess = GaugeObj.SetBackgroundImage(
"Gauge152x101Back_Purple_White_Chroma.png")
```

```
' Set the Foreground Image
bSuccess = GaugeObj.SetForegroundImage(
"Gauge152x101Fore_Purple_White_Chroma.png", 255, 0, 255 )

' Dial Settings
GaugeObj.SetDialColor 0, 0, 0
GaugeObj.DialSweepAngle = 110'Degrees (0-360)
GaugeObj.DialTicks = 4'# of Tick marks on the dial
GaugeObj.DialSizePercent = 120'Percent (0.0 - 200.0+)
GaugeObj.DialFontSize = 14

' Needle Settings
GaugeObj.SetNeedleColor 6, 70, 120'R,G,B
GaugeObj.NeedleThickness = 7
GaugeObj.NeedleYPos = 40'Needle Y Position
GaugeObj.NeedleXPos = 0

' Warning Color Settings
GaugeObj.WarningColorThickness = 7
GaugeObj.EnableWarningGradient = 1

' Green Range (0-50)
GaugeObj.AddWarningColor   0, 255,   0,0, 50'R,G,B,Min,Max

' Yellow Range (50-75)
GaugeObj.AddWarningColor 255, 255,   0,50, 75'R,G,B,Min,Max

' Red Range (75-100)
GaugeObj.AddWarningColor 255,   0,   0,75, 100'R,G,B,Min,Max

' Value Settings
GaugeObj.DecimalPrecision = 0'Number of digits after the
decimal
GaugeObj.SetValueColor 254, 254, 254'R,G,B
GaugeObj.ValueXPos = 0'Needle X Position from Center
GaugeObj.ValueYPos = 34'Needle Y Position from Center
GaugeObj.ValueFontSize = 18'Font Size for Drawing the Value
GaugeObj.ValueSuffix = "%"
GaugeObj.MinValue = 0'Min Needle Value
GaugeObj.MaxValue = 100'Max Needle Value
GaugeObj.Value = 50'Needle Value

GaugeObj.AttachImage "Gauge1.png",
"style=filter:chroma(color=#FF00FF)", ""
```

```
layoutobj.AddVerticalSpace "10pt"
layoutObj.AddText "Gauge 2", "class='Header3'"

'************************************
' Create another Gauge
' but use smaller background/forground images
'************************************

' Set the Background Image
bSuccess = GaugeObj.SetBackgroundImage(
"Gauge110x73Back_Purple_White_Chroma.png")
' Set the Foreground Image
bSuccess = GaugeObj.SetForegroundImage(
"Gauge110x73Fore_Purple_White_Chroma.png", 255, 0, 255 )

GaugeObj.DialSizePercent = 100'Percent (0.0 - 200.0+)
GaugeObj.DialTicks = 4'# of Tick marks on the dial
GaugeObj.DialFontSize = 14
GaugeObj.DialSweepAngle = 110'Degrees (0-360)
GaugeObj.NeedleThickness = 6
GaugeObj.NeedleYPos = 20'Needle Y Position from Center
GaugeObj.ValueYPos = 28'Needle X Position from Center
GaugeObj.ValueFontSize = 16'Font Size for Drawing the Value
GaugeObj.WarningColorThickness = 6
GaugeObj.AttachImage "Gauge2.png",
"style=filter:chroma(color=#FF00FF)", ""

layoutobj.AddVerticalSpace "10pt"
layoutObj.AddText "Gauge 3", "class='Header3'"

'************************************
' Create another Small Gauge (using the same gauge object)
'************************************
' Set the Background Image
bSuccess = GaugeObj.SetBackgroundImage(
"Gauge110x73Back_Purple_White_Chroma.png")
' Set the Foreground Image
bSuccess = GaugeObj.SetForegroundImage(
"Gauge110x73Fore_Purple_White_Chroma.png", 255, 0, 255 )

GaugeObj.DialTicks = 2'# of Tick marks on the dial
GaugeObj.AttachImage "Gauge3.png",
"style=filter:chroma(color=#FF00FF)", ""
layoutobj.EndReport
```

Initialize

Initializes a Layout object.

Example

```
Sub Initialize(layoutObj As Object)
```

Parameters

- Object layoutObj

Return values

- void

Remarks

RemoveWarningColors

Removes all warning colors from the gauge.

Example

```
Sub RemoveWarningColors()
```

Parameters

Return values

- void

Remarks

SaveImage

Saves a PNG image of the gauge to the specified file.

Example

```
Sub SaveImage(strImage As String)
```

Parameters

- String strImage

Return values

- void

Remarks

SetBackgroundImage

Sets the background image. The dial is drawn after the background image.

Example

```
Sub SetBackgroundImage(strPathname As String) As Long
```

Parameters

- String strPathname

Return values

- Long

Remarks

See the AttachImage code sample.

SetDialColor

Sets the RGB color for the values on the dial.

Example

```
Sub SetDialColor(nRed As Integer,nGreen As Integer,nBlue As
Integer)
```

Parameters

- Integer nRed
- Integer nGreen

- Intege rnBlue

Return values

- void

Remarks

See the AttachImage code sample.

SetForegroundImage

Sets the foreground image (with RGB transparency). This image is drawn after the dial.

Example

```
Sub SetForegroundImage(strPathname As String,nRed As
Integer,nGreen As Integer,nBlue As Integer) As Long
```

Parameters

- String strPathname
- Integer nRed
- Integer nGreen
- Integer nBlue

Return values

- Long

Remarks

See the AttachImage code sample.

SetNeedleColor

Sets the RGB color of the needle.

Example

```
Sub SetNeedleColor(nRed As Integer,nGreen As Integer,nBlue As
```

```
Integer)
```

Parameters

- Integer nRed
- Integer nGreen
- Integer nBlue

Return values

- void

Remarks

See the AttachImage code sample.

SetValueColor

Sets the RGB color of the value.

Example

```
Sub SetValueColor(nRed As Integer,nGreen As Integer,nBlue As
Integer)
```

Parameters

- Integer nRed
- Integer nGreen
- Integer nBlue

Return values

- void

Remarks

See the AttachImage code sample.

UpdateGauge

Example

```
Sub UpateGauge()
```

Parameters

Return values

- void

Remarks

Chapter 9

OLAPInfo object

The OLAPInfo object is used to generate an MDX query against a cube, and render the results of the query to an OLAP table based on ADOMD cellsets.

This section covers the following topics:

- OLAPInfo properties
- OLAPInfo methods

OLAPInfo properties

The following properties are available for the OLAPInfo object.

Property	Description
DefaultColumnWidth	Sets or return the default column width of the HTML table.
DemoMode	Sets or returns a boolean value that generates random data to fill the table.
DisableTimeDimensionProcessing	Sets or returns a boolean that disables processing of the time dimension.
DisplayEmptyColumnData	Sets or returns a boolean flag that tells an MDX query to return an empty column.
DisplayEmptyRowData	Sets or returns a boolean flag that tells an MDX query to return an empty row.
Interactive	
Style	Sets or returns the style in which the OLAP table is rendered. The defaut style is Pivot Table.

DefaultColumnWidth

Sets or returns the default column width of the HTML table.

Example

```
Property DefaultColumnWidth As Integer
```

Parameters

- Integer DefaultColumnWidth

Return values

- Integer

Remarks

See the Render code sample.

DemoMode

Sets or returns a boolean value that generates random data to fill the table.

Example

```
Property DemoMode As Long
```

Parameters

- Long DemoMode

Return values

- Long

Remarks

DisableTimeDimensionProcessing

Sets or returns a boolean that disables processing of the time dimension.

Example

```
Property DisableTimeDimensionProcessing As Long
```

Parameters

- Long DisableTimeDimensionProcessing

Return values

- Integer

Remarks

DisplayEmptyColumnData

Sets or returns a boolean flag that tells an MDX query to return an empty column.

Example

```
Property DisplayEmptyColumnData As Long
```

Parameters

- Long DisplayEmptyColumnData

Return values

- Long

Remarks

See the Render code sample.

DisplayEmptyRowData

Sets or returns a boolean flag that tells an MDX query to return an empty row.

Example

```
Property DisplayEmptyRowData As Long
```

Parameters

- Long DisplayEmptyRowData

Return values

- Long

Remarks

See the Render code sample.

Interactive

Example

```
Property Interactive As Long
```

Parameters

- Long Interactive

Return values

- Long

Remarks

Style

Sets or returns the style in which the OLAP table is rendered. The defaut style is Pivot Table.

Example

```
Property Style As Integer
```

Parameters

- Integer Style

Return values

- Integer

Remarks

OLAPInfo methods

The following methods are available for the OLAPInfo object.

Methods	Descriptions
ConvertOLAPToDateTime	Converts or formats the OLAP time to normal date/time.
GenerateMDX	Returns an MDX query that queries the cube.
Initialize	Initializes an OLAPInfo object.
Render	Creates an OLAP table based on the ADOMD cellset that is passed to it.
SetColumn	Sets the column axis of the MDX query.
SetColumnWidth	Sets or returns the width of a specific column.
SetCubeName	Sets the cube name of the MDX query.
SetRow	Sets the row axis of the MDX query.
SetWhere	Sets the *where* clause of the MDX query.

ConvertOLAPToDateTime

Converts or formats the OLAP time to normal date/time.

Example

```
Sub ConvertOLAPToDateTime(strDimension As String) As Date
```

Parameters

- String strDimension

Return values

- Date

Remarks

See the Render code sample.

GenerateMDX

Returns an MDX query that queries the cube.

Example

```
Sub GenerateMDX() As String
```

Parameters

Return values

- String

Remarks

See the Render code sample.

Initialize

Initializes an OLAPInfo object.

Example

```
Sub Initialize(layoutObj As Object)
```

Parameters

- Object layoutObj

Return values

- void

Remarks

Render

Creates an OLAP table based on the ADOMD cellset that is passed to it.

Example

```
Sub Render(connectionObj As Object, cellsetObj As Object)
```

Parameters

- Object connectionObj
- Object cellsetObj

Return values

- void

Remarks

Sample code

```
Const PRM_MDXSERVER = "KIRANJ2000"
Const PRM_MDXCATALOG = "FoodMart 2000"
Const PRM_MDXCUBENAME = "Sales"
Const strOutputDirectory = "d:\reports\OlapInfo"

Dim objLayout
Dim objOLAPInfo
Dim strMDX
Dim strConnection

On Error Resume Next

Set objLayout = CreateObject("AMLAYOUT.Layout")
Set objOLAPInfo = CreateObject("AMLAYOUT.OLAPInfo")

Dim strTime
Dim DateTime
strTime = "[Times].[All Times].[2000].[May].[1]"
Datetime = objOLAPInfo.ConvertOLAPToDateTime("[Times].[All
Times].[2000].[May].[1]")

With objLayout
   .CreateDirectory strOutputDirectory
   .CleanupDirectory strOutputDirectory
   .OutputDirectory = strOutputDirectory
   .DocumentName = "OlapInfo.html"
   .CenterDocument = False
   .EmbedStyleSheet = True
   .Title = "Test Olap Info"
```

```
   .BeginReport vbNullString
   .AddHeader "Test Olap Info", vbNullString
   .AddVerticalSpace "10pt"
End With

strConnection = "DataSource=" & PRM_MDXSERVER & ";Initial
Catalog=" & PRM_MDXCATALOG & ";Provider=msolap"

With objOLAPInfo
    .SetCubeName PRM_MDXCUBENAME
    .DisplayEmptyColumnData = False
    .DisplayEmptyRowData = False
    .SetColumn "[Customers].[All Customers].[USA].Children"
    .SetRow "[Promotion Media].[Media Type].members"
    .SetWhere "[Measures].[Unit Sales]"
    strMDX = .GenerateMDX
End With

ADOMDConnectionQuery objLayout, objOLAPInfo, strConnection,
strMDX

objLayout.WriteLine "<br><P>" & strTime & " = " & DateTime,
TRUE
objLayout.EndReport

Set objLayout = Nothing
Set objOLAPInfo = Nothing

MsgBox "done"

Private Sub ADOMDConnectionQuery(ByRef objLayout, ByRef
objOLAPInfo, ByVal ConnectionStr, ByVal MDXQuery)
    Dim ObjConn
    Dim ObjCellSet

    On Error Resume Next

    Set ObjConn = CreateObject("ADODB.Connection")
    Set ObjCellSet = CreateObject("ADOMD.Cellset")

    ObjConn.Open ConnectionStr
    ObjCellSet.Open MDXQuery, ObjConn

    If (ObjCellSet.Axes(0).Positions.Count <> 0) Then
        With objOLAPInfo
```

```
            .Initialize objLayout
            .DefaultColumnWidth = 100
            .Render ObjConn, ObjCellSet
        End With
    End If

    ADOMDCleanUp ObjConn, ObjCellSet

End Sub

Private Sub ADOMDCleanUp(ByRef ObjConn, ByRef ObjCellSet)
    With ObjConn
        If .State = 1 Then .Close
    End With
    With ObjCellSet
        If .State = 1 Then .Close
    End With
    Set ObjConn = Nothing
    Set ObjCellSet = Nothing
End Sub
```

SetColumn

Sets the column axis of the MDX query.

Example

```
Sub SetColumn(strColumnMDX As String)
```

Parameters

- String strColunmMDX

Return values

- void

Remarks

See the Render code example.

SetColumnWidth

Sets or returns the width of a specific column.

Example

```
Sub SetColumnWidth(nColumnIndex As Integer, nColumnWidth As
Integer)
```

Parameters

- Integer nColumnIndex
- Integer nColumnWidth

Return values

- void

Remarks

SetCubeName

Sets the cube name of the MDX query.

Example

```
Sub SetCubeName(strCubeName As String)
```

Parameters

- String strCubeName

Return values

- void

Remarks

See the Render code example.

SetRow

Sets the row axis of the MDX query.

Example

```
Sub SetRow(strRowMDX As String)
```

Parameters

- String strRowMDX

Return values

- void

Remarks

See the Render code example.

SetWhere

Sets the *where* clause of the MDX query.

Example

```
Sub SetWhere(strWhereMDX As String)
```

Parameters

- String strWhereMDX

Return values

- void

Remarks

See the Render code example.

Chapter 10

Table object

The Table object is used to insert a table in a document. In addition to drawing the table, this object provides methods to parse XML data and render it in tabular form.

This section covers the following topics:

- Table properties
- Table methods

Table properties

The following properties are available for the Table object.

Property	Description
DefaultRowHeight	Sets or return the height in pixels for each row.
FixedTable	Sets or return a flag that indicates whether to use fixed or automatically-sized columns.

DefaultRowHeight

Sets or return the height in pixels for each row.

Example

```
Property DefaultRowHeight As Integer
```

Parameters

- Integer defaultRowHeight

Return values

- Integer

Remarks

FixedTable

Sets or return a flag that indicates whether to use fixed or automatically-sized columns.

Example

```
Property FixedTable As Long
```

Parameters

- Boolean bFixedTable

Return values

- Boolean

Remarks

See the AddColumn sample code.

Table methods

The following methods are available for the Table object.

Methods	Descriptions
AddColumn	Adds a new column to the table.
AddRow	Adds a new row to the table.
AddText	Adds text to the current cell in the table.
EndColumn	Ends the current column.
EndRow	Ends the current row.
EndTable	Ends the table.
Initialize	Initializes the Table object.
MergeSchemaAndParse	Merges schema and data, and renders it into a table.
Parse	Parses XML, and renders it into a table.

Methods	Descriptions
SetColumnWidth	Sets the width for a specified column.
StartTable	Starts a new table.

AddColumn

Adds a new column to the table.

Example

```
Sub AddColumn(strStyle As String)
```

Parameters

- String strStyle

Return values

- void

Remarks

You can leave this parameter empty.

`StrStyle` parameter can be set in various ways, for example:
`“class = ‘TableHeader’ & style = ‘TEXT-ALIGN:center’”`

You can use the following classes:

- TableHeader
- TableHeader2
- TableBodyOdd
- TableBodyEven

You can use the following styles:

- TEXT-ALIGN: left
- TEXT-ALIGN: center
- TEXT-ALIGN: right

Sample code

```
Dim layoutObj
Set layoutobj = CreateObject(“AMLAYOUT.Layout”)

‘###Specify where to get/put files
strOutDir = “d:\reports\table”
layoutObj.CreateDirectory strOutDir
layoutObj.CleanupDirectory strOutDir
layoutObj.OutputDirectory = strOutDir

layoutObj.DocumentName = “Table.html”
layoutObj.CenterDocument = TRUE
layoutObj.BeginReport “”
layoutObj.AddHeader “Report For Table Object”,“”
layoutObj.AddVerticalSpace “10pt”

‘###Set table for two columns
Set TableObj = layoutObj.Table(2)
TableObj.SetColumnWidth 0,200
TableObj.SetColumnWidth 1,150
TableObj.FixedTable = TRUE
TableObj.StartTable “class = ‘Table”

TableObj.AddRow “style = ‘height:auto”
TableObj.AddColumn “class = ‘TableHeader’ & style = _
      ‘TEXT-ALIGN:center’”
layoutObj.WriteLine “FirstName”, tru
TableObj.EndColumn

TableObj.AddColumn “class = ‘TableHeader’ & style = _
      ‘TEXT-ALIGN:center’”
layoutObj.WriteLIne “LastName”, true
TableObj.EndColumn

TableObj.EndRow

TableObj.AddRow “”
TableObj.AddColumn “class = ‘TableBodyEven”
layoutObj.WriteLine “Alfred”, true
TableObj.EndColumn

TableObj.AddColumn “class = ‘TableBodyEven”
layoutObj.WriteLine “Newman”, true
TableObj.EndColumn
```

```
TableObj.EndRow

TableObj.AddRow “”
TableObj.AddColumn “class = ‘TableBodyEven”
layoutObj.WriteLine “Why”, true
TableObj.EndColumn

TableObj.AddColumn “class = ‘TableBodyEven”
layoutObj.WriteLine “Worry”, true
TableObj.EndColumn

TableObj.EndRow

TableObj.EndTable
layoutObj.EndReport
```

AddRow

Adds a new row to the table.

Example

```
Sub AddRow(strStyle As String)
```

Parameters

- String strStyle

Return values

- void

Remarks

See the AddColumn sample code.

AddText

Adds text to the current cell in the table.

Example

```
Sub AddText(strText As String, strStyle As String)
```

Parameters

- String strText
- String strStyle

Return values

- void

Remarks

See the AddColumn sample code.

EndColumn

Ends the current column.

Example

```
Sub EndColumn()
```

Parameters

Return values

- void

Remarks

See the AddColumn sample code.

EndRow

Ends the current row.

Example

```
Sub EndRow()
```

Parameters

Return values

- void

Remarks

See the AddColumn sample code.

EndTable

Ends the table.

Example

```
Sub EndTable()
```

Parameters

Return values

- void

Remarks

See the AddColumn sample code.

Initialize

Initializes the Table object.

Example

```
Sub Initialize(nNumCols As Integer, layoutObj As Object)
```

Parameters

- Integer nNumCols
- Object layoutObj

Return values

- void

Remarks

See the AddColumn sample code.

MergeSchemaAndParse

Merges schema and data, and renders it into a table.

Example

```
Sub MergeSchemaAndParse(schemaPartDocObject As Object,
dataPartDocObject As Object)
```

Parameters

- Object schemaPartDocObject
- Object dataPartDocObject

Return values

- void

Remarks

Parse

Parses XML, and renders it into a table.

Example

```
Sub Parse(MSXMLDocObject As Object)
```

Parameters

- Object MSXMLDocObject

Return values

- void

Remarks

SetColumnWidth

Sets the width for a specified column.

Example

```
Sub SetColumnWidth(nColumnIndex As Integer, nWidth As
Integer)
```

Parameters

- Integer nColumnIndex
- Integer nWidth

Return values

- void

Remarks

See the AddColumn sample code.

StartTable

Starts a new table.

Example

```
Sub StartTable(strStyle As String)
```

Parameters

- String strStyle

Return values

- void

Remarks

See the AddColumn sample code.

Chapter 11

TreeGenerator object

The TreeGenerator object is used to draw an HTML tree based on the structure of the XML binder. It provides the methods of parsing an XML binder and then attaches an HTML tree.

This section covers the following topics:

- TreeGenerator properties
- TreeGenerator methods

TreeGenerator properties

The following properties are available for the TreeGenerator object.

Property	Description
ClosedFolderImage	Sets or returns a closed folder image for a node in a tree.
OpenFolderImage	Sets or returns an open folder image for a node in a tree.

ClosedFolderImage

Sets or returns a closed folder image for a node in a tree.

Example

```
Property ClosedFolderImage As String
```

Parameters

- String ClosedFolderImage

Return values

- String

Remarks

OpenFolderImage

Sets or returns an open folder image for a node in a tree.

Example

```
Property OpenFolderImage As String
```

Parameters

- String OpenFolderImage

Return values

- String

Remarks

TreeGenerator methods

The following methods are available for the TreeGenerator object.

Methods	Descriptions
Attach	Attaches the generated HTML to the current open report.
Initialize	Initializes a Tree object.
ParseXMLBinder	Parses an XML binder and produces a DHTML tree.

Attach

Attaches the generated HTML to the current open report.

Example

```
Sub Attach()
```

Parameters

Return values

- void

Remarks

See the Binder::GetXMLBinder code sample.

Initialize

Initializes a Tree object.

Example

```
Sub Initialize(layoutObj As Object)
```

Parameters

- Object layoutObj

Return values

- void

Remarks

ParseXMLBinder

Parses an XML binder and produces a DHTML tree.

Example

```
Sub ParseXMLBinder(MSXMLDocObject As Object)
```

Parameters

- Object MSXMLDocObject

Return values

- void

Remarks

See the Binder::GetXMLBinder code sample.

Index

C

D

E

F

G

H

I

K

L

M

N

O

P

Q

R

S

Z