# MailEnable
# .NET Reference
# Version 10.0

MailEnable Messaging Services
for Microsoft Windows Server

**MailEnable Pty. Ltd.**
59 Murrumbeena Road
Murrumbeena
VIC 3163
Australia
t: +61 3 9563 4177
f: +61 3 9568 4270
www.mailenable.com

Date last modified March 19, 20199

# Warranty

You should carefully read the following terms and conditions before using this software. Unless you have a different license agreement signed by the respective owners, authors and copyright holders of the MailEnable product suite, herewith referred to as ("ME"), your use, distribution, or installation of this copy of MailEnable indicates your acceptance of this License.

All rights of any kind in MailEnable which are not expressly granted in this License are entirely and exclusively reserved to and by "ME". You may not rent, lease, modify, reverse engineer, translate, decompile and disassemble MailEnable without the permission of its owners, authors and copyright holders of MailEnable.

You are not permitted to commercialize derivative works of MailEnable without a written agreement signed by the respective owners, authors and copyright holders of MailEnable.

All accompanying files, data and materials, are distributed "as is" and with no warranties of any kind, whether express or implied.

This disclaimer of warranty constitutes an essential part of the agreement. Any liability of "ME" will be limited exclusively to refund of purchase price. In no event shall "ME", including but not limited to its principals, shareholders, officers, employees, affiliates, contractors, subsidiaries, or parent organizations, be liable for any incidental, consequential, or punitive damages whatsoever relating to the use of MailEnable, or your relationship with "ME".

In addition, in no event does "ME" authorize you to use MailEnable in applications or systems where "ME"'s failure to perform can reasonably be expected to result in a significant physical injury, or in loss of life. Any such use by you is entirely at your own risk, and you agree to hold "ME" harmless from any claims or losses relating to such unauthorized use.

You are specifically prohibited from charging, or requesting donations, for any copies, however made, and from distributing such copies with other products of any kind, commercial or otherwise, without prior written permission from "ME". "ME" reserves the right to revoke the above distribution rights at any time, for any or no reason.

# 1 Introduction

This document outlines interfaces that are specifically used for integration with MailEnable via the Microsoft .NET framework.

There are two primary assemblies that facilitate tightly coupled integration with .NET applications. These are referred to as the .NET Administration Assembly and the .NET Store Assembly.

The .NET Administration Assembly is used to administrate the configuration of MailEnable. Eg: creation of user mailboxes, creation of domains, postoffices, etc.

The .NET Store Assembly is used to develop applications that need access to the message store and data itself. Example: Creating an Appointment, Task or Contact. The .NET Store API does not currently provide a means of accessing messages themselves.

This API provides a very powerful means of accessing information from the MailEnable message store and making it available to other applications.

# Table of Contents

# 2 .NET Store Assembly

The following section outlines various API functions for managing items within the MailEnable Message Store.

The .NET Assembly hosting these functions is called MEStore.DLL. The namespace of the Store class is MailEnable.Store.

## 2.1 Store Item Functions

### 2.1.1 StoreFolder_Open

This function opens a MailEnable Store Folder so that the items within the folder can be accessed. When you open the folder, you can specify the default message class for the folder. As an example, of you wished to access the Inbox folder, you would specify a value of "0" to denote that the folder contains messages.

**Syntax:**

```
Public Function StoreFolder_Open(ByVal Postoffice As String, ByVal Mailbox As String, ByVal Folder As String, ByVal MessageClass As Integer, Optional ByVal Flags As Integer = 1) As Integer
```

**Parameters:**

| Name | Type | Description |
|---|---|---|
| Postoffice | String | The name of the postoffice owning the respective folder to access. |
| Mailbox | String | The name of the mailbox owning the respective folder to access. |
| Folder | String | The name of the folder to open. Eg: \Inbox |
| MessageClass | Integer | Value of 0,1,2,3 to represent Message, Contacts, Calendar, and Task items |
| Flags | Integer | Optional Value: Must be set to 1 |

**Results:**

| Name | Type | Description |
|---|---|---|
| ReturnValue | Number | Denotes the result of the called method:<br><br>**Value** / **Meaning**<br>0 / Failure<br>1 / Success |

### 2.1.2 StoreFolder_FindFirstItem

This function is fetches the first item in a folder.

**Syntax:**

```
Public Function StoreFolder_FindFirstItem() As Integer
```

**Results:**

| Name | Type | Description |
|---|---|---|
| ReturnValue | Number | Denotes the result of the called method:<br><br>**Value** / **Meaning**<br>0 – Failure<br>1 – Success |

### 2.1.3 StoreFolder_FindNextItem

This function is typically called after a StoreFolder_FindFirstItem call to fetch the next store item from the current folder.

**Syntax:**

```
Public Function StoreFolder_FindNextItem() As Integer
```

**Parameters:**

None

**Results:**

| Name | Type | Description |
|---|---|---|
| ReturnValue | Number | Denotes the result of the called method:<br><br>**Value** / **Meaning**<br>0 – Failure<br>1 – Success |

### 2.1.4 StoreFolder_FindClose

This function is used to release any data and state associated with a StoreFolder_Find/ StoreFolder_FindNext operation.

**Syntax:**

```
Public Function StoreFolder_FindClose() As Integer
```

**Parameters:**

None

**Results:**

| Name | Type | Description |
|---|---|---|
| ReturnValue | Number | Denotes the result of the called method:<br><br>**Value** / **Meaning**<br>0 – Failure<br>1 – Success |

### 2.1.5 StoreFolder_SelectItem

**Syntax:**

```
Public Function StoreFolder_SelectItem(ByVal ItemKey As String) As Integer
```

**Parameters:**

| Name | Type | Description |
|---|---|---|
| ItemKey | String | Optional Parameter that specifies a string that identifies the store item in a given store folder. This can typically be left blank. |

**Results:**

| Name | Type | Description |
|---|---|---|
| ReturnValue | Number | Denotes the result of the called method:<br><br>**Value** / **Meaning**<br>0 – Failure<br>1 – Success |

### 2.1.6 StoreFolder_SelectItemByField

This function allows store items to be located using a designated property/field as the search key. For example, you could use this function to locate an appointment by its "APPOINTMENTUID" property (which is the primary reason that this function exists).

**Syntax:**

```
Public Function StoreFolder_SelectItemByField(ByVal Field As String, ByVal FieldValue As String) As Integer
```

**Parameters:**

| Name | Type | Description |
|---|---|---|
| Field | String | The field/property in which to search |
| FieldValue | String | The value to be searched for in the field. |

**Results:**

| Name | Type | Description |
|---|---|---|
| ReturnValue | Number | Denotes the result of the called method:<br><br>**Value** / **Meaning**<br>0 – Failure<br>1 – Success |

### 2.1.7 StoreFolder_CreateItem

This function is used to create an item in a given folder. After this call has been made, you can then use the SetProperty function to set the properties of the new item. Once the properties have been set, you can call StoreFolder_Save to save changes.

**Syntax:**

```
Public Function StoreFolder_CreateItem(ByVal ItemKey As String) As Integer
```

**Parameters:**

| Name | Type | Description |
|---|---|---|
| ItemKey | String | Optional Parameter that specifies a string that identifies the store item in a given store folder. This can typically be left blank. |

**Results:**

| Name | Type | Description |
|---|---|---|
| ReturnValue | Number | Denotes the result of the called method:<br><br>**Value** – **Meaning**<br>0 – Failure<br>1 – Success |

### 2.1.8 StoreFolder_DeleteItem

This function deletes an item from the current store folder. The ItemKey field is used to specify which item to delete.

**Syntax:**

```
Public Function StoreFolder_DeleteItem(ByVal ItemKey As String) As Integer
```

**Parameters:**

| Name | Type | Description |
|---|---|---|
| ItemKey | String | Optional Parameter that specifies a string that identifies the store item in a given store folder. This can typically be left blank. |

**Results:**

| Name | Type | Description |
|---|---|---|
| ReturnValue | Number | Denotes the result of the called method:<br><br>**Value** – **Meaning**<br>0 – Failure<br>1 – Success |

### 2.1.9 StoreFolder_OpenItem

**Note: This function is no longer supported. Instead, use StoreFolder_Open, then you can use StoreItem_GetProperty or StoreItem_SetProperty. For StoreItem_SetProperty you pass the UID of the item.**

This function is used to access and fetch the properties for an item within a store folder. Typically you would call this function before making calls to StoreItem_GetProperty or StoreItem_SetProperty for the item. It is not

necessary to call StoreFolder_OpenItem if you have already used StoreFolder_FindFirst or StoreFolder_FindNext operation to locate the current store item.

**Syntax:**

```
Public Function StoreFolder_OpenItem(ByVal Postoffice As String, ByVal Mailbox As String,
ByVal Folder As String, ByVal ItemKey As String, ByVal MessageClass As Integer, ByVal Flags As
Integer) As Integer
```

**Parameters:**

| Name | Type | Description |
|---|---|---|
| Postoffice | String | The name of the postoffice owning the respective folder to access. |
| Mailbox | String | The name of the mailbox owning the respective folder to access. |
| Folder | String | The name of the folder to open. Eg: \Inbox |
| ItemKey | String | Optional Parameter that specifies a string that identifies the store item in a given store folder. This can typically be left blank. |
| MessageClass | Integer | A number indicating the type of item to be opened (0 though to 3 respectively as follows):<br><br>**Value** / **StoreItem type**<br>ME_MSG_CLASS_NOTE / Message<br>ME_MSG_CLASS_CONTACT / Contact<br>ME_MSG_CLASS_CALENDAR / Appointment<br>ME_MSG_CLASS_TASK / Task |
| Flags | Integer | Optional Value: Must be set to 1 |

**Results:**

| Name | Type | Description |
|---|---|---|
| ReturnValue | Number | Denotes the result of the called method:<br><br>**Value** / **Meaning**<br>0 / Failure<br>1 / Success |

## 2.1.10 StoreItem_GetProperty

This function returns a named property from the current store item. Named properties are outlined in the MailEnable Store API Reference.

**Syntax:**

```
Function StoreItem_GetProperty(ByVal PropName As String, Optional ByVal ItemKey As String =
"") As Object
```

**Parameters:**

| Name | Type | Description |
|---|---|---|
| PropName | String | The name of the property to be fetched. Property and data types are outlined in the MailEnable Store API Reference. |
| ItemKey | String | Optional Parameter that specifies a string that identifies the store item |

| | | |
|---|---|---|
| | | in a given store folder. This can typically be left blank. |

**Results:**

| Name | Type | Description |
|---|---|---|
| ReturnValue | Object | Returns an object of the type corresponding to the value being sought (Property and data types are outlined in the MailEnable Store API Reference). Since the value could be a string, double, filetime or long, the return type is designated as "object" to represent all cases. The result will be cast to the specified type of the receiving variable. |

### 2.1.11 StoreItem_SetProperty

This function sets a named property from the current store item.

**Syntax:**

```
Function StoreItem_SetProperty(ByVal PropName As String, ByVal PropValue As String, Optional ByVal ItemKey As String = "") As Object
```

**Parameters:**

| Name | Type | Description |
|---|---|---|
| PropName | String | The name of the property to be set. Property and data types are outlined in the MailEnable Store API Reference. |
| PropValue | String | The value to which the property will be set. |
| ItemKey | String | Optional Parameter that specifies a string that identifies the store item in a given store folder. This should be the UID of the item, which is usually the filename. Although an optional parameter, this should be populated. |

**Results:**

| Name | Type | Description |
|---|---|---|
| ReturnValue | Object | Returns an object of the type corresponding to the value being sought (Property and data types are outlined in the MailEnable Store API Reference). Since the value could be a string, double, filetime or long, the return type is designated as "object" to represent all cases. The result will be cast to the specified type of the receiving variable. |

### 2.1.12 StoreFolder_Save

Saves any details relating to modifications made to items within a designated Store Folder
Flags field should be set to 1.

**Syntax:**

```
Public Function StoreFolder_Save(ByVal Flags As Integer) As Integer
```

**Parameters:**

| Name | Type | Description |
|---|---|---|

| | | |
|---|---|---|
| Flags | Integer | Optional Value: Must be set to 1 |

**Results:**

| Name | Type | Description |
|---|---|---|
| ReturnValue | Number | Denotes the result of the called method:<br><br>**Value** / **Meaning**<br>0 – Failure<br>1 – Success |

### 2.1.13 StoreFolder_Close

This function is used to release any data and state associated with a StoreFolder_Open (and subsequent operations).

**Syntax:**

```
Public Function StoreFolder_Close() As Integer
```

**Parameters:**

None

**Results:**

| Name | Type | Description |
|---|---|---|
| ReturnValue | Number | Denotes the result of the called method:<br><br>**Value** / **Meaning**<br>0 – Failure<br>1 – Success |

## 2.2 Directory Functions

### 2.2.1 Directory_ClearFilter

**Syntax:**

```
Function Directory_ClearFilter() As Integer
```

**Parameters:**

None

**Results:**

| Name | Type | Description |
|---|---|---|
| ReturnValue | Number | Denotes the result of the called method: |

| | | | |
|---|---|---|---|
| | | **Value** | **Meaning** |
| | | 0 | Failure |
| | | 1 | Success |

### 2.2.2 Directory_SetFilterProperty

**Syntax:**

```
Function Directory_SetFilterProperty(ByVal PropName As String, ByVal PropValue As String) As Integer
```

**Parameters:**

| Name | Type | Description |
|---|---|---|
| PropName | String | The name of the property to be fetched. Property and data types are outlined in the MailEnable Store API Reference. |
| PropValue | String | The value to which the property will be set. |

**Results:**

| Name | Type | Description |
|---|---|---|
| ReturnValue | Number | Denotes the result of the called method:<br><br>**Value** / **Meaning**<br>0 – Failure<br>1 – Success |

### 2.2.3 Directory_GetEntry

**Syntax:**

```
Function Directory_GetEntry() As Integer
```

**Parameters:**

None

**Results:**

| Name | Type | Description |
|---|---|---|
| ReturnValue | Number | Denotes the result of the called method:<br><br>**Value** / **Meaning**<br>0 – Failure<br>1 – Success |

### 2.2.4 DirectoryEntry_Clear

**Syntax:**

```
Public Function DirectoryEntry_Clear() As Integer
```

**Parameters:**

None

**Results:**

| Name | Type | Description |
|---|---|---|
| ReturnValue | Number | Denotes the result of the called method:<br><br>**Value** / **Meaning**<br>0 – Failure<br>1 – Success |

### 2.2.5 DirectoryEntry_Commit

**Syntax:**

```
Public Function DirectoryEntry_Commit() As Integer
```

**Parameters:**

None

**Results:**

| Name | Type | Description |
|---|---|---|
| ReturnValue | Number | Denotes the result of the called method:<br><br>**Value** / **Meaning**<br>0 – Failure<br>1 – Success |

### 2.2.6 Directory_CreateEntry

**Syntax:**

```
Public Function Directory_CreateEntry() As Integer
```

**Parameters:**

None

**Results:**

| Name | Type | Description |
|---|---|---|
| ReturnValue | Number | Denotes the result of the called method: |

| | | |
|---|---|---|
| | | <table><tr><th>Value</th><th>Meaning</th></tr><tr><td>0</td><td>Failure</td></tr><tr><td>1</td><td>Success</td></tr></table> |

### 2.2.7 Directory_OpenEntry

**Syntax:**

```
Public Function Directory_OpenEntry(ByVal ItemKey As String, Optional ByVal XMLParams As String = "") As Integer
```

**Parameters:**

| Name | Type | Description |
|---|---|---|
| ItemKey | String | Optional Parameter that specifies a string that identifies the store item in a given store folder. This can typically be left blank. |
| XMLParams | String | (Leave Blank) |

**Results:**

| Name | Type | Description |
|---|---|---|
| ReturnValue | Number | Denotes the result of the called method:<br><table><tr><th>Value</th><th>Meaning</th></tr><tr><td>0</td><td>Failure</td></tr><tr><td>1</td><td>Success</td></tr></table> |

### 2.2.8 Directory_SelectEntry

**Syntax:**

```
Public Function Directory_SelectEntry(ByVal URI As String, Optional ByVal XMLParams As String = "") As Integer
```

**Parameters:**

| Name | Type | Description |
|---|---|---|
| URI | String | Optional Parameter that specifies a string that identifies the store item in a given store folder. This can typically be left blank. |
| XMLParams | String | (Leave Blank) |

**Results:**

| Name | Type | Description |
|---|---|---|
| ReturnValue | Number | Denotes the result of the called method:<br><table><tr><th>Value</th><th>Meaning</th></tr></table> |

| | | | | |
|---|---|---|---|---|
| | | 0 | Failure | |
| | | 1 | Success | |

### 2.2.9 Directory_FindFirstEntry

**Syntax:**

```
Public Function Directory_FindFirstEntry() As Integer
```

**Parameters:**

None

**Results:**

| Name | Type | Description |
|---|---|---|
| ReturnValue | Number | Denotes the result of the called method:<br><br>**Value** / **Meaning**<br>0 – Failure<br>1 – Success |

### 2.2.10 Directory_FindNextEntry

**Syntax:**

```
Public Function Directory_FindNextEntry() As Integer
```

**Parameters:**

None

**Results:**

| Name | Type | Description |
|---|---|---|
| ReturnValue | Number | Denotes the result of the called method:<br><br>**Value** / **Meaning**<br>0 – Failure<br>1 – Success |

### 2.2.11 Directory_FindClose

**Syntax:**

```
Public Function Directory_FindClose() As Integer
```

**Parameters:**

None

**Results:**

| Name | Type | Description |
|---|---|---|
| ReturnValue | Number | Denotes the result of the called method:<br><br>**Value** / **Meaning**<br>0 – Failure<br>1 – Success |

### 2.2.12 Directory_DeleteEntry

**Syntax:**

```
Public Function Directory_DeleteEntry(ByVal ItemKey As String, Optional ByVal XMLParams As String = "") As Integer
```

**Parameters:**

| Name | Type | Description |
|---|---|---|
| ItemKey | String | Optional Parameter that specifies a string that identifies the store item in a given store folder. This can typically be left blank. |
| XMLParams | String | (Leave Blank) |

**Results:**

| Name | Type | Description |
|---|---|---|
| ReturnValue | Number | Denotes the result of the called method:<br><br>**Value** / **Meaning**<br>0 – Failure<br>1 – Success |

### 2.2.13 DirectoryEntry_SetProperty

**Syntax:**

```
Function DirectoryEntry_SetProperty(ByVal PropName As String, ByVal PropValue As Object, Optional ByVal ItemKey As String = "") As Long
```

**Parameters:**

| Name | Type | Description |
|---|---|---|
| PropName | String | The name of the property to be set. Property and data types are outlined in the MailEnable Store API Reference. |
| PropValue | String | The value to which the property will be set. |
| ItemKey | String | Optional Parameter that specifies a string that identifies the store item in a given store folder. This can typically be left blank. |

**Results:**

| Name | Type | Description |
|---|---|---|
| ReturnValue | Number | Denotes the result of the called method:<br><br>**Value** / **Meaning**<br>0 – Failure<br>1 – Success |

### 2.2.14 DirectoryEntry_GetProperty

**Syntax:**

```
Function DirectoryEntry_GetProperty(ByVal PropName As String, Optional ByVal ItemKey As String
= "") As Object
```

**Parameters:**

| Name | Type | Description |
|---|---|---|
| PropName | String | The name of the property to be fetched. Property and data types are outlined in the MailEnable Store API Reference. |
| ItemKey | String | Optional Parameter that specifies a string that identifies the store item in a given store folder. This can typically be left blank. |

**Results:**

| Name | Type | Description |
|---|---|---|
| ReturnValue | Object | Returns an object of the type corresponding to the value being sought (Property and data types are outlined in the MailEnable Store API Reference).. Since the value could be a string, double, filetime or long, the return type is designated as “object” to represent all cases. The result will be cast to the specified type of the receiving variable. |

## 2.3 Examples

**Create Contact Example:**

This will create a new contact entry in the mailbox for "Mark" of postoffice "MailEnable"

```
    Private Sub CreateContact()
        Dim oStoreItem As New MailEnable.Store
        If (oStoreItem.StoreFolder_Open("MailEnable", "Mark", "\Contacts",
oStoreItem.ME_MSG_CLASS_CONTACT, 1) = 1) Then
            If oStoreItem.StoreFolder_CreateItem("") = 1 Then
                oStoreItem.StoreItem_SetProperty("PR_GIVEN_NAME", "William")
                oStoreItem.StoreItem_SetProperty("PR_MIDDLE_NAME", "Hattersley")
                oStoreItem.StoreItem_SetProperty("PR_SURNAME", "Thomas")
                oStoreItem.StoreItem_SetProperty("PR_DISPLAY_NAME", "Will H Thomas")
                oStoreItem.StoreItem_SetProperty("PR_DISPLAY_NAME_PREFIX", "Mr.")
                oStoreItem.StoreItem_SetProperty("PR_NICKNAME", "Billyo")
                oStoreItem.StoreItem_SetProperty("PR_EMAIL_ADDRESS", "wht@MailEnable.com")
                oStoreItem.StoreItem_SetProperty("PR_OFFICE_TELEPHONE_NUMBER", "12345678")
                oStoreItem.StoreItem_SetProperty("PR_HOME_TELEPHONE_NUMBER", "98765432")
```

```
                oStoreItem.StoreItem_SetProperty("PR_CELLULAR_TELEPHONE_NUMBER", "0412345678")
                oStoreItem.StoreItem_SetProperty("PR_COMPANY_NAME", "MailEnable")
                oStoreItem.StoreItem_SetProperty("PR_DEPARTMENT_NAME", "IT")
                oStoreItem.StoreItem_SetProperty("PR_NOTE", "This is a note")
                oStoreItem.StoreFolder_Save(1)
            End If
            oStoreItem.StoreFolder_Close()
        End If
    End Sub
```

### Create Appointment Example

This will add a new appointment with the properties specified

```
    Private Sub CreateAppointment()
        Dim oStoreItem As New MailEnable.Store
        If (oStoreItem.StoreFolder_Open("MailEnable", "Mark", "\Calendar",
oStoreItem.ME_MSG_CLASS_CALENDAR, 1) = 1) Then
            If oStoreItem.StoreFolder_CreateItem("") = 1 Then
                Dim APIRes As Long = oStoreItem.StoreItem_SetProperty("PR_SUBJECT", "New
Appointment")
                oStoreItem.StoreItem_SetProperty("PR_START_DATE", Now())
                oStoreItem.StoreItem_SetProperty("PR_END_DATE", DateAdd(DateInterval.Hour, 3,
Now()))
                oStoreItem.StoreItem_SetProperty("PR_CLASS", "APPOINTMENT.PERSONAL")
                oStoreItem.StoreItem_SetProperty("PR_BODY", "This is the body")
                oStoreItem.StoreItem_SetProperty("PR_BNP_AllDayEvent", 0)
                oStoreItem.StoreItem_SetProperty("PR_LOCATION", "Office")
                oStoreItem.StoreItem_SetProperty("PR_BNP_BusyStatus", 2)
                oStoreItem.StoreItem_SetProperty("PR_SENSITIVITY", 2)
                oStoreItem.StoreItem_SetProperty("PR_BNP_ReminderSet", 1)
                oStoreItem.StoreItem_SetProperty("PR_BNP_ReminderMinutesBeforeStart", 0)
                Dim SaveResult As Long = oStoreItem.StoreFolder_Save(1)
            End If
            oStoreItem.StoreFolder_Close()
        End If
    End Sub
```

### Create Task Example

This will add a new task with the properties specified

```
    Private Sub CreateTask()
        Dim oStoreItem As New MailEnable.Store
        If (oStoreItem.StoreFolder_Open("MailEnable", "Mark", "\Tasks",
oStoreItem.ME_MSG_CLASS_TASK, 1) = 1) Then
            If oStoreItem.StoreFolder_CreateItem("") = 1 Then
                oStoreItem.StoreItem_SetProperty("PR_IMPORTANCE", 1)
                oStoreItem.StoreItem_SetProperty("PR_BNP_Status", 3)
                oStoreItem.StoreItem_SetProperty("PR_SENSITIVITY", 2)
                oStoreItem.StoreItem_SetProperty("PR_BNP_StartDate", Now())
                oStoreItem.StoreItem_SetProperty("PR_BNP_DueDate", DateAdd(DateInterval.Hour,
3, Now()))
                oStoreItem.StoreItem_SetProperty("PR_BNP_Mileage", 1)
                oStoreItem.StoreItem_SetProperty("PR_BNP_PercentComplete", 43)
                oStoreItem.StoreItem_SetProperty("PR_BNP_Complete", 0)
                oStoreItem.StoreItem_SetProperty("PR_BNP_TotalWork", 20)
                oStoreItem.StoreItem_SetProperty("PR_BNP_ActualWork", 9)
                oStoreItem.StoreItem_SetProperty("PR_SUBJECT", "subject")
                oStoreItem.StoreItem_SetProperty("PR_BODY", "le body")

                Dim SaveResult As Long = oStoreItem.StoreFolder_Save(1)
            End If
            oStoreItem.StoreFolder_Close()
        End If
```

```
    End Sub
```

**List Directory Example**

This will list all of the email address and display name entries in the "MAILENABLE" postoffice via MessageBoxes

```
    Private Sub ListDirectory()
        Dim oStoreItem As New MailEnable.Store
        oStoreItem.Directory_ClearFilter()
        oStoreItem.Directory_SetFilterProperty(oStoreItem.gc_MEPROP_POSTOFFICE, "MAILENABLE")
        Dim APIResult As Long = 1
        If (oStoreItem.Directory_FindFirstEntry() = 1) Then
            Do
                MsgBox(oStoreItem.DirectoryEntry_GetProperty("", oStoreItem.gc_MEPROP_EMAIL) &
":" & oStoreItem.DirectoryEntry_GetProperty("", oStoreItem.gc_MEPROP_DISPLAYNAME))
            Loop While oStoreItem.Directory_FindNextEntry()
        End If
    End Sub
```

**Fetch Directory Entry Example:**

This opens the entry for Mark@MailEnable and prints the company name property

```
    Sub FetchDirectoryEntry()
        Dim szDLI As String = "VCF://MailEnable/Mark/$VCFROOT/[Mark]"
        Dim sPropVal As String
        Dim oStore As New MailEnable.Store
        If (oStore.Directory_OpenEntry(szDLI, "<PROTOCOL>VCF</PROTOCOL>")) Then
            sPropVal = oStore.DirectoryEntry_GetProperty("", "PR_COMPANY_NAME")
            If (sPropVal <> "") Then
                MsgBox("Result: " + sPropVal)
            End If
        End If
    End Sub
```

**List Messages Example:**

This lists the subject of each message in the inbox of Mark@MailEnable.

```
  Private Sub ListMessages()
    Dim APIResult As Long
    Dim oStore As New MailEnable.Store
    APIResult = oStore.StoreFolder_Open("MailEnable", "Mark", "\Inbox", oStore.ME_MSG_CLASS_NOTE, 1)
    If oStore.StoreFolder_FindFirstItem() = 1 Then
      Do
        Console.Write(oStore.StoreItem_GetProperty("PR_SUBJECT"))
      Loop While (oStore.StoreFolder_FindNextItem() = 1)
    End If
    APIResult = oStore.StoreFolder_FindClose()
    APIResult = oStore.StoreFolder_Close()
  End Sub
```

**Update Messages Example:**

This goes through the the inbox of Mark@MailEnable and changes the subject of each message to “Updated”.

```
Private Sub UpdateMessages()
      Dim APIResult As Long
      Dim oStore As New MailEnable.Store
      APIResult = oStore.StoreFolder_Open("MailEnable", "Mark", "\Inbox", oStore.ME_MSG_CLASS_NOTE, 1)
      If oStore.StoreFolder_FindFirstItem() = 1 Then
         Console.Write(oStore.StoreItem_SetProperty("PR_SUBJECT", "Updated"))
          oStore.StoreFolder_Save(1)
      End If
      APIResult = oStore.StoreFolder_FindClose()
      APIResult = oStore.StoreFolder_Close()
   End Sub
```

# 3 .NET Management Assembly

MailEnable's .NET Management Assembly allows you to manage the system configuration of a MailEnable mail server. As an example, you can use the .NET Management Assembly to programmatically create new mail users or add a new e-mail address to an existing mail user.

## 3.1 Address Map Administration

The AddressMap class is responsible for controlling the list of e-mail addresses that MailEnable is capable of receiving e-mail for.

### 3.1.1 MailEnable AddressMap Class

**Class**
MailEnable.Administration.AddressMap

**Properties**

| Wildcard | Name | Type | Description |
|---|---|---|---|
| Yes | Account | String(1024) | The account/post office |
| Yes | SourceAddress | String(1024) | The address the email was sent to |
| Yes | DestinationAddress | String(1024) | The address to send the email to |
| Yes | Scope | String(1024) | Not used |

**Functions**
GetAddressMap() As Long
FindFirstAddressMap() As Long
FindNextAddressMap() As Long
AddAddressMap() As Long
RemoveAddressMap() As Long
EditAddressMap(ByVal NewAccount As String, ByVal NewSourceAddress As String, ByVal NewDestinationAddress As String, ByVal NewScope As String) As Long

**Remarks**
Functions return a value of 1 for success and 0 for failure. Other status codes may be returned as information on errors. The AddressMap class is used to direct the incoming mail to the correct connector. Connectors would add their own entries to this file, as the MTA uses this to determine which connector is responsible for it.

Catch-all addresses are handled in MailEnable by the use of a wildcard in the email local part, for example *@example.com. Wildcards have to treated carefully when using the API, as they are used as wildcards, so when deleting a catchall address map it has to be renamed before removal, since removing a catchall address with * in it will remove all addresses for the domain. So before removing do an Edit to rename the email address, then remove this renamed entry.

**Example**

```
Dim lResult, oAddressMap

Dim oAddressMap As New MailEnable.Administration.AddressMap

oAddressMap.Account = ""
oAddressMap.DestinationAddress = ""
```

```
oAddressMap.Scope = ""
oAddressMap.SourceAddress = "[SMTP:test@mailenable.com]"

lResult = oAddressMap.GetAddressMap()

If lResult = 0 Then
MsgBox "Failed to get address."
Else
MsgBox "Address is sent to: " & oAddressMap.DestinationAddress
End If
```

## 3.2 Authentication Administration

### 3.2.1 MailEnable Login Class

The MailEnable Login Class is responsible for managing the authentication credentials of MailEnable users/mailbox owners. Whenever a new mailbox is created, an equivalent entry must be added to the Login class so they can authenticate to access the mailbox.

**Class**
MailEnable.Administration.Login

**Properties**

| Wildcard | Name | Type | Description |
|---|---|---|---|
| Yes | Username | String(64) | Username |
| Yes | Status | Long | 0=Disabled, 1=Enabled |
| Yes | Password | String(64) | Password |
| Yes | Account | String(128) | Account/post office |
| Yes | Rights | Rights(128) | Unused |
| Yes | Description | Description(1024) | Unused |
| | LoginAttempts | Long | Unused |
| | LastAttempt | Long | Unused |
| | LastSuccessfulLogin | Long | Unused |

**Functions**
GetLogin() As Long
FindFirstLogin() As Long
FindNextLogin() As Long
AddLogin() As Long
RemoveLogin() As Long
EditLogin(ByVal NewUserName As String, ByVal NewStatus As Long, ByVal NewPassword As String, ByVal NewAccount As String, ByVal NewDescription As String, ByVal NewLoginAttempts As Long, ByVal NewLastAttempt As Long, ByVal NewLastSuccessfulLogin As Long, ByVal NewRights As String) As Long

**Remarks**

Functions return non-zero for success, zero for failure. The authentication class is used to authenticate a users username and password combination. It can, and is, used for a variety of services and connectors. For example, the POP service would use it to validate a user logon.

Use only 30 characters maximum for the password. This is because encrypted passwords take up over twice as many characters, even though the provider will always return the unencrypted passwords. Remember that the encryption key in the registry must be correct, or the password returned will be wrong when using encrypted passwords.

You are able to pattern match on the following properties:

Username
Status
Password
Account
Rights
Description

## 3.3 Directory Administration

### 3.3.1 MailEnable Directory Class

The MailEnable Directory class provides a basic interface for adding items to the Postoffice Directory.

The Store Provider interface is preferred as a means of adding items to the directory, however this interface will suffice for adding basic entries to the Directory.

**Class**
MailEnable.Directory

**Properties**

| Wildcard | Name | Type | Description |
|---|---|---|---|
| | DirectoryEntryID | String(256) | |
| | DisplayName | String(256) | |
| | Account | String(128) | Postoffice |
| | MailAddress | String(1024) | |
| | DirectoryLocatorID | String(2048) | String URI used to locate additional properties of the directory item |
| | EntryType | Long | |
| | Host | String(128) | |

#### 3.3.1.1 .NET Programming Examples

Iterating:

```
Public Sub ListDirectory()
      Dim sPostoffice As String = "MailEnable"
      Dim oDirectory As New MailEnable.Directory
      Dim sDisplayName As String
      Dim sDirectoryEntryID As String
      Dim sResolvedAddress As String

      oDirectory.DirectoryEntryID = ""
      oDirectory.DisplayName = ""
      oDirectory.Account = sPostoffice
      oDirectory.MailAddress = ""
      oDirectory.DirectoryLocatorID = ""
      oDirectory.EntryType = -1
      oDirectory.Host = ""
      oDirectory.SortOrder = ""
      oDirectory.Filter = ""

      If oDirectory.FindFirstDirectory() = 1 Then
            Do
                  sDisplayName = oDirectory.DisplayName
                  sDirectoryEntryID = oDirectory.DirectoryEntryID
                  sResolvedAddress = oDirectory.MailAddress

                  oDirectory.DirectoryEntryID = ""
                  oDirectory.DisplayName = ""
                  oDirectory.Account = sPostoffice
                  oDirectory.MailAddress = ""
                  oDirectory.DirectoryLocatorID = ""
                  oDirectory.EntryType = -1
                  oDirectory.Host = ""
                  oDirectory.SortOrder = ""
                  oDirectory.Filter = ""

            Loop While (oDirectory.FindNextDirectory() = 1)

      End If
End Sub
```

Adding:

```
      Public Const gc_PRS_DISPLAY_NAME As String = "PRS_DISPLAYNAME"
        Public Const gc_PRS_EMAIL_ADDRESS As String = "PRS_EMAIL_ADDRESS"
        Public Const gc_PRS_DLID As String = "PRS_DLID"

        Public Const gc_MEPROP_POSTOFFICE_TAG As String = "POSTOFFICE"
        Public Const gc_MEPROP_ACCOUNT_TAG As String = "ACCOUNT"
        Public Const gc_MEPROP_DIRECTORYID_TAG As String = "DIRECTORYID"
        Public Const gc_MEPROP_DLID_TAG As String = "DLID"
        Public Const gc_MEPROP_MAILBOX_TAG As String = "MAILBOX"
        Public Const gc_MEPROP_FOLDER_TAG As String = "FOLDER"
        Public Const gc_MEPROP_PROTOCOL_TAG As String = "PROTOCOL"
        Public Const gc_MEPROP_EMAIL_TAG As String = "EMAIL"
        Public Const gc_MEPROP_DISPLAYNAME_TAG As String = "DISPLAYNAME"
        Public Const gc_MEPROP_NICKNAME_TAG As String = "PR_NICKNAME"
        Public Const gc_MEPROP_EMAIL_BNP As String = "PR_EMAIL_ADDRESS"

        Public Const gc_PROP_ITEMID As String = "ME_ITEM_ID"
        Public Const gc_PROP_COMPANY_NAME As String = "PR_COMPANY_NAME"
        Public Const gc_PROP_EMAIL_ADDRESS As String = "PR_EMAIL_ADDRESS"
        Public Const gc_PROP_BNP_Email1Address As String =
"PR_BNP_Email1Address"
        Public Const gc_PROP_DEPARTMENT_NAME As String =
"PR_DEPARTMENT_NAME"
        Public Const gc_PROP_DISPLAY_NAME As String = "PR_DISPLAY_NAME"
        Public Const gc_PROP_GIVEN_NAME As String = "PR_GIVEN_NAME"
        Public Const gc_PROP_MIDDLE_NAME As String = "PR_MIDDLE_NAME"
        Public Const gc_PROP_NORMALIZED_SUBJECT As String =
"PR_NORMALIZED_SUBJECT"
        Public Const gc_PROP_SUBJECT As String = "PR_SUBJECT"
        Public Const gc_PROP_SURNAME As String = "PR_SURNAME"
        Public Const gc_PROP_NICKNAME As String = "PR_NICKNAME"
        Public Const gc_PROP_TITLE As String = "PR_TITLE"
        Public Const gc_PROP_DISPLAY_NAME_PREFIX As String =
"PR_DISPLAY_NAME_PREFIX"
        Public Const gc_PROP_CELLULAR_TELEPHONE_NUMBER As String =
"PR_CELLULAR_TELEPHONE_NUMBER"
        Public Const gc_PROP_HOME_TELEPHONE_NUMBER As String =
"PR_HOME_TELEPHONE_NUMBER"
        Public Const gc_PROP_OFFICE_TELEPHONE_NUMBER As String =
"PR_OFFICE_TELEPHONE_NUMBER"
        Public Const gc_PROP_STATE_OR_PROVINCE As String =
"PR_STATE_OR_PROVINCE"
        Public Const gc_PROP_COUNTRY As String = "PR_COUNTRY"
        Public Const gc_PROP_STREET_ADDRESS As String = "PR_STREET_ADDRESS"
        Public Const gc_PROP_LOCALITY As String = "PR_LOCALITY"
        Public Const gc_PROP_POSTAL_CODE As String = "PR_POSTAL_CODE"

        Public Const gc_PROP_ASSISTANT As String = "PR_ASSISTANT"
        Public Const gc_PROP_PAGER_TELEPHONE_NUMBER As String =
"PR_PAGER_TELEPHONE_NUMBER"
        Public Const gc_PROP_BUSINESS_FAX_NUMBER As String =
"PR_BUSINESS_FAX_NUMBER"
        Public Const gc_PROP_BIRTHDAY As String = "PR_BIRTHDAY"
        Public Const gc_PROP_BUSINESS_HOME_PAGE As String =
"PR_BUSINESS_HOME_PAGE"
        Public Const gc_PROP_MSMESSENGER As String = "PR_MSMESSENGER"

        Public Const gc_PROP_NOTE As String = "PR_NOTE"
```

```
Public Function AddDirectoryEntry() As Integer

      Dim sPostoffice As String = "MailEnable"
      Dim oDirectory As New MailEnable.Directory
      Dim sFullName As String = "Joe Smith"
      Dim sMailAddress As String = "joe.smith@mydomain.com"

      oDirectory = New MailEnable.Directory
      oDirectory.Account = sPostoffice
      oDirectory.DirectoryLocatorID = ""
      oDirectory.DisplayName = sFullName
      oDirectory.DirectoryEntryID = ""
      oDirectory.MailAddress = sMailAddress

      GetFieldValues()  ' get all the required field values from the
form

      Dim APIResult As Integer = 0

      oDirectory.DirectoryEntry_Clear()
      oDirectory.DirectoryEntry_SetProperty("",
gc_MEPROP_POSTOFFICE_TAG, sPostoffice)
      oDirectory.DirectoryEntry_SetProperty("", gc_PROP_GIVEN_NAME,
FirstName)
      oDirectory.DirectoryEntry_SetProperty("", gc_PROP_MIDDLE_NAME,
MiddleName)
      oDirectory.DirectoryEntry_SetProperty("", gc_PROP_SURNAME,
LastName)
      oDirectory.DirectoryEntry_SetProperty("", gc_PROP_TITLE, Title)
      oDirectory.DirectoryEntry_SetProperty("",
gc_MEPROP_DISPLAYNAME_TAG, FullName)
      oDirectory.DirectoryEntry_SetProperty("",
gc_MEPROP_NICKNAME_TAG, NickName)   'NICKNAME
      oDirectory.DirectoryEntry_SetProperty("",
gc_PROP_EMAIL_ADDRESS, EMail)
      oDirectory.DirectoryEntry_SetProperty("", gc_MEPROP_EMAIL_TAG,
EMail)
      oDirectory.DirectoryEntry_SetProperty("", gc_PROP_COMPANY_NAME,
Business)

      oDirectory.DirectoryEntry_SetProperty("",
gc_PROP_STREET_ADDRESS, StreetAddress)
      oDirectory.DirectoryEntry_SetProperty("",
gc_PROP_OFFICE_TELEPHONE_NUMBER, PhoneW)
      oDirectory.DirectoryEntry_SetProperty("",
gc_PROP_HOME_TELEPHONE_NUMBER, PhoneH)
      oDirectory.DirectoryEntry_SetProperty("",
gc_PROP_CELLULAR_TELEPHONE_NUMBER, PhoneM)
      oDirectory.DirectoryEntry_SetProperty("", gc_PROP_LOCALITY,
Locality)
      oDirectory.DirectoryEntry_SetProperty("",
gc_PROP_STATE_OR_PROVINCE, State)
      oDirectory.DirectoryEntry_SetProperty("", gc_PROP_POSTAL_CODE,
Postcode)
      oDirectory.DirectoryEntry_SetProperty("", gc_PROP_COUNTRY,
Country)

      oDirectory.DirectoryEntry_SetProperty("", gc_PROP_BIRTHDAY,
TempBirthday)
```

```
	oDirectory.DirectoryEntry_SetProperty("",
gc_PROP_DEPARTMENT_NAME, Department)
	oDirectory.DirectoryEntry_SetProperty("", gc_PROP_ASSISTANT,
Assistant)
	oDirectory.DirectoryEntry_SetProperty("",
gc_PROP_BUSINESS_HOME_PAGE, HomePage)
	oDirectory.DirectoryEntry_SetProperty("",
gc_PROP_BUSINESS_FAX_NUMBER, PhoneF)
	oDirectory.DirectoryEntry_SetProperty("",
gc_PROP_PAGER_TELEPHONE_NUMBER, PhoneP)
	oDirectory.DirectoryEntry_SetProperty("", gc_PROP_MSMESSENGER,
MSMessenger)

	oDirectory.DirectoryEntry_SetProperty("", gc_PROP_NOTE, Notes)


	APIResult = oDirectory.Directory_CreateEntry()

	If APIResult <> 1 Then
		' Failed to create Directory entry
		Return 0
	End If

	Return 1

End Function
```

Removing:

```
Public Function RemoveDirectory() As Integer
	Dim sPostoffice As String = "MailEnable"
	Dim oDirectory As New MailEnable.Directory
	Dim sDirectoryEntryID As String =
"89AC5AD04E324C039AD3A4BDEDDFE832.VCF"

	oDirectory.Directory_ClearFilter()
	oDirectory.Directory_SetFilterProperty("POSTOFFICE",
sPostoffice)
	oDirectory.Directory_SetFilterProperty("DIRECTORYID",
sDirectoryEntryID)

	If oDirectory.Directory_GetEntry() = 1 Then
		If oDirectory.Directory_DeleteEntry() <> 1 Then
			' Delete failed!
			Return 0
		End If

		Return 1
	Else
		' Can't locate Directory entry
		Return 0
	End If

	Return 0
End Function
```

## 3.4 List Server Administration

### 3.4.1 MailEnable List Class

**Class**
MailEnable.Administration.List

**Properties**

| Wildcard | Name | Type | Description |
|---|---|---|---|
| | Description | String(256) | List description |
| | AccountName | String(128) | Account/post office |
| Yes | ListName | String(128) | Name of list |
| | ListType | Long | 0=Unmoderated, 1=Moderated |
| | ListStatus | Long | 0=Disabled, 1=Enabled |
| | ModeratorAddress | String(128) | Moderator address |
| | HeaderAnnotationStatus | Long | 0=no header, 1=Include header file |
| | HeaderAnnotation | String(256) | Name of header file with no extension |
| | FooterAnnotationStatus | Long | 0=no footer, 1=Include footer file |
| | FooterAnnotation | String(256) | Name of footer file with no extension |
| | ListAddress | String(256) | Address of list |
| *(Reserved)* | SubscribeMessageFileStatus | Long | |
| *(Reserved)* | SubscribeMessageFile | String(256) | |
| *(Reserved)* | UnsubscribeMessageFileStatus | Long | |
| *(Reserved)* | UnsubscribeMessageFile | String(256) | |
| *(Reserved)* | SubjectSuffixStatus | Long | |
| *(Reserved)* | SubjectSuffix | String(256) | |
| | SubjectPrefixStatus | Long | 0=Default - List Name<br>1=Don't Modify Subject<br>2=Use Custom Prefix |
| | SubjectPrefix | String(256) | |
| *(Reserved)* | Owner | String(256) | |
| | HelpMessageFileStatus | Long | Will send a list of commands back to the user when they send help as the password help |
| *(Reserved)* | HelpMessageFile | String(256) | |
| *(Reserved)* | RemovalMessageFileStatus | Long | |
| *(Reserved)* | RemovalMessageFile | String(256) | |
| | ReplyToMode | Long | 0=Replies to List<br>1=Replies to Sender<br>2=Replies to Moderator |
| *(Reserved)* | MaxMessageSize | Long | |
| | PostingMode | Long | 0=Members can Post<br>1=Anyone can Post |

| | | | |
|---|---|---|---|
| | | | 2=Password Protected Posting |
| | SubscriptionMode | Long | 0=Anyone can Subscribe<br>1=Subscription is disabled |
| *(Reserved)* | AuthenticationMode | Long | |
| | Password | String(256) | Contains the password if list is password protected- Password is enclosed in [braces] in the subject |
| *(Reserved)* | DigestMode | Long | |
| *(Reserved)* | DigestMailbox | String(256) | |
| *(Reserved)* | DigestAnnotationMode | Long | |
| *(Reserved)* | DigestAttachmentMode | Long | |
| *(Reserved)* | DigestMessageSeparationMode | Long | |
| *(Reserved)* | DigestSchedulingStatus | Long | |
| *(Reserved)* | DigestSchedulingMode | Long | |
| *(Reserved)* | DigestSchedulingInterval | Long | |
| *(Reserved)* | FromAddressMode | Long | |

*Note: Items marked as reserved may not have been implemented in current releases and are provided for forward compatibility.*

**Functions**

FindFirstList() As Long
FindNextList() As Long
AddList() As Long
GetList() As Long
RemoveList() As Long
EditList(ByVal NewDescription As String, ByVal NewAccountName As String, ByVal NewListName As String, ByVal NewListType As Long, ByVal NewListStatus As Long, ByVal NewHeaderAnnotationStatus As Long, ByVal NewHeaderAnnotation As String, ByVal NewFooterAnnotationStatus As Long, ByVal NewFooterAnnotation As String, ByVal NewModeratorAddress As String, ByVal NewListAddress As String, Optional ByVal NewSubscribeMessageFileStatus As Long = -1, Optional ByVal NewSubscribeMessageFile As String = "(Nil)", Optional ByVal NewUnsubscribeMessageFileStatus As Long = -1, Optional ByVal NewUnsubscribeMessageFile As String = "(Nil)", Optional ByVal NewSubjectSuffixStatus As Long = -1, Optional ByVal NewSubjectSuffix As String = "(Nil)", Optional ByVal NewSubjectPrefixStatus As Long = -1, Optional ByVal NewSubjectPrefix As String = "(Nil)", Optional ByVal NewOwner As String = "(Nil)", Optional ByVal NewHelpMessageFileStatus As Long = -1, Optional ByVal NewHelpMessageFile As String = "(Nil)", Optional ByVal NewRemovalMessageFileStatus As Long = -1, Optional ByVal NewRemovalMessageFile As String = "(Nil)", Optional ByVal NewReplyToMode As Long = -1, Optional ByVal NewMaxMessageSize As Long = -1, Optional ByVal NewPostingMode As Long = -1, Optional ByVal NewSubScriptionMode As Long = -1, Optional ByVal NewAuthenticationMode As Long = -1, Optional ByVal NewPassword As String = "(Nil)", Optional ByVal NewDigestMode As Long = -1, Optional ByVal NewDigestMailbox As String = "(Nil)", Optional ByVal NewDigestAnnotationMode As Long = -1, Optional ByVal NewDigestAttachmentMode As Long = -1, Optional ByVal NewDigestMessageSeparationMode As Long = -1, Optional ByVal NewDigestSchedulingStatus As Long = -1, Optional ByVal NewDigestSchedulingMode As Long = -1, Optional ByVal NewDigestSchedulingInterval As Long = -1, Optional ByVal NewFromAddressMode As Long = -1) As Long

**Remarks**

Functions return a value of 1 for success and 0 for failure. Other status codes may be returned as information on errors. The header and footer file needs to be located in the annotations subdirectory of the post office configuration directory.

#### 3.4.1.1 .NET Programming Examples

Iterating:

```
Public Sub ListLists()
	Dim sPostoffice As String = "MailEnable"
	Dim oList As New MailEnable.Administration.List
	Dim sListName As String
	Dim sListDesc As String

	oList.AccountName = sPostoffice
	oList.ListName = ""
	oList.ListStatus = -1
	oList.ListType = -1

	If oList.FindFirstList() = 1 Then
		Do
			sListName = oList.ListName
			sListDesc = oList.Description

			oList.AccountName = sPostoffice
			oList.ListName = ""
			oList.ListStatus = -1
			oList.ListType = -1

		Loop While (oList.FindNextList() = 1)
	End If
End Sub
```

Adding:

```
Public Function CreateAddressMapsForAllDomains(ByVal Postoffice As
String, ByVal AliasName As String, ByVal TargetAddress As String) As
Integer

	Dim oDomain As New MailEnable.Administration.Domain
	Dim oAddressMap As New MailEnable.Administration.AddressMap
	Dim sMappedAddress As String

	oDomain.AccountName = Postoffice
	oDomain.DomainName = ""
	oDomain.Status = -1
	oDomain.DomainRedirectionHosts = ""
	oDomain.DomainRedirectionStatus = -1

	If oDomain.FindFirstDomain() = 1 Then
		Do
			sMappedAddress = "[SMTP:" & AliasName & "@" &
		oDomain.DomainName & "]"

			oAddressMap.Account = Postoffice
			oAddressMap.DestinationAddress = TargetAddress
			oAddressMap.SourceAddress = sMappedAddress
			oAddressMap.Scope = 0

			If oAddressMap.AddAddressMap = 1 Then
				' Address Map added

			End If
```

```
                  ' Now let's try to get the next domain
                  oDomain.AccountName = Postoffice
                  oDomain.DomainName = ""
                  oDomain.Status = -1
                  oDomain.DomainRedirectionHosts = ""
                  oDomain.DomainRedirectionStatus = -1

            Loop While (oDomain.FindNextDomain() = 1)
      Else
            Return 0
      End If

      Return 1
End Function


Public Function AddList() As Integer

      Dim sPostoffice As String = "MailEnable"
      Dim oList As New MailEnable.Administration.List
      Dim sListName As String = "TestList"
      Dim sListDesc As String = "Some description about this List."
      Dim sModeratorAddr As String =
"[SMTP:moderator@mailenable.com]"
      Dim sDefaultAddr As String = "[SMTP:testlist@mailenable.com]"
      Dim sHeaderAnnotation As String = "[Technical Group]"
      Dim sFooterAnnotation As String = "The Technical Team."

      Dim bEnabled As Boolean = True

      oList.AccountName = sPostoffice
      oList.ListName = sListName
      oList.Description = sListDesc

      If bEnabled Then
            oList.ListStatus = 1
      Else
            oList.ListStatus = 0
      End If

      oList.ListType = 0      ' (0 - Unmoderated, 1 - Moderated)
      oList.ModeratorAddress = sModeratorAddr
      oList.ListAddress = sDefaultAddr

      oList.HeaderAnnotationStatus = 1    ' 0 - Disabled, 1 - Enabled
      oList.HeaderAnnotation = sListName & "-HEADER"
      oList.FooterAnnotationStatus = 1    ' 0 - Disabled, 1 - Enabled
      oList.FooterAnnotation = sListName & "-FOOTER"

      If (oList.AddList() = 1) Then
            oList.SetHeader(sPostoffice, sListName,
sHeaderAnnotation)
            oList.SetFooter(sPostoffice, sListName,
      sFooterAnnotation)

            If CreateAddressMapsForAllDomains(sPostoffice, sListName,
      "[LS:" & sPostoffice & "/" & sListName & "]") = 1 Then
                  ' Address Maps created for all domains
                  Return 1
            End If
```

```
                Return 2
            End If

            Return 0

        End Function
```

Removing:

```
        Public Function RemoveList() As Integer
            Dim sPostoffice As String = "MailEnable"
            Dim oList As New MailEnable.Administration.List
            Dim sListName As String = "TestList"

            oList.AccountName = sPostoffice
            oList.ListName = sListName
            oList.ListStatus = -1
            oList.ListType = -1

            If oList.RemoveList() = 1 Then
                ' Remove the Address Map...
                Dim oAddressMap As New
MailEnable.Administration.AddressMap
                oAddressMap.Account = sPostoffice
                oAddressMap.DestinationAddress = "[LS:" & sPostoffice &
            "/" & sListName & "]"
                oAddressMap.SourceAddress = ""
                oAddressMap.Scope = ""

                If oAddressMap.RemoveAddressMap() <> 1 Then
                    ' Address Map removal failed!
                    Return 2
                End If

                Return 1
            End If

            Return 0
        End Function
```

### 3.4.2 MailEnable ListMember Class

**Class**
MailEnable.Administration.ListMember

**Properties**

| Wildcard | Name | Type | Description |
|---|---|---|---|
| Yes | Address | String(256) | The address of the member |
| | AccountName | String(128) | Unused |
| | ListName | String(128) | Unused |
| | ListMemberType | Long | Unused |
| | Status | Long | Unused |

**Functions**
FindFirstListMember() As Long
FindNextListMember() As Long
AddListMember() As Long
GetListMember() As Long
RemoveListMember() As Long
EditListMember(NewAddress, NewAccountName, NewListName, NewListMemberType, NewStatus) As Long

**Remarks**
Functions return a value of 1 for success and 0 for failure. Other status codes may be returned as information on errors.

Adding:

```
Dim oListMember As New MailEnable.Administration.ListMember

oListMember.AccountName = Postoffice
oListMember.Address = "[SMTP:bob@example.com]"
oListMember.ListMemberType = 0
oListMember.ListName = "mylist"
oListMember.Status = 1

oListMember.AddListMember()
```

## 3.5 Post Office Administration

### 3.5.1 MailEnable Group Class

**Class**
MailEnable.Administration.Group

**Properties**

| Wildcard | Name | Type | Description |
|---|---|---|---|
| Yes | RecipientAddress | String(1024) | The address of the group |

| | | | |
|---|---|---|---|
| | Postoffice | String(128) | The account/post office |
| Yes | GroupName | String(128) | Name of the group |
| | GroupFile | String(128) | Unused |
| | GroupStatus | Long | 0=Disabled, 1=Enabled |

**Functions**
FindFirstGroup() As Long
FindNextGroup() As Long
AddGroup() As Long
GetGroup() As Long
RemoveGroup() As Long
EditGroup(ByVal NewRecipientAddress As String, ByVal NewPostoffice As String, ByVal NewGroupName As String, ByVal NewGroupFile As String, ByVal NewGroupStatus As Long) As Long

**Remarks**
Functions return a value of 1 for success and 0 for failure. Other status codes may be returned as information on errors.

#### 3.5.1.1 .NET Programming Examples

Iterating:

```
Public Sub ListGroups()

	Dim sPostoffice As String = "MailEnable"
	Dim oGroup As New MailEnable.Administration.Group
	Dim sGroupName As String
	Dim sRecipientAddress As String
	Dim iGroupStatus As Integer

	oGroup.Postoffice = sPostoffice
	oGroup.GroupFile = ""
	oGroup.GroupName = ""
	oGroup.GroupStatus = -1
	oGroup.RecipientAddress = ""
	oGroup.Host = ""

	If oGroup.FindFirstGroup() = 1 Then
		Do
			' Group Name
			sGroupName = oGroup.GroupName

			' Enabled status
			iGroupStatus = oGroup.GroupStatus

			' Group Recipient Address
			sRecipientAddress = oGroup.RecipientAddress


			oGroup.Postoffice = sPostoffice
			oGroup.GroupFile = ""
			oGroup.GroupName = ""
			oGroup.GroupStatus = -1
			oGroup.RecipientAddress = ""
```

```
            Loop While (oGroup.FindNextGroup() = 1)
      End If
End Sub
```

Adding:

```
Public Function CreateAddressMapsForAllDomains(ByVal Postoffice As
String, ByVal AliasName As String, ByVal TargetAddress As String) As
Integer

      Dim oDomain As New MailEnable.Administration.Domain
      Dim oAddressMap As New MailEnable.Administration.AddressMap
      Dim sMappedAddress As String

      oDomain.AccountName = Postoffice
      oDomain.DomainName = ""
      oDomain.Status = -1
      oDomain.DomainRedirectionHosts = ""
      oDomain.DomainRedirectionStatus = -1

      If oDomain.FindFirstDomain() = 1 Then
            Do
                  sMappedAddress = "[SMTP:" & AliasName & "@" &
            oDomain.DomainName & "]"

                  oAddressMap.Account = Postoffice
                  oAddressMap.DestinationAddress = TargetAddress
                  oAddressMap.SourceAddress = sMappedAddress
                  oAddressMap.Scope = 0

                  If oAddressMap.AddAddressMap = 1 Then
                        ' Address Map added!

                  End If

                  ' Now let's try to get the next domain
                  oDomain.AccountName = Postoffice
                  oDomain.DomainName = ""
                  oDomain.Status = -1
                  oDomain.DomainRedirectionHosts = ""
                  oDomain.DomainRedirectionStatus = -1

            Loop While (oDomain.FindNextDomain() = 1)
      Else
            Return 0
      End If

      Return 1
End Function


Public Function AddGroup() As Integer
```

```
        Dim sPostoffice As String = "MailEnable"
        Dim oGroup As New MailEnable.Administration.Group
        Dim sGroupName As String = "Group1"
        Dim bEnabled As Boolean = True

        oGroup.Postoffice = sPostoffice
        oGroup.GroupName = sGroupName
        oGroup.RecipientAddress = "[SF:" & sPostoffice & "/" &
sGroupName & "]"

        If bEnabled Then
              oGroup.GroupStatus = 1
        Else
              oGroup.GroupStatus = 0
        End If

        If oGroup.AddGroup() = 1 Then ' Success!
              '
              ' Now add the address maps
              '
              If CreateAddressMapsForAllDomains(sPostoffice,
        oGroup.GroupName, oGroup.RecipientAddress) = 1 Then

              End If
        End If
End Function
```

Removing:

```
Public Function RemoveGroup() As Integer
        Dim sPostoffice As String = "MailEnable"
        Dim oGroup As New MailEnable.Administration.Group
        Dim sGroupName As String = "Group1"
        Dim bEnabled As Boolean = True

        oGroup.Postoffice = sPostoffice
        oGroup.GroupName = sGroupName
        oGroup.RecipientAddress = ""
        If bEnabled Then
              oGroup.GroupStatus = 1
        Else
              oGroup.GroupStatus = 0
        End If


        If oGroup.RemoveGroup() = 1 Then    ' Success!
              '
              ' We need to delete any address maps
              '
              Dim oAddressMap As New
        MailEnable.Administration.AddressMap
              oAddressMap.Account = sPostoffice
              oAddressMap.DestinationAddress = "[SF:" & sPostoffice &
        "/" & sGroupName & "]"
              oAddressMap.SourceAddress = ""
              oAddressMap.Scope = ""

              If (oAddressMap.RemoveAddressMap() <> 1) Then
                    ' Address Map removal failed!...
                    Return 0
```

```
                End If

                Return 1
            End If
        End Function
```

### 3.5.2 MailEnable Group Member Class

**Class**
MailEnable.Administration.GroupMember

**Properties**

| Wildcard | Name | Type | Description |
|---|---|---|---|
| Yes | Address | String(256) | The address of the member |
| | Postoffice | String(128) | Unused |
| | Mailbox | String(128) | Unused |

**Functions**
FindFirstGroupMember() As Long
FindNextGroupMember() As Long
AddGroupMember() As Long
GetGroupMember() As Long
RemoveGroupMember() As Long
EditGroupMember(ByVal NewAddress As String, ByVal NewPostoffice As String, ByVal NewMailbox As String) As Long

**Remarks**
Functions return a value of 1 for success and 0 for failure. Other status codes may be returned as information on errors.

### 3.5.3 MailEnable Mailbox Class

**Class**
MailEnable.Administration.Mailbox

**Properties**

| Wildcard | Name | Type | Description |
|---|---|---|---|
| | Postoffice | String(128) | The account/post office |
| Yes | Mailbox | String(64) | The name of the mailbox |
| | RedirectAddress | String(512) | The address list to redirect all inbound email to |
| | RedirectStatus | Long | When the mailbox is redirected. 0=Disabled<br>1=Enabled<br>2=Redirect and keep a copy of the message in |

| | | | |
|---|---|---|---|
| | | | the mailbox |
| | Status | Long | 0=Disabled, 1=Enabled |
| | Limit | Long | Size limit (in kilobytes) of the mailbox |
| | Size | Long | Current size of mailbox in kilobytes. This will either be the inbox folder size or all the mail folders, depending on the quota enumeration settings configured in the administration program. Usually in order to return the current usage for a mailbox you would pass -1 when using the mailbox functions. Other options are available which may increase performance:<br>**Scope** / **Setting**<br>-2 — Don't return the mailbox size if the mailbox has unlimited quota<br>-3 — Only get the mailbox size from cached value (the DIRSIZE.TMP file), do not do any calculations<br>-4 — Don't return the mailbox size |

**Functions**
FindFirstMailbox() As Long
FindNextMailbox() As Long
AddMailbox() As Long
GetMailbox() As Long
RemoveMailbox() As Long
EditMailbox(ByVal NewPostoffice As String, ByVal NewMailbox As String, ByVal NewRedirectAddress As String, ByVal NewRedirectStatus As Long, ByVal NewStatus As Long, ByVal NewLimit As Long, ByVal NewSize As Long) As Long

**Remarks**
Functions return a value of 1 for success and 0 for failure. Other status codes may be returned as information on errors.

RedirectAddress is a semi-colon delimited list of MailEnable formatted email address, eg:
[SMTP:address1@domain.com];[SMTP:address2@domain.com]

### 3.5.3.1 .NET Programming Examples

Iterating:

```
Public Sub ListMailboxes()

    Dim sPostoffice As String = "MailEnable"
    Dim oMailbox As New MailEnable.Administration.Mailbox
    Dim sMailboxName As String

    oMailbox.Postoffice = sPostoffice
    oMailbox.MailboxName = ""
    oMailbox.RedirectAddress = ""
    oMailbox.RedirectStatus = -1
    oMailbox.Size = -1
    oMailbox.Limit = -1
```

```
        oMailbox.Status = -1
        oMailbox.Host = ""

        If oMailbox.FindFirstMailbox = 1 Then
              Do
                    sMailboxName = oMailbox.MailboxName

                    oMailbox.Postoffice = sPostoffice
                    oMailbox.MailboxName = ""
                    oMailbox.RedirectAddress = ""
                    oMailbox.RedirectStatus = -1
                    oMailbox.Size = -1
                    oMailbox.Limit = -1
                    oMailbox.Status = -1

              Loop While (oMailbox.FindNextMailbox() = 1)

        End If

  End Sub
```

Adding:

```
  Public Function AddMailbox()
        Dim sPostoffice As String = "MailEnable"
        Dim oMailbox As New MailEnable.Administration.Mailbox
        Dim oLogin As New MailEnable.Administration.Login
        Dim sMailboxName As String = "Test"
        Dim sPassword As String = "password"
        Dim sRights As String = "USER"      ' USER - User, ADMIN -
  Administrator

        oLogin.Account = sPostoffice
        oLogin.LastAttempt = -1
        oLogin.LastSuccessfulLogin = -1
        oLogin.LoginAttempts = -1
        oLogin.Password = ""
        oLogin.Rights = ""
        oLogin.Status = -1
        oLogin.UserName = sMailboxName & "@" & sPostoffice

        ' If the login does not exist we need to create it
        If oLogin.GetLogin() = 0 Then

              oLogin.Account = sPostoffice
              oLogin.LastAttempt = 0
              oLogin.LastSuccessfulLogin = 0
              oLogin.LoginAttempts = 0
              oLogin.Password = sPassword
              oLogin.Rights = sRights

              oLogin.Status = 1 ' 0 - Disabled, 1 - Enabled
              oLogin.UserName = sMailboxName & "@" & sPostoffice

              If oLogin.AddLogin <> 1 Then
                    ' Error adding the Login
```

```
                Return 0
            End If
        End If

        ' Now we create the mailbox
        oMailbox.Postoffice = sPostoffice
        oMailbox.MailboxName = sMailboxName

        ' Set the Redirect Address if applicable
        'oMailbox.RedirectStatus = 1
        'oMailbox.RedirectAddress = sRedirectAddress

        oMailbox.Size = 0
        oMailbox.Limit = -1     ' -1 - Unlimited OR size value (in KB)
        oMailbox.Status = 1

        If oMailbox.AddMailbox() <> 1 Then
            ' Failed to add mailbox
            Return 0
        End If


        ' Now we need to add the Address Map entries for the Account
        Dim oAddressMap As New MailEnable.Administration.AddressMap
        oAddressMap.Account = sPostoffice
        oAddressMap.DestinationAddress = "[SF:" & sPostoffice & "/" &
sMailboxName & "]"
        oAddressMap.SourceAddress = "[SMTP:" & sMailboxName &
"@mailenable.com]"
        oAddressMap.Scope = 0

        If oAddressMap.AddAddressMap() <> 1 Then
            ' Failed to add Address Map for some reason!
            Return 2
        End If

        Return 1

    End Function
```

Removing:

```
    Public Function RemoveMailbox() As Integer

        Dim sPostoffice As String = "MailEnable"
        Dim oMailbox As New MailEnable.Administration.Mailbox
        Dim sMailboxName As String = "Test"

        oMailbox.Postoffice = sPostoffice
        oMailbox.MailboxName = sMailboxName
        oMailbox.RedirectAddress = ""
        oMailbox.RedirectStatus = -1
        oMailbox.Size = -1
        oMailbox.Limit = -1
        oMailbox.Status = -1

        If oMailbox.RemoveMailbox() = 1 Then
            ' Now remove the corresponding Address Map entry
            Dim oAddressMap As New
MailEnable.Administration.AddressMap
```

```
            oAddressMap.Account = sPostoffice
            oAddressMap.DestinationAddress = "[SF:" & sPostoffice &
      "/" & sMailboxName & "]"
            oAddressMap.SourceAddress = ""
            oAddressMap.Scope = ""

            If oAddressMap.RemoveAddressMap() = 0 Then
                  ' Failed to remove Address Map for some reason!
                  Return 0
            End If

            Return 1
      End If

      Return 0

End Function
```

### 3.5.4 MailEnable Postoffice Class

**Class**
MailEnable.Administration.Postoffice

#### 3.5.4.1 Properties

| Wildcard | Name | Type | Description |
|---|---|---|---|
| Yes | Name | String(128) | Name of the postoffice |
| | Status | Long | 0=Disabled, 1=Enabled |
| Yes | Account | String(64) | Account for the postoffice |

**Functions**
GetMailRootDirectory()
GetConfigurationDirectory() As String
FindFirstPostoffice() As Long
FindNextPostoffice() As Long
AddPostoffice() As Long
GetPostoffice() As Long
RemovePostoffice() As Long
EditPostoffice(ByVal NewName As String, ByVal NewStatus As Long, ByVal NewAccount As String) As Long

**Remarks**
Functions return a value of 1 for success and 0 for failure. Other status codes may be returned as information on errors. Currently, set the Account to be the same as the postoffice name. You are able to match on the following properties:

Name
Account

### 3.5.5 SMTP Administration

**Class**
MailEnable.Administration.Domain

**Properties**

| Wildcard | Name | Type | Description |
|---|---|---|---|
| Yes | DomainName | String(512) | Domain name |
| | Status | Long | 0=Disabled, 1=Enabled |
| | DomainRedirectionStatus | Long | Whether redirection for domain is active<br>0=Disabled<br>1=Enabled<br>2=Redirect from authenticated senders only |
| | DomainRedirectionHosts | String(2048) | List of hosts to redirect to. This is a comma delimited list of the host addresses. |
| Yes | AccountName | String(128) | Account/post office |

**Functions**
AddDomain() As Long
GetDomain() As Long
EditDomain(ByVal NewDomainName As String, ByVal NewStatus As Long, ByVal NewDomainRedirectionStatus As Long, ByVal NewDomainRedirectionHosts As String, ByVal NewAccountName As String) As Long
RemoveDomain() As Long
FindFirstDomain() As Long
FindNextDomain() As Long

**Remarks**
Functions return a value of 1 for success and 0 for failure. Other status codes may be returned as information on errors.

#### 3.5.5.1 .NET Programming Examples

Iterating:

```
Public Sub ListDomains()
      Dim oDomain As New MailEnable.Administration.Domain
      Dim oListItem As ListItem
      Dim sPostoffice As String = "MailEnable"
      Dim sDomainName As String

      oDomain.AccountName = sPostoffice
      oDomain.DomainName = ""
      oDomain.Status = -1
      oDomain.DomainRedirectionHosts = ""
      oDomain.DomainRedirectionStatus = -1

      If oDomain.FindFirstDomain() = 1 Then
```

```
                Do
                        sDomainName = oDomain.DomainName

                        oDomain.AccountName = sPostoffice
                        oDomain.DomainName = ""
                        oDomain.Status = -1
                        oDomain.DomainRedirectionHosts = ""
                        oDomain.DomainRedirectionStatus = -1
                Loop While (oDomain.FindNextDomain() = 1)
        End If
End Sub
```

Adding:

```
Public Function AddDomain() As Integer

        Dim oDomain As New MailEnable.Administration.Domain
        Dim sPostoffice As String = "MailEnable"
        Dim sDomainName As String = "TestDomain"
        Dim bEnabled As Boolean = True
        Dim sRedirectionHost As String = "www.myredirectedhost.com"

        '
        ' RedirectionStatus can be 1 of:
        '       0 - Disabled, 1 - Enabled, 2 - Authenticated
        '
        Dim iRedirectionStatus As Integer = 1

        oDomain.AccountName = sPostoffice
        oDomain.DomainName = sDomainName
        oDomain.Status = -1
        oDomain.DomainRedirectionHosts = ""
        oDomain.DomainRedirectionStatus = -1

        If oDomain.GetDomain() <> 1 Then
                ' Domain doesn't exist so we're free to add it

                If bEnabled Then
                        oDomain.Status = 1
                Else
                        oDomain.Status = 0
                End If

                oDomain.DomainRedirectionHosts = sRedirectionHost
                oDomain.DomainRedirectionStatus = iRedirectionStatus

                If oDomain.AddDomain() = 1 Then
                        ' Add the relevant address map entries (if
                        applicable)...
                        Dim oAddressMap As
                        MailEnable.Administration.AddressMap
```

```
        Dim sPostmasterAccount As String = "Postmaster"
        Dim sAbuseAccount As String = "Abuse"
        Dim sCatchAllAccount As String = "CatchAll"

        '
        ' Add the Postmaster Address Map
        '
        oAddressMap = New
MailEnable.Administration.AddressMap
        oAddressMap.Account = sPostoffice
        oAddressMap.DestinationAddress = "[SF:" &
        sPostoffice & "/" & sPostmasterAccount & "]"
        oAddressMap.SourceAddress = "[SMTP:Postmaster@" &
        sDomainName & "]"
        oAddressMap.Scope = "0"

        If oAddressMap.AddAddressMap() <> 1 Then
              ' Can't add address map for some reason...
        End If

        oAddressMap = Nothing
```

```
                '
                ' Add the Abuse Address Map
                '
                oAddressMap = New
            MailEnable.Administration.AddressMap

                oAddressMap.Account = sPostoffice
                oAddressMap.DestinationAddress = "[SF:" &
                sPostoffice & "/" & sAbuseAccount & "]"
                oAddressMap.SourceAddress = "[SMTP:Abuse@" &
                sDomainName & "]"
                oAddressMap.Scope = "0"

                If oAddressMap.AddAddressMap() <> 1 Then
                      ' Can't add address map for some reason...
                End If

                oAddressMap = Nothing


                '
                ' Add the Catch-All Address Map
                '
                oAddressMap = New
            MailEnable.Administration.AddressMap

                oAddressMap.Account = sPostoffice
                oAddressMap.DestinationAddress = "[SF:" &
                sPostoffice & "/" & sCatchAllAccount & "]"
                oAddressMap.SourceAddress = "[SMTP:*@" &
                sDomainName & "]"
                oAddressMap.Scope = "0"

                If oAddressMap.AddAddressMap() <> 1 Then
                      ' Can't add address map for some reason...
                End If

                oAddressMap = Nothing
                Return 1

            Else
                ' Domain add failed!
                Return 0
            End If

        Else
            ' Domain already exists!
            Return 0
        End If

    End Function
```

Removing:

```
Public Function RemoveDomain() As Integer
      Dim oDomain As New MailEnable.Administration.Domain
      Dim sPostoffice As String = "MailEnable"
      Dim sDomainName As String = "TestDomain"

      oDomain.AccountName = sPostoffice
      oDomain.DomainName = sDomainName
      oDomain.Status = -1
      oDomain.DomainRedirectionHosts = ""
      oDomain.DomainRedirectionStatus = -1

      If oDomain.RemoveDomain() = 1 Then
            ' Domain removed successfully
            Return 1
      Else
            ' Domain removal failed
            Return 0
      End If
End Function
```

### 3.5.6 MailEnable SystemOption Class

**Class**
MailEnable.Administration.SystemOption

**Properties**

| Wildcard | Name | Type | Description |
|---|---|---|---|
| | Scope | Long | 0=System Wide Value<br>1=Post Office Value<br>2=Mailbox Value |
| | Query | String(512) | Query string to denote the URI for the value to be retrieved/set (according to the Scope):<br><br>Scope / Setting<br>0 Category<br>1 PostofficeName<br>2 Postoffice/Mailbox<br><br>If you are trying to read or set a postoffice level option, you would use the postoffice name in this property. If you are after a mailbox level option, you would format this as postoffice/mailbox. The value you use for a system wide option would depend on the option itself. Please see the valid option list for what to use for these. |
| | ValueName | String(512) | The name of the value or setting to be set/retrieved |
| | Value | String(2048) | The value of the setting either to be set or |

| | | | |
|---|---|---|---|
| | | | retrieved. |

**Functions**
SetOption() As Long
GetOption() As Long

**Example:**

This example outlines how to enable Web Administration for a Post Office:

```
Dim oMEOption As New MailEnable.Administration.SystemOption
oMEOption.Query = "MailEnable" 'Postoffice Name
oMEOption.Scope = 1
oMEOption.ValueName = "WebAdmin-Enabled"
oMEOption.Value = 1 '1=On, 0=Off
oMEOption.SetOption
Set oMEOption = Nothing
```

**Remarks**
Functions return a value of 1 for success and 0 for failure. Other status codes may be returned as information on errors. For options which have the same ValueName for a postoffice or mailbox level option, the postoffice option will always overwrite the mailbox level one. For example, if you are setting the SMTP usage restriction for a postoffice, then the mailbox values will never be checked. Only if the postoffice level option is disabled will the mail server check the mailbox level option for usage restrictions.

This API results in settings being stored in the following locations:

- Postoffice specific settings are stored in the POSTOFFICE.SYS file under the Mail Enable\Config\Postoffices\[postoffice] directory.
- Mailbox specific settings are stored in the Mail Enable\Config\Postoffices\[postoffice]\mailboxes\[mailbox].sys file.
- System settings are stored in the {CategoryName}.SYS file under the Mail Enable\Config directory.

The following table contains a list of the value names and their respective meanings for Postoffice level options:

| Postoffice Value Name | Description |
|---|---|
| MappedDomainEnabled | Determines whether the specified postoffice is mapped to a Windows Domain |
| MappedDomain | Specifies the Windows Security Realm to be used for Integrated Authentication |
| UPNEnabled | Specifies whether Integrated Authentication for this postoffice supports UPN (User@securitydomain) formatted usernames |
| WindowsAuthenticationEnabled | Specifies whether the postoffice is mapped to a Windows Security Realm for Integrated Authentication |
| WindowsAccountAutoCreation | Specifies whether accounts are to be automatically created if the user has authenticated using Integrated Authentication |
| WebAdmin-Enabled | Specifies whether Web Administration is available for ADMIN and SYSADMIN users of this Post Office. |
| WebAdmin-CanEditMailboxes | Determines whether ADMIN and SYSADMIN users are able to edit details of Mailboxes |
| WebAdmin-MaxMailboxes | Determines the maximum number of Mailboxes that can be added via Web Administration for this Post Office. |
| WebAdmin-DefaultMailboxSize | Determines the maximum Mailbox Quota Size that can be specified |

| | |
|---|---|
| | when creating or modifying a Mailbox |
| WebAdmin-CanEditMailboxSize | Determines whether ADMIN or SYSADMIN users are able to specify Mailbox sizes |
| WebAdmin-CanEditLists | Determines whether ADMIN or SYSADMIN users are able to edit the details for lists |
| WebAdmin-MaxLists | Determines the maximum number of lists that can be managed by an ADMIN or SYSADMIN user |
| WebAdmin-MaxListMembers | Determines the maximum number of list members that can be assigned to a list |
| WebAdmin-CanEditDomains | Specifies whether ADMIN or SYSADMIN users are able to configure domain details via Web Administration |
| SMTP-Inbound-Message-UsageRestrictionEnabled | 0=Postoffice has no throughput restrictions<br>1=Postoffice has message throughput restrictions |
| SMTP-Inbound-Message-UsageRestriction | The number of messages per hour the postoffice can send |
| SMTP-Inbound-Message-Usage | The number of messages that have been sent since a certain time. This is formatted as:<br>time:count<br>Time is when the message usage count began and is the number of seconds elapsed since midnight, January 1, 1970. Count is the number of recipients sent to by the postoffice since Time. The mail services will reset the time value to the current time when it is more than 60 minutes old. If a recipient could not be sent to since it is has reached the usage restriction value, then the count is still increased. |

The following table contains a list of the value names and their respective meanings for mailbox specific values:

| Mailbox Value Name | Description |
|---|---|
| ActiveSync | 0=ActiveSync disabled<br>1=ActiveSync enabled |
| Antivirus | 0=Antivirus scanning disabled<br>1=Antivirus scanning enabled |
| AutoResponderFinishTime | The end date of when the autoresponder will reply to messages. It is formatted as YYYYMMDDTHHMMSS where T is the letter "T". This time is not UTC, but in the timezone of the mailbox. If the timezone of the mailbox is not set, then the server timezone is applied to determine whether the autoresponder will execute. |
| AutoResponderStartTime | This is the start date of when the autoresponder will reply to messages. It uses the same date and time format as AutoResponderFinishTime. |
| AutoResponderRestrictionState | 0=Autoresponder is not restricted to certain times<br>1=Autoresponder is restricted to certain times |
| CharSet | Default charset |
| ReplyAddress | The default SMTP reply address |
| DisplayName | The friendly name/display name |
| MailBox-DropEventStatus | 0=Mailbox delivery event disabled<br>1=Mailbox delivery event enabled |
| MailBox-DropEvent | The mailbox delivery event to execute |
| SMTP | 0=SMTP disabled<br>1=SMTP enabled |
| POP | 0=POP disabled<br>1=POP enabled |
| HTTPMail | 0=HTTPMail disabled<br>1=HTTPMail enabled |
| WebMail | 0=Webmail disabled<br>1=Webmail enabled |
| IMAP | 0=IMAP disabled<br>1=IMAP enabled |

| | |
|---|---|
| MAPI | 0=Outlook MAPI Connector access disabled<br>1= Outlook MAPI Connector access enabled |
| Priority | 0=Messages are sent normally for this mailbox<br>1=Messages sent are placed in the Priority SMTP queue |
| TimeZone | Timezone for mailbox |
| AutoSignitureStatus | 0=Autosignature disabled<br>1=Autosignature enabled |
| DefaultAddress | Default SMTP address for the mailbox |
| MsgFormat | TEXT=Use text editing for webmail<br>HTML=Use HTML editing for webmail |
| WebMail-UseDeletedItemsFolder | Deleting messages moves them to deleted items folder in webmail |
| WebMail-ClearDeletedOnLogOff | Deletes the contents of the deleted items folder in webmail when the user logs off |
| WebMail-AllowNewWindows | 0=Open messages within inbox frame<br>1=Open messages in a new window |
| SMTP-Inbound-Message-UsageRestrictionEnabled | 0=User has no throughput restrictions<br>1=User has message throughput restrictions |
| SMTP-Inbound-Message-UsageRestriction | The number of recipients per hour the mailbox can send to |
| SMTP-Inbound-Message-Usage | The number of messages that have been sent since a certain time. This is formatted as:<br>time:count<br>Time is when the message usage count began and is the number of seconds elapsed since midnight, January 1, 1970. Count is the number of recipients sent to by the mailbox since Time. The mail services will reset the time value to the current time when it is more than 60 minutes old. If a recipient could not be sent to since it is has reached the usage restriction value, then the count is still increased. |
| MailBox-MailboxRulesSatus | 0=Mailbox filters disabled<br>1=Mailbox filters enabled |

### 3.5.7 Global Registry Options

There are many options for MailEnable that are stored in the Windows registry. The Windows registry stores options that are server specific. So if you are running a cluster of MailEnable servers you can have different settings per server. Most of the options that are server specific would be related to port and IP bindings, which would be unique across servers. The following are some of the options that can be set. The registry key location for MailEnable settings is

For 32bit Windows servers:

HKEY_LOCAL_MACHINE\SOFTWARE\Mail Enable\Mail Enable

For 64bit Windows servers:

HKEY_LOCAL_MACHINE\SOFTWARE\Wow6432Node\Mail Enable\Mail Enable

You can find the options for each server under the following keys under the main registry key branch:

| Service | Key |
|---|---|
| **Synchronisation service** | Services\HTTPMail |
| **IMAP** | Services\IMAP |
| **LDAP** | Services\LDAP |
| **POP3** | Services\POP |
| **Indexing service** | Services\Indexing |
| **Webmail** | Services\Webmail\Options |
| **Web administration** | Services\Webadmin\Options |

| | |
|---|---|
| **List server** | Connectors\LS |
| **SMTP** | Connectors\SMTP |
| **POP Retrieval** | Connectors\POP |
| **Postoffice Connector** | Connectors\SF |
| **SMS Connector** | Connectors\SMS |

| **SMTP Options** | |
|---|---|
| **Activity Log Directory** | The directory the activity logs will be saved to |
| **Activity Log Enabled** | 0=Activity log file disabled<br>1=Activity log file enabled |
| **Debug Enabled** | 0=Debug log file disabled<br>1=Debug log file enabled |
| **Disable Inbound Delivery** | 0=Enabled<br>1=SMTP listener will be disabled |
| **Disable Outbound Delivery** | 0=Enabled<br>1=Emails will not be sent from the SMTP outbound queue |
| **Restrict Inbound Message** | 0=No limit per hour<br>1=Enable per hour limit on all mailboxes |
| **Log Subject** | 0=Don't add email subjects to the log files<br>1=Add the subject of emails to the log files where possible |
| **Inbound Message Restriction** | The number of messages per hour mailboxes can send |
| **Welcome Message** | Welcome message of the SMTP service |