API Introduction Guide

Email Manager 6.0

Version Management

Version History

Version	Date	Author	Distribution
1.0	23 June 2015	L Watts and J Davis	Release

Related Documents

The related documents are located in the Alterian product help.

Name
API Help Files

Table of Contents

1 Introduction

This document is intended for users of the Email Manger API.

The following sections are included in this document:

- What is an API?
- How to call API methods
- What data is sent and returned

You can find full details of all the API calls, and how to use them in the Email Manager EMSDK Help Files.

2 What is an API?

API stands for Application Programming Interface. The Email Manager API is used as an interface to allow Email Manager to communicate with a wide range of clients.

3 Accessing the API Documentation

The API help file is a .chm help file, which you can download onto your computer. To help you, examples written in C#, Visual Basic and Visual C++ are included in the help files.

4 Email Manager Web Services Functional Descriptions

The API is currently broken down into several functional areas by web service giving access to all the functionality available in the Email Manager client software. Methods for these services, as well as additional details, are available in the help files.

EMAuthenticate

Functions for authenticating and initializing. This class allows the use of login credentials to access Email Manager client accounts. Typical steps would be Authenticate, followed byGetClientInfo and then GetModuleCategories/GetModules or GetAllModules.

EMClientManager

Functions for working with clients. A client in Email Manager is a single independent instance of a database and content server assigned to an individual client. For information on managing users and user groups within a client, see the 'EMUserManager' web service.

EMCampaignBuilder

Functions for uploading, creating, modifying, and syntax checking Creatives. A Creative in Email Manager is a term used to define all the physical assets of a campaign – such as Email Contents, Webpage Contents, images, links, variables, and dynamic content rules.

EMListManager

Functions for working with deployment lists and recipients. This class provides functionality for creating, deleting and modifying recipient lists, suppression lists, list fields, and list categories.

EMReporting

Functions for downloading reports like the ones in Quick Totals and Message Tracking modules in the Email Manager client software. Columns, group nestings (or group-by's), nesting order, and conditions may be specified to generate reports on messages, recipients, and deployments. Summary and Detail reports are both available. Report data available for these reports includes system level data such as messages sent, messages opened, bounces, unsubscribes, as well as user defined actions unique to particular Contents.

EMSendMessage

Functions for setting up and creating deployments. Deployments require a Contents template(s), a recipient or list of recipients, field mappings, and personalization. Methods of this class allow for including attachments; for scheduling and throttling; as well as attaching suppression lists. This class is useful in setting up external triggers for automated message deployments.

EMSkinInfoEx

This structure may be used to pass localization or custom skin settings to your client app. Each host name used to reference the webservice may be configured with different values for this structure. How the members of this structure are interpreted is up to the client application.

EMUserManager

Functions for adding users and user groups, modifying user information, creating and editing user permissions. These methods can also be used to set up or change passwords and usernames. Everything in Email Manager is permission based - module access, Contents access, and lists. The User permissions module is set up much like the Windows user model. Users can have various levels of permissions such as read-only access to lists, or send access to a Content, but not modify.

5 API Return Data

5.1 Initial Validation

The standard HTTP protocol return codes are used for significant errors by the EM webservice.

For example:

- Attempting to access a URL that does not exist returns an HTTP 404 / Not found error.
- Sending a string where a number is requested returns an HTTP 400 / Bad Request.
- Sending a blank where a string is required also returns an HTTP 400 / Bad Request.

The EM web service handles errors by giving returning exceptions.

If your exception contains a 500 response code, this exception is generated by the EM web service. The detail of the exception is included in the response. In this example the user has supplied the wrong user name or password. The error details can be parsed from the XML or JSON in the response:

```
HTTP/1.1 500 Internal Server Error
Cache-Control: private
Content-Type: text/xml; charset=utf-8
Server: Microsoft-IIS/7.5
X-AspNet-Version: 2.0.50727
X-Powered-By: ASP.NET
Date: Wed, 17 Apr 2013 09:37:22 GMT
Content-Length: 1457


<?xml version="1.0" encoding="utf-8"?><soap:Envelope
xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"><soap:Body><soap:Fault><faultcode>soap:Server</fau
ltcode><faultstring>System.Web.Services.Protocols.SoapException: incorrect username and
password combination. ---&gt; DMLibrary.DMException: incorrect username and password
combination.
   at DMLibrary.DMLogins.Authenticate(String ConnectionString, Int32 SessionReserve, Int32
MaxAttempts, Int32 MaxAttemptsWindow, String Login, String Password, String NewPassword,
Boolean BumpExisting, String IP, DateTime LocalTime, DMPasswordFormat PasswordFormat) in
e:\DM4Test\DEV_BUILD\DMLibrary\DMLogins.cs:line 389
   at DMWebServices.DMAuthenticate.AuthenticateEx(String Login, String Password, String
NewPassword, Boolean BumpExisting, DateTime LocalTime, DMPasswordFormat PasswordFormat) in
e:\DM4Test\DEV_BUILD\DMWebServices\Authenticate.asmx.cs:line 111
   --- End of inner exception stack trace ---
```

```
   at DMWebServices.DMAuthenticate.AuthenticateEx(String Login, String Password, String
NewPassword, Boolean BumpExisting, DateTime LocalTime, DMPasswordFormat PasswordFormat) in
e:\DM4Test\DEV_BUILD\DMWebServices\Authenticate.asmx.cs:line
120</faultstring><faultactor>https://va-dev-
dmforal.alterian.com/ws/authenticate.asmx</faultactor><detail type="18"
/></soap:Fault></soap:Body></soap:Envelope>
```


5.2 Monitoring API Traffic using Fiddler

In order to watch traffic from the client to the Web Services to see how the client consumes the web services for various features, Alterian recommends using the freeware program Fiddler. To download Fiddler and for more information, see http://www.fiddler2.com/fiddler2/.