



Mydeo Media Manager Platform API

This document contains information about our API, what functionality and data is available through it and an introduction to the core concepts concerning its use.

Nature of this document

This document is intended to provide a non-technical summary of the main functionality offered by our API, including how the API works and an explanation of what data is available through it. A more detailed, technical document and web service specification will be made available at a later date.

Important notice

This document is copyright © 2008 Mydeo Limited, all rights reserved. The information contained herein is commercially sensitive and is not for distribution to any party without the express, written signed consent of the CEO. You are not permitted to use the information in this document to gain any commercial advantage, and you are not permitted to distribute, translate, modify, share, discuss or reduce the contents of this document with any party.

Document History

Date	Author	Revision	Comments
13/10/2008	Richard Parker	1.0	Draft Version
14/10/2008	Richard Parker	1.1	Current Version

Table of Contents

Nature of this document	2
Important notice	2
Document History	2
Introduction.....	5
What can you use the API to do?	5
Whose data can you access?.....	5
How it works	6
What you need to get started.....	6
Choosing your programming language and development environment.....	6
Overview of functionality.....	7
API Functionality: summary of method calls and the data available	8
Signing In and Out	8
Validating a set of credentials.....	8
Plan details: obtaining the current plan allowances and current usage	8
Contact information: getting demographics.....	9
Contact information: setting demographics	9
Billing: getting a list of bills.....	10
Billing: getting data for a specific bill	10
Billing: get the balance outstanding on an account	11
Media Manager: obtaining a list of all the user's tags	11
Media Manager: obtaining a listing of all media files on an account.....	11
Media Manager: Filtering the list of media files by tag	11
Working with a specific media file: get media URLs	12
Working with a specific media file: determine if additional services are available	12
Working with a specific media file: getting a list of currently activated services	12
Working with a specific media file: activating/deactivating a specific service on a file	12
Working with a specific media file: getting/setting the file alias.....	13
Working with a specific media file: getting the list of tags for a specific file.....	13
Working with a specific media file: tagging a specific file.....	13
Working with a specific media file: deleting the file	13
Working with a specific media file: obtaining the current status of a file.....	13
Uploading new media files: upload a new file.....	14
Uploading new media files: uploading a file to replace another	14

Media Manager: getting the current URL table	14
Statistics: global user account statistics	15
Statistics: obtaining statistics by tag(s)	15
Tagging.....	16
Using tagging to separate your own customer's media files	16
What else can you do with the m3api?	16

Introduction

Mydeo Media Manager (or “m3”) is your online suite for hosting and delivering the content that matters to your business. Whether you’re building your brand, communicating a complex message or simply looking to generate more sales, high quality media delivery is paramount.

m3 is built on Mydeo’s Media Management Platform (“MMP”) technology, and utilises a web front-end (available at <http://m3.mydeo.com>) to allow customers to manage their media.

In this document, we introduce the concept of the m3 API (or “m3api”): an XML web services interface to let you harness the power of m3 and MMP in your applications.

What can you use the API to do?

The API is directly connected to the MMP used by m3. Using the API, you can access the same features as a user who is physically utilising the m3 web site, but from within your own software. For security reasons, we do not permit you to carry out credit card transactions or obtain information about a particular transaction via our API.

Whose data can you access?

Using the API, you will have access to your own data as well as any other customer data (provided the customer has supplied their username and password). In most cases, you will be working with your own data but we also offer the ability to impersonate another valid user providing you hold the required credentials to do so.

How it works

The m3api acts as a gateway to the MMP technology, exposing the core functionality of the m3 web interface programmatically to your developers, using a technology known as XML Web Services. This means that you can leverage the power of the m3api from virtually any programming environment or language that can read/send XML messages over the web.

What you need to get started

Before you can use the m3api, access to it must be enabled for your account. To activate, please contact our dedicated customer care team on m3@mydeo.com.

Once you have activated your account, you will be given an “API Key” which is essentially a token that you will need to pass to the API each time you send it an instruction or make a request for data.

Choosing your programming language and development environment

You will need to choose a programming language that is capable of generating, interpreting and sending and receiving XML data over the standard HTTP and HTTPS protocols. We recommend the use of Microsoft® Visual Studio 2008 or later, with either VB.NET or C#. The Visual Studio 2008 Integrated Development Environment will enable you to develop applications more rapidly, as it can be directly integrated with the API – making programming faster and more efficient.

Overview of functionality

This section contains an overview of the key functionality offered by the m3api. This page contains a complete listing of the functionality extended by the m3api. Each section represents particular call to the API.

A detailed breakdown of the information available for each API call can be found in the corresponding section further on in this document.

1. Signing-in and out
2. Validating a set of credentials
3. Plan details
 - a. Obtaining the current plan allowances and current usage
4. Contact information
 - a. Getting demographics
 - b. Setting demographics
5. Billing
 - a. Getting a list of bills
 - b. Getting data for a specific bill
 - c. Get the balance outstanding on an account
6. Media Manager
 - a. Obtaining a list of all the user's tags
 - b. Obtaining a listing of all media files on an account
 - i. Filtering the list by tag
 - c. Working with a specific media file
 - i. Get media URLs
 - ii. Determining if any additional services are available
 - iii. Getting a list of currently activated services
 - iv. Activating/Deactivating additional services
 - v. Getting/Setting the alias
 - vi. Getting/Setting tags
 - vii. Deleting the file
 - viii. Obtain the current status
 - d. Uploading new media files
 - i. Uploading a new media file
 - ii. Upload a file to replace another
 - e. Getting the current URL table
7. Statistics
 - a. Obtaining global account statistics
 - i. Getting requests and throughput data by day (summative)
 - ii. Getting detailed requests and throughput data for a particular date
 - iii. Getting detailed requests and throughput data for a date range
 - b. Obtaining statistics data for all files matching one or more tags:
 - i. Getting requests and throughput data by day (summative)
 - ii. Getting detailed requests and throughput data for a particular date
 - iii. Getting detailed requests and throughput data for a date range

API Functionality: summary of method calls and the data available

This section is intended to provide you with a brief overview of the data available to you via the API, without providing a detailed technical specification.

Signing In and Out

Each time you access the API, you are required to sign in. This is a process of authentication, and if you are successful, you will be given a unique token set to expire in 20 minutes. If you make another valid call to the m3api within this time limit, your token will be renewed for a further 20 minutes, and so on, until you logout or the token expires due to inactivity.

Every other method within the m3api will require this valid token be passed to it as part of the parameter data you send to the method. If the token has expired, you will receive an error message and you will need to repeat the sign-in process again to obtain a different token.

Validating a set of credentials

Through m3api, you can also validate another user's set of credentials. This is useful if you wish to perform method calls 'on behalf of' another m3 user (each subsequent method contains an optional parameter to perform the operation on behalf of the user account specified, instead of your own). To do this, you must supply the customer's registered email address and their password, along with a unique brand identifier (this will be given to you when the API is activated on your account).

It is important to note that any activity undertaken on another user's account will be audited in the security log.

Plan details: obtaining the current plan allowances and current usage

You can perform this operation on your own account, or on another user's account.

This method will return an XML data file containing the following information:

- Unique subscription number
- Plan name
- Bandwidth allowance (in MB)
- Storage allowance (in MB)
- Plan renewal cycle (eg. "Calendar Month").
- Unique plan identification code
- Whether or not the plan is currently 'sellable', i.e. is it active?
- Storage used (in MB)
- Throughput used (in MB)
- When the plan is due for renewal (i.e. next billing date)
- Current status (eg. Active or Expired)

Contact information: getting demographics

You can perform this operation on your own account, or on another user's account.

This method will return an XML data file containing the following information:

- Account holder information
 - Company name
 - First name
 - Last name
 - Email address
 - Contact telephone number
 - Mobile telephone number
 - A flag indicating whether the account holder is the billing contact
- Billing contact information
 - Billing address
 - Billing country

Contact information: setting demographics

You can perform this operation on your own account, or on another user's account.

This method will accept the following data and return either TRUE or FALSE to indicate whether the save operation was successful:

- Account holder information
 - Company name
 - First name
 - Last name
 - Email address
 - Contact telephone number
 - Mobile telephone number
 - A flag indicating whether the account holder is the billing contact
- Billing contact information
 - Billing address
 - Billing country

Billing: getting a list of bills

You can perform this operation on your own account, or on another user's account.

This method will return an XML data set containing a list of all bills produced. The summary information will contain:

- The unique bill number
- The date the bill was raised
- The type of bill ("Service" or "Product", but most will be "Service")
- The service period "from" and "to" dates
- The bill subtotal
- The currency of the bill (eg. "GBP", "USD" or "EUR")
- Whether the bill has been paid (true/false)

Billing: getting data for a specific bill

You can perform this operation on your own account, or on another user's account.

This method will return an XML data set containing detailed information the bill whose unique bill number you supply. The data set will contain:

- The unique bill number
- Payment terms
- Whether the bill has been paid (true/false)
- Bill Lines
 - Code
 - Item/Service Description
 - Quantity
 - Unit Price
 - Line Total
- Subtotal
- Tax (if applicable)
- Payments/Credits already made and applied to this bill
- Balance Outstanding (including any applicable tax)

NOTE: it is not possible to obtain information on the payment method used to pay the bill, or any linked credit card transactions.

Billing: get the balance outstanding on an account

You can perform this operation on your own account, or on another user's account.

This method will return an XML document containing the following information:

- Total amount outstanding
- Currency Code

Media Manager: obtaining a list of all the user's tags

For more information about tagging, see "Tagging".

You can perform this operation on your own account, or on another user's account.

This method will return an XML document containing the following data:

- Tag name
- Tag id

Media Manager: obtaining a listing of all media files on an account

You can perform this operation on your own account, or on another user's account.

This method will return an XML document containing the following data:

- Group Id
- Source file Id
- Source file name (or alias, if applied)
- Source file format
- Source file size (in Bytes)
- Source file uploaded on
- Container status

Media Manager: Filtering the list of media files by tag

For more information about tagging, see "Tagging".

You can perform this operation on your own account, or on another user's account.

This method will return an XML document as described in the method above, but will accept a list of tags as search criteria to use when generating the returned dataset.

Working with a specific media file: get media URLs

You can perform this operation on your own account, or on another user's account.

This method will return an XML document containing the following data:

- Group ID
- File ID
- File name (or alias, if applied)
- URL
- Service name

Working with a specific media file: determine if additional services are available

You can perform this operation on your own account, or on another user's account.

This method will return an XML document containing the following data:

- Group ID
- File Id
- Service ID
- Service name

Working with a specific media file: getting a list of currently activated services

You can perform this operation on your own account, or on another user's account.

This method will return an XML document containing the following data:

- Group ID
- File Id
- Service ID
- Service name

Working with a specific media file: activating/deactivating a specific service on a file

You can perform this operation on your own account, or on another user's account.

This method will return an XML document containing the following data:

- Successful (True/False)

Working with a specific media file: getting/setting the file alias

You can perform this operation on your own account, or on another user's account.

This method will return an XML document containing the following data:

- Successful (True/False)

Working with a specific media file: getting the list of tags for a specific file

You can perform this operation on your own account, or on another user's account.

This method will return an XML document containing the following data:

- File ID
- Tag ID
- Tag Name

Working with a specific media file: tagging a specific file

You can perform this operation on your own account, or on another user's account.

This method will return an XML document containing the following data:

- Successful (True/False)

Working with a specific media file: deleting the file

You can perform this operation on your own account, or on another user's account.

This method will return an XML document containing the following data:

- Successful (True/False)

Working with a specific media file: obtaining the current status of a file

You can perform this operation on your own account, or on another user's account.

This method will return an XML document containing the following data:

- Group ID
- File ID
- Current Status

Uploading new media files: upload a new file

You can perform this operation on your own account, or on another user's account.

This method will return an XML document containing the following data:

- Successful (True/False)
- Data:
 - Group ID
 - File ID
 - Rejection Reason (if applicable)

Uploading new media files: uploading a file to replace another

You can perform this operation on your own account, or on another user's account.

This method will return an XML document containing the following data:

- Successful (True/False)
- Data:
 - Group ID
 - File ID
 - Rejection Reason (if applicable)

Media Manager: getting the current URL table

You can perform this operation on your own account, or on another user's account.

This method will return an XML document containing the following data:

- Group ID
- File ID
- File name (or alias, if applied)
- URL
- Service name

Statistics: global user account statistics

Global Account Statistics are updated regularly to provide a picture of all the traffic (in the form of throughput and requests) on files held within a user account. These statistics will include data for deleted files, too. Most of these functions accept either a date range (for a period not exceeding 32 days in length) or a specific date (representing a 24h period).

Getting throughput and requests by day by date range (summative)

You can perform this operation on your own account, or on another user's account.

This method will return an XML document containing the following data:

- Date
- Aggregate number of requests
- Aggregate throughput (MB)

Getting detailed requests and throughput data for a particular date

You can perform this operation on your own account, or on another user's account.

This method will return an XML document containing the following data:

- Hour of Day
- Aggregate number of requests
- Aggregate throughput (MB)

Getting detailed requests and throughput data per file on a particular date

You can perform this operation on your own account, or on another user's account.

This method will return an XML document containing the following data:

- Hour of Day
- File name (or alias, if set)
- Aggregate number of requests
- Aggregate throughput (MB)

Statistics: obtaining statistics by tag(s)

Each of the methods in the section above will accept an optional list of tags which will be used to filter the results returned to the client.

Tagging

This section contains a description of the tagging service offered on the m3 web site and via m3api.

Tagging is the concept of applying metadata to one or more media files. The user can determine their own custom tags, and apply as many tags as they like to each file. This is the opposite of putting files in categories and subcategories, which are usually hierarchical.

In a category listing, information about the media file is inferred from the category in which it is placed. Tagging, on the other hand, is much more flexible because it is not hierarchical, and you can therefore apply tags in any combination to any file.

Using tagging to separate your own customer's media files

Tagging and the m3api are an extremely powerful combination, because they give you the ability to share your m3 account with as many users as you wish, using your software as the 'front-end'. Consider the following scenario:

- You want to build a media sharing web site, but don't have the time, money or resources to build your own media management platform, or access to a global CDN.
- Using the m3api, you can build your own web site to publish content to and from m3. You could create a single tag for each of your customers, and apply that tag automatically to each file they upload via your site.
- You can then use our 'Statistics By Tag' functionality to return statistical data only for a particular customer, so that, for example, you can bill them for usage.

What else can you do with the m3api?

Providing your use of the m3api conforms to our acceptable usage policy and terms and conditions of service, the only limit to your implementation is your imagination!