Murphy’s Magic Download API

Last updated: 3rd January 2014

Introduction

By fully integrating your website with the Murphy’s Magic Download System, you can give the impression that the downloads are fully hosted on your website. It also allows you to manage the complete user experience.

Note that this service is only suitable for experienced web developers. You would not be able to integrate the system using our API if you do not have web development skills. If you are not an experienced web developer, we suggest that you use our easy integration option which is available from the downloads admin area and takes just a few minutes to set up.

Is the API suitable for my website?

The API is designed specifically for magic shops that meet the following criteria:

- You are able to write code such as C# or PHP, or have a web developer who can do the work for you. A basic understanding of posting form data and parsing JSON is also required.
- You have a member’s area (or can develop one) that allows customers to log in to view their downloads. You will be required to authenticate your customers in order to use the system.
- If you have already integrated Murphy’s Web Services you will find this integration a lot easier as you will need to use that API to import downloads into your website (all downloads act as a normal Murphy’s product and are therefore returned by Web Services).

If you do not feel as though this API is for you, you can still sell downloads and integrate them into your website easily. Follow the instructions on the “Set up your website” area of the downloads admin page.

Understanding the basics of the API

API Key

Each call to the API must include your unique API key. It is essential that you keep this key private as it would allow anyone to make calls to the API and that could cost you money if someone used it to purchase downloads on your account.

Your API key can be found within your account area of the downloads website.

Making a Call to the API

The system using a very basic JSON API. You must simply make a HTTP POST request to the specific call's URL and you will receive JSON data in response.

For example, to get a customer's details, you would submit a POST request containing two fields: **APIKey** (this is required for all calls) and **Email** (the customer's email address) to the following URL:

http://downloads.murphysmagic.com/api/GetCustomer/

An example of the JSON data that you would receive back is:

```
{
     “ID”:1,
     “EmailAddress”:”customer@customer.com”,
     “FirstName”:”Bob”,
     “LastName”:”Customer”,
     “Password”:”asdsds4r”,
     “CustomerId”:2,
     “TimeSignedUp”:”\/Date(1352474848983)\/”
}
```

Each language handles JSON in different ways. The API always returns standard JSON and therfore should be easy to parse into your website.

Handling Errors

Errors are always presented in the same JSON format, as follows:

```
{
     “error”:”Email not passed”
}
```

Error messages are written in such a way that you could display them to the user. However, a full list of errors for each call are included at the end of this document, should you wish to catch them and rewrite them.

Testing the API

The easiest way to get started and to test the API is without our API tester tool which allows you to make real calls (with the exception of a few test calls) to the API is here:

http://downloads.murphysmagic.com/account/api/documentation/

Example User Flow

It is entirely up to you how you display the downloads within your website. However, we suggest that you use the following basic flow:

1. Add downloads during checkout

STEP 1: At the end of your checkout (after payment has been confirmed), call the **AddOrder** command to add a purchase for the customer. If you do not already have a username and password stored for the customer, you would need to have them create on during the checkout process.

STEP 2: Email the customer with a link to your downloads page and/or take them straight to that page.

2. List the customer's downloads within your account area

STEP 1: Autenticate the customer (it is assumed that you already have an account area and therefore do this within your website).

STEP 2: Display a full list of downloads that the customer has purchased using the **GetDownloadsForCustomer** command. You may also like to store this information within your own database.

STEP 3: **Video only:** Display a page that plays the video. You can get the streaming URL for a video with the **GetVideoStreamURL** call. You will need to make this call every time the user plays the video as the URL expires (this is to stop users being able to download the video stream).

STEP 4: Allow the user to download videos or ebooks by calling the GetDownloadLink command. This command will be explained further in the "Downloading videos and ebooks" section as there are a number of things that need to be taken into consideration.

3. Let us know when a customer changes their email address

STEP 1: As all downloads are authenticated based on the customer's email address, please call the **UpdateCustomerEmailAddress** command whenever the customer changes their email address.

Purchasing a Product for a Customer

When a customer purchases a download, simply call the **AddOrder** command passing the following fields:

- FirstName
- LastName
- Email
- ProductIds (a comma separated string of product IDs)

If a user with that email address already exists, we will add the purchases to their account within the download system. Otherwise, we will create a new account for them.

Because **ProductIds** is a comma-separated list, you can add as many downloads to the customers account as you like in one go. The ProductId is Murphy's product ID as provided via webservices.

A successful purchase is returned as a JSON string as follows:

```
{
    “message”:”success”
}
```

Like all calls with the API, any errors are display as follows:

```
{
    “error”:”error description”
}
```

You can view all the possible errors for this call at the end of this document.

Displaying a Customer’s Purchases

To display a customer’s purchases, you simply need to call **GetDownloadsForCustomer** with the following two fields:

- APIKey
- Email (email address of the customer)

You will receive a JSON response with a comma separated list of product IDs in this manner:

```
[45234,34555]
```

(Note that unlike the calls that we have previously discussed, this is a JSON array and not a JSON object. It is possible that your language requires you to parse it differently).

You probably already store the products in your shopping cart with the same IDs and therefore can get information about each product from your database. However, if you prefer, you can get the product info by calling the **GetDownload** command with the following POST fields:

- APIKey
- DownloadID

This returns a JSON object as follows:

```
{
    “ID”:46478,
    ”Name”:”Mirage Et Trois by Eric Jones”,
    ”CreatorName”:”Jones, Eric”,
    ”Price”:25.00,
    ”Type”:”Video”,
    ”LiveStreamStartTime”:null,
    ”NumberOfVideos”:1
}
```

This is a good time to discuss an important concept of the download system. Each download could have any number of files associated with it. For example, some DVD sets have as many as eight DVDs and therefore the download version has eight separate files; one for each DVD. You can find out how many files each download has associated with it by checking the **NumberOfVideos** paramater of the above call.

Each file is considered a “part” and therefore if you want to get the second video you would request part 2. This is covered in detail in the download section.

An ebook, however, has only one file ever associated with it.

Playing a Streaming Video

To embed the download within your website, you will need to use a Flash or HTML5 video player such as Flowplayer or JW Player. In particular, you need to use these players in RTMP mode.

Documentation for this mode for the players cane be found here:

- **Flowplayer:** http://flash.flowplayer.org/plugins/streaming/rtmp.html

- **JWPlayer:**http://www.longtailvideo.com/support/jw-player/jw-player-for-flash-v5/12535/video-delivery-rtmp-streaming

All players will require two things to play the video: the server URL and the video filename. The server URL is always the same:

rtmp://s2tvq8gbnu2j5w.cloudfront.net/cfx/st/

The video filename must be generated each time the user plays the video. This is for security and ensures that only one customer can play the video. Therefore, you must call the command **GetVideoStreamURL** with the following parameters to get the URL:

- APIKey
- DownloadID
- Email (the customer's email address)

You will receive a filename in the following format:

```
“1-1.mp4%3fExpires%3d1352569481%26Signature%3dA-WZMrfMsOSC
yQajbhzLBag4mDUdF0lFZ50AQiHwSMSboA0mhCzTU2icpGM8B%7eQ%7eHr
OEIuccVv-TKbqdRVRvTN9t9jWwPEDssrOSdavCCabaqU0rtUpNxecEVtxm
yH1JizWpwG5Syx2pj1wmQHWxD9dMkaCs1JQ4IWQfQX-Q_%26Key-Pair-
Id%3dBDKAJPA4CHYQ3EKWUVHQ”
```

Note this this is just a string and not a JSON object.

The video link expires 30 seconds after it is generated and therefore you must set your video player to auto play to ensure that the video is played within the correct amount of time.

Ebooks cannot be opened online. They must be downloaded (as per the next section).

Downloading files

Most magic vendors opt for us to watermark their files with the customers' name. Watermarking ebooks is immediate, whereas watermarking videos can take up to an hour to process.

Downloading videos
Your user interface should handle this flow for videos:

1. Customer requests to download video
2. You can display the progress of the download
3. You email the customer when the download is complete (this is achieved via a callback, which we'll cover shortly).

First, you start by calling the **GetDownloadLink** command with the following parameters:

- APIKey
- Email (email address of the customer)
- ProductID
- Part *or* FileID
- CallbackURL (optional)

A note about parts
The part is the part number of the video. For example, if the video has six separate files associated with it (see more info on how to get the number of files in the "Displaying a Customer's Purchases section) and you wanted to watermark the first part, you would enter 1 as the part.

Some downloads have multiple parts. This could be because the video is too long, it was originally on multiple DVDs, or there are different file types (such as an ebook and a video) part of the download. To get full details about the file parts, simply call **GetDownloadFiles** with DownloadID.

For ease, you can normally just pass a part ID. But if the download type (you can see the type in the **GetDownload** call) is Mixed then you would need to call **GetDownloadFiles** so that you can see the file type for each invidual file (current file types: ".mp4" and ".pdf". All functions that accept Part also accept FileID.

How this call works
This function does one of two things. If the download is not yet queued, it will queue it for the customer. If it is queued, it will return the current status of the download.

Important: it is possible that a creator may want to not allow downloading; only streaming (example: if the file is regularly updated). In this instance, you will receive an error of "Download not allowed". You can also check whether a file can be followed by checking the CanDownload parameter of the download object. *At the time of writing, all files can be downloaded but you should still check this, just in case that changes.*

The JSON response from this call is as follows:

```
{
     "PercentComplete:" 0,
     "RequestTime":"\/Date(1352474848983)\/",
     "Status":"queued"
     "DownloadLink":""
}
```

The key here is to always check the **status**. The possible options are:

- **queued** (video is waiting to be watermarked)
- **watermarking** (video is currently being watermarked. Check the PercentComplete column to see the progress)
- **ready** (the file can be downloaded. Send the user to the URL provided in DownloadLink. This will only be populated when the file is in this state)

It is important to note that the URL provided in DownloadLink will expire after a few seconds. We suggest that you create a page that authenticates the user and then redirects them (for example using Response.Redirect for ASP.NET or changing the headers for PHP) to the DownloadLink URL. This way, you can maintain the same URL (such as www.yoursite.com/download/?download=123), authenticate the user and always provide the latest URL.
If you would just like to get the status of a download (and not queue it if it isn't already queued) you can call the **GetDownloadLinkStatus** call with the same paramaters. This will not queue a download. The response is identical, however, the following status might be returned:

- **notqueued** (there is no download queued for the user)

CallbackURL
Once a download has been watermarked, you'll need to let the customer know. The easiest way to do this is to pass a callback URL to a script on your website. This URL can be a page on your site that then runs a script to email the customer.

Whatever URL you pass, we will add three querystring paramaters as follows:

- **CustomerEmail** (the customer's email address)
- **DownloadID** (the ID of the download)
- **Part** (the part number)

You will need to URLDecode the CustomerEmail paramater.

An example script (in C#) could be:

```
protected void Page_Load(object sender, EventArgs e)
{
     // The three paramaters passed by Murphy's
    if (!string.IsNullOrEmpty(Request.
QueryString["CustomerEmail"]) &&               !string.
IsNullOrEmpty(Request.QueryString["DownloadID"]) &&
!string.IsNullOrEmpty(Request.QueryString["Part"]))
    {
        string email = HttpUtility.UrlDecode(Request.
QueryString["CustomerEmail"]);

        Cust cust = GetCustomerByEmail(email);
        if (customer != null)
            {
            // Email the customer here
        }
    }
}
```

Downloading ebooks
This is the easiest option. An ebook will always have the **Status** of ready and the **DownloadLink** will always be in place. Therefore, you can simply forward the customer to the link provided in **DownloadLink**.

Like all other URLs in this system, the link will expire a few seconds after generating is so you will need to create a page to forward the user to the link (being sure to authenticate them within your system first).

Changing a Customer’s Email Address

Customers are identified by an email address. Therefore, if they ever change their address within your system, you will need to identify us. This is a simple call to **UpdateCustomerEmailAddress** with the following paramaters:

- APIKey
- OldEmail
- NewEmail

Other than an error, you will only ever receive this message:

```
{
     “message”:”success”
}
```

Live Lectures

As far as purchasing is concerned, live lectures and season passes using the same API function calls (**AddOrder**). Similarly, the same API is used to list the customer's access to a live lecture.

The only change comes when viewing the live lecture. Simply follow these two steps:

STEP 1: Call **GetLiveLectureLoginURL** with the following info:

- Email (the customer's email address)
- ProductID (the MMS product ID for the lecture)

STEP 2: This call with return a secure URL to allow the customer to view the live lecture. Embed that URL as an iFrame into your page (with a height of at least 1000px.

A few useful API calls

You may find the following API calls useful:

GetSeasonPassLectures (input: ProductID): returns an array of products within a season pass.

IsLectureLive (input: ProductID): returns true/false if a lecture is currently on air.

List of Errors

GetCustomer	
Customer not found	The email address does not exist in the database.
Email not passed	You did not pass the Email post field.
GetCustomers	
Invalid letter	You can optionally pass a letter to filter by (first letter of last name), but you did not pass a correct letter (A-Z only)
GetDownloadsForCustomer	
No downloads	This customer has not purchased any downloads.
Customer not found	The email address does not exist in the database.
Email not passed	You did not pass the Email post field.
GetDownload	
Download not found	Couldn't find a downloading matching the ID you passed.
DownloadID not passed	You did not pass the DownloadID post field.
GetVideoStreamURL	
DownloadID not passed	You did not pass the DownloadID post field.
Email not passed	You did not pass the Email post field.
Part does not exist	The part you give is more than the number of parts in the download.
Not a stream	You can't stream this download (it's likely an ebook).
DownloadID in incorrect format	The DownloadID was not an integer.
Customer not found	The customer does not exist.
GetDownloadLink *and* **GetDownloadLinkStatus**	
Part not found	The part you give is more than the number of parts in the download.
Access denied	Customer does not have permission to download this file.
Customer not found	Could not find a matching customer.
ProductID not passed	You did not pass the ProductID post field.
Email not passed	You did not pass the Email post field.
Download not allowed	The manufacturer has opted for this download to be streaming only
AddOrder	
Invalid email address	You did not pass the Email post field (or invalid address).
No first name passed	You did not pass the FirstName post field.

No last name passed	You did not pass the LastName post field.
No products submitted	You did not pass the ProductIDs post field.
UpdateCustomerEmailAddress	
OldEmail not passed	You didn’t supply the OldEmail field
NewEmail not passed	You didn’t supply the NewEmail field
NewEmail not valid	NewEmail is not a valid email address
Customer not found	The customer you enetered does not exist