ORANGELOGIC support

OrangeDAM Developer's Guide

OrangeDAM API Documentation

Applies to: OrangeDAM
Document Version: 1.4
Last updated: April 2022

Table of Contents

Overview

The OrangeDAM APIs allow developers to safely and securely extend the capabilities of their OrangeDAM platform. Through calls to its API endpoints, a OrangeDAM site can be manipulated and integrated with other software.

For each API, users need a specific security function to run the API. The assets they see in the API Response are determined by their Permissions. For example:

- In order to run the Search API, users need the Search API Security Function.
- When users run the Search API, the assets they can see in the results are assets that they have Permissions to view, access, or edit.

The following APIs are available for your use:

Authenticate API - Used for token or cookie-based authentication to a OrangeDAM site. A required first step for all other API calls.

Search API - Used to query OrangeDAM assets based on a set of criteria , and return specified record data as a response. Used in conjunction with most of APIs.

DataTable APIs - Used to map OrangeDAM “object” structure and allow CRUD (Create, Read, Update, Delete) operations on objects individually or by batches.

Media Upload API (aka “Asset Ingest API”) - Used to upload assets to OrangeDAM.

Extract Original Document API - Used to download the original, high-resolution asset.

Asset Link API - Used to embed assets and download links on external sites.

Lightbox API - Used for managing user lightboxes, which are collections of assets that a user has “favorited.”

Contacts API - Used for managing users in OrangeDAM.

Financial and Order APIs - Used to manage shopping carts, manage transactions, create payments, post royalties, and manage invoices.

Get Keywords API - A dedicated API for retrieving keywords for an asset for OrangeDAM versions prior to Helsinki. For later releases, the Search API can be used.

Package Extractor API - Used to query the content of a package. The Package Extractor API is very similar to the Search API in terms of query structure and response format. Generally, you should be using the Search API, unless you are specifically working with “albums” or “stories.”

Common Parameters

Specifying Date Formats with DateFormat parameter

You can specify the format of the date returned by any API call by adding the parameter `&DateFormat=[format]` to your call.

[format] is a single character "specifier," as determined by the DateTime formats available in the .NET programming language. Different specifiers invoke different date formats.

Example:
`https://www.sitename.com/API/DataTable/v2.2/Document.Image.Default:Read?CoreField.System-ID=DM4321&DateFormat=F`

Result: Tuesday, June 11, 2019 5:50:06 PM

Example:
`https://www.sitename.com/API/Search/v3.0/search?query=SystemIdentifier:OLSI8597&fields=CreateDate&DateFormat=s`

Result: 2019-06-11T17:50:06

The full list of available format specifiers are in the table below:

Specifier	Description	Example
"d"	Short date pattern.	6/15/2009
"D"	Long date pattern.	Monday, June 15, 2009
"f"	Full date/time pattern (short time).	Monday, June 15, 2009 1:45 PM
"F"	Full date/time pattern (long time).	Monday, June 15, 2009 1:45:30 PM
"g"	General date/time pattern (short time).	6/15/2009 1:45 PM
"G"	General date/time pattern (long time).	6/15/2009 1:45:30 PM
"M", "m"	Month/day pattern.	June 15
"O", "o"	Round-trip date/time pattern.	2009-06-15T13:45:30.0000000-07:00

"R", "r"	RFC1123 pattern.	Mon, 15 Jun 2009 20:45:30 GMT
"s"	Sortable date/time pattern.	2009-06-15T13:45:30
"t"	Short time pattern.	1:45 PM
"T"	Long time pattern.	1:45:30 PM
"u"	Universal sortable date/time pattern.	2009-06-15 13:45:30Z
"U"	Universal full date/time pattern.	Monday, June 15, 2009 8:45:30 PM
"Y", "y"	Year month pattern.	June, 2009
Any other single character	Unknown specifier.	Returns date/time in default format, which may be different for each API.

The localization of dates is based on the settings of the server. All OrangeDAM servers are by default set to United States of America (en-US), so the dates and language will therefore return formatted for that locale. For example:

- **6/15/2009** rather than **2009/6/15** (China, Japan)
- **Monday, June 15, 2009** rather than **Montag, 15. Juni 2009** (Germany)
- **1:45 PM** rather than **13:45** (France)

For further reading on the standard date specifiers, please refer to Microsoft .NET docs.

Authentication API

There are two methods for authenticating on the API in order to use a specific account privilege: **token** and **cookie**.

Using a token obtained through the Login API

The Login API is accessible through the page:

```
https://sitename.com/API/Authentication/v1.0/Login?Login=mylogin&Password
=mypassword
```

where:

- sitename.com (or subdomain.sitename.com) is your website address (if the site URL is in the format sitename.com/site please contact Orange Logic to obtain an alternative URL)
- mylogin and mypassword are replaced by your account credentials.

This URL will return a token, for example:

```
<Result>
   <APIRequestInfo>
     <Module>Authentication</Module>
     <APIVersion>v1.0</APIVersion>
     <Resource>Login</Resource>
     <Parameters>
     <Login>mylogin</Login>
     </Parameters>
     <ProviderVersion>5.0.57.0</ProviderVersion>
     <ProviderIdentity>IP-0ADA45BF</ProviderIdentity>
   </APIRequestInfo>
   <APIResponse>
     <Code>SUCCESS</Code>
     <Token>gtvprt45l1mpm5454l1gqqji</Token>
   </APIResponse>
</Result>
```

The token can then be added to the API parameters (for example: `&token=gtvprt45l1mpm5454l1gqqji`) in order to use the account privileges.

If you remain on the same web browser, a session cookie is created and you therefore do not need to add the token for the complete duration of the session (see below: Using the session cookie)

There is a tag in the API response indicating the authentication state (`IsLoggedIn`: True or False).

> Orange Logic recommends that the Login and Password parameters are passed to the Login API using a POST request, as it provides increased security.

Using the session cookie

Whether you sign in on the website (login page) or through the Authenticate API, your web browser will create a Session Cookie that is valid for both API calls and website usage.

The Session Cookie is valid for the complete duration of the session, so once created, you no longer need to add the token to the API parameters if you remain on the same web browser.

If an incorrect token and a correct cookie are provided in the same request, the API will use the session of the cookie.

There is a tag in the API response indicating the authentication state (`IsLoggedIn`: True or False).

Token/Session time-out

In the default configuration, when logging in through the website, your session will expire after 60 minutes of inactivity (includes activity on the website and through the APIs).

There is a tag in the API response with this information:

```
<TimeoutPeriodMinutes type="Numeric">60</TimeoutPeriodMinutes>
```

This time-out delay (aka **Time To Live** or **TTL**) may be different in your site configuration, as it can be changed to any duration up to a maximum of 24 hours.

By default, the 60-minute duration starts from last usage (as opposed to starting from the time of login).

An alternate behavior is to have the token/session valid for a 24-hour period regardless of activity. When this alternate behavior is in place and you log in through the Authentication API, the API response will include the session end date:

```
<SessionEndDate type="DateTime">2014-05-14T12:29:13</SessionEndDate>
```

The tokens can only be used on HTTP requests coming from the IP address they were originally requested from.

Encoding special characters

Certain special characters in parameter values (e.g. in mylogin or mypassword) must be URL encoded using **percent-encoding**.

The reserved characters as defined by RFC 3986 are:

!	#	$	&	'	(	)	*	+	,	/	:	;	=	?	@	[	]
%21	%23	%24	%26	%27	%28	%29	%2A	%2B	%2C	%2F	%3A	%3B	%3D	%3F	%40	%5B	%5D

Other commonly percent-encoded characters are:

newline	space	"	%	-	.	<	>	\	^	_	`	{	|	}	~
%0A *or* %0D *or* %0D%0A	%20	%22	%25	%2D	%2E	%3C	%3E	%5C	%5E	%5F	%60	%7B	%7C	%7D	%7E

For a full list of percent-encoded characters, visit http://www.w3schools.com/tags/ref_urlencode.asp, and refer to the column **From UTF-8**.

Search API

Authentication

Authentication to use the Search API can use a cookie or a token requested through the Authentication API.

Accessing the API

The Search API is accessible through the page
https://www.sitename.com/API/search/v3.0/search?

where **www.sitename.com** is your website address.

To access the Search API using your account privileges, add the Login API token to your request URL.

For example:

https://www.sitename.com/API/search/v3.0/search?query=MediaType:Image&fields=MediaNumber&token=gtvprt45l1mpm5454l1gqqji

Important Note:

In latest OrangeDAM release, the security point "Access to the search API" in family API, sub family "Search" must be given for the API user to be allowed to use the Search API.

In older OrangeDAM releases, the Search API also works without a login, in which case it only returns media available to the general public (based on media ranking levels set by the database administrator)

GET and POST requests

The maximum URL length for a GET request is 260 characters.

The Search API can also be accessed using POST requests: the parameters can be passed in the https header or in the URL. If the same parameter is present in both, the value in **https** header will override the URL.

API query

General Syntax

An API query is made with the following syntax:

- Your query criteria, prefixed with **query=**
- The list of fields you require in the response, prefixed with **&fields=**

Example: https://www.sitename.com/API/search/v3.0/search?**query=**Criteria1:Value Criteria2:Value**&fields=**Field1,Field2

Notes:

1. Do not repeat multiple “query=” calls in a same request. To combine multiple criteria, separate them with operators.

> Correct Syntax:
> https://www.sitename.com/API/search/v3.0/search?
> query=keyword:flower MediaType:Image Color:True Orientation:Portrait
> &fields=SystemIdentifier
>
> Incorrect Syntax:
> https://www.sitename.com/API/search/v3.0/search?
> query=keyword:flower
> query=MediaType:Image Color:True Orientation:Portrait
> &fields=SystemIdentifier

2. Criteria, Parameters, Operators and Fields are **not** case-sensitive. For example, a field parameter could be written as FIELD, Field, FieLD, etc., and it will work fine. Best practice, however, is to write all the elements in lowercase for consistency and readability.

Query Operators

Operators available for combining values within the same criterion, or across different criteria, are listed in the table below.

Operators take the following priority order: NOT > AND > OR

Example:

A AND B OR C AND D will be treated by the Search API as (A AND B) OR (C AND D)

A AND NOT B AND C will be treated by the Search API as A AND (NOT B) AND C

OR	To make an “OR” search, separate criteria by OR (with a space before and after the OR).	MediaType:Video OR Image AND Keyword:beach OR sea OR sun This retrieves Videos or Images with keywords beach or sea or sun.
AND	To make an “AND” search, separate criteria by AND or by a space. AND is the default operator: ● Criteria1:Value Value is equivalent to Criteria1:Value AND Value ● Criteria1:Value Criteria2:Value is equivalent to Criteria1:Value AND Criteria2:Value	
NOT	To exclude a criterion Can be used with AND or OR: ● AND NOT (or space NOT) ● OR NOT	Keyword:beach NOT blue This retrieves media with the keyword “beach” that do not have the keyword “blue”.
( )	You can use brackets to perform advanced boolean searches. For example:	Keyword:(flower AND blue) NOT iris This retrieves media with keywords “flower” and “blue” but that do not have the keyword

	• Criteria1:(Value OR Value) AND Value • (Criteria1:(Value) OR Criteria2(Value)) AND Criteria3:(Value)	"iris" (MediaType:Album OR MediaType:Image) AND (Text:flower AND (red OR green OR blue) NOT tulip) This retrieves images or albums that contain flower and either red, green or blue, but that do not contain tulip

Query Criteria

To retrieve a list of all query criteria available on your site, use the following API call:

https://www.sitename.com/API/search/v3.0/ListCriteria

Standard criteria are detailed in the table below:

Criteria	**Description**	**Example queries**
MediaType	Values: • Image • Video • Audio • Multimedia • Album • Story Notes: • Album = Package / Virtual Folder • Story = Folder • Multimedia = Other (e.g. PDFs, MS documents, Graphic design etc.)	query=MediaType:Media query=MediaType:Video OR MediaType:Image
DocSubType	Each Media Type can have multiple SubTypes. Examples values: Story (Folder) can have the following subtypes: • Library Folder	query=DocSubType:Video segment query=DocSubType:Library Folder &fields=MediaEncryptedIdentifier -> will retrieve the Encrypted Identifier

	• Standard Folder • Upload Folder • User Folder Video can have the following subtypes: • Standard Video • Video segment Other can have the following subtypes: • Other • Release document • Supporting file *Note: this query criteria is available with sites in OrangeDAM Version "Helsinki" and later.*	for the Library Folder
Path	Search for a document (folder/asset) using its Path. Example Path for a folder: Library/California/Diagrams Example Path for an asset: Library/California/Diagrams/Mini-Icons.psd You can also use a wildcard (*) to get all the assets in a given Path. Example: Library/California* *Note: this query criteria is available with sites in OrangeDAM Version "Helsinki" and later.*	query=Path:Library/California/Diagrams query=Path:Library/California/Diagrams/Mini-Icons.psd query=Path:Library/California* query=Path:Library &fields=MediaEncryptedIdentifier -> will retrieve the Encrypted Identifier for the Library Folder
Text	Free text search Adding quotation marks, you can search on exact text	query=Text:red flower Monet -> will search for "red" and "flower" and "Monet" in title, description, keywords, artist, and other searchable fields.

		query=Text:"red flower in a vase painted by Claude Monet" -> will search for assets with the exact phrase in the title, description, and other searchable fields.
Artist	Artist (Photographer, Author, Filmmaker, etc.) for the Media Format: FirstName LastName	query=Artist:John Doe
Keyword	Searches images by a specific keyword. You can specify the Keyword Type (e.g. searching on Orange as a color (Keyword Type "Common"), not Orange as a county (Keyword Type "Geography") When specifying a Keyword Type, use the Keyword Type Code, not the Keyword Type Label. If the Keyword Type Code contains spaces, use quotation marks. For example: Keyword(Point of Interest):Restaurant Note: when a Keyword Type is not specified, the API will search on the keyword in all Keyword Types. For example Keyword:orange will retrieve results when orange is Keyword(Common) or Keyword(Geography).	query=Keyword:beach query=Keyword:beach AND sport query=Keyword:beach OR sea query=Keyword(Common):orange AND NOT Keyword(Geography):orange
NativeKeyword	These are keywords directly assigned to assets (as opposed to inherited by the thesaurus structure). This functionality is not to be used for general searches but only for third party working on the keywording of the images.	If we want to look at the images that have the keyword France specifically added to the image then you would use: query=NativeKeyword:France Even if Paris is a narrow term of France, images with Paris as a keyword would not appear in the NativeKeyword

		search for France. They would appear in a normal Keyword search for France.
SystemIdentifier	Unique System Identifier for the Media generated by the application.	query=SystemIdentifier:OLSI8597
MediaNumber	Media Number specified in the install (this varies from one install to another: “Picture Number”/”Package Number” or “Accession Number” or “URN” etc.). Your site administrator will tell you how the Media Number is labeled on your site.	query=MediaNumber:OLKAFA22-04 query=MediaNumber: OLKAFA22-04 OROLKAFA22-05
Color	Supported values: ● True ● False When set to True, media records are part of the result only if they are color media When set to False, media records are part of the result only if they are Black and White By default, no value is specified.	query=Color:True
Orientation	Supported values : ● Landscape ● Portrait ● Square	query=Orientation:Portrait query=Orientation:Landscape OR Square
OriginalSubmissionNumber	The original Submission that the Media is part of.	query=OriginalSubmissionNumber:OLST25
CreateDate	This is the date the record was created in the database. The CreateDate criteria follows the same behavior as MediaDate. You can also add Time value, in the format YYYY-MM-DDTHH.MM.SS (e.g. 1977-04-22T06.00.00), using	query=CreateDate>:2012-12-30T06.00.00 CreateDate<:2012-13-30T12.00.00 -> will search for Media created between December 30th 2012 at 6am and December 30th 2012 at 12 noon.

<table>
<tr><td></td><td>the 24-hour clock..</td><td></td></tr>
<tr><td>EditDate</td><td>This is the date the record was last edited in the database.

The EditDate criteria follows the same behavior as MediaDate.

You can also add Time value, in the format YYYY-MM-DDTHH.MM.SS (e.g. 1977-04-22T06.00.00), using the 24-hour clock.</td><td>query=EditDate>:2012-12-30T06.00.00 EditDate<:2012-13-30T12.00.00

-> will search for Media last edited between December 30th 2012 at 6am and December 30th 2012 at 12 noon.</td></tr>
<tr><td>MediaDate</td><td>This is the date of the Media itself (not the date the Media was imported into the database)

This is to be searched in the format YYYY-MM-DD
MM and DD are optional:

● If DD is not provided, the search is made on the full month for YYYY-MM
● If MM-DD are not provided, the search is made on the full year for YYYY (01-01 to 12-31)

The operators are:

<table><tr><td>:</td><td>Equal to, including the value/date provided</td></tr><tr><td>></td><td>Higher than/Posterior to, excluding the value/date provided</td></tr><tr><td><</td><td>Lower than/Prior to, excluding the value/date provided</td></tr><tr><td>>:</td><td>Higher than/Posterior to, including the value/date provided</td></tr><tr><td><:</td><td>Lower than/Prior to, including the value/date provided</td></tr></table></td><td>Exact date:
query= MediaDate:2012-12-31

-> will search for Media dated December 31st 2012

query=MediaDate:1955

-> will search for Media dated from January 1st (included) to December 31st 1955 (included).

Posterior to / Prior to:
query=MediaDate<1995

-> will search for Media with a date prior to January 1st 1995 (excluding Media dated January 1st 1995)

query=MediaDate>:2000-12-01

-> will search for Media with a date after December 1st 2000 (including Media dated December 1st 2000)

Date interval:
query=MediaDate>:2012-12-30 MediaDate<:2013-01-02

-> will search for Media on 4 dates: December 30th and 31st 2012 and January 1st and 2nd 2013</td></tr>
</table>

	To search on date interval, you can combine MediaDate criteria.	query=MediaDate>2012-12-30 MediaDate<2013-01-02 -> will search for Media on 2 dates: December 31st 2012 and January 1st 2013

Fields

To retrieve a list of all fields available on your site, use the following API call:

https://www.sitename.com/API/search/v3.0/ListFields

Fields include media formats and media metadata.

Multiple fields must be separated by commas (,).

Standard Fields for Media Metadata

Criteria	**Description**	**Examples**
Path	The Path for the document (e.g. asset/folder), including the document filename. Example Path: Library/California/Diagrams/Mini-Icons.psd	&fields=Path
Directory	The Directory for the document (e.g. asset/folder), excluding the document filename. Example Path: Library/California/Diagrams	&fields=Directory
Title	The title in the default language	&fields=Title
Title_French	The title in French	&fields=Title_French,Title_German,Title_Italian,Title_Japanese
Title_German	The title in German	

Title_Italian	The title in Italian	
Title_Japanese	The title in Japanese	
Title_Portuguese	The title in Portuguese	
Title_Spanish	The title in Spanish	
SystemIdentifier	Unique System Identifier for the Media generated by the application.	&fields=SystemIdentifier
MediaNumber	Media Number specified in the install (this varies from install to install: “Picture Number”/”Package Number” or “Accession Number” or “URN” etc.). Your site administrator will tell you how the Media Number is referred to on their install	&fields=MediaNumber
MediaEncryptedIdentifier	This field is protected by security level. Ask you site administrator if you need access.	&fields=MediaEncryptedIdentifier
FileMD5	The MD5 checksum for the asset.	&fields=FileMD5
Caption	The Description field (your site administrator will tell you which caption field(s) are used on their install)	&fields=Title,Caption
CaptionShort	The Short Description field (the site administrator will tell you which caption field(s) are used on their install)	&fields=Title,CaptionShort
CaptionLong	The Long Description field (the site administrator will tell you which caption field(s) are used on their install)	&fields= Title,CaptionShort,CaptionLong
MediaDate	Media date in the format: MM/DD/YYYY HH:MM:SS AM/PM	&fields=MediaDate
CreateDate	Creation date in the format: MM/DD/YYYY HH:MM:SS AM/PM	&fields=CreateDate
EditDate	Last Ediy date in the format: MM/DD/YYYY HH:MM:SS AM/PM	&fields=EditDate

MediaType	The Media Type retrieved can include Image/Video/Audio/Package/Graphic	&fields=Title,MediaDate,MediaType
Artist	The Artist is retrieved in the format: FirstName LastName	&fields=Title,Artist
ArtistShortID	The Artist Short ID or Code (if in use in the install)	&fields=Title,Artist,ArtistShortID
Link	This “Link” field can be used in Packages	&fields=Link
OriginalSubmissionTitle	The title of the original Submission/Folder that the Media is part of	&fields=Title,OriginalSubmissionTitle
OriginalSubmissionNumber	The unique System Identifier of the original Submission/Folder that the Media is part of	&fields=Title,MediaNumber,OriginalSubmissionTitle,OriginalSubmissionNumber
OriginalSubmissionEncryptedIdentifier	This field is protected by security level. Ask you site administrator if you need access.	
MaxWidth	This is the width in pixels of the original Media (Image)	&fields=MaxWidth,MaxHeight
MaxHeight	This is the height in pixels of the original Media (Image)	&fields=MaxWidth,MaxHeight
MediaCount	Only relevant in a query on Packages. This retrieves the number of media of any type (Image/Video/Audio/Package) included in a Package.	In the context of a query on Packages (query=MediaType:Packages), you can require to retrieve the number of media in each package: &fields=Title,MediaCount
ImageCount	Only relevant in a query on Packages. This retrieves the number of images included in a Package.	In the context of a query on Packages (query=MediaType:Packages), you can require to retrieve the number of images in each package: &fields=Title,MediaCount,ImageCount

VideoCount	Only relevant in a query on Packages. This retrieves the number of videos included in a Package.	In the context of a query on Packages (query=MediaType:Packages), you can require to retrieve the number of videos in each package: &fields=Title,VideoCount
PackageCount	Only relevant in a query on Packages. This retrieves the number of Packages included in a Package.	In the context of a query on Packages (query=MediaType:Packages), you can require to retrieve the number of packages in each package: &fields=Title,PackageCount

Media Formats

The Media formats and dimensions used on your site (such as TR1, TR7, Crop etc.) differ from one install to another and cannot therefore be listed here.

The Media formats and dimensions you have access to are driven by your account credentials and Security Profile.

Please contact the site administrator of the site you are connecting to. Site administrators can see the list of all Media formats used on the site on: Administration > Archives > Documents Formats.

Note that the original high-resolution file is protected and cannot be retrieved with the Search API. Please ask for the “Extract Original Document API” if you need access.

Examples:
TR1 = Resolution used in front-end detailed asset page
TR7 = Thumbnail resolution used in front-end search results

If the Media exists or can be generated in the requested format, its URL, Width and Height will be included in the response.

Example:
&fields=Path_TR1
will return:

<Path_TR1>

```
        <URI>https://www.sitename.com/Media/Images/TR1/OL15390.jpg</URI>
        <Width type="Numeric">500</Width>
        <Height type="Numeric">406</Height>
    </Path_TR1>
```

If you are requesting a non-watermarked format but your Security Profile does not authorize you to access non-watermarked assets, the API will provide you with a substitute watermarked format and display the following message:

```
    <Warning> You do not have the security clearance to access the requested format.
    </Warning>
    <SubstituteFormat>TR1_ WATERMARKED</SubstituteFormat>
```

For videos, response can return:

- the path for the WebHigh format of the video file.
- the path for the WebLow format of the video file.

Example:
&fields=Path_WebHigh,Path_WebLow
will return:

```
<Path_WebHigh>
        <URI>rtmp://s3098o8h1l.cloudfront.net/OL/WebHigh/e/4/8/8/OL60825.mp4</U
        RI>
        <Width type="Numeric">640</Width>
        <Height type="Numeric">360</Height>
</Path_WebHigh>
<Path_WebLow>
        <URI>rtmp://s3098o8h1l
        .loudfront.net/OL/WebLow/1/f/1/4/OL60825.mp4</URI>
        <Width type="Numeric">320</Width>
        <Height type="Numeric">180</Height>
</Path_WebLow>
```

From OrangeDAM version “Florence”, the path returned for videos is available in 2 URL formats:

- the RMTP URL (to be consumed by a Flash Player)
- the HTTP URL

To obtain the representative image for a video (as displayed in the player prior to playing the video), query the appropriate media format (in most cases Path_TR1)

Example:
&fields=Path_WebHigh, Path_TR1

will return:

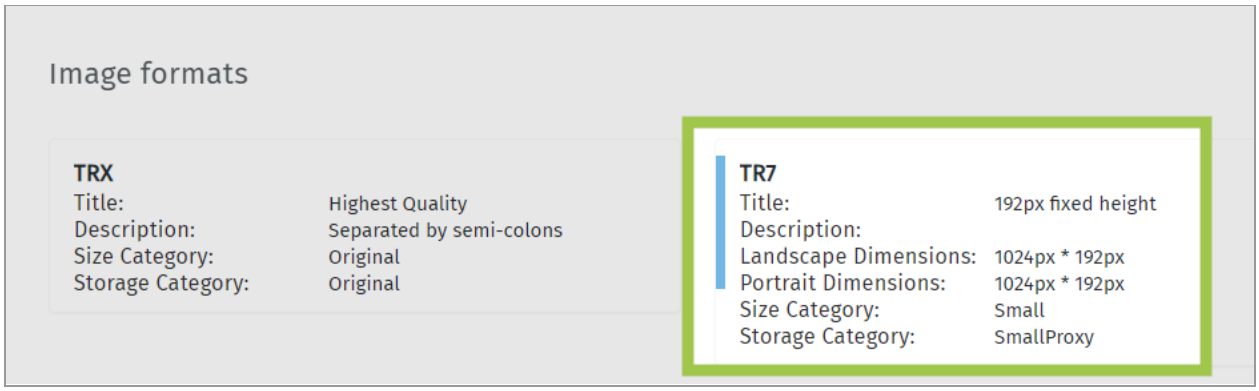
```
<Path_WebLow>
        <URI>rtmp://s3098o8h1l
        .loudfront.net/OL/WebLow/1/f/1/4/OL60825.mp4</URI>
        <Width type="Numeric">320</Width>
        <Height type="Numeric">180</Height>
</Path_WebLow>
<Path_TR1>
    <URI>https://www.sitename.com/Media/TR1/OL14567.jpg</URI>
    <Width type="Numeric">512</Width>
    <Height type="Numeric">288</Height>
</Path_TR1>
```

Generate Formats On Demand

If a specific asset format has not been generated yet, you can generate it on demand. To view a list of available on-demand formats, go to **Administration** > **Archives** > **Documents Formats**.

Formats that are generated on import are highlighted with a blue line. Some examples of these types of formats are:

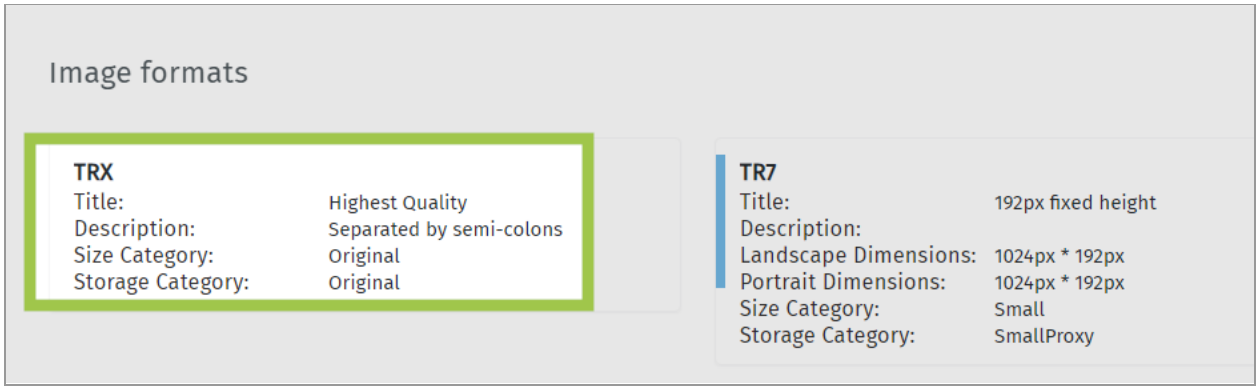
- **TR1**: Medium res. (1200px x 1200px). This format is used on the Overview tab.
- **TR4**: Small thumbnail (352px fixed height). This format is used in the Search Results.
- **TR7**: Large thumbnail (192px fixed height). This format is used in the Search Results.



Formats Highlighted in Blue Are Generated on Import

Formats that are generated on demand are not highlighted. Some examples of these types of formats are:

- **TRX**: Highest quality
- **CMS1**: 2000px x 2000px
- **CMS2**:- 1000px x 1000px
- **CMS3**: 500px x 500px



Formats That Are Not Highlighted Are Generated On Demand

For example, let's say you uploaded a logo and OrangeDAM generated several formats of that logo. However, you need a format that is 2000px x 2000px. You can call a Search API and include **&generateformatifnotexists=1** to generate the CMS1 format, which is 2000px x 2000px.

Syntax:
https://www.sitename.com/API/search/v3.0/search?query=[*Unique ID*]&fields=[*List of Fields*],**Path_**[*Format*]&generateformatifnotexists=1

Example:
https://www.sitename.com/API/search/v3.0/search?query=CTL238597&fields=Title,Caption, Photographer,**Path_**CMS1&generateformatifnotexists=1

Note: You can also generate new proxies for all assets. For more information, go here.

Including the details of the Search API interpretation of your query

You can include **&verbose=1** in your query to get in the response details on how the Search API conducted your query.

> **For example:**
> https://www.sitename.com/API/search/v3.0/search?query=MediaType:Image AND Keyword:(Marilyn Monroe OR Greta Garbo) AND Keyword:Cinema AND Color:False AND Orientation:Portrait&fields=Title,SystemIdentifier,MediaDate&Verbose=1
>
> Will return the following Query Interpetration:
> <QueryInterpretation>
> <QueryPart>
> <QueryPart>
> <QueryPart>
> <QueryPart>

```
        <Criterion>
          <Name>MediaType</Name>
          <Comparator>:</Comparator>
          <Value>Image</Value>
        </Criterion>
    <And/>
        <QueryPart2>
            <Criterion>
                <Name>Keyword</Name>
                <Comparator>:</Comparator>
                <Value>Marilyn Monroe</Value>
                  <ValueMappedTo type="List">
                  <Item>2C04AUL8B6X</Item>
                  </ValueMappedTo>
            </Criterion>
            <Or/>
            <Criterion2>
                <Name>Keyword</Name>
                <Comparator>:</Comparator>
                <Value>Greta Garbo</Value>
                  <ValueMappedTo type="List">
                  <Item>2C04AUL7B6X</Item>
                  </ValueMappedTo>
            </Criterion2>
        </QueryPart2>
    </QueryPart>
    <And/>
    <Criterion2>
        <Name>Keyword</Name>
        <Comparator>:</Comparator>
        <Value>Cinema</Value>
        <ValueMappedTo type="List">
        <Item>2C04AUL7B6H</Item>
        <Item>2C0BLIXXRX_W</Item>
        </ValueMappedTo>
    </Criterion2>
</QueryPart>
<And/>
    <Criterion2>
        <Name>Color</Name>
        <Comparator>:</Comparator>
        <Value>False</Value>
    </Criterion2>
</QueryPart>
<And/>
    <Criterion2>
```

```
                <Name>Orientation</Name>
                <Comparator>:</Comparator>
                <Value>Portrait</Value>
            </Criterion2>
        </QueryPart>
    </QueryInterpretation>
```

Managing the Page Numbers and Count Results Per Page

You can include parameters in your query to manage the page numbers you wish to retrieve in your response and the number of results you want to have on each page.

COUNTPERPAGE
The default value is 100
The format to use is : **&countperpage=200**

> **For example:**
> https://www.sitename.com/API/search/v3.0/search?query=flower&fields=Title,CaptionLong,Path_TR1&countperpage=200
>
> Will return the result with 200 media per page.

The maximum countperpage is 300. Specifying a countperpage higher than 300 will return the results with 300 media per page.

PAGENUMBER
The default value is 1.
The format to use is : **&pagenumber=12**

> **For example:**
> https://www.sitename.com/API/search/v3.0/search?query=flower&fields=Title,CaptionLong,Path_TR1**&pagenumber=**12
>
> Will retrieve page 12 of the results (with 100 media per page since 100 is the default).

Changing the Sort Order

You can include parameters in your query to specify the order in which you want to retrieve results.

RELEVANCY
This is the default sort order.

The format to use is : **&sort=relevancy** or **&sort=sort1** or **&sort=sort2** or **&sort=sort 3** , etc.

This sort order depends on the order in these parameters: SortOrderSelector_VForm.Data.UserSortOrderList and SortOrderSelector_VForm.Data.UserSortOrderList. For example, if site A has the Relevancy of sort3, then the syntax here will not work.

The sort order “Relevancy” algorithm is based on a number of factors, such as tags in the context of the total number of tags applied to the asset, the title and description of the asset, keywords of the parent folder(s), tags extrapolated from the keyword tree/thesaurus, etc. Each of these elements have different weightings applied to them.

> **For example:**
> https://www.sitename.com/API/search/v3.0/search?query=flower&fields=Title,CaptionLong,Path_TR1**&sort=relevancy**
>
> OR
>
> https://www.sitename.com/API/search/v3.0/search?query=flower&fields=Title,CaptionLong,Path_TR1**&sort=sort1**
>
> will display assets in the order driven by their score computed by the algorithm (best score first).

NEWEST
The format to use is : **&sort=newest first** or **&sort=sort3**

This retrieves assets based on Create Date and displays the most recent date first.

> **For example:**
> https://www.sitename.com/API/search/v3.0/search?query=flower&fields=Title,CaptionLong,Path_TR1**&sort=newest first**
>
> OR
>
> https://www.sitename.com/API/search/v3.0/search?query=flower&fields=Title,CaptionLong,Path_TR1**&sort=sort3**
>
> will display results with assets that have the most recent Create Date first.

OLDEST
The format to use is : **&sort=oldest first** or **&sort=sort4**

This retrieves assets based on Create Date and displays the oldest date first.

For example:
https://www.sitename.com/API/search/v3.0/search?query=flower&fields=Title,CaptionLong,Path_TR1**&sort=oldest first**

OR

https://www.sitename.com/API/search/v3.0/search?query=flower&fields=Title,CaptionLong,Path_TR1**&sort=sort4**

will display results with assets that have the oldest Create Date first.

EDITOR CHOICE
The format to use is : **&sort=Editor choice** or **&sort=sort2**

This retrieves assets based on the ranking of an asset and displays assets with a higher ranking first. For example, an image with the ranking of “Public + (C)" will come before an image with the ranking of “Public (E)”.

For example:
https://www.sitename.com/API/search/v3.0/search?query=flower&fields=Title,CaptionLong,Path_TR1**&sort=Editor choice**

OR

https://www.sitename.com/API/search/v3.0/search?query=flower&fields=Title,CaptionLong,Path_TR1**&sort=sort2**

will display results with assets that have a higher ranking first.

Output format

Results are returned in XML or JSON.
To get the answer in JSON, add **&format=json** at the end of the URL.

For example:

https://www.sitename.com/API/search/v3.0/search?query=flower&fields=Title,CaptionLong,Path_TR1&format=json

Encoding special characters in parameter values

Certain special characters in parameter values must be URL encoded using **percent-encoding**.

The reserved characters as defined by RFC 3986 are:

!	#	$	&	'	(	)	*	+	,	/	:	;	=	?	@	[	]
%21	%23	%24	%26	%27	%28	%29	%2A	%2B	%2C	%2F	%3A	%3B	%3D	%3F	%40	%5B	%5D

Other commonly percent-encoded characters are:

newline	space	"	%	-	.	<	>	\	^	_	`	{	|	}	~
%0A *or* %0D *or* %0D%0A	%20	%22	%25	%2D	%2E	%3C	%3E	%5C	%5E	%5F	%60	%7B	%7C	%7D	%7E

The percent-encoding of non-reserved characters is optional. For example, a “space” character encoded into %20 is optional.

“Ben & Jerry’s” can be either:

`“Ben %26 Jerry%27s”`

`“Ben%20%26%20Jerry%27s”`

Allow search on all documents

The Search API only returns documents that are visible in the front end.
For example, by default stacked assets are not visible in the front-end, only the Lead assets are visible. Stacked assets are therefore not returned by Search API calls.

If you do want the Search API to return assets that are not visible in the front end, you need to grant the API user with the security function "Search API - Allow search on all documents".

Note that this security function does not override permissions so the API user will still only be able to return the assets he has permission to access.

API response information

The API response contains the expected results, with the following additional information:

- API Request Information:
  - Boolean tag indicating the authentication state (IsLoggedIIn : True or False)
  - Parameters : Query, Fields, Token
- API Response Global Information:
  - Total results count
  - Request duration (in milliseconds)
  - Sort Order
  - Link to the next page of results

For example:

```
<APIRequestInfo>
   <Module>search</Module>
   <APIVersion>v3.0</APIVersion>
   <Resource>search</Resource>
   <IsLoggedIn type="Boolean">True</IsLoggedIn>
   <Parameters>
      <query>MediaType:Image,Keyword:Paris</query>
      <fields>Title,CaptionLong,Path_TR1</fields>
      <token>gtvprt45l1mpm5454l1gqqji</token>
   </Parameters>
<ProviderVersion>5.0.57.0</ProviderVersion>
<ProviderIdentity>IP-0ADA45BF</ProviderIdentity>
</APIRequestInfo>

<APIResponse>
   <GlobalInfo>
      <TotalCount type="Numeric">3552</TotalCount>
      <QueryDurationMilliseconds type="Numeric">192</QueryDuratio
      nMilliseconds>
      <Sort>Ranking</Sort>
      <NextPage href="https://www.sitename.com/API/search/v3.0/se
      arch?query=MediaType%3aImage%2cKeyword%3aParis&fields=Title
      %2cCaptionLong%2cPath_TR1&token=gtvprt45l1mpm5454l1gqqji&pa
      genumber=2" rel="next"/>
   </GlobalInfo>
<Items type="List">
<Item>
```

Failure messages

Error codes common to all OrangeDAM REST APIs:

https return code	Message	Notes
401	Security violation	Your Security Profile does not give you access to the given API
403	This API cannot connect using a Secure Sockets Layer (SSL). Please change your URL from http to https.	
503	The server is currently in a state that does not allow API calls... [Client specific message]	
500	The server encountered an unexpected condition which prevented it from fulfilling the request. [Client specific message]	This is the fallback error message/code for all unhandled errors

Warning/Errors codes specific to the Search API:

Warning Messages

Https return code	Message	Notes
200	Your queries are temporarily throttled (response time is degraded) to preserve server resources.	Reason: This warning message is displayed when too many requests are made in a short period of time. Note: This is just a warning message and the results are still provided.

Client-side Error Messages

Https return code	Message	Notes

400	Error. We were not able to match some criteria values in the database. <IgnoredCriterion>[CRITERION NAME] [CRITERION VALUE]…… <IgnoredCriterion>	Reason: Some of the query criteria you passed in the request do not match any values. Solution: This message will include the criterion which was not matched within <IgnoredCriterion> tags. Correct the value of the criterion mentioned.
400	Unsupported field: [FIELD NAME]	Reason: The field format or metadata you requested in the response is not supported. Solution: Supply a field that is supported. See topic Fields for more details.
400	Unauthorized format: [FORMAT NAME]	Reason: You do not have security clearance for the media format you requested. Solution: Request a format which is allowed for your credentials or contact your Account Manager if you need to change your access privileges.
400	Query parsing error: Error at position [POSITION OF THE ERROR]	Reason: The query parameter does not follow the expected syntax. Solution: Look for the error at the position mentioned and correct the syntax.
400	The text is too vague. Please provide a more specific query.	Reason: This error will be displayed when the search is composed exclusively of noise words. Solution: Provide a more specific query.

Server-side Error Messages

Https return code	**Message**	**Notes**
503	The server is currently in a state that does not allow API calls...	Reason: This error will be displayed when the site is closed to the public (this is the case of the DR site before a switch). Solution: Contact Technical Support

500	The server encountered an unexpected condition which prevented it from fulfilling the request.	Reason: This is the fallback error message/code for all other unhandled errors. Note: When you issue a Search API call when logged out (no token provided or expired/invalid token provided), you will get the above message as well. Solution: ● If you have logged out or provided an invalid token, retry with a valid token. ● For all other cases, contact Technical Support.

Retrieving Keywords

For OrangeDAM versions prior to Helsinki, use the Get Keywords API.

For OrangeDAM versions from Helsinki and later, use the Search API v3.0.

Retrieving Keywords with Search API

For sites updated to OrangeDAM Helsinki on later version after September 25th 2018, keywords can be extracted using the Search AP, using the field ‘Keywords”. The Get Keywords API is therefore no longer required.

The Keywords directly attached to the asset are included in the response, as well as the keyword type for each keyword.

Example:

```
https://www.sitename.com/API/search/v3.0/search?query=SystemIdentifier
:ABC12345&fields=Title,Keywords
```

Response:

```
<Item>
        <Title>ABC12345</Identifier>
        <Keywords type="List">
                <Keyword>
                        <Label>Sport</Label>
                        <KeywordType>Category</KeywordType>
                        <LinkType/>
       </Keyword>
       <Keyword>
                        <Label>sport shoes</Label>
                        <KeywordType>common</KeywordType>
                        <LinkType/>
     </Keyword>
     <Keyword>
                        <Label>Quality Control required</Label>
                        <KeywordType>Workflow</KeywordType>
                        <LinkType>Workflow</LinkType>
             </Keyword>
        </Keywords>
</Item>
```

Retrieving Keywords with Get Keywords API

To retrieve keywords for an asset, there is a dedicated API, Get Keywords API, available on https://www.sitename.com/htm/GetKeywordsAPI.aspx

See documentation: Get Keywords API

The security point "Can extract document keywords" in family API, sub family "Keywords" must be given at least "Read" basis for the API user to be allowed to use the Get Keywords API.

This a 2-step process:

1. Obtain the encrypted ID for the image using the Search API

> https://www.sitename.com/API/search/v3.0/search?query=SystemIdentifier:ABC12345&fields=MediaEncryptedIdentifier
>
> API response: 2K7O3R3GDZTR

2. Get the list of keywords for this asset using the GetKeywordsAPI

> https://www.sitename.com/htm/GetKeywordsAPI.aspx?DocID=2K7O3R3GDZTR

Optional parameter: IncludeInheritedKeywords

Add &IncludeInheritedKeywords=1 (or any true string (e.g. True, Yes), to return the list of all keywords directly applied to a given document ('native keywords'), as well as all keywords applicable to the document through inheritance.

Example:
`https://sitename.orangelogic.com/htm/GetKeywordsAPI.aspx?DocID=2UVRYDM1H4CO&IncludeInheritedKeywords=1`

Usage Examples

Example 1 - Return assets of any media type with free text containing ‘flower’.

https://www.sitename.com/API/search/v3.0/search?
query=Text:flower
&fields=Title,CaptionLong,Path_TR1

Result: Provides in the response the Title, Long Caption/Description and Path to the TR1 image format

Example 2 - Return ONLY Image Assets with keyword Paris but NOT keyword car.

https://www.sitename.com/API/search/v3.0/search?
query=MediaType:Image Keyword:Paris AND NOT Keyword:car
&fields=Title,CaptionLong,Path_TR3&format=json

Result: Provides in the response (given in JSON format) the Title, Long Caption/Description and Path to the TR1 image format

Example 3 - Return Images with Keyword criterion / Image Orientation type created between two dates.

https://www.sitename.com/API/search/v3.0/search?
query=ComputedMediaType:Picture Keyword:beach AND NOT Keyword:France
Orientation:Portrait MediaDate>:2010-01-01
&fields=SystemIdentifier,MediaNumber,CaptionShort,CaptionLong,MediaDate,Artist,ArtistShortID,OriginalSubmissionTitle,OriginalSubmissionNumber,MaxWidth,MaxHeightPath_TR1,Path_TR7

Result: Provides in the response the System Identifier, Media Number, Short Caption/Description, Long Caption/Description, Shoot Date, Artist, Artist Short ID, Title of the Original Submission, Number of the Original Submission, Image dimensions (MaxWidth and MaxHeight) and Paths to the TR1 and TR7 formats.

Example 4 - Return Albums containing ‘flower’ in free text search.

https://www.sitename.com/API/search/v3.0/search?
query=ComputedMediaType:Album Text:flower
&&fields=Title,SystemIdentifier,MediaNumber,CaptionShort,CaptionLong,MediaDate,Artist,ArtistShortID,Link,ImageCount

Result: Provides in the response the Title, System Identifier, Media Number, Short Caption/Description, Long Caption/Description, Shoot Date, Artist, Artist Short ID, Link field for

the album and the number of images contained in each album.

Example 5 - Return black&white Images containing keywords of a specific keyword type and Orientation type Portrait.

https://www.sitename.com/API/search/v3.0/search?
query=MediaType:Image AND Keyword(PersonalNames):(Marilyn Monroe OR Greta Garbo) AND Keyword:Cinema AND Color:False AND Orientation:Portrait
&fields=Title,SystemIdentifier,MediaDate

Result: Provides in the response the Title, System Identifier, Media Date and the Query Interpretation

Example # - Return list of all “related assets” for a particular image asset

This example takes TWO separate Search API calls in order to grab the list of “Related Assets”

Call 1:
Search for images with keyword flower and request the "Story" field and MediaType to be returned.

https://www.sitename.com/API/search/v3.0/search?
query=keyword:flower
&fields=Story,SystemIdentifier,MediaType

This snippet is part of the response:

```
<Item>
  <Story>OTS001357</Story>
  <SystemIdentifier>OT133177</SystemIdentifier>
  <MediaType>Image</MediaType>
</Item>
<Item>
  <Story>OTS001357</Story>
  <SystemIdentifier>OT133176</SystemIdentifier>
  <MediaType>Image</MediaType>
</Item>
```

Call 2
Using the <Story> element you can now populate another Search API query that looks like:

https://www.sitename.com/API/search/v3.0/search?
query=(text:OTS001357) and (Mediatype:Image)
&fields=SystemIdentifier,MediaType,Story

This will return ALL Image Assets with "OTS001357" as it's Parent Folder. Thus granting you the list of all "Related Assets".

```
<Item>
  <SystemIdentifier>OT133180</SystemIdentifier>
  <MediaType>Image</MediaType>
  <Story>OTS001357</Story>
</Item>
<Item>
  <SystemIdentifier>OT133179</SystemIdentifier>
  <MediaType>Image</MediaType>
  <Story>OTS001357</Story>
</Item>
<Item>
  <SystemIdentifier>OT133178</SystemIdentifier>
  <MediaType>Image</MediaType>
  <Story>OTS001357</Story>
</Item>
<Item>
  <SystemIdentifier>OT133177</SystemIdentifier>
  <MediaType>Image</MediaType>
  <Story>OTS001357</Story>
</Item>
```

Example 7 - Search within a specific folder.

To search for assets/folders within a particular folder, add to your query ParentFolderIdentifier (the System ID for the Parent Folder) or ParentFolderTitle (the Title for the Parent Folder)) or ParentFolderNumber (the Client Number/Legacy Identifier for the Parent Folder)

Example: Search assets/folders with the keyword "United Kingdom" within Folder "Conferences" (System ID: DMOSTO65)

https://www.sitename.com/API/search/v3.0/search?
query=Keyword:United Kingdom AND ParentFolderIdentifier:DMOSTO65
&fields=Title,SystemIdentifier,MediaType

Example 8 - Search for tasks (workflow steps) values assigned to assets.

Example: Search assets/folders with the task (workflow step) 'Additional Retouching Needed' when status is 'To be decided'

https://www.sitename.com/API/search/v3.0/search?
query=Task:"Additional Retouching Needed [To be decided]"
&fields=Title,SystemIdentifier,MediaType

Example: Search assets/folders with the task (workflow step) 'Legal to approve' and status 'Approved' OR the task (workflow step) 'John to review' and status 'Approved'

https://www.sitename.com/API/search/v3.0/search?
query=Task:"Legal to approve [Approved]" OR "John to review [Approved]"
&fields=Title,SystemIdentifier,MediaType

Data Tables API v2.2

Introduction

The Data Table API is designed to map OrangeDAM “objects” structure and allow CRUD (Create/Add Read Update Delete) operations on objects individually or by batches.

Objects are defined as Assets (Images, Videos, Multimedia), Keywords, Folders, virtual folders, Links between Assets and virtual folders, Links between Assets and Folders.

DataTable v2.2 is used with entities which have the custom forms activated (currently available on documents: assets, folders, etc.).

DataTable v.2.1 remains available for managing data in other DataTables: Keywords, Contacts, etc.

Authentication

Authentication to use the Data Table API can use a cookie or a token requested through the Authentication API.

DataTables (Resources)

https://www.sitename.com/API/DataTable/v2.2
→ lists all Data Table resources available on your installation.

> When some calls/DataTable are not yet available in v2.2, it is possible to mix DataTable API v2.1 and DataTable API v2.2 in the same query, by making a sub-call (using bracket) within the main call.

Example:
https://www.sitename.com/API/DataTable/V2.1/Contact.Client:Update?Contact.CoreField.OldNumber=12345&Contact.CoreField.Country:=[DataTable/v2.2/ReferenceTable.Country:Read?ReferenceTable.CoreField.OldID=BELG&ReturnField=ReferenceTable.CoreField.CountryCode]

DataTables for managing Documents

Examples:

- Managing Assets of type "**Image**" →
  https://www.sitename.com/API/DataTable/v2.2/Documents.Image.Default
  This resource allows you to manage (Create, Read, Update, Delete) Image records

- Managing Assets of type "**Video**" →
  https://www.sitename.com/API/DataTable/v2.2/Documents.Video.Default
  This resource allows you to manage (Create, Read, Update, Delete) Video records

- Managing Assets of type "**Audio**" →
  https://www.sitename.com/API/DataTable/v2.2/Documents.Audio.Default
  This resource allows you to manage (Create, Read, Update, Delete) Audio records

- Managing Assets of type "**Multimedia**" →
  https://www.sitename.com/API/DataTable/v2.2/Documents.Multimedia.Default
  This resource allows you to manage (Create, Read, Update, Delete) Multimedia records

- Managing Records of type "**Folders**" →
  https://www.sitename.com/API/DataTable/v2.2/Documents.Folder.Default
  This resource allows you to manage (Create, Read, Update, Delete) Folder records

- Managing Records of type "**Virtual Folder**" →
  https://www.sitename.com/API/DataTable/v2.2/Documents.Virtual-folder.Default
  This resource allows you to manage (Create, Read, Update, Delete) Virtual Folders records

Document Subtypes

If you have multiple subtypes for any of the above assets.records, all subtypes will be listed. For example, if you have Folders of subtypes Event, Shoot, and Upload Folder, you will have the following list of DataTables for Folders:

- https://www.sitename.com/API/DataTable/v2.2/Documents.Folder.Event
- https://www.sitename.com/API/DataTable/v2.2/Documents.Folder.Shoot
- https://www.sitename.com/API/DataTable/v2.2/Documents.Folder.Upload-Folder

Documents.All Read-Only Resource

Security Function required: *Family API: Datatable API - "Document (Read only)"* (Code: APITableDocument)
WARNING: This security point gives access to all documents as it does not take into account Permissions/Restrictions/Embargo dates etc

https:/www.sitename.com/API/DataTable/v2.2/Documents.All:Read?CoreField.Identifier=[ID12345]

The Document.All resource is 'read only'. It is useful to retrieve information about a document, without knowing the document type (i.e. Image, Video, Folder, etc.).

You can use the following fields to search for the document: RecordID, CoreField.Identifier, CoreField.Id_Client, CoreField.OtherNum and CoreField.OriginalFileName.

If you know the document type, you can use the dedicated DataTable.

DataTables for managing Contacts

Examples:

- Managing Contacts of type **"Client"** → https://www.sitename.com/API/DataTable/v2.2/Contacts.Client.Default
- Managing Contacts of type **"Staff"** → https://www.sitename.com/API/DataTable/v2.2/Contacts.Staff.Default
- Managing Contacts of type **"Source"** → https://www.sitename.com/API/DataTable/v2.2/Contacts.Source.Default
- Managing Contacts of type **"Source agent"** → https://www.sitename.com/API/DataTable/v2.2/Contacts.Source-agent.Default
- Managing Contacts of type **"Agent"** → https://www.sitename.com/API/DataTable/v2.2/Contacts.Agent.Default
- Managing Contacts of type **"Billing account"** → https://www.sitename.com/API/DataTable/v2.2/Contacts.Billing-account.Default
- Managing Contacts of type **"Company"** → https://www.sitename.com/API/DataTable/v2.2/Contacts.Company.Default

Contacts.All Read-Only Resource

Security Function required: *Family API: Datatable API - Contact (Read only)*

https://www.sitename.com/API/DataTable/v2.2/Contacts.All:Read?Contact.CoreField.Email1=[EMAIL_ADDRESS]

The Contact.All resource is 'read only'. It is useful to retrieve information about a contact, without knowing the contact type (i.e. Client, Staff, Source, etc.).

If you know the contact type, you can use the dedicated DataTable.

DataTables for managing Keywords

Examples:

- Managing Keywords → https://www.sitename.com/API/DataTable/v2.2/Keywords
- Managing Links between Keywords (Thesaurus hierarchy) → https://www.sitename.com/API/DataTable/v2.2/Keywords-links

DataTables for managing Links between Documents

Example:

- Managing Links between assets and virtual folders → https://www.sitename.com/API/DataTable/v2.2/Documents-links

Create Relationship Between Contact and Keyword

Note: This option needs to be enabled by an Orange Logic Implementation Specialist before it can be used. To make this option available for your organization's OrangeDAM site, please submit a support request here.

https://www.sitename.com/API/DataTable/v2.1/Contact.Source:Update?Contact.CoreField.Email1={PhotographerEmail}&Contact.CoreField.KW_RecordID:=[Datatable/v2.1/Tags.Keyword:Read?Tags.CoreField.Keyword_English={Keyword}]

Remove relationships from Document to Keyword

Note: This API may require an update to OrangeDAM. If this API does not work, please submit a support request here to schedule a OrangeDAM update.

https://www.sitename.com/API/DataTable/v2.1/Document.Keywords.Link:Delete?Document.CoreField.DO_RecordID=[DataTable/v2.2/Documents.Image.Default:Read?CoreField.Identifier=DMW116470]
&Document.CoreField.KW_RecordID=[DataTable/v2.1/Tags.Keyword:Read?Tags.CoreField.Keyword_English=Computer]

Specify the KeyType to Search for or Create Keywords

To specify the KeyType when searching for or creating Keywords (via API and referencing a CSV column), use a call like this:

https://www.sitename.com/API/v2.2/DataTable/Documents.All:Update?Corefield.OriginalFileName=[File Name] &CoreField.Keywords++=[Keywords]

You need to edit the values in the CSV to specify the KeyType for search and creation. For example, find "|" in the Keywords column, and replace it with:

{"SearchTypes":"CustomCommon","CreationType":"CustomCommon"}|

In this situation, you would also add this to the last Keyword in each row:

{"SearchTypes":"CustomCommon","CreationType":"CustomCommon"}

Remove relationships from Keyword to Keyword

Note: This API may require an update to OrangeDAM. If this API does not work, please submit a support request here to schedule a OrangeDAM update.

`Keyword to Keyword (Tags,Keyword.Link) →`
https://www.sitename.com/API/DataTable/v2.1/Tags.Keyword.Link:Delete?Tags.CoreField.KeyidFather=[DataTable/v2.1/Tags.Keyword:Read?Tags.CoreField.Keyword_English=Computer]
&Tags.CoreField.KeyidSon=[DataTable/v2.1/Tags.Keyword:Read?Tags.CoreField.Keyword_English=PC

GET and POST requests

For API testing purposes, every action can be performed using POST or GET HTTP requests.

- The maximum URL length for a GET request is 260 characters.
- For POST requests, the parameters are passed in the HTTP header or in the URL. If the same parameter is present in both, the value in HTTP header will override the URL.

Data Table APIs operations and parameters

Each of these resources/data tables supports the following operations:

• ListFields	List the fields that are available for further read, update or add operations
• Read	Can perform a search based on a given set of criteria and return the requested fields for each matching object
• Update	To update the given fields with the given values for a given object based on its RecordID value
• Create	To create an object with the given fields and the given values. The RecordID of the newly created object is returned
• CreateOrUpdate	To create an object with the given fields and the given values or update an existing object if found in the data table
• Delete	To delete the object that has the given RecordID value

The operations are prefixed by a colon (:)

Example:
www.sitename.com/API/DataTable/V2.2/Documents.Image.Default:ListFields

www.sitename.com/API/DataTable/V2.2/Contacts.Client.Default:ListFields

LIST - Listing fields available for a given data table

ListFields will return the list of fields that are available for the given resource.

For example:
www.sitename.com/API/DataTable/V2.2/Documents.Image.Default:ListFields
will list all fields available for Assets of type “Default Image”

Documents.Image.Default is the default image type. Administrator of OrangeDAM can create additional custom Asset Types of images. For example, if an custom asset type of Images is created — for example: 'Born Analog Image' — The syntax will then be: Documents.Image.Born-Analog-Image

The call to list its fields would then be:
www.sitename.com/API/DataTable/V2.2/Documents.Image.Born-Analog-Image:ListFields

The following attributes are included in the response:

Read-Only	<CanBeUsedAsAReadCriterion type="Boolean">True</CanBeUsedAsAReadCriterion>
Multi-Valued	<IsMultiValue type="Boolean">True</IsMultiValue>
Label	`<Label> - The field label, as it appears in Administration screens`
The field type	`<DataType> - e.g. Text, DateTime, Boolean, Numeric, etc.`
The field length in number of characters (for text fields)	<DataLength>
Whether the field is **based on a reference table**	IsBasedOnAuthorityTable <type="Boolean">False</IsBasedOnAuthorityTable>
Whether the **field is searchable**	<IsSearchable type="Boolean">True</IsSearchable>

Whether the **field is visible**	<IsVisible type="Boolean">True</IsVisible>

READ - Reading records

Read will return all fields for the requested objects.
Specifying filter conditions =

For example:
www.sitename.com/API/DataTable/V2.2/Documents.Image.Default:Read?CoreField.Identifier=DM123

Blank fields are not included in the responses unless **&Verbose=1** has been specified in the query (see Verbose parameter).

The following operators are available. You can use a field value or a field name.

=	**Is equal to** Examples: www.sitename.com/API/DataTable/v2.2/Contact.Client:**Read?** Contact.CoreField.Last_Name=a* → will return clients with Last Name starting with the letter 'a' www.sitename.com/API/DataTable/v2.2/Documents.Image.Default:**Read?** CoreField.System-ID=DM123&Verbose=1 → will return image DM123, with all fields including blank fields www.sitename.com/API/DataTable/v2.2/Documents.Image.Default:Read? CoreField.CreateDate=CoreField.EditDate → will return assets where the CreateDate is the same as the EditDate
>	**Is greater than / Is posterior to** Examples: www.sitename.com/API/DataTable/v2.2/Documents.Image.Default:Read? CoreField.CreateDate>2000-02-19 → will return assets created on or after 19th February 2000 Date format: Use one of these 2 ISO 8601 date formats: YYYY-MM-DD or

	YYYY-MM-DDTHH:MM:SS www.sitename.com/API/DataTable/v2.2/Documents.Image.Default:Read?CoreField.EditDate>CoreField.ExpirationDate → will return assets where the EditDate is posterior to the Expiration Date
<	**Is smaller than / Is anterior to** Examples: www.sitename.com/API/DataTable/v2.2/Documents.Image.Default:Read?CoreField.CreateDate<2000-02-19 → will return assets created on or before 19th February 2000 Date format: Use one of these 2 ISO 8601 date formats: YYYY-MM-DD or YYYY-MM-DDTHH:MM:SS www.sitename.com/API/DataTable/v2.2/Documents.Image.Default:Read?CoreField.CreateDate<Document.CoreField.DocDate → will return assets created in the system before the date specified in the field "Document Date"
<>	**Is different from** Examples: www.sitename.com/API/DataTable/v2.2/Contacts.Client.Default:Read?CoreField.Country<>France → will return clients with a Country other than France www.sitename.com/API/DataTable/v2.2/Documents.Image.Default:Read?CoreField.ModelReleaseExpirationDate<>CoreField.PropertyReleaseExpirationDate → return assets where the Model Release Expiration Date and the Property Release Expiration Date are different
!=	**Is different from** works the same way as <> above

UPDATE - Updating records

Update will update a record for given condition.

For example:
https://www.sitename.com/API/DataTable/V2.2/Documents.Image.Default:Update?CoreField.Identifier=DM123

Assigning and Removing operators

:= → The operator for assigning a value to a single value field

For example:
&CoreField.Title:=My new title

/API/DataTable/V2.2/Documents.Image.Default:Update?CoreField.Identifier=DM123&CoreField.Title:=My new title

To remove the value, make a call in which you do not assign anything:

&CoreField.Title:=

/API/DataTable/V2.2/Documents.Image.Default:Update?CoreField.Identifier=DM123&CoreField.Title:=

+:= → The operator for assigning a value to a single valued field based on an Authority List

For example:
&CustomNameSpace.Video-Format+:=PAL

/API/DataTable/V2.2/Documents.Image.Default:Update?CoreField.Identifier=DM123&CustomNameSpace.Video-Format+:=PAL

To remove the value, make a call in which you do not assign anything:

For example:
&CustomNameSpace.Video-Format+:=

/API/DataTable/V2.2/Documents.Image.Default:Update?CoreField.Identifier=DM123&CustomNameSpace.Video-Format+:=

+= → The operator for assigning a value to a multi valued field based on an Authority List (if value already exists in the Authority List)

For example:
&CoreField.Keywords+=Flower

/API/DataTable/V2.2/Documents.Image.Default:Update?CoreField.Identifier=DM123&CoreField.Keywords+=Flower

Using the above example, if the keyword “Flower” does not exist in your thesaurus you will get the following response from the API call:

<Error>

The requested value was not found in the authority list and you did not request to create it if needed (operator ++=)

</Error>

-= → The operator to remove the value to a multi valued field based on an Authority List

For example:
&CoreField.Keywords-=Flower

/API/DataTable/V2.2/Documents.Image.Default:Update?CoreField.Identifier=DM123&CoreField.Keywords-=Flower

++= → The operator to create and assign a value to a multi valued field based on an Authority List

For example:
&CoreField.Keywords++=Flower

/API/DataTable/V2.2/Documents.Image.Default:Update?CoreField.Identifier=DM123&CoreField.Keywords++=Flower

> *- The keyword type for keyword created with the operator ++= is “NewlyCreated”*
>
> *- The application is setup by default to only add or remove keywords of type 'NewlyCreated’ or ‘Common’*

--= → The operator to remove ALL the value assigned to a multi valued field based on an Authority List

For example:

&CoreField.Keywords--=

/API/DataTable/V2.2/Documents.Image.Default:Update?CoreField.Identifier=DM123&CoreField.Keywords--=

> See important information about Enable / Disable Automated Email when Creating/Updating User Accounts via API

CREATE - Creating records

Create will create a new record.

If the operation is successful, then the RecordID of the newly created object is returned.

If the operation failed, the response will include details of the fields with unexpected or required values.

Example
/API/DataTable/V2.2/Documents.Image.Default:Create?CoreField.Title:=An API Created Record

> See important information about Enable / Disable Automated Email when Creating/Updating User Accounts via API

CREATEORUPDATE - Create Or Update records

CreateOrUpdate allows you to perform the Update and Add operations in one single operation.

The object is updated if found in the database. If it is not found, the object is created.

For example:
API/DataTable/V2.2/Documents.Image.Default:CreateOrUpdate?CoreField.Title=A Record Created by API&CoreField.Keywords+=Flower

Will Create Or Update image with Title = A Record Created by API & Add the Keyword Flower

> See important information about Enable / Disable Automated Email when Creating/Updating User Accounts via API

DELETE - Deleting records

Delete the object with the given RecordID or condition

For example:
/API/DataTable/V2.2/Documents.Image.Default:Delete?CoreField.Identifier=FD110408

Will delete image with Identifier FD110408

Encoding special characters in parameter values

Special characters in parameter values must be URL encoded using percent-encoding.

Refer to http://en.wikipedia.org/wiki/Percent-encoding for the percent-encoding of common characters, such as:

Special character	Percent-encoding
&	%26
=	%3D

> The percent-encoding of the “space” characters into %20 is optional.

For example

/API/DataTable/V2.2/Documents.Image.Default:Read?Title=Ben %26 Jerry&ShowAllFields=1

or

/API/DataTable/V2.2/Documents.Image.Default:Read?Title=Ben%20%26%20Jerry&ShowAllFields=1

where Ben %26 Jerry

or Ben%20%26%20Jerry

is Ben & Jerry

> For a list of percent-encoding, visit http://www.w3schools.com/tags/ref_urlencode.asp (Column “From UTF-8”)

MaxRecordsAffected parameter

When trying to create or update more than 1 object in a single operation without using the MaxRecordsAffected parameter, you will get the following failure response : “*If executed, this operation would have updated [RECORD_COUNT] records. The current syntax authorizes the update of up to 1 record.*”

For example if you are trying to updated metadata on images based on Original filename, and there are multiple assets with this filename in your database.

To upgrade more than 1 record, use the parameter MaxRecordsAffected=100.

The maximum number of objects that can be created or updated in a single operation is 100.

Example:

www.sitename.com/API/DataTable/v2.2/Documents.Image.Default:CreateorUpdate?CoreField.OriginalFileName=beach20150402.jpg&CoreField.DocMark:=E&MaxRecordsAffected=100

FieldLengthSafety parameter

If you create or update fields with values that exceed the maximum length for the field, you will get a failure response indicating the field concerned by the error and its maximum length.

Example:

<Message>Update failed for the given records</Message>
<ErrorList>
<Document.CoreField.Title>This field cannot contain more than 200 characters</Document.CoreField.Title>
</ErrorList>

Note: Information about the max length for a given field can be retrieved using a ListFields call (attribute: <DataLength>)

Adding the parameter FieldLengthSafety=1 to an API call will:
- Update the fields by truncating the values provided to the max length.
- Append the full values provided to a Notes field

Example:

www.sitename.com/API/DataTable/v2.2/Documents.Video.Default:CreateorUpdate?CoreField.System-identifier=DMO1234&CoreField.Title:=This is such a long title that it exceeds 200 characters and would make the API call fail. A title really should be succinct, further details about the video would be better added to the Caption field. To truncate this title after the first 200 characters, use the Field Length Safety parameter&FieldLenghtSafety=1

Verbose parameter

Adding the parameter Verbose=1 to an API call will cause all fields of a record to be included in the response even if they are blank.

IndexInBackground parameter

By default, OrangeDAM indexes newly created and updated items at the time of the API call, which means there will be a delay to account for processing time. Adding the parameter IndexInBackground=1 to an API call will speed up the API call by delaying item indexing.

For example:

https://www.sitename.com/API/Datatable/v2.2/Documents.Folder.Default:Create?CoreField.Title=My new folder &IndexInBackground=1

Data Table APIs Response

Standard response codes

The Data Table API returns the following response codes:

- Listing Fields:
  - If successful, the response returns the fields for the specified data table
- When Reading:
  - If successful, the response returns the fields for the specified object(s)
  - If the object/record is not found, the response returns 0 results
- When Updating:
  - Success

Response>
<RecordsAffected type="List">
<Result>
<RecordID>KRFQ7UHR39</RecordID>
<Code>SUCCESS</Code>
</Result>
</RecordsAffected>

- Record not found

<Response>
<RecordsAffected type="List"/>
<Code>RECORD_NOT_FOUND</Code>
<Message>Found no record with the given criteria</Message>
</Response>

- Update failed

<Response><RecordsAffected type="List"><Result>
<RecordID>KRFQ7UHR39</RecordID>
<Code>UPDATE_FAILED</Code>
<Message>Update failed for the given records</Message>
<ErrorList>
<Contact.CoreField.Email1>Email address is not valid</Contact.CoreField.Email1>
</ErrorList>

- When Creating:
  - Success

Response>
<RecordsAffected type="List">
<Result>
<RecordID>KRFQ7UHR39</RecordID>
<Code>SUCCESS</Code>
</Result>
</RecordsAffected>

- Creation failed

<Response>
<Code>CREATION_FAILED</Code>
<Message>Creation failed for the given record</Message>
<ErrorList>

```
<Contact.CoreField.Last_Name>This field must be
filled</Contact.CoreField.Last_Name>
</ErrorList>
</Response>
```

- When Deleting:
  - Success

Failure messages

Error codes common to all OrangeDAM REST APIs

Http return code	Message	Notes
401	Security violation	Your Security Profile does not give you access to the given API
403	This API cannot connect using a Secure Sockets Layer (SSL). Please change in your URL, HTTPS to HTTP	
403	This API can only connect using a Secure Sockets Layer (SSL). Please change in your URL, HTTP to HTTPS	
503	The server is currently in a state that does not allow API calls... [Client specific message]	
500	The server encountered an unexpected condition which prevented it from fulfilling the request. [Client specific message]	This is the fallback error message/code for all unhandled errors

Errors codes specific to the DataTable APIs

Http	Message	Notes

return code		
412	Update failed for the given records	
404	Found no record with the given criteria	
400	Wrong value set for parameter MaxRecordsAffected. Should be less than...	When trying to create or update 100 objects in a single operation using the 100MaxRecordsAffected parameter, but there are more than 100 objects concerned with the query condition
412	Creation failed for the given record	
400	Some parameters were missing while trying to call the requested function	
400	Some of the parameters don't have the expected type	
400	If executed, this operation would have updated [RECORD_COUNT] records. The current syntax authorizes the update of up to 1 record. To upgrade more than 1 record, please use the parameter MaxRecordsAffected	When trying to create or update more than 1 object in a single operation without using the MaxRecordsAffected parameter (note: the maximum number of objects that can be created or updated in a single operation is 100)
400	Some errors were found in the requested assignments or requested criteria. Please check messages.	A detailed description of the error for each parameter is included in the request summary. Including: • The given field is unknown • The given field is not searchable for Read operations • The given operator is not compatible with a single-value field. Please use assignment operator ":=" instead • The given operator is not compatible with a multi-value field. Please use multi-value operators instead (+=, ++=, -=, --=) • The given field is not supported • The requested value was not found in the authority list and you did not request to create it if needed (operator ++=)

		• The requested value was not found in the authority list and you don't have the right to create it • An error occurred during the creation of the reference value
500	Failed	This is the fallback error message/code for all DataTable API unhandled errors

Additional information in the response

In addition to the HTTP code, the expected results, with the following additional information:

- API Request Information:
    - Resource requested (see section 3- Example Data Tables (Resources))
    - Boolean tag indicating the authentication state (IsLoggedIIn : True or False)
    - User Login
    - Timeout Period (in minutes)

- API Request Interpretation
    - Query field
    - Operator used with description of operator
    - Field value

- API Response Summary
    - Current page number
    - Number of results per page
    - Total results count
    - Link to the next page of results

For example:

<Result>

<APIRequestInfo>

```
        <Module>DataTable</Module>
        <APIVersion>V2.2</APIVersion>
        <Resource>Documents.Image.Default:Create</Resource>
        <IsLoggedIn type="Boolean">True</IsLoggedIn>
        <ProviderVersion>5.5.20.036</ProviderVersion>
        <ProviderIdentity>OL-SERV-NAM</ProviderIdentity>
        <Status>LoggedIn</Status>
        <UserLogin>MyLoginName</UserLogin>
        <TimeoutPeriodMinutes type="Numeric">20</TimeoutPeriodMinutes>
    </APIRequestInfo>
    <RequestInterpretation>
        <Updates type="List">
        <Update>
            <Field>CoreField.Title</Field>
            <Operator>:=</Operator>
            <OperatorDescription>Assign a value to a single-value field</OperatorDescription>
            <Value>An API Created Record</Value>
        </Update>
        </Updates>
    </RequestInterpretation>
    <Response>
        <RecordID>29XDHUFLFLF</RecordID>
        <Code>SUCCESS</Code>
    </Response>
</Result>
```

JSON Parameter

Every CRUD operation has the same XML schema as a result. Only specific operations (such as creating links in the “Documents.Link” data table) will have a different result format.

The output is in XML by default but can be turned into JSON using the URL parameter format=JSON.

For example:
www.sitename.com/API/DataTable/V2.2/Documents.Image.Default:Read?
CoreField.Identifier=DM123&format=JSON

SortField Parameter

Results are ordered by RecordID by default.

They can be sorted by any CoreField using the API parameter SortField.

Add **SortField=[FIELD_API_NAME]** to the URL.
Replace [FIELD_API_NAME] with the API name of the field to sort by.

For example:
www.sitename.com/API/DataTable/V2.2/Documents.Image.Default:Read?
CoreField.EditDate>2018-01-31
&SortField=CoreField.CreateDate

Use Case Scenarios

Listing fields

For Data Table of assets of type “Image” data table (Documents.Image.Default)

www.sitename.com/API/DataTable/V2.2/Documents.**Image**.Default:**ListFields**

Reading records created or edited before/after a given date/time

To get the list of Client record last edited at or after 10th March 2018 8 pm:

www.sitename.com/API/V2.2/DataTable/Contacts.**Client**.Default:Read?
CoreField.EditDate>2018-03-10T20:00:00

To sort the results by EditDate, add &SortField=CoreField.EditDate:

www.sitename.com/API/V2.2/DataTable/Contacts.**Client**.Default:Read?
CoreField.EditDate>2018-03-10T20:00:00
&SortField=CoreField.EditDate

To get a list of Client records created on or before 31st December 2014:

www.sitename.com/API/V2.2/DataTable/Contacts.**Client**.Default:Read?
CoreField.CreateDate<2014-12-31

Create or Update an Image record based on your internal system identifier

www.sitename.com/API/DataTable/V2.2/Documents.**Image**.Default:CreateOrUpdate?
Document.CoreField.Id_Client**=**OL123456
&Document.CoreField.Title**:=**This is the title for this image
&Document.CoreField.CaptionLong**:=**This is the description for this image
&Document.CoreField.Title**:=**Title

Add keywords (multi-valued field) to an image (only if keyword already exists in authority list)

www.sitename.com/API/DataTable/V2.2/Documents.**Image**.Default:Update?
Document.CoreField.Id_Client=OL123456
&CoreField.Keywords+=Keyword1|Keyword2|Keyword3

Add keywords (multi-valued field) to an image (create keyword if does not exist and assign)

www.sitename.com/API/DataTable/V2.2/Documents.**Image**.Default:Update?
Document.CoreField.Id_Client=OL123456
&CoreField.Keywords++=Keyword1|Keyword2|Keyword3

Remove one keyword from an image

www.sitename.com/API/DataTable/V2.2/Documents.**Image**.Default:Update?
Document.CoreField.Id_Client=OL123456&
&CoreField.Keywords-=Keyword1

Remove all keywords from an image

www.sitename.com/API/DataTable/V2.2/Documents.**Image**.Default:Update?
Document.CoreField.Id_Client=OL123456&
&CoreField.Keywords--=

Remove all keywords and add a keyword

www.sitename.com/API/DataTable/V2.2/Documents.**Image**.Default:Update?
Document.CoreField.Id_Client= OL123456
&CoreField.Keywords--=
&CoreField.Keywords+=Keyword4

Moving a folder or an asset into another folder

- Specify which Folder/Asset you want to Update or CreateAndUpdate (i.e. which Folder/Asset you want to move to another Parent Folder)

- Specify the new Parent Folder using the field: CoreField.Mother

Example of moving an image to another parent folder:

www.sitename.com/API/DataTable/V2.2/Documents.**Image**.Default:Update?
CoreField.Identifier=DM5555
&CoreField.Parent-folder:=[Documents.**Folder**.Default:CoreField.Unique-Identifier=DMSTO122]

Example of moving a folder to another parent folder:

www.sitename.com/API/DataTable/V2.2/Documents.**Folde**r.Default:Update?
CoreField.Identifier=DMSTO123
&CoreField.Parent-folder:=[Documents.**Folder**.Default:CoreField.Unique-Identifier=DMSTO122]

The call in between the square brackets [..] is a sub call to locate the Parent Folder in the database based on its Identifier(CoreField.Identifier) or other criteria.

Adding assets into virtual folders

- Specify the asset you want to add into the virtual folder (ChildRecords)
- Specify the Virtual Folder (ParentRecord)

Example of adding an image to a virtual folder:

www.sitename.com/API/DataTable/v2.2/Documents-links:ParentChildLink**?**
ParentRecord=[Documents.Folder.Default:CoreField.Unique-Identifier=DMSTO122]
&ChildRecords=[Documents.Image.Default:CoreField.Unique-Identifier=123456789]

Example of adding a video to a virtual folder:

www.sitename.com/API/DataTable/v2.2/Documents-links:ParentChildLink**?**
ParentRecord=[Documents.Folder.Default:CoreField.Unique-Identifier=DMSTO122]
&ChildRecords=[Documents.Video.Default:CoreField.Unique-Identifier=9876543]

The calls in between the square brackets [..] are sub calls to locate the asset and Virtual Folder in the database, based on their Unique Identifier(CoreField.Unique-Identifier) or other criteria.

If you want to add a large number of assets to the Virtual Folder you can use iAPI for batch applying the API call to all records in a CSV file.

Removing Assets from Virtual Folders

- Specify the Unique Identifier of the Virtual Folder you want to remove the asset from.

- Specify the Unique Identifier of the asset you want to remove from the Virtual Folder.

Example of removing an image from a Virtual Folder:

https://www.sitename.com/API/DataTable/v2.1/Document.Container.Link:Delete?Document.CoreField.DO_RecordIDTop=[Datatable/v2.2/Documents.Virtual-folder.Default:Read?CoreField.Unique-Identifier=CTL47478]&Document.CoreField.DO_RecordIdBot=[Datatable/v2.2/Documents.Image.Default:Read?CoreField.Unique-Identifier=CTL51171]

The calls in between the square brackets [..] are sub calls to locate the asset and Virtual Folder in the database, based on their Unique Identifier(CoreField.Unique-Identifier).

If you want to remove a large number of assets from the Virtual Folder, you can use iAPI for batch applying the API call to all records in a CSV file.

Designating a Lead asset and relating other assets as alternative versions of this Lead asset

AssignLeadDocument can be used to assign the “Lead asset” field to several assets at once.

It takes 2 parameters:

- LeadIdentifier: the identifier of the Lead asset
- TrailIdentifiers: the identifiers of the assets to assign as related versions, separated by commas.

For example:
www.sitename.com/API/DataTable/v2.2/Documents.Image.Default:AssignLeadDocument?LeadIdentifier=DM123
&TrailIdentifiers=DM456

Assign Representative Image

The Assign Representative Image API call allows you to assign a Representative Image to a specific folder. This call uses two parameters:

- CoreField.ID_Client: The folder’s Legacy Identifier. This is located on the Details tab of a folder.
- CoreField.Identifier: The asset’s Unique Identifier. This is located on the Details tab of an

asset.

www.sitename.com/API/DataTable/v2.2/Documents.Folder.Default:Update?CoreField.ID_Client=CTLCT33&CoreField.Representative_DO:=[DataTable/v2.2/Documents.Image.Default:Read?CoreField.Identifier=CTL39449]

The call between the brackets is a sub call to locate the image based on the image's Unique Identifier (CoreField.Identifier).

Assigning the Source to an asset (Contact of type Photographer / Creator / Source)

www.sitename.com/API/DataTable/v2.2/Documents.Video.Default:CreateorUpdate?CoreField.OriginalFileName=beach 20150215.jpg
&CoreField.CT_RecordidSource:=[Contacts.Source.Default:CoreField.Identifier=DMOCT12]

The call in between the square brackets [..] is a sub call to locate the Photographer in the database based on its **FastID** (CoreField.Identifier) or other criteria (e.g. email)

Assigning keywords with Roles in a multi-valued field

www.sitename.com/API/DataTable/v2.2/Documents.Video.Default:CreateorUpdate?CoreField.System-Identifier=DM23478
&Entities.Cast++=Chantal Muller{"Role":"Animation"}

Example with multiple keywords/roles:

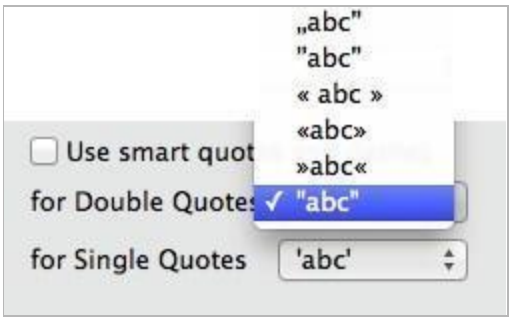
www.sitename.com/API/DataTable/v2.2/Documents.Video.Default:CreateorUpdate?CoreField.System-Identifier=DM23478
&Entities.Cast++=Elsa{"Role":"Actor"}|Olaf{"Role":"Special effects"}

This will populate both the keyword list and roles list on the fly (operator ++=)

If you only want to add the keyword if the role already exists, use the operator +=

> The above calls **do not** work with the curly quotation mark/inverted comma (“). Make sure you are using the straight quotation mark (").

Your keyboard may be set up to automatically convert double and single quotes to the curly characters. On a Mac, the feature “Use smart quotes and dashes” is found under System Preferences > Keyboard > Text.



Set Purposes on Large Amount of Image Assets Using Datatable and iAPI

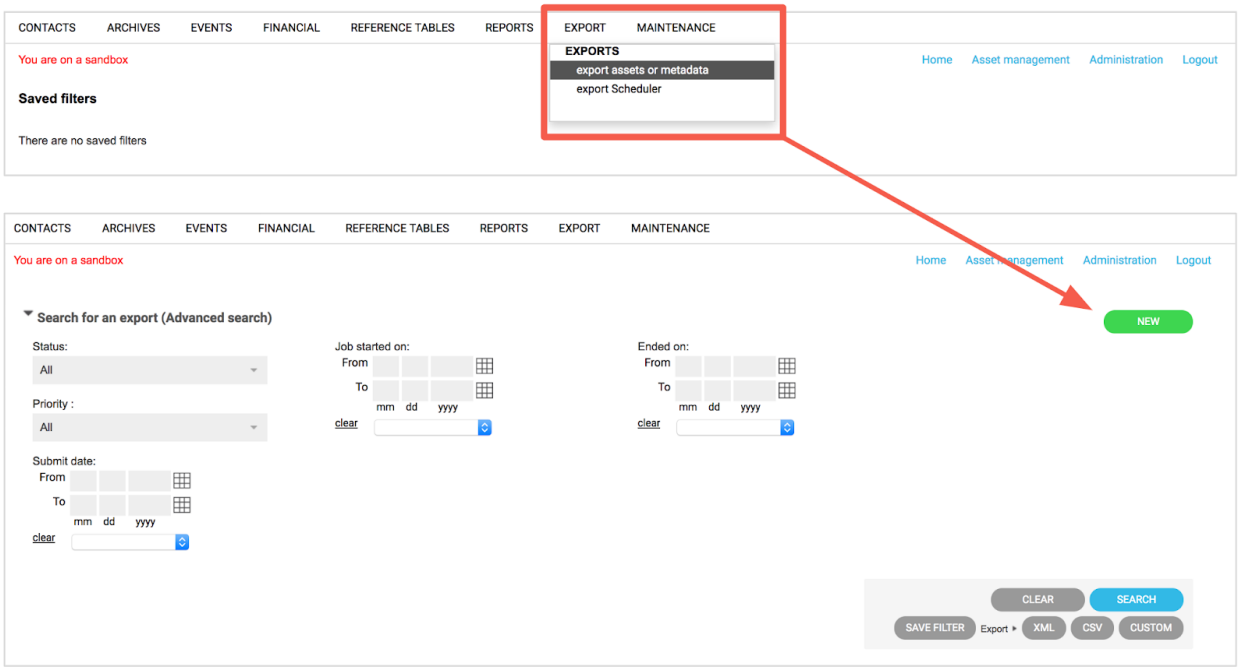
Example: You want to repurpose a large number of images in the DAM

> This instruction works on Windows only. Before you begin, you will need the iAPI tool.

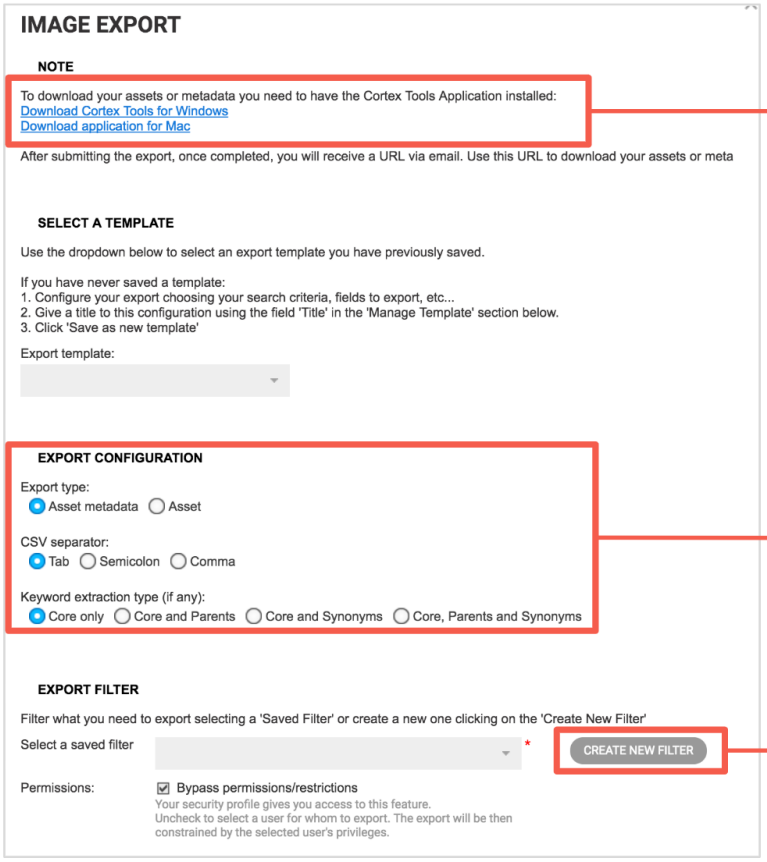
To repurpose a large number of image assets:

Step 1 - Export the Metadata BASED on a saved search

Administration ⇒ Exports ⇒ Export assets or metadata ⇒ New



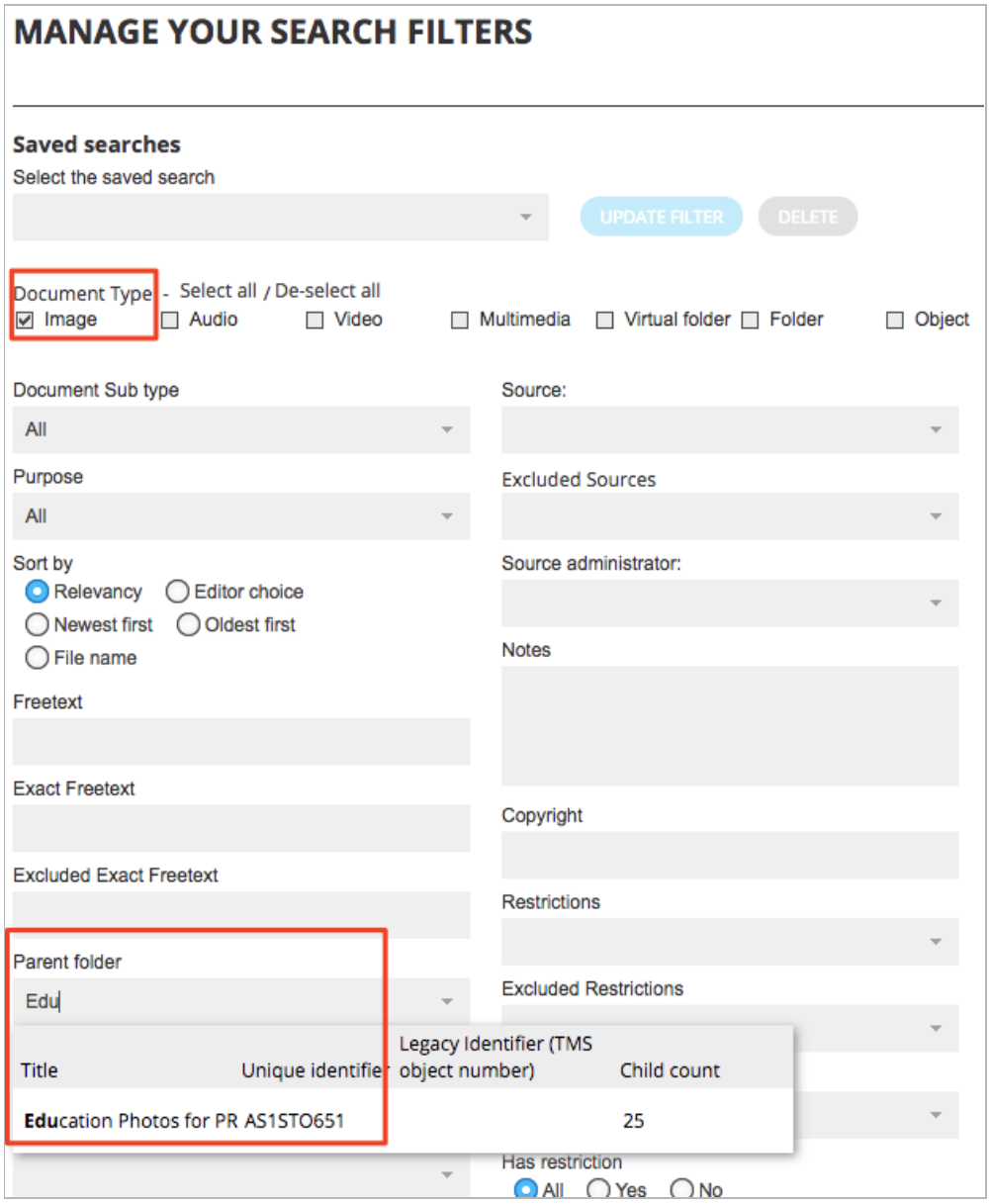
This will open the **Image Export** pop-up window. On this window, do the following:



1. Click on the links to download Cortex Tools
2. Select **Asset Metadata** as Export type and leave other radio buttons set to default
3. Click on this button to create new filter

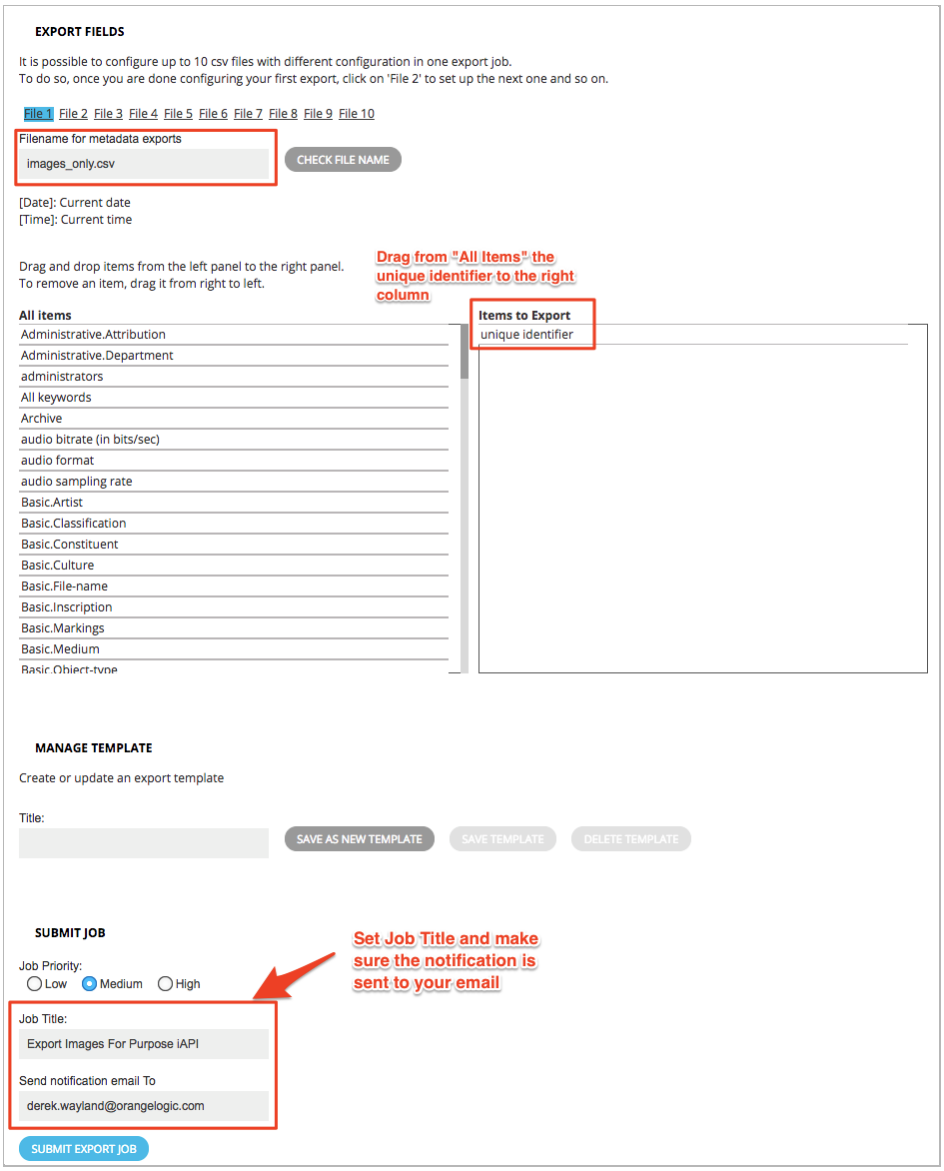
On this pop-up window, do the followings:

1- Be sure to have the Cortex Downloader installed on your machine. (download from the “Cortex Applications” page on your website)
2 - Make sure that the "Export Type" is set to "Asset Metadata" under Export Configuration. (Leave other radio buttons set to default).
3 - Configure a Filter to grab all image assets contained in the top-level folder you want to capture asset metadata.



Click CREATE NEW FILTER at the bottom and save your filter.

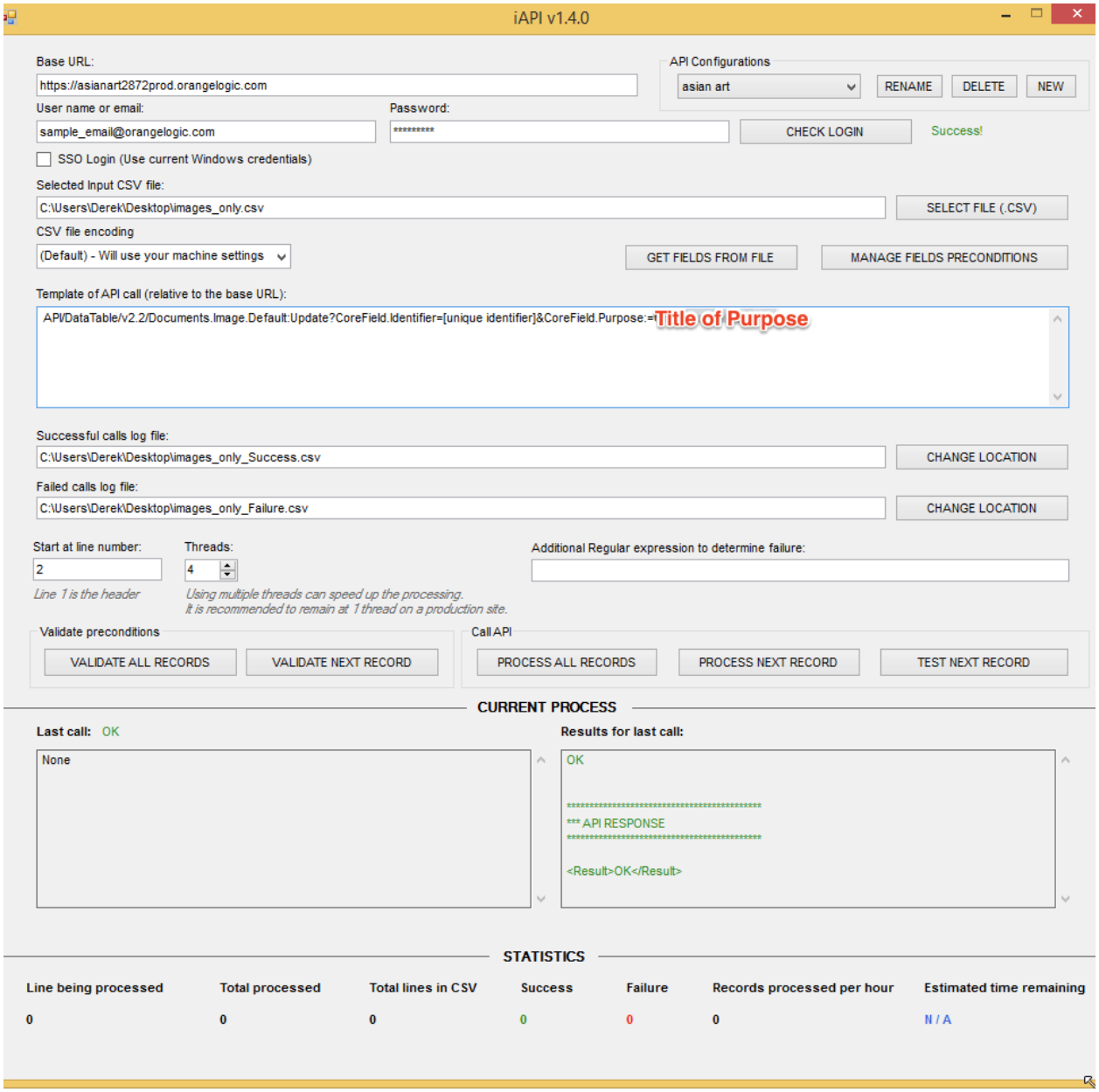
4 - Complete the "Export metadata as follows"



This will be put into a queue which will generate a CSV file - Once it is complete it will email you (may take a few mins -> longer depending on asset count

Step 2 - Populate iAPI call with template API Call and account details.

Fill in the URL and credentials for your user for your site. Take the “images_only.csv” metadata file and import it into API.



Replace the API Template with the following (In this example we will be using Pending Process)
/API/DataTable/v2.2/Documents.Image.Default:Update?CoreField.Identifier=[unique identifier]&CoreField.Purpose:=Pending Process

Re-applying Import Mapping Template to Already Ingested Assets

This API allows re-apply the metadata mapping template to already ingested assets, using the

metadata embedded. This API call can be used in iAPI with a CSV file for a mass update.

Syntax:
www.sitename.com/API/UploadProcess/v1.0/ReprocessDocumentMetadata?DocIdentifier=[Unique identifier]&IngestMappingName=The exact name of the mapping template

For how can you run a call using iAPI, see iAPI User Guide.

> This call requires *OrangeLogic privileges, which are granted by SSO. This can therefore only be used with **iAPI v1.5.0,** which allows users to log by SSO.

To retrieve a CSV file:

Option1: `Run a report from Administration ⇒ Reports ⇒ Documents List`

Option 2: `Select the folder on Asset Management workspaceCustom Report ⇒ Run report`

Option 3: Export data (see Export search results as CSV)

Enable / Disable Automated Email when Creating/Updating User Accounts via API

If you are using APIs to create accounts on a Test or Pre-prod site using CSVs, the events that are generated by your calls are - by default - NOT sending email notifications.

If you DO want emails to be sent, add **&AllowNotifications=1** in your API call.

Examples :

1. You are creating new accounts by API: Event type “Administrator created a new Contact" will be triggered and logged, but will not send email notifications to new Contacts. To enable emails to be sent, add **&AllowNotifications=1** in your API call.

   Emails are sent to the email addresses specified in the field “Email to” field of your Event Type.

2. You are batch updating the “Editor’s Rating” for assets: Event type “Document level changed" will be triggered and logged, but will not send email notifications. To enable emails to be sent, add **&AllowNotifications=1** in your API call.

Emails are sent to the email addresses specified in the field “Email to” field of your Event Type.

Sample call:
API/Datatable/v2.1/Document.Asset.Image:Update?Document.CoreField.Identifier=IMAGEID1&Document.CoreField.DocMark:=I&AllowNotifications=1

Utility Parameters (for Advanced Users/Developers)

UseSystemNames Parameter

Including the API parameter **UseSystemNames=1** will return the system name of all fields rather than the user defined names.

ReturnField Parameter

Including the API parameter **ReturnField=[FIELD_API_NAME]** will only return in the response that field and the encrypted recordID for matching records.

Only one field can be declared in one API call.

Example: set the country for a Client Account

www.sitename.com/API/DataTable/V2.1/Contacts.Client.Default:Update?
Contact.CoreField.Email1=john.doe@orangelogic.com
&Contact.CoreField.Country:=
[DataTable/v2.1/ReferenceTable.Country:Read?ReferenceTable.CoreField.Country_English=France&**ReturnField**=ReferenceTable.CoreField.CountryCode]

Example: set the parent Company Account for a Staff Account

https://www.sitename.com/API/DataTable/v2.2/Contacts.Staff.Default:Update?
CoreField.Email-address=john.doe@orangelogic.com
&CoreField.Company:=[DataTable/v2.2/Contacts.Company.Default:Read?CoreField.Company-name=Orange Logic&**ReturnField**=RecordID]

Media Upload API

Terminology

Base Name - The name of a computer file without its file extension. The base name of “example.1001.cr2” is “example.1001” and the extension is “.cr2”

Proxies - Video files which have a lower quality than the original, and are designed for downloading or streaming.

Thumbnail - A representative, low resolution image, taken from the sequential frames of a video or downsized from a larger source image. Used to visually distinguish one file from another.

Background

Programmatically uploading an asset to your OrangeDAM DAM is a 3-step process:

1) Authenticate the session utilizing the Authentication API.
2) Identify the destination folder utilizing the Search API.
3) Upload the assets to the folder utilizing the Media Upload API.

Media Upload API does not support chunked file transfer and it does not support uploading files that are larger than 2 GB. To upload a file that is larger than 2 GB, it is recommended to use Cortex Tools or Cortex Uploader. If you wish to upload large files using OrangeDAM APIs, please contact Orange Logic support for more instructions on the workflow.

> The 2 GB limitation is imposed by Microsoft IIS and it varies based on your OrangeDAM installation as well as the IIS configuration in your system. Please contact your system administrator for information about your website limit.

The **Media Upload API** (also referred to as the “Asset Ingest API”) uses a standard HTTP POST request to upload the asset to the appropriate directory and then returns the System Identifier in the response. This method is available in Kyoto and later releases of OrangeDAM. For earlier versions of OrangeDAM, please refer to:

- Geneva, Helsinki, Idlewild, and early Kyoto - Media Upload API v2.0 Documentation
- Florence and prior - Media Upload API v1.0 Documentation

Authenticating the Session

The Media Upload API requires authentication using a token obtained through the Authentication API. The token is valid for the duration of the session.

Identifying the Destination Folder

Before transmitting the file, you must obtain the `MediaEncryptedIdentifier` of the folder you want to upload into. This is accomplished through the OrangeDAM user interface or by using the MediaEncryptedIdentifier query as part of the Search API.

Using the OrangeDAM User Interface

You can use the OrangeDAM user interface to find a folder's MediaEncryptedIdentifier.

In Asset Management, right-click on the folder you wish to upload into, and select **Get Link**. The link will be in the form: https://www.sitename.com/asset-management/**2RODHUWEFI8**

2RODHUWEFI8 is the `MediaEncryptedIdentifier` of the folder.

Using the Search API

Either the folder's **unique identifier** (if known) or its **title** (if unique) are passed along to the Search API with a `fields` query for `MediaEncryptedIdentifier`. If successful, the response will include the folder's MediaEncryptedIdentifier.

Example: A query is made for a folder with the unique identifier ABC12345:

```
https://www.sitename.com/API/Search/v3.0/search?
query=ABC12345
&fields=MediaEncryptedIdentifier
```

Response:

```
<MediaEncryptedIdentifier>2RODHUWEFI8</MediaEncryptedIdentifier>
```

> When passing along a folder title containing an ampersand (&) or equals sign (=), be sure to escape those characters in the query. See Search API failure codes for troubleshooting.

Assembling the POST Parameters

Token - Login token retrieved from the Authentication API

Example: Token: OrangeDAMVxe3YevGSVl2ZW7QZbLbtCq.1fgE3m68nENBSPRPjBY*

FolderRecordID - The MediaEncryptedIdentifier of the destination folder (obtained from the Search API)
Example: FolderRecordID: 2RODHUWEFI8

FileName - Base name and filetype extension of the asset to upload into OrangeDAM.
Example: FileName: test.jpg

UploadMode - a required parameter which defines how the asset will be processed once on the server. Valid values are:

> **ProcessInLine** - Fully process the asset before the API returns. This will take the longest amount of time, but proxies and representatives will be generated before the API returns.
>
> **ProcessFullyInBackground** - The API returns immediately after creating the document in OrangeDAM. This will take a short amount of time, but there will be a delay before the asset can be downloaded, proxies are available, and thumbnails are generated.
>
> **ProcessFullyInBackgroundLite** - The API returns immediately after uploading the file in OrangeDAM but before the document is created. This will take the shortest amount of time, but there will be a delay before the document is created in OrangeDAM.
>
> **ProcessProxiesInBackground** - The API returns after making the asset available for download in OrangeDAM, but not before generating proxies and thumbnails. This mode is slightly faster than ProcessFullyInBackground.

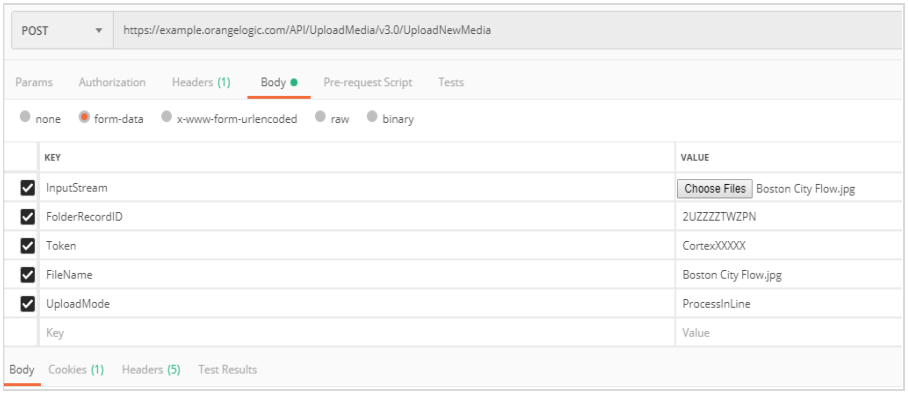
InputStream - HTTP POST stream of file data.

Using Postman to Send POST Requests

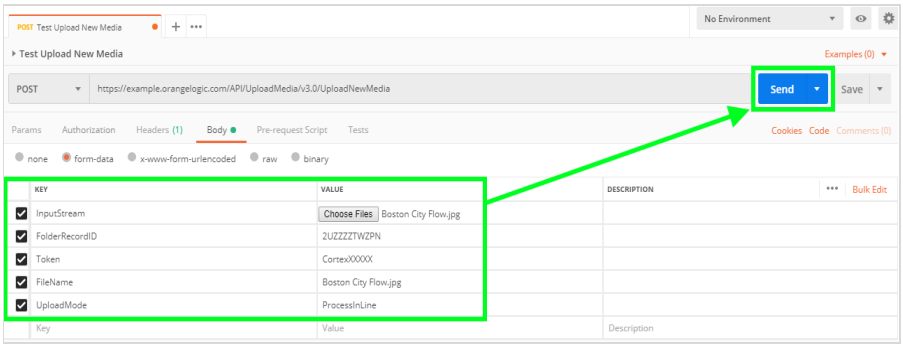
Postman (www.getpostman.com) is a free desktop application or Chrome extension which allows you to make HTTP POST requests.

To send Media Upload API POST commands using Postman:

1. Create a new command.
2. Select POST and enter the API URL:
   https://www.sitename.com/API/UploadMedia/v3.0/UploadNewMedia
3. Add the following form-data in the “Body”:
   - InputStream - Click “Choose Files” to select an asset to upload.
   - FolderRecordID
   - Token
   - FileName
   - UploadMode



4. Click the blue button ‘Send’



Example Code

C#

```
// Example C# code calling API with Microsoft.Net.Http library
using (var webClient = new HttpClient())
using (var multiPartFormData = new MultipartFormDataContent())
{
      // MediaEncryptedIdentifier obtained from API Search/v3.0/search API.
      multiPartFormData.Add(new StringContent("2RODHUWEFI8"),
      "FolderRecordID");
      // Token obtained through Authentication/v1.0/Login API
      multiPartFormData.Add(new StringContent("abc123"), "Token");
      // File to upload
      multiPartFormData.Add(new StreamContent(File.OpenRead(@"C:\test.jpg")),
      "InputStream", "InputStream");
      // File name of the file to upload
      multiPartFormData.Add(new StringContent("test.jpg"), "FileName");
      // How you would like the uploaded file to be processed
      multiPartFormData.Add(new StringContent("ProcessFullyInBackground"),
      "UploadMode");

      var response =
      webClient.PostAsync("https://www.sitename.com/API/UploadMedia/v3.0/Uploa
      dNewMedia", multiPartFormData).Result;
}
```

Java

```
// Example Java code using Apache HTTP Client library
(https://hc.apache.org/).
CloseableHttpClient httpClient = HttpClients.createDefault();

HttpPost httpPost = new
HttpPost("https://www.sitename.com/API/UploadMedia/v3.0/UploadNewMedia");
FileBody file = new FileBody(new File("C:\\test.jpg"));
HttpEntity entity = MultipartEntityBuilder.create()

      // MediaEncryptedIdentifier obtained from API Search/v3.0/search API.
      .addPart("FolderRecordID", new StringBody("2RODHUWEFI8",
      ContentType.TEXT_PLAIN))
      // Token obtained through Authentication/v1.0/Login API
      .addPart("Token", new StringBody("abc123", ContentType.TEXT_PLAIN))
      // How you would like the uploaded file to be processed
      .addPart("UploadMode", new StringBody("ProcessInLine",
      ContentType.TEXT_PLAIN))
      // File name of the file to upload
      .addPart("FileName", new StringBody("test.jpg", ContentType.TEXT_PLAIN))
      // File to upload
      .addPart("InputStream", file)
      .build();

httpPost.setEntity(entity);

CloseableHttpResponse response = httpClient.execute(httpPost);

response.close();
httpClient.close();
```

Example Success Response

The Response includes the Unique System Identifier of the uploaded file.

```
<Result>
<APIRequestInfo>
        <Module>UploadMedia</Module>
        <APIVersion>v3.0</APIVersion>
        <Resource>UploadNewMedia</Resource>
        <IsLoggedIn type="Boolean">True</IsLoggedIn>
        <ProviderVersion>KYOTO.A1.1215A.46000</ProviderVersion>
            /* This is the version of Cortex. */
        <ProviderIdentity>OLXXXX</ProviderIdentity>
            /* This is the server the response came from. */
        <Status>LoggedIn</Status>
            /* This confirms that the login was successful. */
        <UserLogin>userXXXX</UserLogin>
            /* This confirms the User logged in */
        <TimeoutPeriodMinutes type="Numeric">60</TimeoutPeriodMinutes>
</APIRequestInfo>
<APIResponse>
        <RequestInterpretation>
            <ParametersInformation>
                <FolderRecordID>
                  <Value>AB124ABF</Value>
                  /* The MediaEncryptedIdentifier for the Folder to upload
                  into (obtained from the Search API)*/
                  <IsOptional type="Boolean">False</IsOptional>
                  <ValueCount type="Numeric">1</ValueCount>
                </FolderRecordID>
                <InputStream>
                  <Value>Binary data</Value> /* The Input Stream */
            <IsOptional type="Boolean">False</IsOptional>
            <ValueCount type="Numeric">0</ValueCount>
                </InputStream>
                <FileName>
                  <Value>file.jpg</Value>
                  /* The filename of the file to upload. */
                  <IsOptional type="Boolean">False</IsOptional>
                  <ValueCount type="Numeric">1</ValueCount>
                </FileName>
                <UploadMode>
                  <Value>ProcessFullyInBackground</Value>
                  <IsOptional type="Boolean">False</IsOptional>
```

```
                <ValueCount type="Numeric">1</ValueCount>
            </UploadMode>
        </ParametersInformation>
    </RequestInterpretation>
    <Response>
        <ExecutionStatus>Executed</ExecutionStatus>
        <HTTPCode type="Numeric>200</HTTPCode>
        <Code>SUCCESS</Code>
        /* This confirms that the upload was successful. */
        <SystemIdentifier>ID123</SystemIdentifier>
        /* The Unique System Identifier assigned by Cortex to the uploaded
        file */
    </Response>
</APIResponse>
</Result>
```

Example Error Message

```
<Response>
      <ErrorList>
            <ErrorDetails>Upload was successful, however the file could not be
            ingested. This will be retried by the agent in the background.
            Details: The file has been rejected by the system, it seems that
            this is not a valid .doc file.</ErrorDetails>
      </ErrorList>
      <ExecutionStatus>Executed</ExecutionStatus>
      <HTTPCode type="Numeric>200</HTTPCode>
      <Code>FAILURE</Code>
</Response>
```

> The API URL is *https://www.sitename.com/API/***UploadMedia***/v3.0/UploadNewMedia* by default. However, the “UploadMedia” URL string can be modified through the **Media_DbBO** parameter in your installation of OrangeDAM, making your particular URL unique.

Generate or Regenerate Proxies for Assets

When you upload an asset, OrangeDAM generates multiple files for the asset. For one asset, you can have the following files:

- **Original File**: The Original file that is uploaded.
- **Large Proxies**: These are used for alternate download formats and large asset preview files. These are typically lower resolution versions of the original file.
- **Small Proxies**: These are used as thumbnails in the Search Results.

Orange Logic can configure specific preset formats for viewing and downloading assets. After Orange Logic has configured these formats, you can call an API to create or regenerate these formats for a specific asset or for all assets in OrangeDAM.

> 💡 **Tip**
>
> To view the list of formats, go to **Administration** > **Archives** > **Documents Formats**.

Generate a Format for One Image

Let's say Orange Logic configures the **CMS1** format for **images**, and you want to generate this format for the asset with this Unique ID: **CTL232017**.

Syntax:
https://www.sitename.com/API/V1.0/Media_DbBO/RegenerateProxiesForAsset?AssetIdentifier=[*Unique ID*]&FileFormat=[*Format*]&DocType=[*Asset Type*]

Example:
https://www.sitename.com/API/V1.0/Media_DbBO/RegenerateProxiesForAsset?AssetIdentifier=CTL232017&FileFormat=CMS1&DocType=Image

Regenerate a Format for All Assets of One Type

Let's say Orange Logic modifies the **TR10** format for **images**, and you want to update this format for all images in OrangeDAM.

Options:

- You can use the asterisk wildcard (*) instead of a specific Unique Identifier to apply the API call to all assets.
- You can use **Priority=-100** so that the processing takes place in the background and does not impact new uploads.
- You can add comments to reference a task or provide the format that is being generated. These comments will be kept in the database and can be viewed at **Administration** > **Maintenance** > **Show Processing Tasks**.

Syntax:
https://www.sitename.com/API/Media_DbBO/v1.0/RegenerateProxiesForAsset?AssetIdentifier=[*Unique Identifier*]&FileFormat=[*Format*]&DocType=[*Asset Type*]&Priority=-100&Comments=[*Comment - Freetext*]

Example:
https://www.sitename.com/API/Media_DbBO/v1.0/RegenerateProxiesForAsset?AssetIdentifier=*&FileFormat=TR10&DocType=Image&Priority=-100&Comments=Generating the new TR10 proxy format for all images

Generate a Format for All Videos That Do Not Already Have That Format

Let's say you want to generate a high-resolution, watermarked format for all videos.

Options:

- You can use **&GenerateOnlyIf=TheFormatDoesntExist** to skip videos that already have the format.
- You can use **Priority=-100** so that the processing takes place in the background and does not impact new uploads.

Syntax:
https://www.sitename.com/API/Media_DbBO/v1.0/RegenerateProxiesForAsset?AssetIdentifier=*&FileFormat=[*Format*]&DocType=Video&GenerateOnlyIf=TheFormatDoesntExist&Priority=-100&Comments=[*Comment - Freetext*]

Example:
https://www.sitename.com/API/Media_DbBO/v1.0/RegenerateProxiesForAsset?AssetIdentifier=*&FileFormat=WebHigh_Watermarked&DocType=Video&GenerateOnlyIf=TheFormatDoesntExist&Priority=-100&Comments=Computing missing format

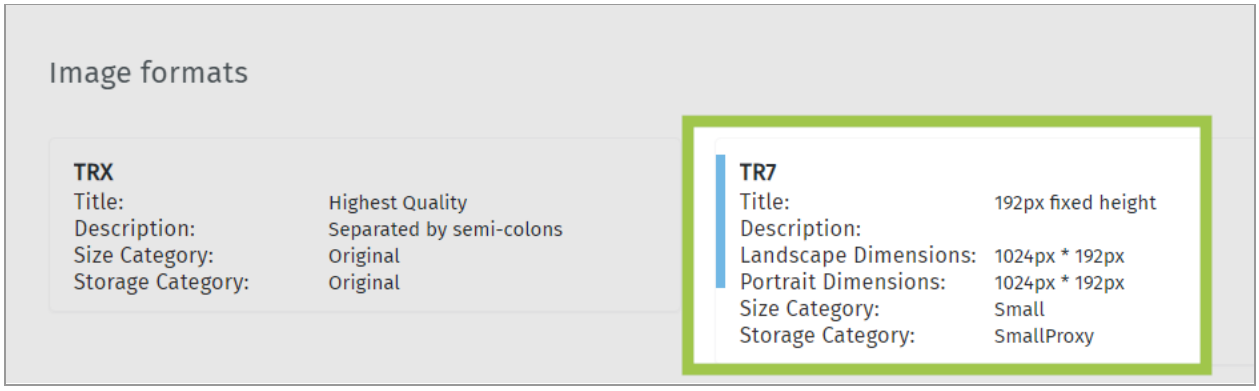
Extract Original Document API

The Extract Original Document API allows you to download the Original, high-resolution format (TRX).

You can also use this API to generate other formats on demand. To view a list of available on-demand formats, go to **Administration** > **Archives** > **Documents Formats**.

Formats that are generated on import are highlighted with a blue line. Some examples of these types of formats are:

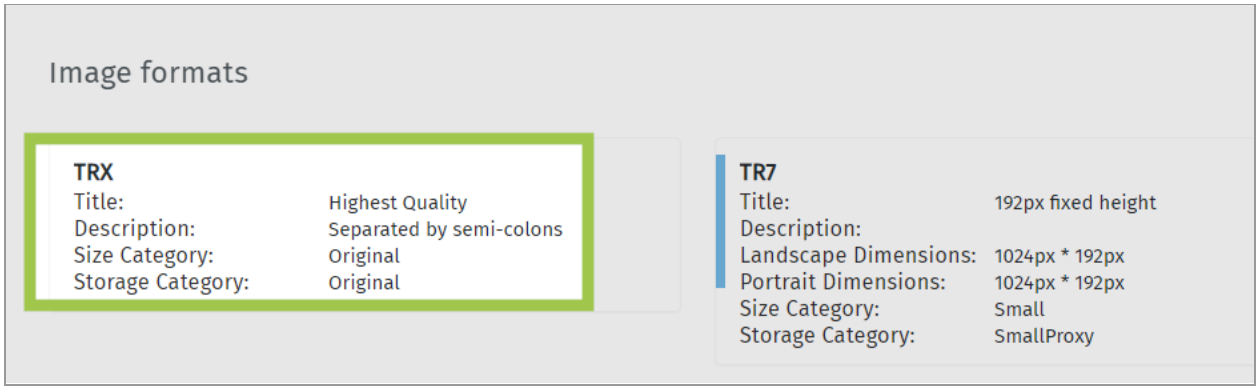
- **TR1**: Medium res. (1200px x 1200px). This format is used on the Overview tab.
- **TR4**: Small thumbnail (352px fixed height). This format is used in the Search Results.
- **TR7**: Large thumbnail (192px fixed height). This format is used in the Search Results.



Formats Highlighted in Blue Are Generated on Import

Formats that are generated on demand are not highlighted. Some examples of these types of formats are:

- **TRX**: Highest quality
- **CMS1**: 2000px x 2000px
- **CMS2**:- 1000px x 1000px
- **CMS3**: 500px x 500px



Formats That Are Not Highlighted Are Generated On Demand

The Extract Original Document API uses these parameters:

- Format
- Unique Identifier

Syntax:

Use one of the following API syntaxes:

- https://www.sitename.com/htm/GetDocumentAPI.aspx?F=[*Format*]&DocID=[*Unique Identifier*]
- https://www.sitename.com/htm/GetDocumentAPI.aspx?F=[*Format*]&DocID=[*Encrypted RecordID*]

Example:
https://www.sitename.com/htm/GetDocumentAPI.aspx?F=TRX&UniqueID=CTL238597

The API retrieves the file.

Authentication

Authentication to use the Extract Original Document API can use a cookie or a token requested through the Authentication API.

Important Note:

- To be authorized to download the original, high-resolution document, the logged in user must have the following Security Function:
    - API > Download media > Can download high res.

- ~~The following Security Functions restrict high resolution images that can be downloaded based on image ranking/level:~~
    - ~~API > Download media > Can download media at level 'Public +++ (A)'~~
    - ~~API > Download media > Can download media at level 'Public ++ (B)'~~
    - ~~API > Download media > Can download media at level 'Public + (C)'~~
    - ~~API > Download media > Can download media at level 'Public (E)'~~
    - ~~API > Download media > Can download media at level 'Purgatory (F)'~~
    - ~~API > Download media > Can download media at level 'Internal (G)'~~
    - ~~API > Download media > Can download media at level 'Bin (I)'~~

~~For example, a user with the Security Functions "Can download high res" and "Can download media at level 'Public +++ (A)' will only be able to download the original, high-resolution image if the image is assigned the ranking/level 'Public +++ (A)'.~~

Asset Link API

Create a Link to an Asset (CreateFormatLink)

The **Create Format Link operation** allows you to create hyperlinks to assets in OrangeDAM. The link can be configured to:

- View or download the asset.
- Define whether users need to log in to use the link.
- Display the format of your choice (if the format is available for that asset type).
- Always link to the current version or link to the newest version when an asset is updated in OrangeDAM. Different versions of an asset in OrangeDAM are stored under the same Unique Identifier which means the API calls will always have the same syntax. The API allows you to define which version you want to use.

In Universal View (Asset Management), you can create these links using the Get Link button. This section provides instructions on how to create these links using the **Asset Link APIs, Create Format Link** operation.

Create Format Link API Syntax

https://sitename.com**/API/AssetLink/V1.0/CreateFormatLink?**
Identifier=[UniqueIdentifier]
&CreateDownloadLink=[1or0]
&StickToCurrentVersion=[1or0]
&AssetFormat=[AssetFormat]
&LogAccess=[1or0]
&ExpirationDate=[ExpirationDate]

Parameters

- **Identifier:** Asset's Unique Identifier, System Identifier, or System ID.
- **CreateDownloadLink**: Use to set the function for the URL:
    - Enter **true** if you want to create a link that will download the asset when clicked.
    - Enter **false** if you want to create a link that allows users to view the asset when clicked (view only).
- **StickToCurrentVersion**: Define which version of the asset you want to link to:

  - Enter **true** if you want to create a URL that will always link to the current version of the asset, even when there are newer versions of that asset in OrangeDAM.
  - Enter **false** if you want to create a URL that will link to the newest version of the asset.
- **AssetFormat**: Define the format that the URL will link to. Enter the API name of the format that you want to query such as TR1, TR1_Comp. To see the list of available formats in OrangeDAM, go to **Administration** > **Archive** > **Assets** > **Document Formats**.
- **LogAccess**: Define whether or not users must be logged in to use the URL:
  - Enter **true** to create links that requires users to log in to use.
  - Enter **false** to create links that does not require users to log in to use. This will allow unregistered users to see or download the asset.
- **ExpirationDate (optional)**: Set an expiration date for the URL. If omitted, the URL will not have an expiration date. Available formats (ISO 8601 standard) are:
  - 2018-07-24T07:08:04+00:00
  - 2018-07-24T07:08:04Z
  - 2018-07-25

> **Example: Embed a Video from OrangeDAM on Your External Website**
>
> On your external homepage, you want to create a link that will display the video CTL10969 from OrangeDAM when clicked. You have these additional requirements:
> - Users are not required to log in to see the video.
> - The video should have the original format that was uploaded to OrangeDAM (TRX).
> - The link should always show the newest version of the video.
> - The link should not have an expiration date.
>
> To create this link:
> 1. Use the Asset Link API to retrieve the embeddable link to the video. The parameters in the syntax address all your requirements. :
>
>   **Syntax**
>   http://OrangeDAM-acmeassets.com/API/AssetLink/V1.0/CreateFormatLink?Identifier=CTL10969
>   &CreateDownloadLink=0
>   &StickToCurrentVersion=0
>   &AssetFormat=TRX
>   &LogAccess=0

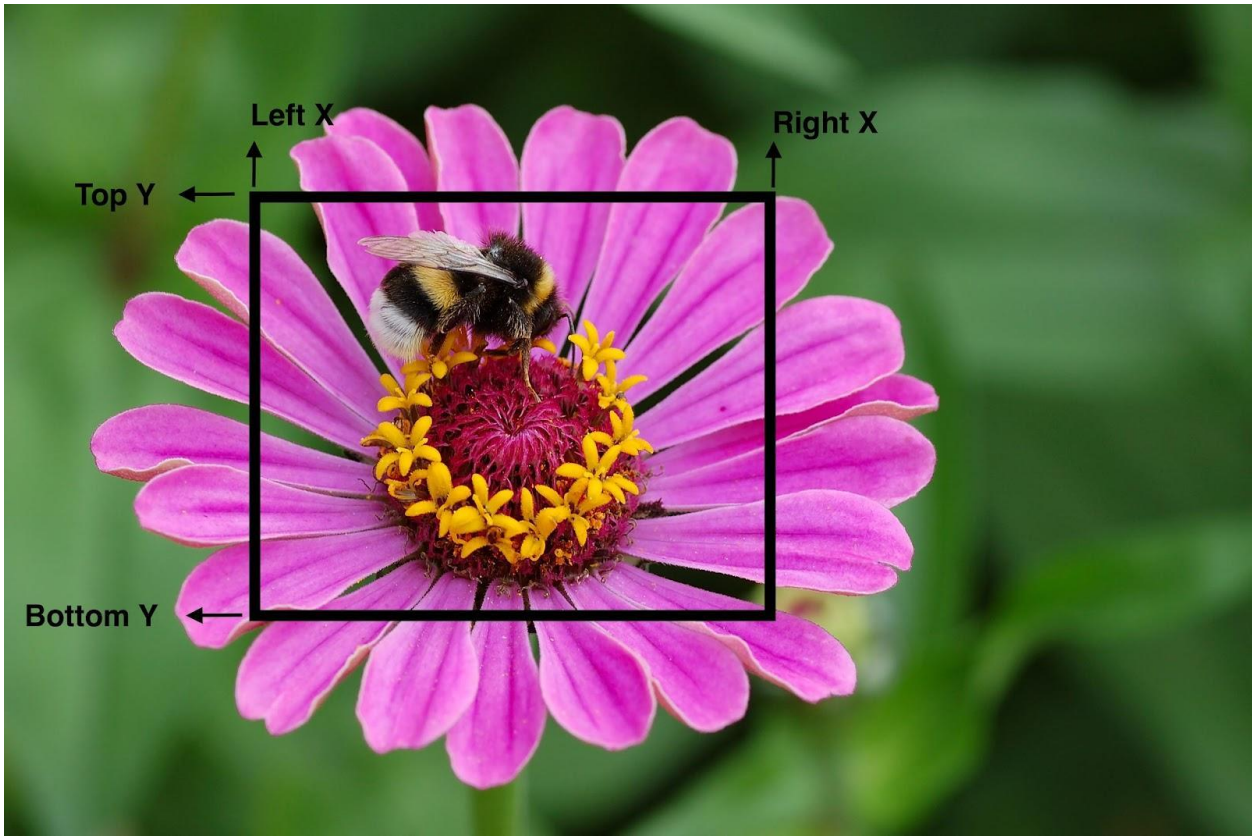
> **Response**
> <Link>**https://OrangeDAM-acmeassets.com/AssetLink/ocm6u1o32l7xo6055u0hq04t5451u15q.mp4**</Link>
>
> 2. Once you have the link, embed it on your homepage.

Create a Link to a Cropped Asset (CreateCropLink)

A call with CreateCropLink enables you to create links to a cropped version of the asset.

Parameters	Value
Identifier	[OrangeDAM Unique Identifier]
CreateDownloadLink	**1:** Create Link to downloadable asset on external site **0:** Create Link to embed asset on external site (view only link)
Height	**Value in pixels** Height of the original image To get the height and width of the original image, use the Search API: https://www.sitename.com/API/Search/v3.0/Search?query=SystemIdentifier:[OrangeDAM Unique Identifier]&fields=maxwidth,maxheight
Width	**Value in pixels** Width of the original image To get the height and width of the original image, use the Search API: https://www.sitename.com/API/Search/v3.0/Search?query=SystemIdentifier:[OrangeDAM Unique Identifier]&fields=maxwidth,maxheight
OutputHeight	**Value in pixels** Required height for the cropped image
OutputWidth	**Value in pixels**

	Required width of the cropped image
TopY	**Value from 0 to 1** Ratio of Top Y coordinate of crop rectangle to the height of the original image.
BottomY	**Value from 0 to 1** Ratio of Bottom Y coordinate of crop rectangle to the height of the original image.
LeftX	**Value from 0 to 1** Ratio of Left X coordinate of crop rectangle to the width the original image.
RightX	**Value from 0 to 1** Ratio of Right X Y coordinate of crop rectangle to the width of the original image.
ApplyWatermark	**1:** A watermark will be applied to the bottom right of the cropped image **0:** No watermark will be applied to the cropped image
LogAccess	**1:** Link should be logged **0:** Link should NOT be logged The link URL and the X-Request-Id of the user creating the link are logged in: ● "*View Shares*" (Details tab for an asset > Actions) ● "*My Shared Assets*" (user menu in the top right of the screen) ● "*Full Logs of Views and Downloads*" (Details tab for an asset > Actions) ● The following reports (Administration » Reports » Full logs of views and downloads): ○ *"Full logs of views and downloads Full"* ○ *"Full logs of views and downloads Full (Visual)"*
ExpirationDate	Use ExpirationDate to define the expiration date for the link, using any of the following formats (ISO 8601 standard): ● 2018-07-24T07:08:04+00:00 ● 2018-07-24T07:08:04Z ● 2018-07-25


Left X
Right X
Top Y
Bottom Y

Examples

To create a link to download the watermarked cropped asset on an external site:

```
https://www.sitename.com/API/AssetLink/V1.0/CreateCropLink?
Identifier=DM1300166
&CreateDownloadLink=1
&Height=2362
&Width=3543
&OutputHeight=500
&OutputWidth=500
&TopY=0.1
&BottomY=0.9
&LeftX=0.2
&RightX=0.6
&ApplyWatermark=1
&LogAccess=1
&ExpirationDate=2018-07-25
```

Response:

```
<Link>https://www.sitename.com/AssetLink/w4g82d56624q662l5k62vq362f1dm0mb.jpg<
/Link>
```

To create a view link to embed the **non-**watermarked cropped asset on an external site:

```
https://www.sitename.com/API/AssetLink/V1.0/CreateCropLink?
Identifier=DM1300166
&CreateDownloadLink=0
&Height=2362
&Width=3543
&OutputHeight=500
&OutputWidth=500
&TopY=0
&BottomY=0.5
&LeftX=0
&RightX=0.5
&ApplyWatermark=0
&LogAccess=1
&ExpirationDate=2018-07-25
```

Response:

```
<Link>https://www.sitename.com/AssetLink/115123w7qpg751t076q635a83ao2abcd1.jpg
</Link>
```

Lightbox API

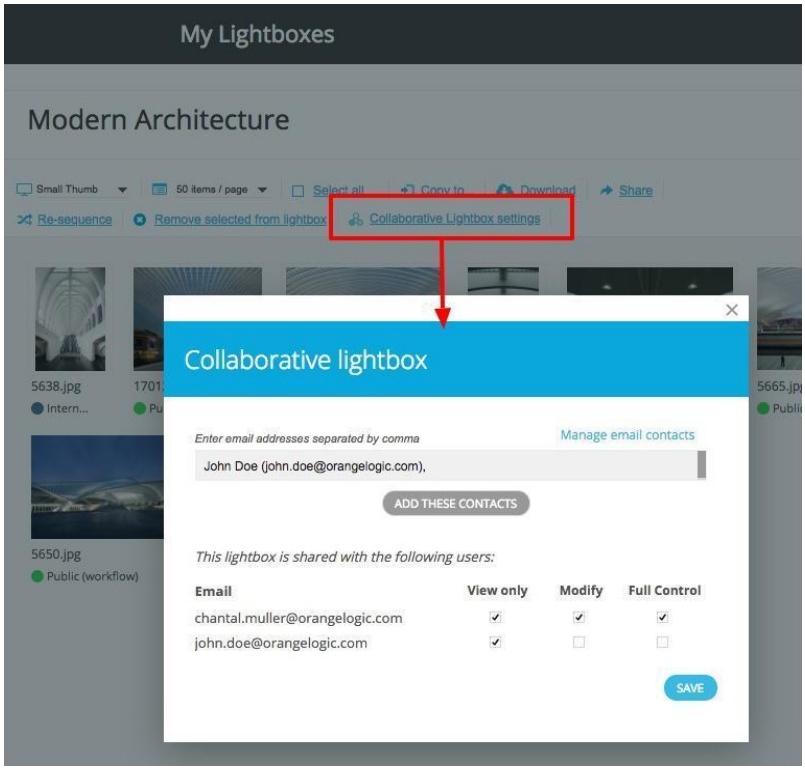
Introduction

The Lightbox API can be used for:

- searching lightboxes (LightboxSearch)
- listing lightboxes' content (LightboxListContent)
- listing lightboxes' "share" settings (LightboxListShares) (Collaborative Lightbox settings)

Lightboxes' "share" settings are those in the "Collaborative Lightbox" feature: a lightbox can be shared with users, granting them the following access privileges:

- 'View only" (Allow users to view any changes to the lightbox)
- 'Modify' (Allow users to delete, add, and reorder contents of the lightbox)
- 'Full Control' (Allow users to change the Collaborative settings, delete, add, and reorder contents of the lightbox)



Authentication and Security

Authentication

Authentication to use the Lightbox API can use a cookie or a token requested through the Authentication API.

If you do not have access, please "Request access" (blue button below).

Security

If the security point "API > Lightbox APIs > Can see all lightboxes for all users through APIs" is given to the API user, all lightboxes can be retrieved with a Lightbox API search.

If the security point is not given to the API user, only lightboxes owned or shared with the API user will be retrieved.

Accessing the API

The Lightbox API is accessible through the page www.sitename.com/API/Lightbox/v1.0 where www.sitename.com (the base URL) is your website address.

All example calls in this documentation are given without the base URL.

GET and POST requests

For API testing purposes, every action can be performed using POST or GET HTTP requests.

- The maximum URL length for a GET request is 260 characters.
- For POST requests, the parameters are passed in the HTTP header or in the URL. If the same parameter is present in both, the value in HTTP header will override the URL.

Search for lightboxes (LightboxSearch)

You can search for all lighboxes in the application (search with no criteria, or “blank search”):

/API/Lightbox/v1.0/LightboxSearch

You can also search for lightboxes based on any of the following criteria:

/API/Lightbox/v1.0/LightboxSearch?LightboxIdentifier=2V08M1S_U

/API/Lightbox/v1.0/LightboxSearch?Name=Full name
/API/Lightbox/v1.0/LightboxSearch?Name=Prefix*
/API/Lightbox/v1.0/LightboxSearch?NumberOfItems
/API/Lightbox/v1.0/LightboxSearch?EditedFrom=2015
/API/Lightbox/v1.0/LightboxSearch?EditedFrom=2012-06-07
/API/Lightbox/v1.0/LightboxSearch?EditedTo=2015
/API/Lightbox/v1.0/LightboxSearch?EditedTo=2012-06-07
/API/Lightbox/v1.0/LightboxSearch?CreatedFrom=2015
/API/Lightbox/v1.0/LightboxSearch?CreatedFrom=2012-06-07
/API/Lightbox/v1.0/LightboxSearch?CreatedTo=2015
/API/Lightbox/v1.0/LightboxSearch?CreatedTo=2012-06-07
/API/Lightbox/v1.0/LightboxSearch?CreatedFrom=2012-06-01&CreatedTo=2012-06-30
/API/Lightbox/v1.0/LightboxSearch?ContainsAsset=DMO128
/API/Lightbox/v1.0/LightboxSearch?SharedWithUserName=John
/API/Lightbox/v1.0/LightboxSearch?SharedWithUserName=Doe
/API/Lightbox/v1.0/LightboxSearch?SharedWithUserName=John Doe
/API/Lightbox/v1.0/LightboxSearch?SharedWithUserName=Doe John
/API/Lightbox/v1.0/LightboxSearch?SharedWithUserName=Doe
/API/Lightbox/v1.0/LightboxSearch?SharedWithUserEmail=j.doe@gmail.com

Extract from the response:

<Lightbox>

<LightboxIdentifier>2V08M1N_X</LightboxIdentifier>

<Name>Chris demo</Name>

<NumberOfItems/>

<CreateDate type="DateTime">2014-09-17T23:49:58</CreateDate>

<LastEditDate type="DateTime">2014-09-17T23:50:04</LastEditDate>

</Lightbox>

Listing lightboxes' content (LightboxListContent)

LightboxListContent is used to retrieve the list of documents contained in the lightbox.

Documents are listed in the same order they appear in the user's lighbox.

Example
/API/Lightbox/v1.0/LightboxListContent?Lightbox.Identifier=2V08M1S_U
Lightbox.Identifier can be recovered from the LightboxSearch call.

For each document the Identifier and Title are provided

Listing lightboxes' share settings (Collaborative Lightbox) - (LightboxListShares)

Provides the list of users with whom a lightbox is shared, including the share privileges for each user (View only, Edit/Modify or Admin/Full Control).

Example:

/API/Lightbox/v1.0/LightboxListShares?Lightbox.Identifier=2V08M1S_U

Extract from the response:

<Item>

<UserIdentifier>KU1B2PK_D</UserIdentifier>

<UserEmail>dev@orangelogic.com</UserEmail>

<UserFirstName>Clark</UserFirstName>

<UserLastName>Kent</UserLastName>

<UserCompany/>

<UserRights>Admin</UserRights>

</Item>

Note:

Values for the tag <UserRights> are:

- ViewOnly ('View only" access to the lightbox)
- Edit ('Modify' privileges to the lightbox)
- Admin ('Full Control' privileges to the lightbox)

Encoding special characters in parameter values

Special characters in parameter values must be URL encoded using percent-encoding.

Refer to http://en.wikipedia.org/wiki/Percent-encoding for the percent-encoding of common characters, such as:

Special character	Percent-encoding
&	%26

=	%3D
+	%2B

The percent-encoding of the “space” characters into %20 is optional.

Example:

john.doe%2Bemail2@gmail.com

must be used instead of:

john.doe+email2@gmail.com

/API/Lightbox/v1.0/LightboxSearch?SharedWithUserEmail=john.doe%2Bemail2@gmail.com

Pagination

Pagination is handled by the CurrentPage parameter:

/API/Lightbox/v1.0/LightboxSearch?CurrentPage=2

/API/Lightbox/v1.0/LightboxSearch?CurrentPage=3

Lightbox API Response

Additional information in the response

The following information is provided in addition to the results:

- API Request Information:

  <Module>Lightbox</Module>

  <APIVersion>v1.0</APIVersion>

  <Resource>LightboxSearch</Resource>

  <IsLoggedIn type="Boolean">True</IsLoggedIn>

  <Status>LoggedIn</Status>

  <UserLogin>chantal_admin</UserLogin>

  <TimeoutPeriodMinutes type="Numeric">60</TimeoutPeriodMinutes>

- API Request Interpretation (e.g. Lightbox.Identifier for LightboxListContent and LightboxListShares)
- API Response Summary

Output format

Every CRUD operation has the same XML schema as a result. Only specific operations (such as creating links in the “Documents.Link” data table) will have a different result format.

The output is in XML by default, but can be turned into JSON using the URL parameter "&format=JSON"

Example:

/API/v1.0/Lightbox/LightboxSearch?&format=JSON

Failure messages

Error codes common to all OrangeDAM REST APIs

Http return code	Message	Notes
401	Security violation	Your Security Profile does not give you access to the given API
403	This API cannot connect using a Secure Sockets Layer (SSL). Please change your URL from HTTP to HTTPS.	
503	The server is currently in a state that does not allow API calls... [Client specific message]	
500	The server encountered an unexpected condition which prevented it from fulfilling the request. [Client specific message]	This is the fallback error message/code for all unhandled errors

Errors codes specific to the Lightbox API

Http return code	Message	Notes

Contacts API

Introduction

Starting with the Helsinki release of OrangeDAM | DAM solution, you can change the Contact type for a contact record using 'ChangeContactType' in the Contacts v1.0 API.

Authentication

Authentication to use the Contacts API can use a cookie or a token requested through the Authentication API.

Security functions required

To use the API '/API/Contacts/v1.0/ChangeContactType', user needs the following security functions:

1. Mandatory - User needs 'Full' privileges

Category / Sub-category	**SubCategory**	**Name**
API > Contact	Contact	Contact API - Change contact type

2. User needs 'Edit' privileges for each type of Contacts he wants to use in the API call (change from and change to)

Category / Sub-category	**SubCategory**	**Name**
Contacts	Client accounts	Client accounts
Contacts	Company accounts	Company accounts
Contacts	Staff accounts	Staff accounts
Contacts	Source accounts (aka Photographer accounts)	Source accounts (aka Photographer accounts)

Contacts	Source agent accounts	Source agent accounts
Contacts	Agent accounts	Agent accounts
Contacts	Model accounts	Model accounts
Contacts	Billing accounts	Billing accounts

Use ‘ChangeContactType”

Example call

https://www.sitename.com/API/Contacts/V1.0/ChangeContactType?
Email=john.doe@orangelogic.com
&NewContactType=Staff

Search parameters

There are 4 parameters to retrieve the Contact to be updated. Use at least one of these parameters:

- RecordID
- Identifier
- Email
- LoginID

NewContactType

NewContactType is the new type of Contact to assign to the Contact record:

Response

Financial and Transactions APIs

Introduction

OrangeDAM Financial and Transactions APIs allow you to create Transactions (Invoice Header and associated Invoice Items), finalize transactions, create payments, pay invoices, credit invoices or invoice items, and post royalties.

Using these API calls, you can also import legacy invoices and payment records in your database. You can also add items to a user's cart, list items in a user's cart and request items in a user's cart.

The APIs referred to in this documentation are:

- DataTable/v2.1
- Financial/v1.0
- Order/v1.0

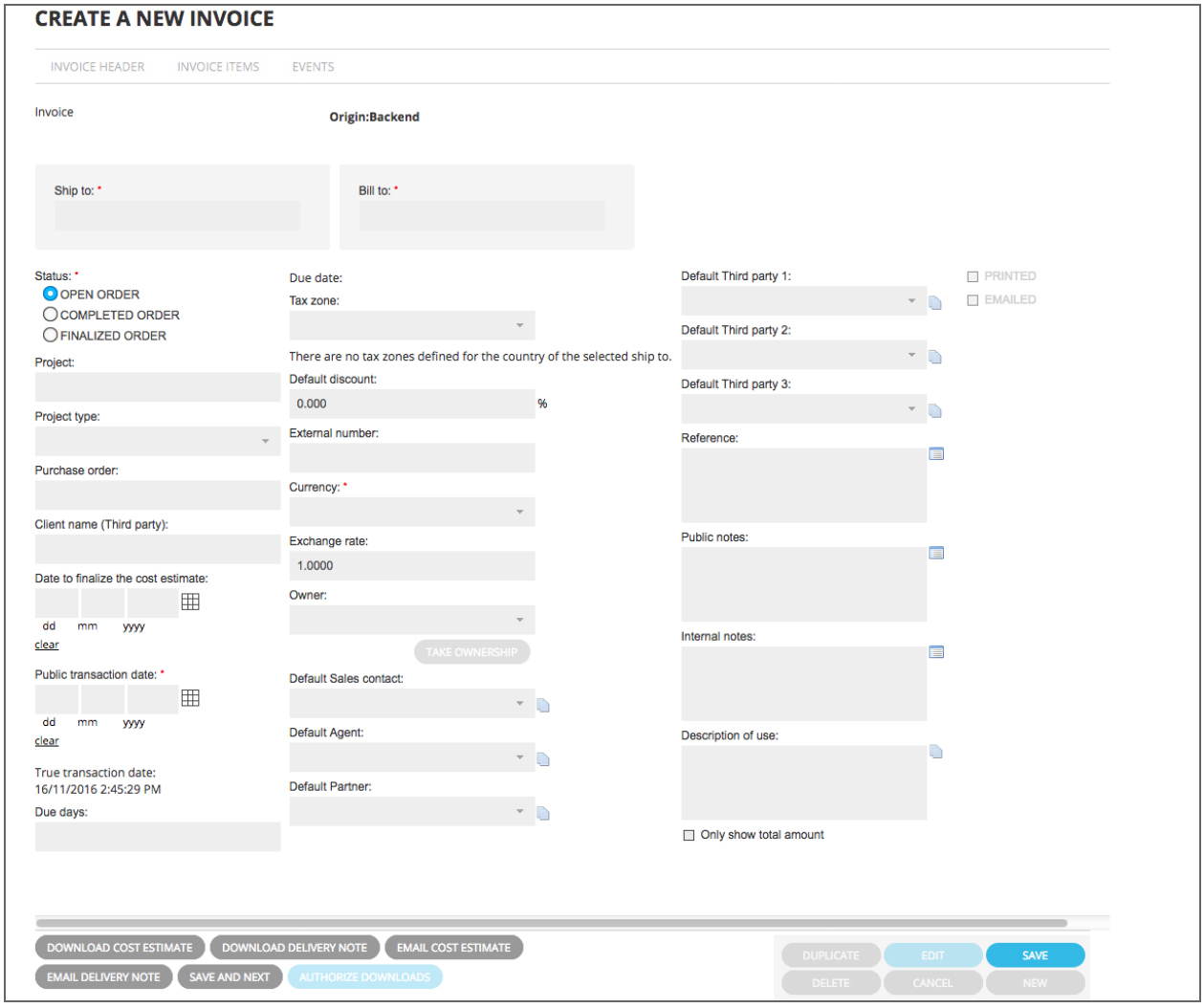
Authentication

The Data Table API is secured using a token that is provided by the Authentication API.

Manage Transactions

Create An Invoice Header

Security Point required: *Family API: Datatable API - Financial transactions*



List fields:

ListFields will return the list of fields available for the given Data Table (Financial.Invoice)

www.sitename.com/API/DataTable/v2.1/Financial.Invoice:ListFields

Sample call:

www.sitename.com/API/DataTable/v2.1/Financial.Invoice:Create?

Financial.CoreField.CT_BillTo:=[DataTable/v2.1/Contact.Client:Read?Contact.CoreField.Email1=*client@mycompany.com*]

&Financial.CoreField.CT_ShipTo:=[DataTable/v2.1/Contact.Client:Read?Contact.CoreField.Email1=*client@mycompany.com*]

&Financial.CoreField.ProjectTitle:=*My Project Title*

&Financial.CoreField.ProjectType:=*My Project Type*

&Financial.CoreField.PurchaseOrder:=*My Purchase Order*

&Financial.CoreField.UsageDate:=*2015-12-24*

&Financial.CoreField.InvoiceDate:=*2015-05-03*

&Financial.CoreField.DueDays:=*30*

&Financial.CoreField.CurrencyCode:=*EUR*

&Financial.CoreField.CT_Owner:=[DataTable/v2.1/Contact.Client:Read?Contact.CoreField.Email1=*owner@orangelogic.com*]
&Financial.CoreField.CT_Salesman:=[DataTable/v2.1/Contact.Client:Read?Contact.CoreField.Email1=*salesman@orangelogic.com*]
&Financial.CoreField.CT_Agent:=[DataTable/v2.1/Contact.Client:Read?Contact.CoreField.Email1=*agent@agentcompany.com*]
&Financial.CoreField.CT_Partner:=[DataTable/v2.1/Contact.Client:Read?Contact.CoreField.Email1=*partner@partnercompany.com*]
&Financial.CoreField.CT_ThirdParty1:=[DataTable/v2.1/Contact.Client:Read?Contact.CoreField.Email1=*other1@mycompany.com*]
&Financial.CoreField.CT_ThirdParty2:=[DataTable/v2.1/Contact.Client:Read?Contact.CoreField.Email1=*other2@mycompany.com*]
&Financial.CoreField.CT_ThirdParty3:=[DataTable/v2.1/Contact.Client:Read?Contact.CoreField.Email1=*other3@mycompany.com*]

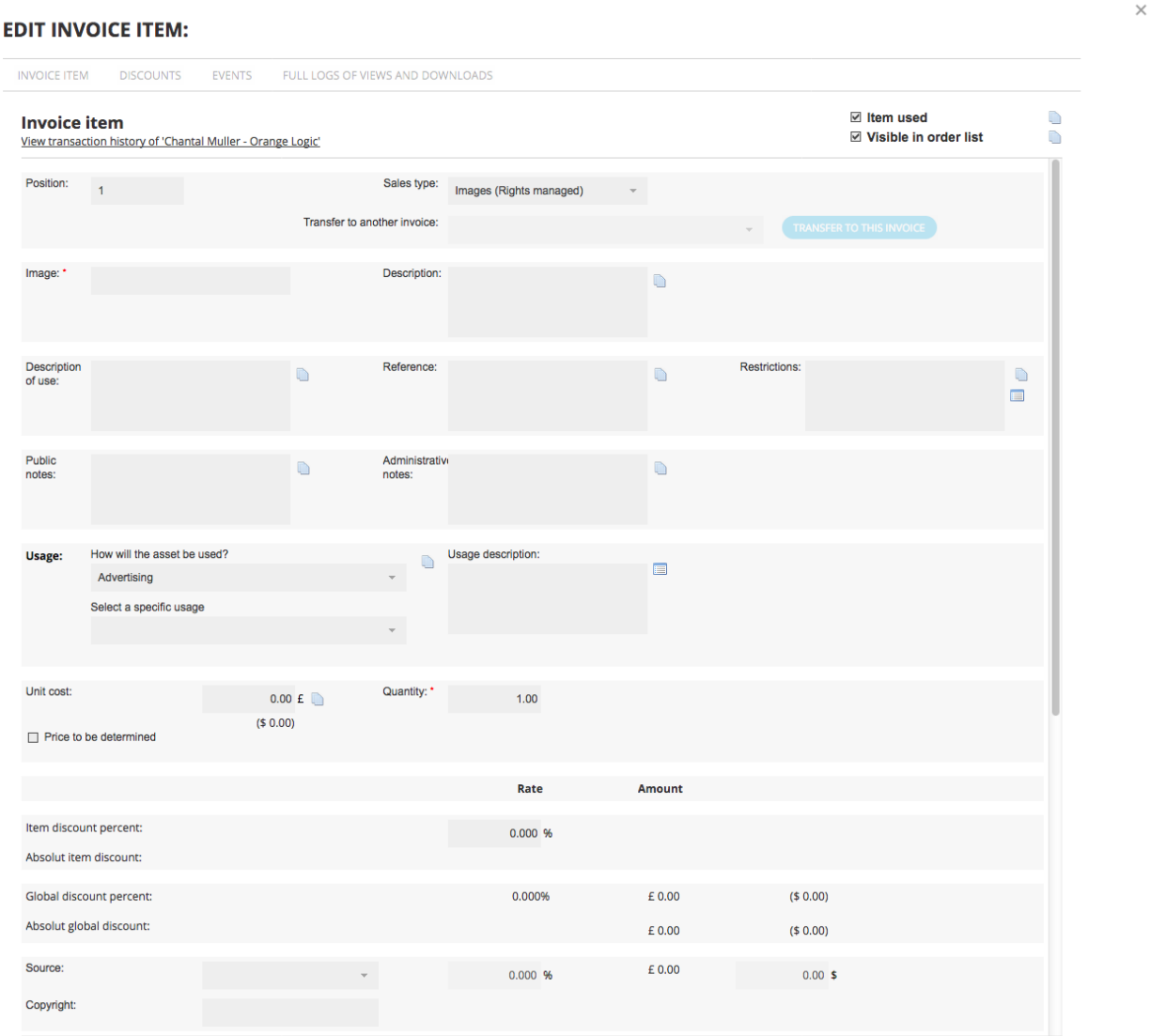
&Financial.CoreField.ClientReference:=*The Client Reference*
&Financial.CoreField.Reference:=*My Reference*

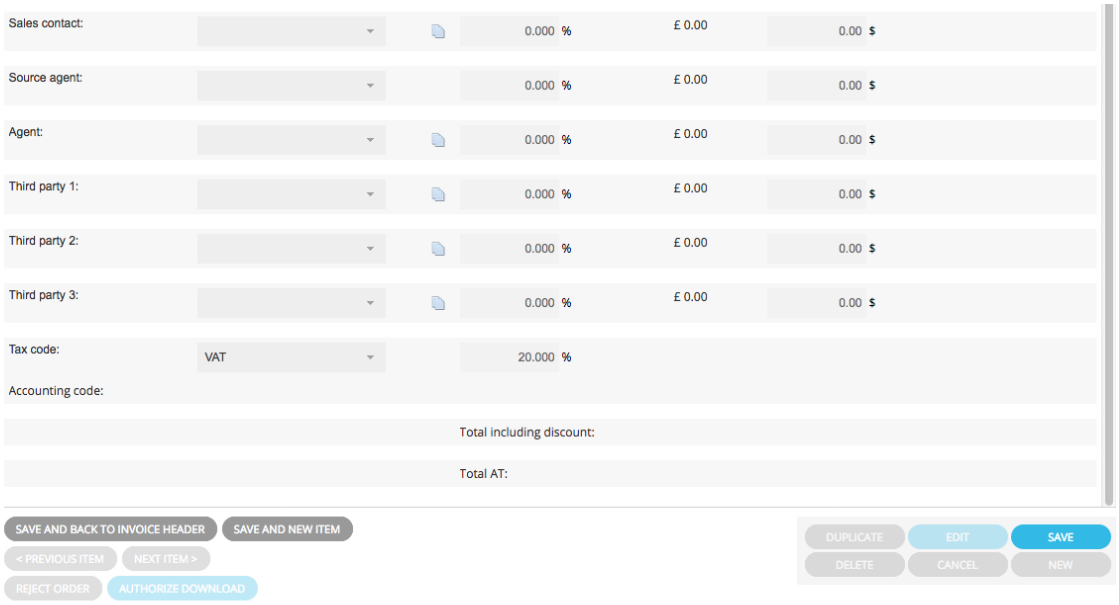
&Financial.CoreField.PublicNotes:=*My Public Notes*
&Financial.CoreField.InternalNotes:=*My Internal Notes*
&Financial.CoreField.UsageDescription:=*Description of Use*

> The calls in between the [brackets] are sub-calls to locate the Contacts in the Contact.Client datatable, based on the contacts email addresses

Create an invoice item

Security Point required: *Family API: Datatable API - Financial transaction items*





List fields:

ListFields will return the list of fields available for the given Data Table (Financial.InvoiceItem)

www.sitename.com/API/DataTable/v2.1/Financial.InvoiceItem:ListFields

Create an invoice item with a freetext price

www.sitename.com/API/DataTable/v2.1/Financial.InvoiceItem:Create?

Financial.CoreField.IH_Invoice:=[DataTable/v2.1/Financial.Invoice:Read?Financial.CoreField.Identifier=*INV1234*]

&Financial.CoreField.VisibleInOrderList:=1

&Financial.CoreField.IT_Code:=*ItemTypeImageRM*

&Financial.CoreField.DO_RecordID:=[DataTable/v2.1/Document.Asset.Image:Read?Document.CoreField.Identifier=*OL9876*]

&Financial.CoreField.TermsConditions:=1

&Financial.CoreField.AuthorizationRequired:=0

&Financial.CoreField.Description:=*My Description*

&Financial.CoreField.Reference:=*My Reference*

&Financial.CoreField.PublicNotes:=*My Public Notes*

&Financial.CoreField.InternalNotes:=*My Internal Notes*

&Financial.CoreField.Restrictions:=*My Restrictions*

&Financial.CoreField.UnitCostInitial_Local:=*250*

&Financial.CoreField.Quantity:=1

Notes

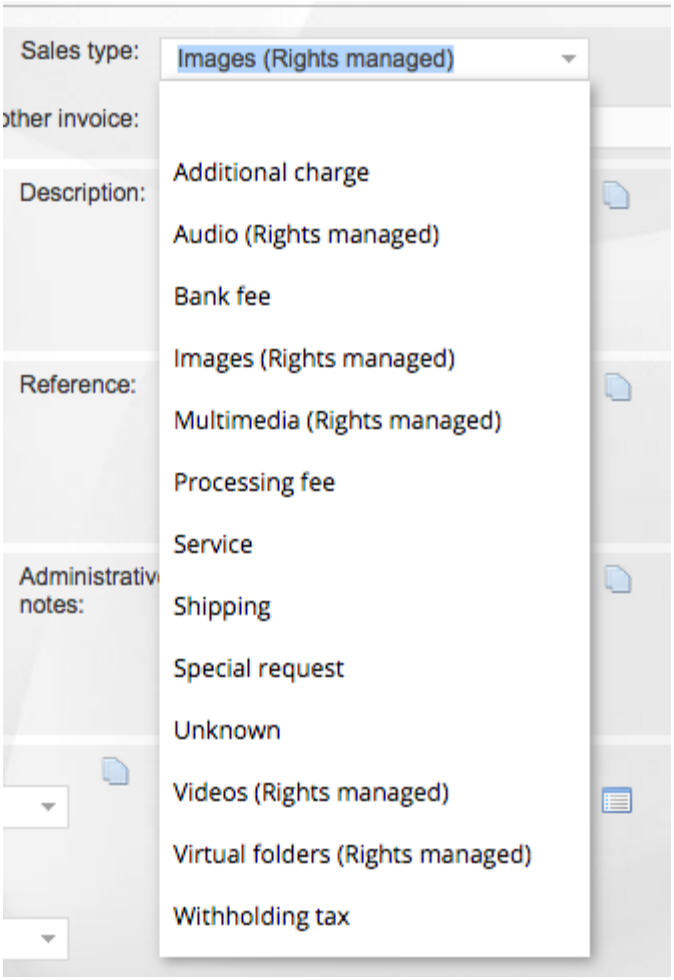
1. The calls in between the [brackets] are sub-calls to locate:
    - The Invoice header in the Financial.Invoice datatable
    - The image in the asset in Document.Asset.Image datatable

2. Financial.CoreField.IT_Code (Sales type code):
    ItemTypeImageRM is the sales type Code for sales type with the Display Name "*Images (Rights Managed)*" in the interface (see screenshot below)

    To retrieve the sales type code for other item types ("*Videos (Rights Managed)*", "*Shipping*", "*Service*", etc.), use the SalesType datatable.

    You can also use a subcall to get the sales type code from the Display Name:

    &Financial.CoreField.IT_Code:=[Financial/v1.0/SalesType.GetCode?DisplayName =*Images (Rights managed)*]



SalesType datatable

Security Point required: *Family API: Datatable API - Financial transaction items*

List fields

The following call will return the list of all available sales types

www.sitename.com/API/Financial/v1.0/SalesType.GetList

Response:
<Code>ItemTypeVideoRM</Code>
<DisplayName>Videos (Rights managed)</DisplayName>

<Code>ItemTypeShipping</Code>
<DisplayName>Shipping</DisplayName>

<Code>ItemTypeService</Code>
<DisplayName>Service</DisplayName>
etc.

Get the sales type code from the display name

www.sitename.com/API/Financial/v1.0/SalesType.GetCode?DisplayName=Images (Rights managed)

Response: ItemTypeImageR

Create an invoice item using a usage agreement

Security Point required: *Family API: Datatable API - Usage agreements*

Sample call:

www.sitename.com/API/DataTable/v2.1/Financial.InvoiceItem:Create?

Financial.CoreField.IH_Invoice:=[DataTable/v2.1/Financial.Invoice:Read?Financial.CoreField.Identifier=*INV1234*]

&Financial.CoreField.VisibleInOrderList:=1

&Financial.CoreField.IT_Code:=*ItemTypeImageRM*

&Financial.CoreField.DO_RecordID:=[DataTable/v2.1/Document.Asset.Image:Read?Document.CoreField.Identifier=*OL9876*]

&Financial.CoreField.PU_RecordID:=[DataTable/v2.1/Financial.UsageAgreement:Read?Financial.CoreField.Title=*Subscription 2015*]

&Financial.CoreField.CT_RecordID=[DataTable/v2.1/Contact.Client:Read?Contact.CoreField.Email1=*client@mycompany.com*]

Notes: The calls in between the [brackets] are sub-calls to locate:

- The Invoice header in the Financial.Invoice datatable
- The image in the asset in Document.Asset.Image datatable
- The User Usage Agreement in the Financial.UsageAgreement datatable
- The Contact in the Contact.Client datatable
- Financial.CoreField.IT_Code (Sales type code): see note in "Sample call to Create an Invoice item with a freetext price"

Create an invoice item using a calculator sequence

Security Point required: *Family API: Financial - Access to the calculator*

Sample call:

www.sitename.com/API/DataTable/v2.1/Financial.InvoiceItem:Create?

Financial.CoreField.IH_Invoice:=[DataTable/v2.1/Financial.Invoice:Read?Financial.CoreField.Identifier=*INV1234*]

&Financial.CoreField.VisibleInOrderList:=1

&Financial.CoreField.IT_Code:=*ItemTypeImageRM*

&Financial.CoreField.DO_RecordID:=[DataTable/v2.1/Document.Asset.Image:Read?Document.CoreField.Identifier=OL9876]

&Financial.CoreField.CalculatorSequenceCodes:=[Financial/v1.0/Calculator.ConvertAnswersToRecordIDs?AnswerList=*Answer1|Answer2*]

Notes:

1. The calls in between the [brackets] are sub-calls to locate:
   - The Invoice header in the DataTable API, Financial.Invoice resource
   - The image in the asset in the DataTable API, Document.Asset.Image resource
   - The calculator sequence using the Financial API, Calculator.ConvertAnswersToRecordIDs resource
2. To convert answer codes or labels into RecordIDs, see Calculator.

Finalize an invoice

Security Point required: *Family API: Financial - Can finalize invoices*

Sample call:
www.sitename.com/API/Financial/v1.0/Transaction.Invoice.Finalize?
InvoiceID=*INV1234*

Create a payment

Sample call:

www.sitename.com/API/DataTable/v2.1/Financial.Payment:Create?

Financial.CoreField.CT_BillTo:=[DataTable/v2.1/Contact.Client:Read?Contact.CoreField.Email1=*client@mycompany.com*]

&Financial.CoreField.CT_ShipTo:=[DataTable/v2.1/Contact.Client:Read?Contact.CoreField.Email1=*client@mycompany.com*]

&Financial.CoreField.Identifier:=*PAY1234*

&Financial.CoreField.InvoiceDate:=*2016-11-01*

&Financial.CoreField.PaymentType:=WireTransfert

&Financial.CoreField.CurrencyCode:=*EUR*

&Financial.CoreField.PaymentReceived_Local:=*250*

> Financial.CoreField.PaymentType: Can be Check,ACH or WireTransfert

Pay an invoice (full or partial payment)

Sample call:

www.sitename.com/API/Financial/v1.0/Transaction.Invoice.Pay?

InvoiceID=*INV1234*

&PaymentID=*PAY1234*

&Amount=*200*

> This API call is used to assign a payment (PaymentID) to an invoice

Pay an invoice in full

Sample call:

www.sitename.com/API/Financial/v1.0/Transaction.Invoice.PayInFull?

InvoiceID=*INV1234*

&PaymentType=*WireTransfert*

Pay a specific invoice item

Sample call:

www.sitename.com/API/Financial/v1.0/Transaction.Invoice.Pay?

InvoiceID=*INV1234*

&PaymentID=*PAY5678*

&Amount=*150*

&InvoiceItemID=*ORD765*

> This API call is used to assign a payment (PaymentID) to a specific invoice item

Credit an invoice (full or partial credit)

Security Point required: *Family API: Financial - Can credit invoices*

Sample call:

www.sitename.com/API/Financial/v1.0/Transaction.Invoice.Credit?

InvoiceID=*INV1234*

&Amount=*120*

&Date=*2015-01-01*

&PublicNotes=*Public notes for the Credit record*

&InternalNotes=*Internal notes for the Credit record*

> - InvoiceID is the invoice RecordID or Unique Identifier/System Identifier

- InvoiceID can be replaced by a sub call, as follows: InvoiceID=[Datatable/v2.1/Financial.Invoice:Read?Financial.CoreField.ExternalNumber=EXT-12345]
- Date = the date of the Credit record. This is a 'user date', can be set to any date, even in the past

Credit a specific invoice item (full or partial credit)

Security Point required: *Family API: Financial - Can credit invoices*

Sample call:

www.sitename.com/API/Financial/v1.0/Transaction.Invoice.Credit?

InvoiceID=*INV234*

&InvoiceItemID=*ORD765*

&Amount=*120*

&Date=*2015-01-01*

&PublicNotes=*Public notes for the Credit record*

&InternalNotes=*Internal notes for the Credit record*

Notes:

- InvoiceID is the invoice RecordID or Unique Identifier/System Identifier
- InvoiceItemID is the invoice item RecordID or Unique Identifier/System Identifier of the invoice item
- InvoiceID and InvoiceItemID can be replaced by sub calls, as follows: InvoiceID=[Datatable/v2.1/Financial.Invoice:Read?Financial.CoreField.ExternalNumber=EXT-12345] InvoiceItemID=[Datatable/v2.1/Financial.InvoiceItem:Read?Financial.CoreField.ExternalID=IT-EXT-98]
- Date = the date of the Credit record. This is a 'user date', can be set to any date, even in the past

Print a financial transaction

Security Point required: *Family API: Financial - Can print transaction*

Sample call:
www.sitename.com/API/Financial/v1.0/Transaction.Print?TransactionID=*INV1234*

Manage Royalty Distributions

Security Point required: Family API: *Financial - Can manage royalties* (Can post and finalize royalties through API calls)

Post royalty distribution

Sample call:

www.sitename.com/API/Financial/v1.0/Transaction.Royalty.Post?
ContactID=*CT112233*
&Date=*2015-04-04*
&ItemIDList=ID1234|ID5678

Notes:

- This call will post payment allocations into a royalty distribution header (a payment from royalty distribution).
- It is equivalent to posting items in 'Financial > Post royalty distributions'
- ContactID is the Contact Record ID.
  This can be replaced by a sub call as follows:
  [Datatable/v2.1/Contact.Client:Read?Contact.CoreField.Email1=*johndoe@gmail.com*]
- ItemIDList is the list of items to post for this contact, separated by pipes
  If not provided, the API will post all pending transactions.
  ItemIDLIst can be replaced by a sub call as follows:
  [Datatable/v2.1/Financial.InvoiceItem:Read?Financial.CoreField.AgentCommissionExternalID=1234]
- Date = The date of the royalty distribution posting. This is a 'user date', can be set to any date, even in the past

Finalize a royalty distribution

Sample call:

www.sitename.com/API/Financial/v1.0/Transaction.Royalty.Finalize?
RoyaltyID=*DM2PAY40*

Notes:

- This call will close the royalty distribution
- RoyaltyID is the RecordID of the royalty distribution to close.

This can be replaced by a sub call as follows:
[Datatable/v2.1/Financial.Payment:Read?Financial.CoreField.Identifier=ROY12345].

Refund a royalty distribution

Sample call:

www.sitename.com/API/Financial/v1.0/Transaction.Royalty.Refund?

RoyaltyID=[ROYALTY ID]
&PaymentType=Check

Notes:

- This call will refund the royalty distribution (after payment to the third party has been made)
- RoyaltyID is the RecordID of the royalty distribution to refund.
  This can be replaced by a sub call as follows:
  [Datatable/v2.1/Financial.Payment:Read?Financial.CoreField.Identifier=ROY12345]..

Retrieve Calculator Answers

Security Point required: Family API: Financial - Access to the calculator

Get answers from a given question

Sample call:

www.sitename.com/API/Financial/v1.0/Calculator.GetAnswers?Question=Q1

Get questions from a given answer

Sample call:

www.sitename.com/API/Financial/v1.0/Calculator.GetQuestions?Answer=A1

Convert answer codes or labels into RecordID (To be used in Invoice item API)

Sample call:

www.sitename.com/API/Financial/v1.0/Calculator.ConvertAnswersToRecordIDs?AnswerList=*Answer1|Answer2*

Manage User's Cart

Add items to a user's cart

Sample call:

www.sitename.com/API/Order/v1.0/Cart.AddItem?
Contact=[DataTable/v2.1/Contact.All:Read?Contact.CoreField.Email1=*client@mycompany.com*]
&Asset=*OL1234*
&Calculator=Answer1_RecordID|Answer2_RecordID

- &Calculator: answers are separated by pipes
- Refer to the Calculator section for retrieving the RecordIDs of the Calculator Answers.

List items in a user's cart

Sample call:

www.sitename.com/API/Order/v1.0/Cart.List?
Contact=[DataTable/v2.1/Contact.All:Read?Contact.CoreField.Email1=*client@mycompany.com*]

Request items that are in a user's cart

Sample call:

www.sitename.com/API/Order/v1.0/RequestItems?Items=*ORD123|ORD456*

- Items are separated by pipes
- Items can be listed using the RecordID, or the Order.CoreField. Identifier that can be obtained from a call to List items in a user's cart.

Import Legacy Invoices And Payment Records

You can import legacy invoices and payment records in your database.

> **Do not switch your live/primary site to “import” state***
> - Switch your site to 'Import' state (Administration > Maintenance > fallback procedure administration)
> - Add '&IsLegacyData=1' in the API calls.

In ‘Import’ mode, some fields that are normally locked (such as the invoice finalized date) can be updated.
These fields are highlighted in yellow in the examples below.

Values between brackets in the examples below can be pulled based on the column headers of your CSV spreadsheet.
See documentation on Ingesting/Managing data with Cortex iAPI v1.4

Create an invoice header

www.sitename.com/API/DataTable/v2.1/Financial.Invoice:CreateOrUpdate?
&Financial.CoreField.Identifier=[Invoice Number]
&Financial.CoreField.Identifier:=[Invoice Number]
&Financial.CoreField.CT_BillTo:=[DataTable/v2.1/Contact.Client:Read?Contact.CoreField.Email1=[Contact Email]]
&Financial.CoreField.CT_ShipTo:=[DataTable/v2.1/Contact.Client:Read?Contact.CoreField.Email1=[Contact Email]]
&Financial.CoreField.ProjectTitle:=[Opportunity]
&Financial.CoreField.InvoiceDate:=[Invoice Date]
&Financial.CoreField.CurrencyCode:=[Invoice Currency]
&IsLegacyData=1

Create an invoice item

www.sitename.comAPI/DataTable/v2.1/Financial.InvoiceItem:CreateOrUpdate?
&Financial.CoreField.Identifier=[Invoice Line Item ID]
&Financial.CoreField.Identifier:=[Invoice Line Item ID]

&Financial.CoreField.IH_Invoice:=[DataTable/v2.1/Financial.Invoice:Read?Financial.CoreField.Identifier=[Invoice Number]]
&Financial.CoreField.IT_Code:=ItemTypeImageRM
&Financial.CoreField.DO_RecordID:=[DataTable/v2.1/Document.Asset.Image:Read?Document.CoreField.Identifier=[Asset #]]
&Financial.CoreField.TermsConditions:=1
&Financial.CoreField.AuthorizationRequired:=0
&Financial.CoreField.UnitCostInitial_Local:=[Unit Price]
&Financial.CoreField.Quantity:=1
&IsLegacyData=1

> If needed to assign the source (photographer, Artist) to an invoice item. You can use the following API call:
> &Financial.CoreField.CT_Source:=[DataTable/v2.1/Contact.Source:Read?Contact.CoreField.FastID=[Source FastI D]]

Finalize an invoice

www.sitename.com/API/Financial/v1.0/Transaction.Invoice.Finalize?
InvoiceID=[Invoice Number]
&ForcedIdentifier=[Invoice Number]
&DateFinalized=[Invoice Date]
&IsLegacyData=1

Create a payment

www.sitename.com/API/DataTable/v2.1/Financial.Payment:Create?
Financial.CoreField.CT_BillTo:=[DataTable/v2.1/Contact.Client:Read?Contact.CoreField.Email1=[Contact Email]]
&Financial.CoreField.CT_ShipTo:=[DataTable/v2.1/Contact.Client:Read?Contact.CoreField.Email1=[Contact Email]]
&Financial.CoreField.Identifier:=[Payment Number]
&Financial.CoreField.InvoiceDate:=[Payment Date]
&Financial.CoreField.LockDate:=[Payment Date]
&Financial.CoreField.PaymentType:=WireTransfert
&Financial.CoreField.CurrencyCode:=[Invoice Currency]
&Financial.CoreField.PaymentReceived_Local:=[Net Value]
&IsLegacyData=1

Pay an invoice

www.sitename.com/API/Financial/v1.0/Transaction.Invoice.Pay?
InvoiceID=[Invoice Number]
&PaymentID=[Payment Number]
&Amount=[Payment Amount]
&DateFinalized:=[Invoice payment date]
&IsLegacyData=1

Pay an invoice in full

www.sitename.com/API/Financial/v1.0/Transaction.Invoice.PayInFull?
InvoiceID=[Invoice Number]
&PaymentType=WireTransfert
&DateFinalized=[Invoice payment date]
&IsLegacyData=1

Credit an invoice (full or partial credit)

www.sitename.com/API/Financial/v1.0/Transaction.Invoice.Credit?
InvoiceID=[INVOICE ID]
&Amount=[Credit Amount]
&Date=[Date of Credit record - this is a 'user date', can be set to any date, even in the past]
&PublicNotes=*Public notes for the Credit record*
&InternalNotes=*Internal notes for the Credit record*
&DateFinalized=[Invoice payment date]
&IsLegacyData=1

Credit a specific invoice item (full or partial credit)

www.sitename.com/API/Financial/v1.0/Transaction.Invoice.Credit?
InvoiceID=[INVOICE ID]
&InvoiceItemID=[ITEM ID]
&Amount=[Credit Amount]
&Date=[Date of Credit record - this is a 'user date', can be set to any date, even in the past]
&PublicNotes=*Public notes for the Credit record*
&InternalNotes=*Internal notes for the Credit record*

&DateFinalized=[Invoice payment date]
&IsLegacyData=1

Add To Cart and Download API (in Honoring Usage Agreement)

The operator **AddToCartAndDownload API** allows you to add an asset to the cart and download the asset immediately, without going through the check-out process in OrangeDAM. The download session made from this API call will be subtracted directly from your usage agreement (or subscription).

With this API, you can allow users to download assets via API. They will not have to go to OrangeDAM to download the assets but they will need to use the Authentication API to retrieve their access token. As an Administrator, you can track the number of downloads made from their usage agreement.

AddToCartAndDownload API Syntax
https://sitename.com**/API/v2.0/ORDERS/AddToCartAndDownload?documentid=[Media Encrypted Identifier]**

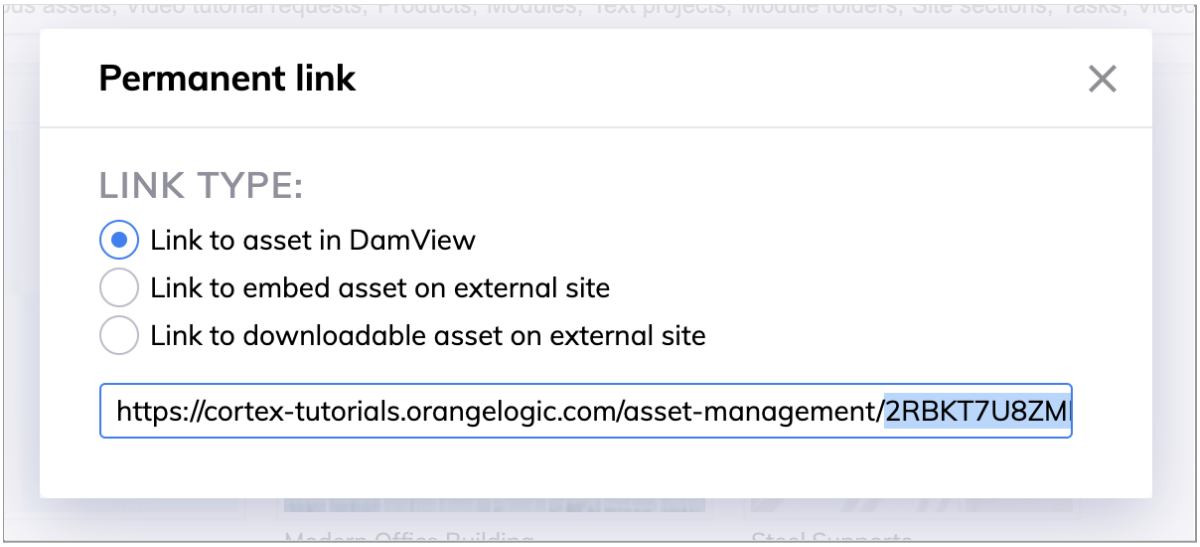
Parameter

DocumentID is the Media Encrypted Identifier (or Encrypted Record ID) of the asset that you want to download. There are two ways to retrieve the Media Encrypted Identifier:

- Option 1: Use the Get Link Feature
- Option 2: Use the Search API

Option 1: Use the **Get Link** Feature

1. Right-click the asset.
2. Select **Get Link**.
3. Copy the Document ID, which is the series of letters and numbers after “asset-management/”.



Option 2: Use the Search API

You can use the Search API to search for the Media Encrypted Identifier. The syntax is:
`https://www.sitename.com/API/search/v3.0/search?query=SystemIdentifier:[Unique Identifier]&fields=MediaEncryptedIdentifier`

> **Example**
>
> As a sales representative at Acme Assets, you receive a request from your client, David Rosales, to retrieve the image “Modern Office Building”. David wants the image from OrangeDAM using API, so that he can embed it in his company’s editorial system. To fulfill this request, you need to provide a tool for David to download the asset using API. The number of downloads made by David using API needs to be subtracted from his Usage Agreement.
>
> To do this, David first retrieves the Document ID of this image using Get Link or Search API. The Document ID of this image is 2RBKT7U8ZMK.
>
> Then he uses the **AddToCartAndDownload** API to download the image directly using API.
>
> **Syntax**
>
> https://acmeassets.com/API/v2.0/ORDERS/AddToCartAndDownload?documentid=2RBKT7U8ZMK

> **Result**
>
> The image will be downloaded directly from OrangeDAM and this download session subtracts one item from David's usage agreement. If David wants to download this image again in the future, another item will be subtracted from his usage agreement.

Summary Of Related Security Functions

Security Function (Family: API)	**Required for:**
Datatable API - Contact (Read only)	Retrieving information about a contact, without knowing the contact type (i.e. Client, Staff, Source, etc.).
Datatable API - Document (Read only)	Retrieving information about a document, without knowing the document type (i.e. Image, Video, Folder, etc.). WARNING: This security point gives access to all documents as it does not take into account Permissions/Restrictions/Embargo dates etc.
Datatable API - Order item	Working with order items using the Datatable API and the Cart APIs.
Datatable API - Financial transactions	Working with invoice headers
Datatable API - Financial transaction items	Working with invoice item and accessing the Sales Type list
Datatable API - Usage agreements	Working with invoice items associated with a usage agreement
Financial - Access to the calculator	Retrieving calculator answers, and working with invoice items associated with a calculator sequence
Financial - Can finalize invoices	Finalizing invoices
Financial - Can credit invoices	Crediting invoices or specific invoice items

Financial - Can manage royalties	Managing royalties

Useful Datatable Api Resources

Contact API - “Contact.All” Resource

Security Point required: *Family API: Datatable API - Contact (Read only)*

www.sitename.com/API/DataTable/v2.1/Contact.All:Read?Contact.CoreField.Email1=[EMAIL_ADDRESS]

The Contact.All resource is 'read only'. It is useful to retrieve information about a contact, without knowing the contact type (i.e. Client, Staff, Source, etc.).

If you know the contact type, you can use the dedicated DataTable.

Document API - “Document.All” Resource

Security Point required: *Family API: Datatable API - Document (Read only)*
WARNING: This security point gives access to all documents as it does not take into account Permissions/Restrictions/Embargo dates etc

www.sitename.com/API/DataTable/v2.1/Document.All:Read?Document.CoreField.Identifier=[ID 12345]

The Document.All resource is 'read only'. It is useful to retrieve information about a document, without knowing the document type (i.e. Image, Video, Folder, etc.).

If you know the document type, you can use the dedicated DataTable.

Get Keywords API

Authentication

Authentication to use the Search API can use a cookie or a token requested through the Authentication API.

Accessing the API

Get Keywords API is a dedicated API for retrieving keywords for an asset.

It is accessible through the page https://www.sitename.com/htm/GetKeywordsAPI.aspx

where www.sitename.com is your website address.

To access the Search API using your account privileges, add the Login API token to your request URL.

Example:

https://www.sitename.com/API/search/v3.0/search?query=MediaType:Image&fields=MediaNumber&token=gtvprt45l1mpm5454l1gqqji

> The security point "Can extract document keywords" in family API, subfamily "Keywords" must be given at least "Read" basis for the API user to be allowed to use the Get Keywords API.

Retrieving keywords using the Get Keywords API

This a 2-step process:

1. Obtain the encrypted ID for the image using the Search API

https://www.sitename.com/API/search/v3.0/search?query=SystemIdentifier:ABC12345&fields=MediaEncryptedIdentifier

API response: 2K7O3R3GDZTR

See documentation on Search API.

2. Get the list of keywords for this asset using the GetKeywordsAPI

https://www.sitename.com/htm/GetKeywordsAPI.aspx?DocID=2K7O3R3GDZTR

The response is an XML listing **keywords** and the keyword type for each keyword:

```
<Root>
<Keyword>
<Text>Group of people</Text>
<KeyType>Common</KeyType>
</Keyword>
<Keyword>
<Text>Motorbike</Text>
<KeyType>Common</KeyType>
</Keyword>
<Keyword>
<Text>Paris</Text>
<KeyType>Location</KeyType>
</Keyword>
<Keyword>
<Text>Validate by Art Dept.</Text>
<KeyType>Workflow</KeyType>
</Root>
```

Optional parameter: IncludeInheritedKeywords

Add &IncludeInheritedKeywords=1 (or any true string (e.g. True, Yes), to return the list of all keywords directly applied to a given document ('native keywords'), as well as all keywords applicable to the document through inheritance.

Example:
https://sitename.orangelogic.com/htm/GetKeywordsAPI.aspx?DocID=2UVRYDM1H4CO&IncludeInheritedKeywords=1

Virtual Folder Extractor API (Package Extractor API)

Authentication

The Package Extractor API is secured using a token that is provided by the Authentication API.

Note: The Package Extractor API also works without a login, in which case it only returns media available to the general public (based on based on access rights set by the database administrator set by the database administrator).

Package Extractor API

The **Package Extractor** API allows you to query the content of a virtual folder (package), including:

- All assets contained in a virtual folder, with corresponding metadata (Title, Caption, Photographer, etc.) and the URL path to the requested asset format.
- The Representative Image of the virtual folder or subfolders contained in the virtual folder.

To use the Package Extractor API to query OrangeDAM assets, use the Search API to retrieve the Media Encrypted Identifier of the virtual folder first, and then use that identifier with the Package Extractor API to extract data.

GET and POST requests

The maximum URL length for a GET request is 260 characters.

The Package Extractor API can also be accessed using POST requests: the parameters can be passed in the HTTP header or in the URL. If the same parameter is present in both, the value in HTTP header will override the URL.

Package Extractor API Parameters

Use Search API to Get the Media Encrypted Identifier

The **Search API** allows you to search for assets and query their system information using different criteria, such as System Identifier, keywords, and media type. In this scenario, you need to use Search API to retrieve a virtual folder's Media Encrypted Identifier.

Search API Syntax - MediaEncryptedIdentifier

https://sitename.com/**API/search/v3.0/search?query**=[VirtualFolderIdentifier]&Fields=MediaEncryptedIdentifier

Parameters

- **sitename.com** is your website address.
- **VirtualFolderIdentifier** is the System ID or Unique Identifier of the virtual folder that you want to retrieve assets from.

> **Example**
>
> You want to query information about assets in a virtual folder whose Unique Identifier is CTXALB27. In order to do so, you need to retrieve the virtual folder's Media Encrypted Identifier from OrangeDAM-acmeassets.com.
>
> **Syntax**
>
> https://OrangeDAM-acmeassets.com/**API/search/v3.0/search?query=CTXALB27&Fields=MediaEncryptedIdentifier**
>
> **Response**
>
> <MediaEncryptedIdentifier>**2RBV7CU_E**</MediaEncryptedIdentifier>

Use Package Extractor API to Retrieve Information

The **Package Extractor API** allows you to query attributes related to the content of a virtual folder (package) using Media Encrypted Identifier.

Note: The Package Extractor API also works without a login, in which case it only returns assets that are accessible to the *Unregistered contact group.

Package Extractor API Syntax

http://sitename.com/**API/PackageExtractor/v1.0/Extract?Package**=[MediaEncryptedIdentifier]
&PackageFields=[DataPointsNeeded]
&RepresentativeFields=[DataPointsNeeded]
&ContentFields=[DataPointsNeeded]

Parameters:

- **sitename.com** is your website address.
- **MediaEncryptedIdentifier** is automatically generated by OrangeDAM and can be retrieved using Search API.
- **PackageFields** is the list of metadata fields related to the virtual folder itself (Title, Caption, Date, etc).
- **RepresentativeFields** is the list of metadata fields related to the Representative Image of the virtual folder. For example, the image's title, and image's identifier.
- **ContentFields** is the list of metadata fields related to the assets in the virtual folder. For example, the asset's system identifier, format, URL, width, and height.

Note:

- Field name values used in an API call must be exact as system field names. For example, the system field name of the Identifier field is SystemIdentifier. To see all the system field names available for a specific asset type, see List API.
- Multiple fields must be separated by commas (,).

Example

You want to query information about assets in a virtual folder whose Unique Identifier is CTXALB27. You already retrieved the virtual folder's Media Encrypted Identifier from OrangeDAM-acmeassets.com. Now, you want to retrieve:

- System Identifier and Title of the virtual folder (Package fields).
- System Identifier and Title of the representative image of this virtual folder.

- System Identifier, Title, Artist, Media Type, Path_TR1, and Path_WebHigh of all contents in this virtual folder.

Syntax

http://OrangeDAM-acmeassets.com/API/PackageExtractor/v1.0/Extract?Package=2RBV7CU_E
&PackageFields=SystemIdentifier,Title
&RepresentativeFields=SystemIdentifier,Title
&ContentFields=SystemIdentifier,Title,Artist,MediaType,Path_TR1,Path_WebHigh

Response

- Information about the virtual folder (PackageFields):

  <SystemIdentifier>**CTXALB27**</SystemIdentifier>
  <Title>**Homepage slideshow**</Title>

- Information about the virtual folder's representative (RepresentativeFields): None, as the virtual folder does not have a representative image

- Information about the contents of the virtual folder (ContentFields):

  <SystemIdentifier>**CTL8351**</SystemIdentifier>
  <Title>**Stone-floor Conference Room**</Title>
  <MediaType>**Image**</MediaType>
  <URI>**https://OrangeDAM-acmeassets.com/Assets/V2/Y3vwbBLZ3WROfciZLPOxB.jpg**</URI>
  <Width type="Numeric">**1200**</Width>
  <Height type="Numeric">**801**</Height>

  <SystemIdentifier>**CTL4626**</SystemIdentifier>
  <Title>**Danish National Aquarium**</Title>
  <Artist>**Sandro Katalina**</Artist>
  <MediaType>**Image**</MediaType>
  <URI>**https://OrangeDAM-acmeassets.com/Assets/V2/Y3vwbBLZ3WROfciZL.jpg**</URI>
  <Width type="Numeric">**1200**</Width>
  <Height type="Numeric">**801**</Height>

Note:

- The asset formats and dimensions used on your site (such as TR1, TR2, TR3, TR6, Crop etc.) are different for each installation and cannot be listed here. To see the list of all media formats used on your site, go to **Administration** > **Archive** > **Assets** > **Document Formats**.

> **Example**
>
> Image formats: TRX (original), TR1 (medium res), etc.
> Video formats: TRX (original), Web High (720p), Web Low, Proxy.
> Audio formats: TRX (original), Web (web streaming, mp3, 128kbps).

- The asset formats and dimensions you have access to are driven by your account privileges and related Security Group. If you cannot retrieve the format you want, please contact your system administrator for further assistance.
- If the Media exists and can be generated in the requested format, then its URL, Width, and Height will automatically be included in the response. The URL is the public URL that allows unregistered users to see the asset.

Change Log

2019-12-17

- Added: New API to allow users to add to cart and download assets in honoring the Usage Agreement (AddToCartAndDownload API).

2019-11-27

- Changed: Virtual Folder Extractor API.

2019-06-11

- Added: example to set the parent Company Account for a Staff Account (using ReturnField parameter)

2019-06-11

- Added: Specify date and time formats with "DateFormat" parameter.
- Removed: "GetRestrictions" parameter removed.

2019-01-21

- Added: Speed up DataTable API calls by delaying asset indexing with the "IndexInBackground" parameter.

2020-07-09

- Added: "Remove Relationships from Document to Keyword" and "Remove Relationships from Keyword to Keyword"

2020-07-30

- Added: "Create Relationship Between Contact and Keyword"
- Added: "Removing Assets from Virtual Folders"

2020-08-12

- Added: "Assign Representative Image"

2020-08-17

- Added: "Specify the KeyType to Search for or Create Keywords"

2020-09-08

- Changed: "Changing the Sort Order"

2022-03-21

- Added: “Generate Formats On Demand”
- Added: “Generate or Regenerate Proxies for Assets ”
- Changed: “Extract Original Document API”

2022-03-21

- Changed: “Extract Original Document API”

2022-04-07

- Changed: Updated terminology.