

Version 10.2
Connector Guide

2024-03-14

Index

1	Integration of digital assets using GET requests	1
1.1	Images	2
1.2	Videos	4
1.3	Documents	4
2	OpenSearch	5
2.1	Introduction	5
2.2	OpenSearch Description Document	5
2.2.1	The "Url" element	5
2.2.2	Fetching the OpenSearch Description Document	6
2.3	Accessing the service	6
2.4	Different options to use the search-service	7
2.4.1	Object based search	7
2.4.2	Container based search	7
2.4.3	Base search	7
2.5	Pagination	9
2.6	OpenSearch Response	10
3	RESTful OpenAPI	11
3.1	Introduction	11
3.2	REST	11
3.3	Preparations	12
3.4	Authentication	13
3.4.1	authenticate (POST)	13
3.4.2	close (GET)	14
3.5	GET Functions	14
3.5.1	download	14
3.5.2	getobjectlist	15
3.5.3	getprojectstructure	19
3.5.4	getuserinformation	19
3.6	POST Functions	23
3.6.1	upload	23
3.6.2	createaccesslink	24
3.6.3	createobjectaccesslink	25
3.6.4	createwrapperlink	25
3.6.5	createdownloadlink	26
3.6.6	createmultiaccesslink	26
3.6.7	createmultidownloadlink	26
3.6.8	createpublication	27
3.6.9	createuser	27
3.6.10	creategroup	28
3.6.11	createfolder	28
3.6.12	createobject	28
3.6.13	sendmessage	29
3.6.14	createproject	29
3.6.15	createtask	29
3.7	PUT Functions	31
3.7.1	editpublication	31
3.7.2	editpublicationsetting	32
3.7.3	edituser	33
3.7.4	editgroup	33
3.7.5	renamefolder	35
3.7.6	editmediaobject	35

3.7.7	renameobject	36
3.7.8	cutobject	36
3.7.9	copyobject	37
3.7.10	copyconnectedobject	37
3.7.11	pasteobject	37
3.7.12	lockobject	38
3.7.13	unlockobject	38
3.7.14	publish	38
3.7.15	unpublish	39
3.7.16	savecontent	39
3.7.17	acceptworkflowobject	40
3.7.18	rejectworkflowobject	40
3.7.19	editproject	41
3.7.20	edittask	41
3.8	DELETE Functions	42
3.8.1	deletepublication	42
3.8.2	deleteuser	42
3.8.3	deletegroup	42
3.8.4	deletefolder	43
3.8.5	deleteobject	43
3.8.6	deleteproject	44
3.8.7	deletetask	44
4	SOAP OpenAPI	45
4.1	Introduction	45
4.2	SOAP	45
4.3	WSDL	45
4.4	(re)generate the WSDL	45
4.5	Types	45
4.5.1	download_object	45
4.5.2	Group	46
4.5.3	Grouplist	46
4.5.4	Permission	46
4.5.5	Permissionlist	47
4.5.6	Folderlist	47
4.5.7	Textcontent	47
4.5.8	Fieldlist	47
4.6	Functions	48
4.6.1	authenticate	48
4.6.2	close	49
4.6.3	download	49
4.6.4	upload	49
4.6.5	user_create	50
4.6.6	user_edit	50
4.6.7	user_delete	51
4.6.8	group_create	51
4.6.9	group_edit	51
4.6.10	group_delete	52
4.6.11	publish / unpublish	52
4.6.12	save_content	53
4.6.13	folder_create	54
4.6.14	folder_rename	54
4.6.15	folder_delete	54
4.6.16	object_create	54
4.6.17	object_rename	55
4.6.18	object_delete	55
4.6.19	project_create	55

4.6.20	project_edit	56
4.6.21	project_delete	56
4.6.22	task_create	56
4.6.23	task _edit	57
4.6.24	task _delete	57
4.6.25	workflow_accept	58
4.6.26	workflow_reject	58
5	Authentication	59
5.1	Configuration of OpenLDAP and Apache	60
5.2	Configuration of the LDAP / MS Active Directory Connector	61
5.2.1	Requirements	61
5.2.2	Configuration	61
5.3	Implement Single Sign-On (SSO) using Microsoft Active Directory	65
5.3.1	SSO with the OAuth remote client	65
5.3.2	SSO with the Apache module	66
5.4	Workplace Integration and LDAP/AD	67
6	Assetbrowser	68
6.1	Introduction	68
6.2	Configuration	68
6.3	Open the Assetbrowser	69
7	YouTube Connector	70
8	Adobe Creative Cloud Connector	80
8.1	Enable the LinkrUI Connector	80
8.2	Download and install the Connector	80
8.3	Using the Connector	81
8.3.1	Quick Start Guide	81
8.3.2	Add assets to your Adobe Content	81
8.3.3	Upload your Adobe files	82
8.3.4	Change the view	82
8.3.5	Search and filter Assets inside the Connector	82
8.3.6	View Asset Metadata in the Connector	82
8.4	Other features of the LinkrUI Connector	83
9	Microsoft Office Connector	85
9.1	About the integration	85
9.2	Enable the Microsoft Office Connector	85
9.3	Install the Microsoft Office Connector	85
9.4	How to use the Microsoft Office Connector?	86
10	Legal reference / flag	87
10.1	Questions and suggestions	87
10.2	Imprint	87
10.3	Legal information	87

1 Integration of digital assets using GET requests

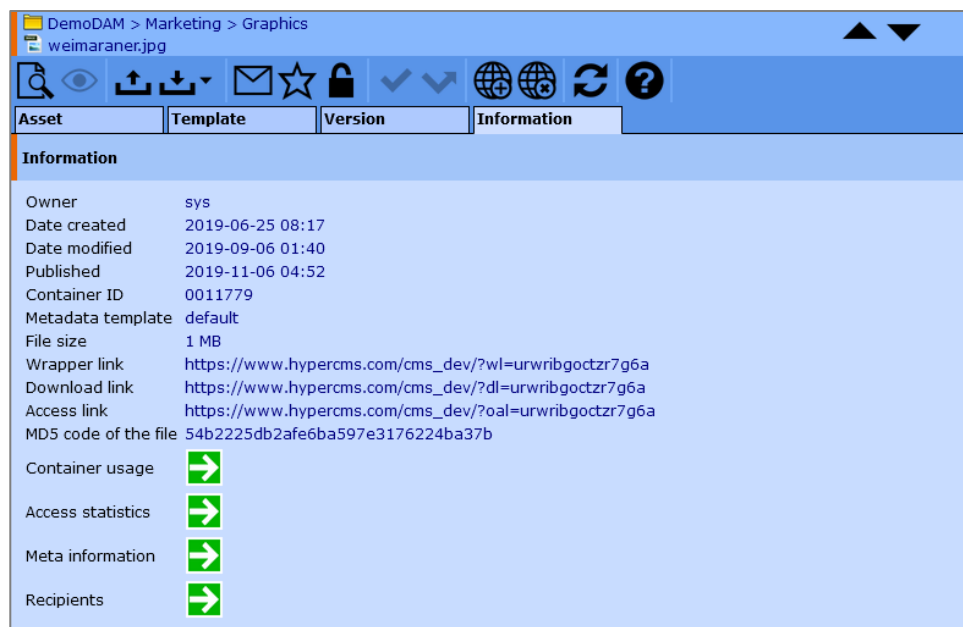
Besides the code snippets for the more advanced integration of assets in websites or web application, a simple GET request can be used to access digital assets stored in the system.

This approach is mainly interesting for developers that want to use images, documents or videos in their application without the use of a code snippet, or the want to provide their own snippet and refer to the digital assets.

The system provides wrapper links for this purpose. These are links that will provide the requested resource, no matter if it is an image, document, video or any other file format.

You can access the wrapper links in the graphical user interface of the system when you open an asset and click on the info tab. The wrapper links use unique and secure hash codes for the access of a digital asset. This way the assets in the system can't be located by guessing of their location and name or enumeration.

The wrapper link will still be working even if the object has been moved or renamed.



Example of a wrapper link:

<https://yourdomain.com/hypercms/?wl=k0aq3mbgc6e6hdly>

This wrapper link will provide the original file based on its unique hash code as identifier.

If you want to request another version or format of the same file you can add additional information to the request:

- type ... requested format (file extension)
- mediacfg ... media configuration parameter (defines the format and size of the asset, see main configuration settings of your system)
- user ... external user ID provided for tracking

1.1 Images

You can request an image in a certain format and size as defined by the systems main configuration settings in hypercms/config/config.inc.php:

```
// Define additional download formats besides the original image:
$mgmt_imageoptions['.jpg.jpeg']['1920x1080px'] = '-s 1920x1080 -f jpg';
$mgmt_imageoptions['.jpg.jpeg']['1024x768px'] = '-s 1024x768 -f jpg';
$mgmt_imageoptions['.jpg.jpeg']['640x480px'] = '-s 640x480 -f jpg';
```

If you want your image to be a JPEG file that fits in a frame of 1024 x 768 pixels, you can request the image with this wrapper link:

<https://yourdomain.com/hypercms/?wl=k0aq3mbgc6e6hdl&type=jpg&mediacfg=1024x768px>

Advanced image editing options

In order to edit an image on the fly, advanced options can be provided in the GET request:

-s ... output size in width x height in pixel (WxH), e.g. -s 1028x768

-f ... output format (file extension without dot: jpg, png, gif), e.g. -f png

-d ... image density (DPI) for vector graphics and EPS files, common values are 72, 96 dots per inch for screen, while printers typically support 150, 300, 600, or 1200 dots per inch, e.g. -d 300

-q ... quality for compressed image formats like JPEG (1 to 100), e.g. -q 95

-c ... crop x and y coordinates (XxY), e.g. -c 100x100

-g ... gravity (NorthWest, North, NorthEast, West, Center, East, SouthWest, South, SouthEast) for the placement of an image, e.g. -g west

-ex ... extent/enlarge image, unfilled areas are set to the background color, to position the image, use offsets in the geometry specification or precede with a gravity setting, -ex 1028x768

-bg ... background color, the default background color (if none is specified or found in the image) is white, e.g. -bg black

-b ... image brightness from -100 to 100, e.g. -b 10

-k ... image contrast from -100 to 100, e.g. -k 5

-cs ... color space of image, e.g. RGB, CMYK, gray, e.g. -cs CMYK

-rotate ... rotate image in positive degrees, e.g. -rotate 90

-fv ... flip image in the vertical direction (no value required)

-fh ... flip image in the horizontal direction (no value required)

-sh ... sharpen image, e.g. one pixel size sharpen, e.g. -sh 0x1.0

-bl ... blur image with a Gaussian or normal distribution using the given radius and sigma value, e.g. -bl 1x0.1

-pa ... apply paint effect by replacing each pixel by the most frequent color in a circular neighborhood whose width is specified with radius, e.g. -pa 2

-sk ... sketches an image, e.g. -sk 0x20+120

-sep ... apply sepia-tone on image from 0 to 99.9%, e.g. -sep 80%

-monochrome ... transform image to black and white (no value required)

-wm ... watermark image->positioning->geometry,
e.g. /logo/image.png->topleft->+30

Keep in mind that you need to URL-encode the options string.

Request the image as cropped PNG file in gray scale:

[https://yourdomain.com/hypercms/?wl=k0aq3mbgc6e6hdl&type=png&options=urlencode\('-s 640x320 -c 32x10 -cs gray'\)](https://yourdomain.com/hypercms/?wl=k0aq3mbgc6e6hdl&type=png&options=urlencode('-s 640x320 -c 32x10 -cs gray'))

1.2 Videos

Video files can be provided as original video file, a preview video of the original file, a series of images from the video as JPEG or PNG images packed in a ZIP file.

Original video:

[*https://yourdomain.com/hypercms/?wl=v0aqee45c6e7hdl*](https://yourdomain.com/hypercms/?wl=v0aqee45c6e7hdl)

Preview video:

[*https://yourdomain.com/hypercms/?wl=v0aqee45c6e7hdl&type=origthumb*](https://yourdomain.com/hypercms/?wl=v0aqee45c6e7hdl&type=origthumb)

Images taken from the video as JPEG:

[*https://yourdomain.com/hypercms/?wl=v0aqee45c6e7hdl&type=jpg*](https://yourdomain.com/hypercms/?wl=v0aqee45c6e7hdl&type=jpg)

1.3 Documents

Documents (MS Word, Excel, Powerpoint, OpenOffice, ...) can be provided in their original format or as PDF or in a corresponding format, for instance MS Powerpoint will be provided as OpenOffice Presentation.

Please see the definition in the main configuration file `hypercms/config/config.inc.php` for the definitions:

```
// Define the mapping of the source and target formats for documents:
$mgmt_docconvert['.doc'] = array('.png', '.pdf', '.odt');
$mgmt_docconvert['.docx'] = array('.png', '.pdf', '.odt');
$mgmt_docconvert['.xls'] = array('.png', '.pdf', '.csv', '.ods');
$mgmt_docconvert['.xlsx'] = array('.png', '.pdf', '.csv', '.ods');
$mgmt_docconvert['.ppt'] = array('.png', '.pdf', '.odp');
$mgmt_docconvert['.pptx'] = array('.png', '.pdf', '.odp');
$mgmt_docconvert['.odt'] = array('.png', '.pdf', '.doc');
$mgmt_docconvert['.ods'] = array('.png', '.pdf', '.csv', '.xls');
$mgmt_docconvert['.odp'] = array('.png', '.pdf', '.ppt');
$mgmt_docconvert['.rtf'] = array('.png', '.pdf', '.doc', '.odt');
$mgmt_docconvert['.txt'] = array('.png', '.pdf', '.doc', '.odt');
```

Request a document (MS Powerpoint presentation) as PDF file:

[*https://yourdomain.com/hypercms/?wl=jej6jflmmgbra4b&type=pdf*](https://yourdomain.com/hypercms/?wl=jej6jflmmgbra4b&type=pdf)

2 OpenSearch

2.1 Introduction

OpenSearch is a collection of technologies that allow publishing of search results in a format suitable for syndication and aggregation. It is a way for websites and search engines to publish search results in a standard and accessible format.

OpenSearch mainly consists of two main components:

1. OpenSearch description files: XML files that identify and describe a search engine.
2. OpenSearch response: providing open search results.

More detailed information about the OpenSearch standard could be found at <http://www.opensearch.org>

Requirements: The Connector module is required.

2.2 OpenSearch Description Document

The OpenSearch Description Document is an XML-file which describes the search-service in detail. It contains information about the service-provider, attribution, adult content, syndication right, language and much more. The most valuable information is delivered by the "Url" element, which describes an interface by which a client can make requests for an external resource, such as search results, search suggestions, or additional description documents.

2.2.1 The "Url" element

As mentioned above this element characterizes how clients can access the search-service and what kind response will be delivered. One can reach the service via URL and define the search via GET parameter. The URL and the GET parameters are described in the template attribute of the "Url" element.

Example

```
<Url    type="text/html"
        indexOffset="0"
        pageOffset="0"
        rel="results"
        template="http://yourdomain.com/opensearch/?action=base_search&search_ex
pression={searchTerms}&startPage={startPage?}&count={count?}" />
```

The example shows that the template attribute contains the URL to the service "**`http://yourdomain.com/connector/opensearch/`**" and the GET Parameters with placeholders for their corresponding value "**`?action=base_search&search_expression={searchTerms}&startPage={startPage?}&count={count?}`**".

2.2.2 Fetching the OpenSearch Description Document

Calling the URL to the service with a GET Parameter "desc" with a value "1", returns the OpenSearch Description Document:

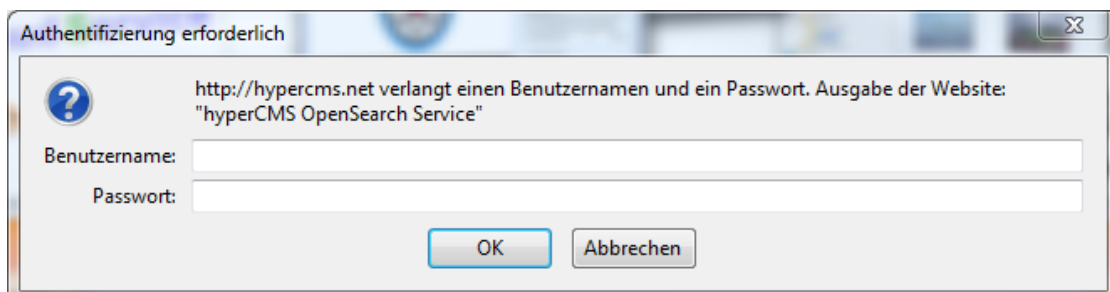
`http://yourdomain.com/hypercms/connector/opensearch/?desc=1`

2.3 Accessing the service

As already mentioned in chapter 2.1 the search-service is reachable via a simple URL and the search-query is specified via GET parameters. Considering the "Url" element example in chapter 2.1 such a query-URL can look like this:

`http://yourdomain.com/hypercms/connector/opensearch/?action=base_search&search_expression=example&startPage=2`

The search-service is secured via HTTP Basic Authentication; therefore a search can be triggered only with a valid username and password combination.



2.4 Options to use the search service

2.4.1 Object based search

In case of known object-ID, it is possible to retrieve the corresponding object directly.

Input parameters

- `object_id` ... Unique ID or hash of an object.

Example

`http://yourdomain.com/hypercms/connector/opensearch/?object_id=3ed2vmtmac43f1zq`

2.4.2 Container based search

The search-service also enables to query for objects of the same container via the container-ID.

Input parameters

- `container_id` ... Unique ID or hash of a content container.

Example

`http://yourdomain.com/hypercms/connector/opensearch/?container_id=0003983`

2.4.3 Base search

The base search is probably the common way to search for content in hyperCMS. One can search for textual phrases, attributes of media files, modification date etc.

Input parameters

- `search_expression` ... A string, which could be a part of an object name, file path, textual phrase

Example

`http://yourdomain.com/hypercms/connector/opensearch/?search_expression=test`

- `search_cat` ... If the search should only consider object names or a part of the file path, then the value of this parameter has to be "file".

Example

`http://yourdomain.com/hypercms/connector/opensearch/?search_expression=test&search_cat=file`

- `search_textnode` ... An array of strings, where each string is representing a search expression and its corresponding key is standing for the `text_id`. The `text_id` indicates where the given expression has to be looked at.

Example

`http://yourdomain.com/hypercms/connector/opensearch/?search_textnode[test]=test&search_textnode[test]=example`

- `search_dir` ... Path to a folder which only should be considered for the search.

Example

`http://yourdomain.com/hypercms/connector/opensearch/?search_expression=test&search_dir=%page%/ExamplePublication/`

- `search_format` ... An array of strings, which defines what kind of objects should be considered. These can be: page, comp, audio, document, text, image, video, compressed, flash, binary

Example

`http://yourdomain.com/hypercms/connector/opensearch/?search_expression=test&search_format[]=document&search_format[]=video`

- `date_modified_from` and `date_modified_to` ... If the search should filter for the modification date, then the from and/or to date interval has to be defined via following input parameters:
 - `date_modified_from` (format YYYY-MM-DD)
 - `date_modified_to` (format YYYY-MM-DD)

Example

`http://yourdomain.com/hypercms/connector/opensearch/?search_expression=test&date_modified_from=2021-03-01&date_modified_to=2022-02-01`

- `template` ... Name of a template, to retrieve all objects, which are using this template.

Example

`http://yourdomain.com/hypercms/connector/opensearch/?template=BoxStandard.com.p.tpl`

- `search_imagesize` ... An interval of pixels, which reduces the result only to media files, where at least one side (whether horizontal or vertical) is in the given interval.

Example

`http://yourdomain.com/hypercms/connector/opensearch/?search_imagesize=500-700`

- `search_imagewidth` ... Exact amount of pixels a media files width has to have to be considered as a hit

Example

`http://yourdomain.com/hypercms/connector/opensearch/?search_imagewidth=1024`

- `search_imageheight` ... Exact amount of pixels a media files height has to have to be considered as a hit

Example

`http://yourdomain.com/hypercms/connector/opensearch/?search_imageheight=768`

- `search_imagetype` ... Defines which format should the searched image have. The values of the parameter can be: `landscape`, `portrait` or `square`.

Example

`http://yourdomain.com/hypercms/connector/opensearch/?search_imagetype=square`

- `search_imagecolor` ... A color code that defines which main color should be considered for the search. These can be: K for Black, W for White, E for Grey, R for Red, G for Green, B for Blue, C for Cyan, M for Magenta, Y for Yellow, O for Orange, P for Pink and N for Brown.

Example

`http://yourdomain.com/hypercms/connector/opensearch/?search_imagecolor=K`

- `search_result` ... The format of the search result that will be returned. Use `"rss"` for RSS 2.0 feed and `"json"` for JSON.

Example

`http://yourdomain.com/hypercms/connector/opensearch/?search_result=json`

2.5 Pagination

The OpenSearch standard specifies parameters for implementing pagination. The hyperCMS OpenSearch service is using of them to page through the results:

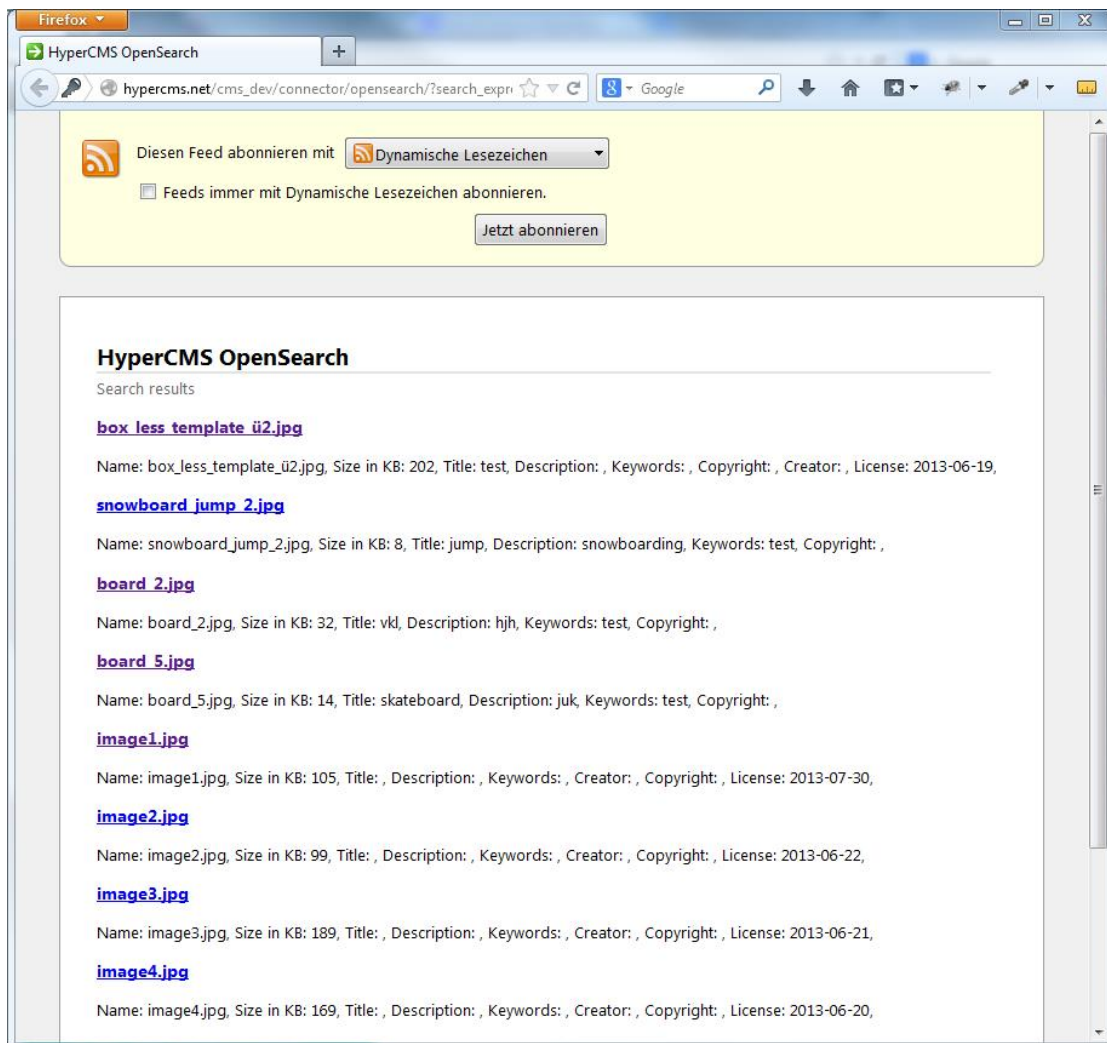
- `count` ... Defines the page size – amount of elements. If not given, the default value for this parameter is 10.
- `startPage` ... Defines which page should be displayed. If not given, the default value for this parameter is 1.

Example

`http://yourdomain.com/hypercms/connector/opensearch/?search_expression=test&count=5&page=2`

2.6 OpenSearch Response

The OpenSearch service is using the RSS 2.0 format or a JSON encoded string to display the search results.



3 RESTful OpenAPI

3.1 Introduction

The RESTful API of the system enables third party software to execute system functions, like create, modify, and delete objects, folders, groups and users. REST is used as the technology layer for the communication between the client and server.

Requirements: The Connector module is required and the API need to be enabled in the publication settings, see Administrators Guide.

3.2 REST

A short Introduction into REST (Representational State Transfer) can be found here: <http://en.wikipedia.org/wiki/REST>

The REST API is used to call a system API function and provide the required input. This is usually done via a HTTP GET/POST/PUT or DELETE request.

The request needs to be posted to: [URL of hyperCMS]/connector/rest/

The RESTful API supports GET parameters for functions that mainly read and provide the requested information, and JSON encoded data for most of the other requests that include sensitive data or writes data to the server.

General rule for the HTTP request method:

- Use GET when you need to access a resource and retrieve data, and you don't have to modify or alter the state of this data.
- Use POST when you need to send some data to the server. Ex. from a form to save these data somewhere.
- Use PUT when you need to replace the state of some data already existing on that system.
- Use DELETE when you need to delete a resource (relative to the URI you've sent) on that system.

3.3 Preparations

In order to prepare the webserver, all requests to the RESTful API need to be forwarded to a small script (PHP file).

The following example can be used for the forward of all invalid requests (request that do not have a valid resource on the webserver) to the file "[hypercms-root]/connector/rest/index.php" inside your webserver's root directory. You may also use another name or place the PHP file anywhere else inside the webserver's root directory. Make sure that the reference to the main configuration file is valid.

Please add these lines to your Apache host file:

```
# This will check if the requested filename either exists or is a
# directory, if that is NOT the case, it will call the file index.php
FallbackResource /index.php
```

If your webserver has not been configured to use an index page you should also add this line to your Apache host:

```
DirectoryIndex index.php
```

The content of the PHP script in the file "index.php":

```
<?php
// Main configuration
require_once ("[/webserver-root]/hypercms/config.inc.php");

// System API
require_once
($mgmt_config['abs_path_cms']."function/hypercms_api.inc.php");

// execute RESTful API
if (function_exists ('executeREST')) executeREST ();
?>
```

The function executeREST will handle all the requests to the Restful API and will return the proper HTTP headers and JSON encoded results.

An example REST client can be found in "[hypercms-root]/connector/rest/test/index.php". You can view and use the source code for your own client.

In order to run the tests you need to provide a valid user account, see these lines in the source code of the test file:

```
// user hash code for authentication (please provide a valid user hash
code, user must have the right permissions)
$login_user = '';
$login_password = '';
$login_hash = '';
```


3.4 Authentication

3.4.1 authenticate (POST)

Authenticate a user in the system in order to be able to use the OpenAPI. It expects the hash of a valid user to be provided. After successful login the authenticate functions returns the session ID that is needed for the call of other functions.

The standard session lifetime is 8 hours. The timeout value can be changed in the file hypercms/connector/rest/library/config.inc.php

Input parameters

- hash ... user hash code [string]
OR
- user user name [string]
- password ... password [string]

Output

JSON encoded result

Example using CURL providing the user name and password

```
curl -L -X POST 'https://[domain]/hypercms/connector/rest/authenticate' -H  
'Content-Type:application/json;charset=UTF-8;' -d '{"user\":"pparker\  
","password\":"hyperCMS123\"}'
```

Example using PHP providing the user hash

```
<?php  
$url = 'https://[domain]/hypercms/connector/rest/authenticate';  
  
$input = array(  
    'hash' => '[user-hash-code]'  
);  
  
$json = json_encode ($input, JSON_UNESCAPED_SLASHES);  
  
$ch = curl_init ($url);  
curl_setopt($ch, CURLOPT_URL, $url);  
curl_setopt ($ch, CURLOPT_HTTPHEADER, array('Content-type:  
application/json'));  
curl_setopt ($ch, CURLOPT_RETURNTRANSFER, 1);  
curl_setopt ($ch, CURLOPT_POST, 1);  
curl_setopt ($ch, CURLOPT_POSTFIELDS, $json);  
$response = curl_exec ($ch);  
curl_close ($ch);  
  
$result = json_decode ($response, true);  
  
if (!empty ($result['result'])) echo 'My sessionID is:  
' . $result['sessionID'];  
else echo 'Error: ' . $result['message'];  
?>
```

3.4.2 close (GET)

Closes the session with the session id you provided, so further attempts on function calls with this id will be rejected. As a security measure you should always close a session id you don't need it any more.

Input parameters

- sessionID ... user session ID [string]

Output

JSON encoded result

Example

```
https://[domain]/hypercms/connector/rest/close/[sessionID]
```

3.5 GET Functions

Executing a function using a GET request follows the URL syntax:

```
https://[domain]/hypercms/connector/rest/[function-name]/[sessionID]/[object-hash]
```

Keep in mind that you can also provide the object path instead of the object hash code, e.g. %comp%/Publication/Folder/Object. However, the object path must to be URL encoded for the GET request:

```
https://[domain]/hypercms/connector/rest/[function-name]/[sessionID]/urlencode([object-path])
```

3.5.1 download

The function retrieves the requested object or folder identified by the object hash (GET) or object path (POST). It creates and provides the wrapper and download link for the requested object or folder. This function supports GET and POST requests.

Function

download

Input parameters

- sessionID ... Session ID provided by function authenticate [string]
- objecthash ... object hash code for GET request [string]
- OR
- objectpath ... object path for POST request [string]

Output

JSON encoded result with wrapper and download links, meta data content, file size, mime-type.

Example using CURL

```
curl -L -X POST 'https://[domain]/hypercms/connector/rest/download'
-H 'Content-Type:application/json;charset=UTF-8;'
-d '{"sessionID":"5651441211",
"objectpath":"%comp%/Publication/Test/song.mp3"}'
```

Example GET request

GET:

```
https://[domain]/hypercms/connector/rest/download/[SessionID]/[objecthash]
```

Example POST request using PHP

```
<?php
$url = 'https://[domain]/hypercms/connector/rest/download';

$input = array(
    'sessionID' => '[sessionID]',
    'objectpath' => '%comp%/Publication/Test/song.mp3'
);

$json = json_encode ($input, JSON_UNESCAPED_SLASHES);

$ch = curl_init ($url);
curl_setopt($ch, CURLOPT_URL, $url);
curl_setopt ($ch, CURLOPT_HTTPHEADER, array('Content-type:
application/json'));
curl_setopt ($ch, CURLOPT_RETURNTRANSFER, 1);
curl_setopt ($ch, CURLOPT_POST, 1);
curl_setopt ($ch, CURLOPT_POSTFIELDS, $json);
$response = curl_exec ($ch);
curl_close ($ch);
$result = json_decode ($response, true);

if (!empty ($result['result'])) echo 'Link: '.$result['wrapperlink'];
else echo 'Error: '.$result['message'];
?>
```

Example JSON result of an object

```
{ "objectpath": "%comp%/DemoDAM/Marketing/Graphics/big-wave.jpg", "wrapperlink": "https://www.hypercms.com/cms/?wl=x6gifmx4t6e3htmlt", "content": { "Title": "Ocean wave", "Keywords": "Ocean, Surfing, Water, Wave", "Description": "Inside the wave", "Quality": "Print", "Copyright": "Ocean", "Creator": "Ocean" }, "size": "185.964", "downloadlink": "https://www.hypercms.com/cms/?dl=x6gifmx4t6e3htmlt", "modified": "2024-03-14 16:20", "mime-type": "image/jpeg" }
```

Example JSON result of an object converted to a readable format

```
objectpath => "%comp%/DemoDAM/Marketing/Graphics/big-wave.jpg"
wrapperlink => "https://www.hypercms.com/cms/?wl=x6gifmx4t6e3htmlt"
content => '{"Title": "Ocean wave", "Keywords": "Ocean, Surfing, Water, Wave", "Description": "Inside the wave", "Quality": "Print", "Copyright": "Ocean", "Creator": "Ocean"}'
size => "185.964"
downloadlink => "https://www.hypercms.com/cms/?dl=x6gifmx4t6e3htmlt"
modified => "2024-03-14 16:20"
mime-type => "image/jpeg"
```

3.5.2 getObjectlist

Get all objects of a location on the same level including their meta data and the users permissions. This is a simplified wrapper for function `rdbms_searchcontent`. In order to get all publications a user has access to you can set the value `"%publication%"` for the location.

The response will only include folders and objects the user has access to due to the access permissions by his group membership(s).

Function

`getObjectlist`

Input parameters

- sessionID ... Session ID provided by function authenticate [string]
- folderhash ... hash code of a folder for GET request [string] or [array]
OR
location ... location path for POST request [string]
- search ... search parameters for POST request [array]
 - search['expression'] ... search expression for text [string]
 - search['filename'] ... file or folder name [string]
 - search['format'] ... format values: "page", "comp", "image", "document", "text", "video", "audio", "flash", "compressed", "binary", "folder", "unknown" [array]
 - search['fileextension'] ... file extensions without dot [array]
 - search[' date_modified_from'] ... modified from date [YYYY-MM-DD]
 - search[' date_modified_to'] ... modified to date [YYYY-MM-DD]
 - search['imagewidth'] ... exact image width in pixel [integer] or image min. and max. width or height in pixels using dash as separator [min-max] the parameter search['imageheight'] will be ignored in this case
 - search['imageheight'] ... exact image height in pixel [integer]
 - search['imagecolor'] ... image primary colors: K for Black, W for White, E for Grey, R for Red, G for Green, B for Blue, C for Cyan, M for Magenta, Y for Yellow, O for Orange, P for Pink, N for Brown [array]
 - search['imagetype'] ... image type: portrait, landscape, square
 - search['object_id'] ... object ID [integer] or hash [string]
 - search['container_id'] ... container ID [integer]
 - search['geo_border_sw'] ... South/West coordinates [geo-coordinates]
 - search['geo_border_ne'] ... North/East coordinates [geo-coordinates]
 - search['samelocation'] ... only search in the same location without including objects from subfolders [boolean]
 - search['limit'] ... limit for the max. search result entries use #start,#results for a range or pagination [integer]

Output

JSON encoded result (associative result array / false)

The hash provided for the result type "folder" can be used to navigate the folder and its objects.

The value of "usedby" provides the user name if an object has been locked for exclusive editing by a user. The "permission" provides all permissions values for an object, with 1 means granted and 0 means not granted.

Example GET request

`https://[domain]/hypercms/connector/rest/getobjectlist/[SessionID]/[folder hash]`

Example POST request using PHP

```
<?php
$url = 'https://[domain]/hypercms/connector/rest/getobjectlist;

$input = array(
    'sessionID' => '[sessionID]',
    'location' => '%comp%/publication-name/folder-name/'
);

$json = json_encode ($input, JSON_UNESCAPED_SLASHES);
$ch = curl_init ($url);
curl_setopt($ch, CURLOPT_URL, $url);
curl_setopt ($ch, CURLOPT_HTTPHEADER, array('Content-type:
application/json'));
curl_setopt ($ch, CURLOPT_RETURNTRANSFER, 1);
```

```

curl_setopt ($ch, CURLOPT_POST, 1);
curl_setopt ($ch, CURLOPT_POSTFIELDS, $json);
$response = curl_exec ($ch);
curl_close ($ch);
$result = json_decode ($response, true);

if (!empty ($result)) echo 'Success: '.print_r($result, true);
else echo 'Nothing found';
?>

```

Example POST search query using PHP

```

<?php
$url = 'https://[domain]/hypercms/connector/rest/getobjectlist';

$input = array(
    'sessionID' => '[sessionID]',
    'search' => array('expression'=>'dog', 'format'=>'image',
    'imagewidth'=>'640-1028', 'limit'=>300)
);

$json = json_encode ($input, JSON_UNESCAPED_SLASHES);
$ch = curl_init ($url);
curl_setopt($ch, CURLOPT_URL, $url);
curl_setopt ($ch, CURLOPT_HTTPHEADER, array('Content-type:
application/json'));
curl_setopt ($ch, CURLOPT_RETURNTRANSFER, 1);
curl_setopt ($ch, CURLOPT_POST, 1);
curl_setopt ($ch, CURLOPT_POSTFIELDS, $json);
$response = curl_exec ($ch);
curl_close ($ch);
$result = json_decode ($response, true);

if (!empty ($result)) echo 'Success: '.print_r($result, true);
else echo 'Nothing found';
?>

```

Example JSON result of an object

```

{"jngrkm2i5mkj4gox":{"objectpath":"%comp%/Publication/Folder/test~20has
h.jpg","type":"object","container_id":"0010408","media":"weimaraner-
143753_hcm0010408.jpg","date":"2022-02-10 18:28:20","createdate":"2017-09-
05
09:13:43","user":"sys","filesize":"398","width":"1217","height":"999","loc
ation":"%/Publication/Folder/","object":"test
hash.jpg","wrapperlink":"https://domain.com/cms/?wl=jngrkm2i5mkj4gox",
"downloadlink":"https://domain.com/cms/?dl=jngrkm2i5mkj4gox","thumbnai
l":"https://domain.com/cms/service/mediawrapper.php?site=Publication&
media=Publication%2Fweimaraner-
143753_hcm0010408.thumb.jpg&token=86f4ee0d5acee6c80cdec5f196ed47ae93ab28ea
","text:Title":"Test image","text:Description":"This is just a test
image","text:Keywords":"test,image,dog,beach","usedby":
""},"permission":{"root":1,"upload":1,"download":1,"sendlink":1,"foldercrea
te":1,"folderdelete":1,"folderrename":1,"create":1,"delete":1,"rename":1,"
publish":1}},"count":"1"}

```

Example JSON result of an object converted to an array

```

array (
    'jngrkm2i5mkj4gox' =>
        array (
            'objectpath' => '%comp%/Publication/Folder/test~20image.jpg',
            'type' => 'object',
            'container_id' => '0010408',

```

```

'media' => 'weimaraner-143753_hcm0010408.jpg',
'date' => '2022-02-10 18:28:20',
'createdate' => '2017-09-05 09:13:43',
'user' => 'sys',
'filesize' => '398',
'width' => '1217',
'height' => '999',
'location' => '/Publication/Folder/',
'object' => 'test hash.jpg',
'wrapperlink' => 'https://domain.com/cms/?wl=jngrkm2i5mkj4gox',
'downloadlink' => 'https://domain.com/cms/?dl=jngrkm2i5mkj4gox',
'thumbnail' =>
'https://domain.com/cms/service/mediawrapper.php?site=Publication&media=
weimaraner-143753_hcm0010408.thumb.jpg&token=86f4...',
'text:Title' => 'Test image',
'text:Description' => 'This is just a test image',
'text:Keywords' => 'test,image,dog,beach',
'usedby' => '',
'permission' =>
array (
    'root' => 1,
    'upload' => 1,
    'download' => 1,
    'sendlink' => 1,
    'foldercreate' => 1,
    'folderdelete' => 1,
    'folderrename' => 1,
    'create' => 1,
    'delete' => 1,
    'rename' => 1,
    'publish' => 1,
)
),
'count' => '1'
)

```

Example JSON result of a folder converted to an array

```

array (
    '4hb3c3nmlpa4a4kb' =>
array (
    'objectpath' => '%comp%/Publication/Multimedia/.folder',
    'type' => 'folder',
    'container_id' => '0001101',
    'media' => '',
    'date' => '2013-06-04 00:00:00',
    'createdate' => '0000-00-00 00:00:00',
    'user' => 'admin',
    'location' => '/Publication/',
    'object' => 'Multimedia',
    'text:Title' => 'Multimedia Folder',
    'text:Description' => 'Collection of images and documents',
    'usedby' => '',
    'permission' =>
array (
    'root' => 1,
    'upload' => 1,
    'download' => 1,
    'sendlink' => 1,
    'foldercreate' => 1,
    'folderdelete' => 1,
    'folderrename' => 1,
)
)
)

```

```

        'create' => 1,
        'delete' => 1,
        'rename' => 1,
        'publish' => 1,
    )
),
'count' => '1'
)

```

3.5.3 getprojectstructure

This function creates an associative array presenting the project structure (project -> subprojects -> tasks).

Function

getprojectstructure

Input parameters

- sessionID ... Session ID provided by function authenticate [string]
- project_id ... project ID [integer]
- user ... projects for specific user name [string] (optional)

Output

JSON encoded result (associative result array / false)

Example GET request

`https://[domain]/hypercms/connector/rest/getprojectstructure/[SessionID]/[project_id]/[user]`

3.5.4 getuserinformation

This function provides the information of a user including the group memberships and permissions.

Function

getuserinformation

Input parameters

- sessionID ... Session ID provided by function authenticate [string]
- user ... user name [string] (optional)

Output

JSON encoded result (associative result array / false)

Example GET request

`https://[domain]/hypercms/connector/rest/getuserinformation/[SessionID]/[user]`

Example JSON result of a user

```
{ "Publication": { "Username": { "nologon": 0, "email": "info@hypercms.net", "realname": "Tomy Tesla", "signature": "", "language": "en", "usergroup": "|Demo|", "permissions": { "Groupname": { "globalpermission": { "user": 0, "usercreate": 0, "userdelete": 0, "useredit": 0, "group": 0, "groupcreate": 0, "groupdelete": 0, "groupedit": 0, "pers": 0, "perstrack": 0, "perstrackcreate": 0, "perstrackdelete": 0, "perstrackedit": 0, "persprof": 0, "persprofcreate": 0, "persprofdelete": 0, "persprofedit": 0, "workflow": 0, "workflowproc": 0, "workflowproccreate": 0, "workflowprocdelete": 0, "workflowprocedit": 0, "workflowprocfolder": 0, "workflowscript": 0, "workflowscriptcreate": 0, "workflowscriptdelete": 0, "workflowscriptedit": 0, "template": 0, "tpl": 0, "tplcreate": 0, "tpldelete": 0, "tpledit": 0, "tplmedia": 0, "tplmediacatcreate": 0, "tplmediacatdelete": 0, "tplmediacatrename": 0, "tplmediaupload": 0, "tplmediadelete": 0, "component": 1, "page": 0 }, "localpermission": { "component": 1, "compupload": 1, "compdownload": 1, "compsendlink": 1, "compfoldercreate": 1, "compfolderdelete": 1, "compfolderrename": 1, "compcreate": 1, "compdelete": 1, "compprename": 1, "comppublish": 1, "page": 0, "pageupload": 1, "pagesendlink": 0, "pagefoldercreate": 0, "pagefolderdelete": 0, "pagefolderrename": 0, "pagecreate": 0, "pagedelete": 0, "pagerename": 0, "pagepublish": 0 }, "rootpermission": { "desktop": 1, "desktopsetting": 1, "desktopprojectmgmt": 1, "desktoptaskmgmt": 1, "desktopcheckedout": 1, "desktoptimetravel": 1, "desktopfavorites": 1, "site": 0, "sitecreate": 0, "sitedelete": 0, "siteedit": 0, "user": 0, "usercreate": 0, "userdelete": 0, "useredit": 0 }, "pageaccess": [], "compaccess": [ "%comp%\\Boarderline\\" ] } } } }
```

Example JSON result of a user converted to an array

Array

```
(
  [Publication] => Array
  (
    [Username] => Array
    (
      [nologon] => 0
      [email] => info@hypercms.net
      [realname] => Tomy Tesla
      [signature] =>
      [language] => en
      [usergroup] => |Demo|
      [permissions] => Array
      (
        [Groupname] => Array
        (
          [globalpermission] => Array
          (
            [user] => 0
            [usercreate] => 0
            [userdelete] => 0
            [useredit] => 0
            [group] => 0
            [groupcreate] => 0
            [groupdelete] => 0
            [groupedit] => 0
            [pers] => 0
            [perstrack] => 0
            [perstrackcreate] => 0
            [perstrackdelete] => 0
            [perstrackedit] => 0
            [persprof] => 0
            [persprofcreate] => 0
            [persprofdelete] => 0
            [persprofedit] => 0
            [workflow] => 0
          )
        )
      )
    )
  )
```



```

[workflowproc] => 0
[workflowproccreate] => 0
[workflowprocdelete] => 0
[workflowprocedit] => 0
[workflowprocfolder] => 0
[workflowscript] => 0
[workflowscriptcreate] => 0
[workflowscriptdelete] => 0
[workflowscriptedit] => 0
[template] => 0
[tpl] => 0
[tplcreate] => 0
[tpldelete] => 0
[tpledit] => 0
[tplmedia] => 0
[tplmediacatcreate] => 0
[tplmediacatdelete] => 0
[tplmediacatrename] => 0
[tplmediaupload] => 0
[tplmediadelete] => 0
[component] => 1
[page] => 0
)

[localpermission] => Array
(
    [component] => 1
    [compupload] => 1
    [compdownload] => 1
    [compsendlink] => 1
    [compfoldercreate] => 1
    [compfolderdelete] => 1
    [compfolderrename] => 1
    [compcreate] => 1
    [compdelete] => 1
    [comprename] => 1
    [comppublish] => 1
    [page] => 0
    [pageupload] => 1
    [pagesendlink] => 0
    [pagefoldercreate] => 0
    [pagefolderdelete] => 0
    [pagefolderrename] => 0
    [pagecreate] => 0
    [pagedelete] => 0
    [pagerename] => 0
    [pagepublish] => 0
)

[rootpermission] => Array
(
    [desktop] => 1
    [desktopsetting] => 1
    [desktopprojectmgmt] => 1
    [desktoptaskmgmt] => 1
    [desktopcheckedout] => 1
    [desktoptimetravel] => 1
    [desktopfavorites] => 1
    [site] => 0
    [sitecreate] => 0
    [sitedelete] => 0
)

```

```
[siteedit] => 0
[user] => 0
[usercreate] => 0
[userdelete] => 0
[useredit] => 0
)

[pageaccess] => Array
(
)

[compaccess] => Array
(
    [0] => %comp%/Publication/
)
)
)
)
)
```

3.6 POST Functions

These functions require a JSON encoded string to be posted to the RESTful API.

Executing a function using a POST request follows the URL syntax:

`https://[your-domain]/hypercms/connector/rest/[function-name]`

3.6.1 upload

This function allows you to upload a single file to the provided object path or hash. If you upload a zip file the function can extract its content into the folder where it is uploaded into, if requested. It is also possible to create a preview image of supported files. If you upload an image you will be able to optionally specify the resize of the image by the resize percentage (value is valid between 0 and 200).

You can also check for duplicates in order to avoid uploading the same file content again.

If you want to update a files content you need to overwrite the existing content and you might want to create a new version of its content as well.

Function

upload

Input parameters

- sessionID ... Session ID provided by function authenticate [string]
- objectpath ... object path [string]
- OR
- objecthash ... object hash (only for existing objects) [string] (optional)
- mediacontent ... media content (base64 encoded or binary) [string or binary]
- unzip ... extract zip file [boolean] (optional)
- createthumbnail ... create only a new thumbnail from the uploaded image file [boolean] (optional)
- imageresize ... resize image [boolean] (optional)
- imagepercentage ... resize percentage (100 for original size) [integer] (optional)
- checkduplicates ... check for duplicates [boolean] (optional)
- overwrite ... overwrite existing file [boolean] (optional)
- versioning ... versioning of file [boolean] (optional)

Output

JSON encoded result

Example POST file upload (base64 encoded) using PHP

```
<?php
$url = 'https://[domain]/hypercms/connector/rest/upload';
$media_content = base64_encode (file_get_contents ('/files/image.jpg'));

$input = array(
    'sessionID' => '[sessionID]',
    'objectpath' => '%comp%/Publication/Test/image.jpg',
    'mediacontent' => $media_content,
    'unzip' => false,
    'createthumbnail' => false,
    'imageresize' => 1,
    'imagepercentage' => 50,
    'checkduplicates' => 0,
    'overwrite' => 1,
    'versioning' => 1
);
```

```

$json = json_encode ($input, JSON_UNESCAPED_SLASHES);

$ch = curl_init ($url);
curl_setopt($ch, CURLOPT_URL, $url);
curl_setopt ($ch, CURLOPT_HTTPHEADER, array('Content-type:
application/json'));
curl_setopt ($ch, CURLOPT_RETURNTRANSFER, 1);
curl_setopt ($ch, CURLOPT_POST, 1);
curl_setopt ($ch, CURLOPT_POSTFIELDS, $json);
$response = curl_exec ($ch);
curl_close ($ch);
$result = json_decode ($response, true);

if (!empty ($result['result'])) echo 'Success';
else echo 'Error: ' . $result['message'];
?>

```

Example POST file upload (binary) using CURL

```

curl -L -X POST 'https://[domain]/hypercms/connector/rest/upload'
-F 'objectpath="%comp%/Publication/Test/image.jpg"'
-F 'sessionID="[sessionID]"'
-F 'mediacontent=@"/files/image.jpg"'
-F 'checkduplicates=0'

```

3.6.2 createaccesslink

Creates an access link to any object based on the permissions of the provided user account.

Function

createaccesslink

Input parameters

- sessionID ... Session ID provided by function authenticate [string]
- objectpath ... object path [string] (optional)
- OR
- objecthash ... object hash [string] (optional)
- login ... user name of the user account to be used for access [string]
- type ... link type [al, dl] (optional)
- lifetime ... token lifetime in seconds, use 0 for unlimited life time [integer] (optional)
- formats ... formats defined in main configuration [JSON-string] (optional)

Output

JSON: URL for access to given object / false on error

Example POST request using PHP

```

<?php
$url = 'https://[domain]/hypercms/connector/rest/createwrapperlink';

$input = array(
    'sessionID' => '[sessionID]',
    'objectpath' => '%comp%/Publication/Test/image.jpg',
    'login' => '[username]',
    'type' => 'al',
    'formats' => '{"image":{"jpg":{"1920x1080px":1,"1024x768px":1}}}'
);

```

```

$json = json_encode ($input, JSON_UNESCAPED_SLASHES);

$ch = curl_init ($url);
curl_setopt($ch, CURLOPT_URL, $url);
curl_setopt ($ch, CURLOPT_HTTPHEADER, array('Content-type:
application/json'));
curl_setopt ($ch, CURLOPT_RETURNTRANSFER, 1);
curl_setopt ($ch, CURLOPT_POST, 1);
curl_setopt ($ch, CURLOPT_POSTFIELDS, $json);
$response = curl_exec ($ch);
curl_close ($ch);
$result = json_decode ($response, true);

if (!empty ($result['result'])) echo 'Success';
else echo 'Error: ' . $result['message'];
?>

```

3.6.3 createobjectaccesslink

Creates an access link to any object based on the permissions of the user account defined in the publication settings.

Function

createobjectaccesslink

Input parameters

- sessionID ... Session ID provided by function authenticate [string]
- objectpath ... object path [string] (optional)
- OR
- objecthash ... object hash [string] (optional)
- OR
- container_id ... container-ID or repository media file [string] (optional)

Output

JSON encoded result (URL for download of the multimedia file of the given object or folder / false on error)

3.6.4 createwrapperlink

In order to track and include external user IDs in the daily statistics you need to manually add the 'user' parameter to the link in the form of: &user=[user-ID]

Function

createwrapperlink

Input parameters

- sessionID ... Session ID provided by function authenticate [string]
- objectpath ... object path [string] (optional)
- OR
- objecthash ... object hash [string] (optional)
- OR
- container_id ... container-ID or repository media file [string] (optional)
- type ... media file extension or type based on main config settings [string] (optional)
- mediaconfig ... media configuration based on main config settings [string] (optional)

Output

JSON encoded result (URL for download of the multimedia file of the given object or folder / false on error)

3.6.5 createdownloadlink

In order to track and include external user IDs in the daily statistics you need to manually add the 'user' parameter to the link in the form of: &user=[user-ID]

Function

createdownloadlink

Input parameters

- sessionID ... Session ID provided by function authenticate [string]
- objectpath ... object path [string] (optional)
OR
objecthash ... object hash [string] (optional)
OR
container_id ... container-ID or repository media file [string] (optional)
- type ... media file extension or type based on main config settings [string] (optional)
- mediaconfig ... media configuration based on main config settings [string] (optional)

Output

JSON encoded result (URL for download of the multimedia file of the given object or folder / false on error)

3.6.6 createmultiaccesslink

Creates an access link to any object based on the permissions of the provided user account.

Function

createmultiaccesslink

Input parameters

- sessionID ... Session ID provided by function authenticate [string]
- multiobject ... multiobject [path1|path2|path3] or [array]
- login ... user name of the user account to be used for access [string]
- type ... link type [al, dl] (optional)
- lifetime ... token lifetime in seconds, use 0 for unlimited life time [integer] (optional)
- formats ... formats defined in main configuration [JSON-string] (optional)

Output

JSON encoded result (URL for access to the requested objects / false on error)

3.6.7 createmultidownloadlink

Generates a download link of a single media file, folder or multi objects.
Priority if multiple input parameters for media file, folder or multi objects are given:

1st...multiobjects
2nd...media file
3rd...folder

Function

createmultidownloadlink

Input parameters

- sessionID ... Session ID provided by function authenticate [string]
- publication ... publication name [string]
- multiobject ... multiobject using | as separator [string] or [array] (optional)
OR
media ... media file name [string] (optional)
OR
location ... location path [string] (optional)
- name ... presentation name [string] (optional)
- user ... user name [string]
- type ... conversion type example: jpg [string]
- mediacfg ... media configuration used for conversion (e.g.: 1024x768px) [string]
- linktype ... type [wrapper, download] (optional)
- flatzip ... flat hierarchy means no directories [true,false] (optional)

Output

JSON encoded result (URL for download of the requested objects / false on error)

3.6.8 createpublication

This function creates a new publication with all its files.

Function

createpublication

Input parameters

- sessionID ... Session ID provided by function authenticate [string]
- publication ... publication name [string]

Output

JSON encoded result

3.6.9 createuser

This function creates a new user. Use *Null* for publication name to remove access to all publications.

Function

createuser

Input parameters

- sessionID ... Session ID provided by function authenticate [string]
- publication ... publication name [string] (optional)
- login ... user login name [string]
- password ... password [string]

Output

JSON encoded result

3.6.10 creategroup

This function creates a new user group.

Function

creategroup

Input parameters

- sessionID ... Session ID provided by function authenticate [string]
- publication ... publication name [string]
- group_name ... group name [string]

Output

JSON encoded result

3.6.11 createfolder

This function creates a new folder. The folder name must not match any temp file pattern.

Function

createfolder

Input parameters

- sessionID ... Session ID provided by function authenticate [string]
- location ... location [string]
- folder ... folder name [string]
- user ... user name [string]

Output

JSON encoded result

3.6.12 createobject

This function creates a new page or component.

Function

createobject

Input parameters

- sessionID ... Session ID provided by function authenticate [string]
- location ... location [string]
- object ... object name without file extension [string]
- template ... template name [string]
- user ... user name [string]

Output

JSON encoded result

3.6.13 sendmessage

Sends a message via e-mail to a user.

Function

sendmessage

Input parameters

- sessionID ... Session ID provided by function authenticate [string]
- from_user ... from user name [string] (optional)
- to_user ... to user name [string]
- title ... title [string]
- message ... message [string]
- object_id ... object hash or object path [string] (optional)
- publication ... publication name [string] (optional)

Output

JSON encoded result (true/false)

3.6.14 createproject

This function creates a new project.

Function

createproject

Input parameters

- sessionID ... Session ID provided by function authenticate [string]
- subproject_id ... ID of main project (only if the project is a subproject) [integer]
- object_id ... object ID or path to object [string] (optional)
- user ... user name of sub/project owner [string]
- projectname ... project name [string]
- description ... project description [string] (optional)

Output

JSON encoded result

3.6.15 createtask

Creates a new user task and send optional e-mail to user.
Since version 5.8.4 the data will be stored in RDBMS instead of XML files.

Function

createtask

Input parameters

- sessionID ... Session ID provided by function authenticate [string]
- publication ... publication name [string] (optional)
- from_user ... from_user name [string]
- from_email ... from_email [email-address] (optional)
- to_user ... to_user name [string]
- to_email ... to_email [email-address] (optional)
- startdate ... start date [yyyy-mm-dd] (optional)
- finishdate ... finish date [yyyy-mm-dd] (optional)

- category ... category [link, user, workflow] (optional)
- object_id ... object hash or object path [string]
- taskname ... task name [string]
- message ... message [string] (optional)
- sendmail ... sendmail [boolean]
- priority ... priority [high, medium, low] (optional)
- project_id ... project/subproject ID if the task should be assigned to a project [integer] (optional)
- planned ... planned effort in taskunit [integer] (optional)
- dependency ... list of comma-separated task IDs that the new task depends on [comma-separated integer values] (optional)

Output

JSON encoded result

3.7 PUT Functions

These functions require the data to be put to the RESTful API.

Executing a function using a PUT request follows the URL syntax:

`https://[your-domain]/hypercms/connector/rest/[function-name]`

The request data can be provided as query or JSON encoded string.

The functions will provide a JSON encoded result in the response.

3.7.1 editpublication

This function saves all settings of a publication. It is a good advice to load the settings of a publication and manipulate the values in order to provide all settings as input.

Settings (keys for setting array):

- inherit_obj ... [boolean]
- inherit_comp ... inherit components [boolean]
- inherit_tpl ... inherit templates [boolean]
- youtube_token ... Youtube token [string]
- registration ... enable registration of new users [boolean]
- registration_group ... assign new users to the group [string]
- registration_notify ... notify the users whenever a new user registered [string]
- site_admin0 ... enable publication management [true]
- url_path_page ... URL to page root on management server [string]
- abs_path_page ... absolute path to page root on management server [string]
- exclude_folders ... exclude folders from the system [string]
- allow_ip ... only enable access for the provided IP addresses [string]
- webdav ... enable WebDAV access [boolean]
- webdav_dl ... enable download links for WebDAV [boolean]
- webdav_al ... enable access links for WebDAV [boolean]
- default_codepage ... default character set of the publication [string]
- sendmail ... enable send mail [boolean]
- mailserver ... mailserver name [string]
- portalaccesslink ... enable portal access via public link [boolean]
- accesslinkuser ... user to be used for access links [string]
- watermark_image ... watermark settings for images (-wm /home/hypercms/public_html/logo_watermark.png->toleft->10) [string]
- watermark_video ... watermark settings for videos (-wm /home/hypercms/public_html/logo_watermark.png->toleft->10) [string]
- sharesociallink ... enable social media link sharing [boolean]
- youtube ... enable Youtube upload [boolean]
- theme ... standard design theme of the publication [string]
- translate ... languages that can be used for automated translation (en,fr,de) [string]
- ocr ... languages that can be used for OCR (en,de, ...) [string]
- crypt_content ... encrypt content [boolean]
- connector_rest ... Enable RESTful API [boolean]
- connector_soap ... enable SOAP API [boolean]
- storage_limit ... storage limit in MB [integer]
- gs_access_json ... Google Cloud service JSON access code [string]
- gs_analyze_image ... Enable automated image tagging [boolean]
- gs_analyze_video ... Enable automated video tagging [boolean]
- gs_speech2text ... Enable automated speech to text translation for video and audio files [boolean]

- gs_speech2text_langcode ... language code to be used for Google Speech2Text Cloud service (en-US) [string]
- url_publ_page ... URL to page root on publication server [string]
- abs_publ_page ... absolute path to page root on publication server [string]
- url_publ_rep ... URL to repository root on publication server [string]
- abs_publ_rep ... absolute path to repository root on publication server [string]
- abs_publ_app ... absolute path to application root on publication server [string]
- publ_os ... operating system used (UNIX or WIN) [string]
- remoteclient ... URL to remote client on remote webserver [string]

Function

editpublication

Input parameters

- sessionID ... Session ID provided by function authenticate [string]
- publication ... publication name [string]
- setting ... publication settings with setting name as key and parameter as value [array]

Output

JSON encoded result

3.7.2 editpublicationsetting

This function can be used to edit a single setting of a publication.

Function

editpublicationsetting

Input parameters

- sessionID ... Session ID provided by function authenticate [string]
- publication ... publication name [string]
- setting ... publication settings with setting name as key and setting parameter as value (see publication config file for details) [array]

Output

JSON encoded result

Example PUT request using PHP

```
<?php
$url = https://[domain]/hypercms/connector/rest/editpublicationsetting';

$input = array(
    'sessionID' => '[sessionID]',
    'publication' => 'Publication',
    'setting' => array('storage_limit'=>1024)
);

$ch = curl_init ($url);
curl_setopt($ch, CURLOPT_URL, $url);
curl_setopt ($ch, CURLOPT_RETURNTRANSFER, 1);
curl_setopt ($ch, CURLOPT_PUT, 1);
curl_setopt ($ch, CURLOPT_POSTFIELDS, http_build_query($input));
$response = curl_exec ($ch);
curl_close ($ch);
$result = json_decode ($response, true);
```

```
if (!empty ($result['result'])) echo 'Success';
else echo 'Error: ' . $result['message'];
?>
```

3.7.3 edituser

This function edits a user. Use *Leave* as input if a value should not be changed. Use *Null* for publication name to remove access to all publications. Use *Null* for user group to remove user from all user groups of the publication.

Function

edituser

Input parameters

- sessionID ... Session ID provided by function authenticate [string]
- publication ... publication name [string]
- login ... user login name [string]
- old_password ... new login name [string] (optional)
- password ... password [string] (optional)
- confirm_password ... confirmed password [string] (optional)
- superadmin ... super administrator [0, 1] (optional)
- realname ... real name [string] (optional)
- language ... language setting [en, de, ...] (optional)
- timezone ... time zone [string] (optional)
- theme ... theme name (optional)
- email ... email [string] (optional)
- phone ... phone [string] (optional)
- signature ... signature [string] (optional)
- usergroup ... member of usergroup string [group1|group2] (optional)
- usersite ... member of publications string [site1|site2] (optional)
- validdatefrom ... valid date from [date] (optional)
- validdateto ... valid date to [date] (optional)

Output

JSON encoded result

3.7.4 editgroup

This function edits the settings of a user group.

The following permission names are currently recognized by the system:

desktopglobal	persprofcreate	tplmediaupload
desktopsetting	persprofdelete	tplmediadelete
desktoptaskmgmt	persprofedit	componentglobal
desktopcheckedout	workflowglobal	compupload
desktoptimetravel	workflowproc	compdownload
userglobal	workflowproccreate	compsemlink
usercreate	workflowprocdelete	compfoldercreate
userdelete	workflowprocedit	compfolderdelete
useredit	workflowprocfolder	compfolderrename
groupglobal	workflowscrip	compcreate
groupcreate	workflowscripcreate	compdelete

groupdelete	workflowscripdelete	comprenam
groupedit	workflowscripedit	comppublish
sitglobal	templateglobal	pageglobal
sitecreate	Tpl	pagesendlink
sitedelete	tplcreate	pagefoldercreate
siteedit	tpldelete	pagefolderdelete
persglobal	tpledit	pagefolderrename
perstrack	tplmedia	pagecreate
perstrackcreate	tplmediacatcreate	pagedelete
perstrackdelete	tplmediacatdelete	pagerename
perstrackedit	tplmediacatrename	pagepublish
persprof		

The value of each parameter can be 1 for granted permission or 0 for no permission. If you need further explanation about the permissions, please have a look at the Administrators Guide.

Function

editgroup

Input parameters

- sessionID ... Session ID provided by function authenticate [string]
- publication ... publication name [string]
- group_name ... group name [string]
- pageaccess ... page folder access array [array]
- compaccess ... component folder access array [array]
- permission ... permissions [array]
- user ... user name [string] (optional)

Output

JSON encoded result

Example PUT request using PHP

```
<?php
$url = 'https://[domain]/hypercms/connector/rest/editgroup';

$input = array(
    'sessionID' => $sessionID,
    'groupname' => "openAPI_testGroup",
    'publication' => "Publication",
    'pageaccess' => array("%page%/Publication/05_Kontakt/",
"%page%/Publication/02_Unternehmen/"),
    'compass' => array("%comp%/Publication/openapi/",
"%comp%/Publication/News/"),
    'permission' => array('desktopglobal' => 1, 'desktopsetting' => 1,
'pageglobal' => 1, 'pagesendlink' => 1, 'componentglobal' => 1,
'compdownload' => 1)
);

$ch = curl_init ($url);
curl_setopt ($ch, CURLOPT_RETURNTRANSFER, 1);
curl_setopt ($ch, CURLOPT_CUSTOMREQUEST, "PUT");
curl_setopt ($ch, CURLOPT_POSTFIELDS, http_build_query($input));
$response = curl_exec ($ch);
curl_close ($ch);
$result = json_decode ($response, true);
```

```

if (!empty ($result['result'])) echo 'Success';
else echo 'Error: ' . $result['message'];
?>

```

3.7.5 renamefolder

This function renames a folder.

Function

renamefolder

Input parameters

- sessionID ... Session ID provided by function authenticate [string]
- objectpath ... folder path [string] (optional)
- foldernew ... new folder name [string]
- user ... user name [string] (optional)

Output

JSON encoded result

Example PUT request using PHP

```

<?php
$url = 'https://[domain]/hypercms/connector/rest/renamefolder';

$input = array(
    'sessionID' => '[sessionID]',
    'objectpath' => '%comp%/Publication/myFolder',
    'foldernew' => 'myNewFolder',
    'user' => 'ckent'
);

$ch = curl_init ($url);
curl_setopt($ch, CURLOPT_URL, $url);
curl_setopt ($ch, CURLOPT_RETURNTRANSFER, 1);
curl_setopt ($ch, CURLOPT_PUT, 1);
curl_setopt ($ch, CURLOPT_POSTFIELDS, http_build_query($input));
$response = curl_exec ($ch);
curl_close ($ch);
$result = json_decode ($response, true);

if (!empty ($result['result'])) echo 'Success';
else echo 'Error: ' . $result['message'];
?>

```

3.7.6 editmediaobject

This function mainly uses function createmedia to render the objects media, but at the same time takes care of versioning and the object name, if the file extension has been changed.

Function

editmediaobject

Input parameters

- sessionID ... Session ID provided by function authenticate [string]

- objectpath ... object path [string]
- format ... format (file extension w/o dot) [string] (optional)
- type ... type of image/video/audio file [thumbnail, orighumb (thumbnail made from original video/audio), original, any other string present in \$mgmt_imageoptions] (optional)
- user ... user name [string] (optional)

Output

JSON encoded result (result array / false on error)

Saves original or thumbnail media file of an object, for thumbnail only jpeg format is supported as output.

3.7.7 renameobject

This function renames a page, component or asset.

Function

renameobject

Input parameters

- sessionID ... Session ID provided by function authenticate [string]
- objectpath ... object path [string] (optional)
- objectnew ... new object name without file extension [string]
- user ... user name [string] (optional)

Output

JSON encoded result

3.7.8 cutobject

This function cuts a page, component or asset.

Function

cutobject

Input parameters

- sessionID ... Session ID provided by function authenticate [string]
- objectpath ... object path [string] (optional)
- user ... user name [string]
- clipboard_add ... add to existing clipboard entries [boolean] (optional)
- clipboard_session ... save clipboard in session [boolean] (optional)

Output

JSON encoded result

Example PUT request using CURL:

```
curl --location --request PUT
'https://[domain]/hypercms/connector/rest/cutobject' \
--header 'Content-Type: application/json' \
--data-raw '{"sessionID": "0415796060",
"objectpath": "%comp%/Demo/Flexboard.php", "user": "pparker",
"clipboard_add": true, "clipboard_session": true}'
```


JSON Response:

(Notice the "clipboard" value that will be required for the pasteobject operation):

```
{ "result": true, "message": "Objects are copied to clipboard.",  
  "location": "%comp%/Demo/", "object": "Flexboard.php", "objecttype": "Component",  
  "clipboard": [ "cut|Demo|comp|%comp%/Demo/|Flexboard.php|Flexboard.php|Component" ] }
```

3.7.9 copyobject

This function copies page, component or asset.

Function

copyobject

Input parameters

- sessionID ... Session ID provided by function authenticate [string]
- objectpath ... object path [string] (optional)
- user ... user name [string] (optional)
- clipboard_add ... add to existing clipboard entries [boolean] (optional)
- clipboard_session ... save clipboard in session [boolean] (optional)

Output

JSON encoded result

3.7.10 copyconnectedobject

This function makes a connected copy of a page, component or asset.

Function

copyconnectedobject

Input parameters

- sessionID ... Session ID provided by function authenticate [string]
- objectpath ... object path [string] (optional)
- user ... user name [string] (optional)
- clipboard_add ... add to existing clipboard entries [boolean] (optional)
- clipboard_session ... save clipboard in session [boolean] (optional)

Output

JSON encoded result

3.7.11 pasteobject

This function pastes an object.

Function

pasteobject

Input parameters

- sessionID ... Session ID provided by function authenticate [string]
- location ... location path [string] (optional)
- user ... user name [string] (optional)
- clipboard_array ... clipboard entries [array] (optional)

Output

JSON encoded result

Example PUT request using CURL:

(Keep in mind that you need to use the "clipboard_array" string provided by the response of the cutobject, copyobject, or copyconnectedobject operation)

```
curl --location --request PUT
'https://[domain]/hypercms/connector/rest/pasteobject' \ --header
'Content-Type: application/json' \ --data-raw '{"sessionID": "0415796060",
"location": "%comp%/Demo/openapi/", "user": "demo", "clipboard_array":
["cut|Demo|comp|%comp%/Demo\|Flexboard.php|Flexboard.php|Component"]}'
```

3.7.12 lockobject

This function locks an object for a specific user.

Function

lockobject

Input parameters

- sessionID ... Session ID provided by function authenticate [string]
- objectpath ... object path [string] (optional)
- user ... user name [string]

Output

JSON encoded result

3.7.13 unlockobject

This function unlocks an object of a specific user.

Function

unlockobject

Input parameters

- sessionID ... Session ID provided by function authenticate [string]
- objectpath ... object path [string] (optional)
- user ... user name [string]

Output

JSON encoded result

3.7.14 publish

This function publishes a page, component or asset.

Function

publish

Input parameters

- sessionID ... Session ID provided by function authenticate [string]
- objectpath ... object path [string] (optional)
- user ... user name [string] (optional)

Output

JSON encoded result

3.7.15 unpublish

This function unpublishes a page, component, or asset and calls the function manipulateobject.

Function

unpublish

Input parameters

- sessionID ... Session ID provided by function authenticate [string]
- objectpath ... object path [string] (optional)
- user ... user name [string] (optional)

Output

JSON encoded result

3.7.16 savecontent

Saves the provided content for a specific object. Only the provided content based on its ID will be saved. Existing content with a different ID will not be deleted.

Example of a page content array as content input:

```
$content = array(
    array(
        "pagetitle"=>"My Page",
        "pageauthor"=>"Thomas Tester",
        "pagedescription"=>"Just an example"
    ),
    array(
        "id"=>"Title",
        "textu"=>"This is my page title"
    ),
    array(
        "id"=>"MyImage",
        "mediaobject"=>"%comp%/Publication/DSC_1.jpg",
        "mediaalttext"=>"This is my image",
        "mediaalign"=>"top",
        "mediawidth"=>"260",
        "mediawidth"=>"140",
    ),
    array(
        "id"=>"MyLink",
        "linkhref"=>"http://www.hypercms.com",
        "linktarget"=>"_SELF",
        "linktext"=>"Home Page Link"
```

```

    ),
    array(
        "id"=>"Related",
        "componentm"=>"%comp%/Publication/A.html|%comp%/Publication/B.html",
        "condition"=>" "
    )
);

```

Function

savecontent

Input parameters

- sessionID ... Session ID provided by function authenticate [string]
- objectpath ... object path [string] (optional)
- content ... content array with 1st key as index number and 2nd keys as id or name according to the template tags [array]
- charset ... character set [string] (optional)
- user ... user name [string]
- db_connect ... DB connectivity file name [string] (optional)

Output

JSON encoded result

3.7.17 acceptworkflowobject

Accepts an object that is managed by a workflow.

Function

acceptworkflowobject

Input parameters

- sessionID ... Session ID provided by function authenticate [string]
- objectpath ... object path [string]
- user ... user name of executing user [string]
- message ... task message [string] (optional)
- priority ... priority [high, medium, low] (optional)

Output

JSON encoded result

3.7.18 rejectworkflowobject

Rejects an object that is managed by a workflow.

Function

rejectworkflowobject

Input parameters

- sessionID ... Session ID provided by function authenticate [string]
- objectpath ... object path [string]
- user ... user name of executing user [string]
- message ... task message [string] (optional)
- priority ... priority [high, medium, low] (optional)

Output

JSON encoded result

3.7.19 editproject

This function saves data of an existing project.

Function

editproject

Input parameters

- sessionID ... Session ID provided by function authenticate [string]
- project_id ... project ID [integer]
- subproject_id ... ID of main project (only if project is a subproject) [integer]
- object_id ... object ID or path to object [string] (optional)
- user ... user name of sub/project owner [string] (optional)
- projectname ... project name [string] (optional)
- description ... project description [string] (optional)

Output

JSON encoded result

3.7.20 edittask

Saves data of a user task and send optional e-mail to user.

Function

edittask

Input parameters

- sessionID ... Session ID provided by function authenticate [string]
- task_id ... task ID [integer]
- object_id ... object ID pr path to object [string] (optional)
- to_user ... to_user name [string] (optional)
- startdate ... start date [yyyy-mm-dd] (optional)
- finishdate ... finish date [yyyy-mm-dd] (optional)
- taskname ... name of task [string] (optional)
- message ... task message/description [string] (optional)
- sendmail ... sendmail [boolean]
- priority ... priority [high, medium, low] (optional)
- status ... status in percent [0-100] (optional)
- planned ... planned effort in taskunit [integer] (optional)
- actual ... actual effort in taskunit [integer] (optional)
- project_id ... project/subproject ID the task belongs to [integer] (optional)
- dependency ... list of comma-separated task IDs that the new task depends on [comma-separated integer values] (optional)

Output

JSON encoded result

3.8 DELETE Functions

These functions require a simple request to the RESTful API without any additional data.

Executing a function using a DELETE request follows the URL syntax:

`https://[your-domain]/hypercms/connector/rest/[function-name]/[sessionID]/[identifier]`

3.8.1 deletepublication

This function deletes a publication with all its files.

Function

deletepublication

Input parameters

- sessionID ... Session ID provided by function authenticate [string]
- publication ... publication name [string]

Output

JSON encoded result

Example DELETE request

`https://[domain]/hypercms/connector/rest/deletepublication/[SessionID]/[publication]`

3.8.2 deleteuser

This function removes a user.

Function

deleteuser

Input parameters

- sessionID ... Session ID provided by function authenticate [string]
- publication ... publication where the user should be removed, use *Null* for all publications [string]
- user ... user name [string]

Output

JSON encoded result

Example DELETE request

`https://[domain]/hypercms/connector/rest/deleteuser/[SessionID]/[publication]/[user]`

3.8.3 deletegroup

This function removes a user group.

Function

deletegroup

Input parameters

- sessionID ... Session ID provided by function authenticate [string]
- publication ... publication name [string]
- groupname ... group name [string]

Output

JSON encoded result

Example DELETE request

`https://[domain]/hypercms/connector/rest/deletegroup/[SessionID]/[publication]/[groupname]`

3.8.4 deletefolder

This function removes a folder. The folder must be empty in order to be removed from the system.

Function

deletefolder

Input parameters

- sessionID ... Session ID provided by function authenticate [string]
 - objectpath ... folder path [string] (optional)
 - folderhash ... folder hash code for DELETE request [string]
- OR
- objectpath ... object path for POST request [string]

Output

JSON encoded result

Example DELETE request

`https://[domain]/hypercms/connector/rest/deletefolder/[SessionID]/[folderhash]`

3.8.5 deleteobject

This function removes a page, asset, or component.

Function

deleteobject

Input parameters

- sessionID ... Session ID provided by function authenticate [string]
 - objecthash ... object hash code for DELETE request [string]
- OR
- objectpath ... object path for POST request [string]

Output

JSON encoded result

Example DELETE request

`https://[domain]/hypercms/connector/rest/deleteobject/[SessionID]/[objecthash]`

Example POST request

```
curl -L -X POST 'https://[domain]/hypercms/connector/rest/deleteobject -F  
'objectpath=%comp%/publication/folder/file.jpg' -F 'sessionID=[sessionID]'
```

3.8.6 deleteproject

This function removes a project.

Function

deleteproject

Input parameters

- sessionID ... Session ID provided by function authenticate [string]
- project_id ... project ID or array of project IDs to be deleted [string or array]

Output

JSON encoded result

Example DELETE request

```
https://[domain]/hypercms/connector/rest/deleteproject/[SessionID]/[project_id]
```

3.8.7 deletetask

Deletes user tasks.

Function

deletetask

Input parameters

- sessionID ... Session ID provided by function authenticate [string]
- task_id ... task ID or array of task IDs to be deleted [integer]

Output

JSON encoded result

Example DELETE request

```
https://[domain]/hypercms/connector/rest/deletetask/[SessionID]/[task_id]
```


4 SOAP OpenAPI

4.1 Introduction

The REST API of the system enables third party software to execute system functions, like create, modify, and delete objects, folders, groups and users. SOAP and WSDL is used as the technology layer for the communication between the client and server.

Requirements: The Connector module is required and the API need to be enabled in the publication settings, see Administrators Guide.

4.2 SOAP

A short Introduction into SOAP can be found here: <http://en.wikipedia.org/wiki/SOAP>
SOAP is supported in the version 1.1 and 1.2.

4.3 WSDL

A short Introduction into WSDL can be found on Wikipedia here:
http://en.wikipedia.org/wiki/Web_Services_Description_Language

The file which describes the OpenAPI functions is accessible under the following link:
<URL to hyperCMS>/connector/openapi/?wsdl.

4.4 (re)generate the WSDL

Whenever you need to generate or regenerate the WSDL file (for Example after an update to the OpenAPI) you can do that by opening the following link:
<URL to hyperCMS>/connector/openapi/?wsdl=generate
The newly generated WSDL file will be shown afterwards.

4.5 Types

There are several types defined in WSDL for use with the OpenAPI.

4.5.1 download_object

Describes an object located inside the system. An object is either a component, multimedia asset, page or folder. This object has the following attributes:

objectpath

Complete path to the object, it has the format (%comp% for Components or %page% for Pages) followed by the publication name and then the exact location of the object.

type

The mime-type of the file

modifiedtime

Date of the last change to this object in the format „YYYY-MM-DD HH:SS“

downloadlink

A link in order to download the file. This link is only provided for multimedia assets.

wrapperlink

A link to the file which can be used to view the file inside the browser,

content

Contains the JSON-encoded text content of an object. For multimedia files this is usually metadata, for other files these are the fields you can change when editing an object.

4.5.2 Group

Is used to identify a group when changing user settings.

Name: Name of the group

Publication: The publication from which the group should be used

4.5.3 Grouplist

A list of groups.

4.5.4 Permission

A permission that is assigned to the group.

Name:

Name of the permission assigned to the group.

The following names are currently recognized by the system:

desktopglobal	persprofcreate	tplmediaupload
desktopsetting	persprofdelete	tplmediadelete
desktoptaskmgmt	persprofedit	componentglobal
desktopcheckedout	workflowglobal	compupload
desktoptimetravel	workflowproc	compdownload
userglobal	workflowproccreate	compsendlink
usercreate	workflowprocdelete	compfoldercreate
userdelete	workflowprocedit	compfolderdelete
useredit	workflowprocfolder	compfolderrename
groupglobal	workflowscrip	compcreate
groupcreate	workflowscripcreate	compdelete
groupdelete	workflowscripdelete	comprename
groupedit	workflowscripedit	comppublish
siteglobal	templateglobal	pageglobal
sitecreate	tpl	pagesendlink
sitedelete	tplcreate	pagefoldercreate
siteedit	tpldelete	pagefolderdelete
persglobal	tpledit	pagefolderrename
perstrack	tplmedia	pagecreate

perstrackcreate	tplmediacatcreate	pagedelete
perstrackdelete	tplmediacatdelete	pagerename
perstrackedit	tplmediacatrename	pagepublish
persprof		

If you need further explanation about the permissions, please have a look at the Administrators Guide.

Set:

Either 1 or 0. If the value is 1 the permission will be granted to the group.

4.5.5 Permissionlist

A list of permissions.

4.5.6 Folderlist

This is a list of folders to which the group has access. It consists of "%page%/" for pages, "%comp%/" for components, then the publication name followed by "/" and the subsequent folders. For example "%page%/Publication/Kontakt/" grants access to the folder "Kontakt" in the pages of the Publication "Publication".

4.5.7 Textcontent

A text content field that is saved into an object.

Name: The name (ID) of the text content

Value: The value that is stored

4.5.8 Fieldlist

A list of text fields which are stored to the specified object.

4.6 Functions

These are functions of the SOAP OpenAPI which can be used to manipulate and view objects, folder, users, and user groups. You can use any common programming language to create a SOAP client.

See the following example of a SOAP client written in PHP:

```
try
{
    // PHP Version 5.01 is required to be able to use the SoapClient class
    $hypercms = new SoapClient
    ("http://[domain]/hypercms/connector/soap/?wsdl=generate", array ('trace'
=> 1, 'exception' => 1));
    $sessionID = $hypercms->authenticate
    ('f6f9ae656204c2bc24321cbaffef060a6');
    $hypercms->folder_create ($sessionID, "%comp%/Publication/Folder/");
    $file = "/tmp/test.mp3";
    $hypercms->upload ($sessionID, "%comp%/Publication/Folder/test.mp3",
base64_encode(file_get_contents($file)), filesize($file), 0, 0, 0, 0);
    $download = $hypercms->download ($sessionID,
"%comp%/Publication/Folder/test.mp3");
    $hypercms->close ($sessionID);
}
catch (Exception $e)
{
    // Do your error handling here
    var_dump ($e);
}
```

An example SOAP client can be found in hypercms/connector/soap/test/. You can view and use the source code for your own client.

4.6.1 authenticate

Authenticate a user in the system in order to be able to use the OpenAPI. It expects the user name and password or hash of a valid user to be provided. After successful login the authenticate functions returns the session ID that is needed for other functions. The standard session lifetime is 8 hours. The timeout value can be changed in the file hypercms/connector/soap/library/config.inc.php

Input parameters

- login name
- password
- OR
- user hash

Example

```
$sessionID = $hypercms->authenticate('', '',
'f6f9ae656204c2bc2521cbaffef060a6');
```

4.6.2 close

Closes the session with the session id you provided, so further attempts on function calls with this id will be rejected. As a security measure you should always close a session id you don't need it any more.

Input parameters

- sessionID

Example

```
$hypercms->close($sessionID);
```

4.6.3 download

With this function you retrieve the download_object for the object or folder identified by the objectpath. Which in this case can either be the path to the object, or the hash code of the object / folder.

Input parameters

- sessionID
- object path OR object hash

Example

```
$hypercms->download(15895748, "%comp%/Publication/video.avi");
$hypercms->download(15895748, "zjm3quu0ahqd3k6y");
$hypercms->download(15895748, "%comp%/Publication/Test/song.mp3");
$hypercms->download(15895748, "%comp%/Publication/Test/");
```

4.6.4 upload

This function allows you to upload a single file to the objectpath. This objectpath must lead to a non-existing file. If you upload a zip file the function can extract its content into the folder where it is uploaded into, if requested. It is also possible to create a preview image of supported files and if you upload an image you can specify if you want to resize the image and the resize percentage (value is valid between 0 and 200)

Input parameters

- sessionID
- object path OR object hash (only for existing objects)
- media content (base64 encoded)
- extract zip file (0/1)
- create preview image (0/1)
- resize image (0/1)
- resize percentage (1 - 200) or 0 or 100 for the original image size
- check for duplicates (0/1)
- overwrite existing file (0/1)
- versioning of file (0/1)

Example

```
$hypercms->upload(15895748, "%comp%/Publication/Test/song.mp3", "[base64 encoded content of the file]", 0, 0, 0, 100, 0, 1, 1);
```

4.6.5 user_create

Create a new user with the provided data for the specified publication. If you don't specify a publication or it is *Null* then the user will be created without a publication assignment. Login name and password are used to authenticate with the system

Input parameters

- sessionID
- publication name
- login name
- password

Example

```
$hypercms->user_create(15895748, "", "User1", "password");  
$hypercms->user_create(15895748, "*Null*", "User2", "password");  
$hypercms->user_create(15895748, "Publication", "User3", "password");
```

4.6.6 user_edit

This function can change the password, real name, language, theme, email, signature and the group membership of the user identified by his login name. If you change your own user data you must provide the current password as a security measure, else you can leave it empty.

For the supported languages, time zones, and design themes have a look at the edit user feature in your system.

Input parameters

- sessionID
- login name
- password
- real name
- language (see hypercms/language)
- timezone (use a valid timezone, see PHP function `timezone_identifiers_list`)
- name of the design theme (see hypercms/theme)
- email
- phone
- signature
- valid date from (YYYY-MM-DD)
- valid date to (YYYY-MM-DD)
- list of groups
- current password

Example

```
$groups = array(array("publication" => "Publication", "name" =>  
"Designer"), array("publication" => "DemoDAM", "name" => "ChiefEditor"));
```

```
$hypercms->user_edit(15895748, "TestUser", "Testpassword1", "Test User",  
"de", "*Leave*", "Standard", "test@user.com", "0123456", "This is a  
Signature to test", "*Leave*", "*Leave*", $groups, "");
```

```
# If the authenticated user is TestUser  
$hypercms->user_edit(15895748, "TestUser", "Testpassword1", "Test User",  
"de", "*Leave*", "Standard", "test@user.com", "0123456", "This is a  
Signature to test", "*Leave*", "*Leave*", $groups, "TestingTest");
```

4.6.7 user_delete

This function deletes a user from the user database. If you provide an empty publication or *Null* as publication the user will be deleted from the system else from the publication only.

Input parameters

- sessionID
- publication name
- login name

Example

```
$hypercms->user_delete(15895748, "*Null*", "TestUser");  
$hypercms->user_delete(15895748, "", "TestUser2");  
$hypercms->user_delete(15895748, "Publication", "TestUser3");
```

4.6.8 group_create

Create a new group for a publication.

Input parameters

- sessionID
- publication name
- name of the group

Example

```
$hypercms->group_create(15895748, "Publication", "TestGroup");
```

4.6.9 group_edit

Change the allowed paths and permissions associated with a group.

Input parameters

- sessionID
- publication name
- group name
- folder access to pages
- folder access to components
- permissions

Example

```
$accesslistpage = array("%page%/Publication/Kontakt/",  
"%page%/Publication/Unternehmen/");  
  
$accesslistcomp = array("%comp%/Publication/_denis_test/",  
"%comp%/Publication/News/");  
$permissionlist = array(array('name' => 'desktopglobal', 'set' => 1),  
array('name' => 'desktopsetting', 'set' => 1));  
  
$hypercms->group_edit(15895748, "Publication", "TestGroup",  
$accesslistpage, $accesslistcomp, $permissionlist);
```

4.6.10 group_delete

Deletes a group out of a publication

Input parameters

- sessionID,
- publication name
- group name

Example

```
$hypercms->group_delete(15895748, "Publication", "TestGroup");
```

4.6.11 publish / unpublish

Publishes/unpublish the object identified by objectpath. If the object is a folder all files contained in it will also be published / unpublished. Its also possible to only apply this to already published objects.

Input parameters

- sessionID
- object path OR object hash
- only publish objects that have been already been published before (0/1)

Example

```
$hypercms->publish(15895748, "%comp%/Publication/Test/test.php", 0);
```

```
$hypercms->publish(15895748, "%comp%/Publication/Test/", 1);
```

```
$hypercms->unpublish(15895748, "%comp%/Publication/Test/test.php", 0);
```

```
$hypercms->unpublish(15895748, "%comp%/Publication/Test/", 1);
```


4.6.12 save_content

You can change the value of any content stored with an object with this function. Please be aware that you can set the content for any ID, but it won't be displayed in the system if the respective template doesn't define the tag ID.

Input parameters

- sessionID
- object path OR object hash
- list of fields (see savecontent function reference)
- character set for the object

Example

```
$fields = array(
    array(
        "pagetitle"=>"My Page",
        "pageauthor"=>"Thomas Tester",
        "pagedescription"=>"Just an example"
    ),
    array(
        "id"=>"Title",
        "textu"=>"This is my page title"
    ),
    array(
        "id"=>"MyImage",
        "mediaobject"=>"%comp%/Publication/DSC_1.jpg",
        "mediaalttext"=>"This is my image",
        "mediaalign"=>"top",
        "mediawidth"=>"260",
        "mediawidth"=>"140",
    ),
    array(
        "id"=>"MyLink",
        "linkhref"=>"http://www.hypercms.com",
        "linktarget"=>"_SELF",
        "linktext"=>"Home Page Link"
    ),
    array(
        "id"=>"Related",
        "componentm"=>"%comp%/Publication/A.html|%comp%/Publication/B.html",
        "condition"=>" "
    )
);

$hypercms->save_content(15895748, "%comp%/Publication/Test/test.php",
$fields, "UTF-8");
```

4.6.13 folder_create

Create a new folder at the specified path.

Input parameters

- sessionID
- folder path (Path including the folder to be created)

Example

```
$hypercms->folder_create(15895748, "%comp%/Publication/Test/Test2/");
```

4.6.14 folder_rename

Rename the folder at the specified path to new name which should just be the new name.

Input parameters

- sessionID
- folder path
- new name

Example

```
$hypercms->folder_rename(15895748, "%comp%/Publication/Test/Test2/",  
„test3“);
```

4.6.15 folder_delete

Deletes the specified folder

Input parameters

- sessionID
- folder path (Ordner der gelöscht werden soll)

Example

```
$hypercms->folder_delete(15895748, "%comp%/Publication/Test/Test3/");
```

4.6.16 object_create

Creates a new object based on a defined template. The template must exist and must be of the correct type (page templates for pages, component template for components). The objectpath should contain the filename without an extension because the extension is defined by the template

Input parameters

- sessionID
- object path
- template (Name of the template which should be used)

Example

```
$hypercms->object_create(15895748, "%comp%/Publication/Test/Test2",  
"Template");
```

4.6.17 object_rename

Rename the object to another name. Object should contain the file extension of the file, but the new name must not contain any extension.

Input parameters

- sessionID
- object
- new name

Example

```
$hypercms->object_rename(15895748, "%comp%/Publication/Test/Test2.php",  
"Test3");
```

4.6.18 object_delete

Delete the object and all associated data from the System. It can then only be restored from the backup

Input parameters

- sessionID
- object

Example

```
$client->object_delete(15895748, „%comp%/Publication/Test/Test3.php“);
```

4.6.19 project_create

Create a new project or subproject.

Input parameters

- sessionID
- ID of main project (only if the project is a subproject)
- object hash OR object path (could be a folder as well)
- user name of sub/project owner
- project name
- project description

Example

```
$hypercms->project_create(15895748, "", "%comp%/Publication/ProjectABC",  
"peterparker", "Project ABC", "Defining the alphabet");
```

4.6.20 project_edit

Edit the project attributes.

Input parameters

- sessionID
- ID of project
- ID of main project (only if the project is a subproject)
- Object hash or object path (could be a folder as well)
- user name of sub/project owner
- project name
- project description

Example

```
$hypercms->project_edit(15895748, 123, "",  
"%comp%/Publication/ProjectABC", "peterparker", "Project NEW ABC",  
"Defining the new alphabet");
```

4.6.21 project_delete

Deletes a project and does not touch associated data like subprojects or tasks.

Input parameters

- sessionID
- ID of project

Example

```
$client->project_delete(15895748, 123);
```

4.6.22 task_create

Create a new task.

Input parameters

- sessionID
- publication name
- from user name
- from user email address
- to user name
- to user email address
- start date [YYYY-MM-DD]
- finish date [YYYY-MM-DD]
- category [link, user, workflow]
- object hash or object path
- task name
- task description
- priority [high, medium, low]
- project ID if the task should be assigned to a project
- planned effort in task unit
- list of comma-separated task IDs that the new task depends on

Example

```
$hypercms->task_create(15895748, "Publication", "peterparker", "",  
"clarkkent", "", "2016-05-01", "2016-05-30", "user",  
"%comp%/Publication/ProjectABC/tasks/do-something.doc", "Design Draft",  
"Create design draft for the customer", "medium", 123, 15, "12,17");
```

4.6.23 task _edit

Edit the tasks attributes.

Input parameters

- sessionID
- ID of task
- object hash or object path
- to user name
- start date [YYYY-MM-DD]
- finish date [YYYY-MM-DD]
- name of task
- task description
- priority [high, medium, low]
- status in percent [0-100]
- planned effort in task unit
- actual effort in task unit
- project or subproject ID the task belongs to
- list of comma-separated task IDs that the new task depends on

Example

```
$hypercms->task_edit(15895748, 411,  
"%comp%/Publication/ProjectABC/tasks/do-something.doc", "clarkkent",  
"2016-05-10", "2016-06-05", "Design Draft", "Create design draft for  
customer", "medium", 0, 15, 0, 123, "12,17");
```

4.6.24 task _delete

Delete a task.

Input parameters

- sessionID
- ID of task

Example

```
$client->task_delete(15895748, 411);
```

4.6.25 workflow_accept

Accept/approve an object after review and execute the workflow.

Input parameters

- sessionID
- user name (user executing the workflow)
- message to next user
- priority [high, medium, low]

Example

```
$hypercms->workflow_accept(15895748,  
"%comp%/Publication/ProjectABC/tasks/do-something.doc", "clarkkent", "The  
design draft for the customer looks good", "medium");
```

4.6.26 workflow_reject

Reject an object after review and execute the workflow.

Input parameters

- sessionID
- user name (user executing the workflow)
- message to next user
- priority [high, medium, low]

Example

```
$hypercms->workflow_reject(15895748,  
"%comp%/Publication/ProjectABC/tasks/do-something.doc", "clarkkent", "The  
chapter for the management is missing in the design draft for the  
customer", "medium");
```

5 Authentication

If you are using LDAP / MS Active Directory, or any other user directory, you can specify the file name without extension to be used for the connector in the main configuration file `config.inc.php` located in `hypercms/config`.

For instance, specify the file name `"ldap_connect"` located in the Connector module or in `data/connect/` if you are using your own connector. The function `authconnect` in the connector file is used to connect to an LDAP/AD directory and to `snyc/update` the user data.

Keep in mind that the settings will be applied to all publications. If you want to use LDAP or Active Directory for a specific publication, you need define the settings in the publication management.

The function `authconnect` in the file `hypercms/connector/authconnect/ldap_connect.inc.php` can be copied to `data/connect/` and can be modified. The connector file saved in `data/connect/` will not be updated by the system. This means that all your modifications in the code will be kept.

The function `authconnect` authenticates a user by username and password using LDAP/AD. Authenticating to LDAP does not actually set up a session or perform any sort of login functionality.

Users that are not found in the LDAP/AD directory will be logged in if the provided valid user credentials and the user exists in the system (this is the case for external users that are not included in LDAP/AD).

If you don't like this default behaviour, since you don't want persons that are not valid LDAP/AD users to be able to logon, you should delete invalid users if they exist in the system (uncomment lines in the code).

The user data of users which can be found in LDAP/AD will be updated with the data provided by LDAP/AD.

Keep in mind that a mapping for the publication and group membership need to be defined in the main configuration or publication configuration.

If a per publication LDAP/AD connection is used, the users will be created automatically as well if the same user name and password will be accepted by different LDAP/AD directories.

For LDAP or Active Directory connection you need to uncomment this line in the main configuration file (`hypercms/config/config.inc.php`):

```
$mgmt_config['authconnect'] = "ldap_connect";
```

You can create your own file in `data/connect` and code for the function `authconnect` for connecting to any kind of user directory.

Please open the file `ldap_connect.inc.php` for more information.

5.1 Configuration of OpenLDAP and Apache

Add the following to your `ldap.conf` file, usually located in `/etc/openldap/ldap.conf`:

```
#--begin--
# Instruct client to NOT request a server's cert.
#
# WARNING: This will open up the server for Man-in-the-middle
# attacks and should *not* be used on production systems or outside
# of test-scenarios!
#
# If you use this setting you will not need any other settings as
# no certificate is requested and therefore will not be validated
TLS_REQCERT never

# Define location of CA Cert
TLS_CACERT /usr/local/ssl/certs/AD_CA_CERT.pem
TLS_CACERTDIR /usr/local/ssl/certs
#--end--
```

You also need to place those same settings in a file within the Apache Web user homedir called `".ldaprc"`:

```
cp /usr/local/openldap/etc/openldap/ldap.conf ~www/.ldaprc
```

You can then test that you're able to establish a LDAPS connection to Active Directory from the OpenLDAP command tools:

```
ldapsearch -H "ldaps://ldap.server.com"
```

This should return some output in extended LDIF format and will indicate no matching objects, but it proves the connection works.

5.2 Configuration of the LDAP / MS Active Directory Connector

5.2.1 Requirements

To use the LDAP / MS Active Directory Connector, your server must support:

- PHP 5.5.9 or greater
- PHP LDAP Extension
- An LDAP / MS Active Directory Server

If your AD server requires SSL, your server must support the following libraries:

- PHP SSL Libraries (<http://php.net/openssl>)

5.2.2 Configuration

Please define the LDAP/AD connection parameters in the main configuration file `hypercms/config/config.inc.php` for all publications or use a definition per publication provided by the publication management, see Administrators Guide.

```
// LDAP/AD Integration
// If you are using LDAP, Active Directory, or any other user directory, you can
// specify the file name without extension to be used for the connector.
// The standard connector file "ldap_connect.inc.php" is located in
// hypercms/connector/authconnect/.
// Besides the standard connector, you can make a copy of the file
// "ldap_connect.inc.php" and paste it in /data/connect/ in order to modify the code
// and define your own connector.
// The system will look in /data/connect/ for the specified connector file and
// will fallback to /hypercms/connector/authconnect/.

// Use "ldap_connect" for LDAP/AD
// Specify the file name "ldap_connect" located in /data/connect/ or
// hypercms/connector/authconnect/ in order to connect to an LDAP or AD directory
// and verify users.
// Alternatively you can create your own connector file in /data/connect/ and
// refer to it. Make sure you use the file extension .inc.php and use the same
// function name and parameters.
$mgmt_config['authconnect'] = "";

// Enable (true) or disable (false) the connectivity for all publications
// If enabled the below AD/LDAP settings need to be defined.
// If disabled the AD/LDAP settings need to be defined in the publication
// management (per publication).
$mgmt_config['authconnect_all'] = false;

// Define a LDAP/AD user with general read permissions
// This LDAP/AD user is only required if you want to use SSO using the OAuth
// remoteclient
$mgmt_config['ldap_admin_username'] = "";
$mgmt_config['ldap_admin_password'] = "";

// Define the connection parameters
// Port 389 is for LDAP over TLS. Port 636 is for LDAP over SSL, which is
// deprecated.
// LDAP works from port 389 and when you issue the StartTLS (with
// ldap_start_tls()) it encrypts the connection.
// Example: ldapserver.name
$mgmt_config['ldap_servers'] = "ldapserver.name";
// Example: domain.com
```

```

$mgmt_config['ldap_userdomain'] = "";
// Example: OU=Departments,DC=MYDOMAIN,DC=COM
$mgmt_config['ldap_base_dn'] = "";
// Example: 2 or 3
$mgmt_config['ldap_version'] = 3;
// Example: 389 (for TLS) or 636 (for SSL)
$mgmt_config['ldap_port'] = "";
$mgmt_config['ldap_follow_referrals'] = false;
$mgmt_config['ldap_use_ssl'] = true;
$mgmt_config['ldap_use_tls'] = false;

// Define the LDAP/AD user filter for the LDAP bind or leave empty if the LDAP
server support the user name for the bind
// This setting will not be applied if MS AD or a user domain is used.
// Use %user% as placeholder for the user name.
// Example: uid=%user%,cn=users
$mgmt_config['ldap_username_dn'] = "";

// Define the user filter for the search in LDAP/AD
// For Active Directory define "sAMAccountName"
$mgmt_config['ldap_user_filter'] = "sAMAccountName";

// Enable (true) or disable (false) the sync of LDAP users with the system users
// The user information such as name, email, telephone is queried and
synchronized.
// If the user's publication and group membership should also be synchronized
according to certain rules,
// this must be specified in the $mgmt_config['ldap_sync_publications_mapping']
and $mgmt_config['ldap_sync_groups_mapping']
// otherwise the memberships are retained as stored in the system.
$mgmt_config['ldap_sync'] = false;

// Define the user attributes you want so sync with LDAP/AD
// Supported attributes for the sync are 'memberof', 'givenname', 'sn',
'telephonenumber', and 'mail'
// memberof ... user memberships in LDAP/AD
// givenname ... firstname
// sn ... surname/lastname
// telephonenumber ... phone
// mail ... e-mail address
$mgmt_config['ldap_user_attributes'] = array('memberof', 'givenname', 'sn',
'telephonenumber', 'mail');

// Delete the user if it does not exist in the LDAP/AD directory (true) or leave
user (false)
$mgmt_config['ldap_delete_user'] = false;

// Keep existing group memberships of user (true) or not (false)
// Enable this setting if groups are defined manually and by LDAP/AD (mix of
groups)
// Keep in mind that enabling this setting has security implications, since
LDAP/AD groups will not be removed anymore once assigned
$mgmt_config['ldap_keep_groups'] = false;

// Synchronize AD/LDAP groups with publications of the user
// Define mapping based on a search string that defines the users publication
membership, use "," as separator for the assignment to multiple publications
// Mapping: "LDAP search string" => "Publication-name-A,Publication-name-B"
// Example: $mgmt_config['ldap_sync_publications_mapping'] =
array("DC=domain,DC=de"=>"Publication-name-A,Publication-name-B",
"DC=domain,DC=uk"=>"Publication-name-C");
$mgmt_config['ldap_sync_publications_mapping'] = array();

// Synchronize AD/LDAP groups with user groups of the user
// Define mapping based on a search string that defines the users group
membership, use "," as separator for the assignment to multiple groups
// Mapping: "LDAP search string" => "Publication-name-A/Group-name-A,Publication-
name-B/Group-name-B"

```

```
// Example for general groups for all publications:
$mgmt_config['ldap_sync_groups_mapping'] = array("OU=MANAGER
GROUP"=>"ChiefEditor", "OU=ALL GROUPS"=>"Editor");
// Example for specific groups per publication:
$mgmt_config['ldap_sync_groups_mapping'] = array("OU=MANAGER
GROUP"=>"Publication/ChiefEditor", "OU=ALL GROUPS"=>"Publication/Editor");
$mgmt_config['ldap_sync_groups_mapping'] = array();

// Signature template
// If the user data should be used to create a signature for the e-mails you can
use the following template.
// Leave empty or comment if you don't want to use the signature template.
// Use %firstname%, $lastname%, %email%, and %phone% for the provided user data
from LDAP/AD.
$mgmt_config['ldap_user_signature'] = "Best regards
%firstname% %lastname%
E: %email%
T: %phone%

This message is intended for the individual named above and is confidential and
may also be privileged. If you are not the intended
recipient, please do not read, copy, use or disclose this communication to others.
For electronically send information and pieces of advice,
which are not confirmed by following written execution, in principle no adhesion
is taken over.
";
```

LDAP Servers (required)

The domain controllers option is an array of servers located on your network that serve Active Directory. You insert as many servers or as little as you'd like depending on your forest (with the minimum of one of course).

For example, if the server name that hosts AD on my network is named `ACME-DC01`, then I would insert `[ACME-DC01.corp.acme.org]` inside the domain controllers option array.

User Domain (required)

The account suffix option is the suffix of your user accounts in AD. For example, if your domain DN is `DC=corp,DC=acme,DC=org`, then your account suffix would be `@corp.acme.org`. This is then appended to the end of your user accounts on authentication.

For example, if you're binding as a user, and your username is `jdoe`, then Adldap would try to authenticate with your server as `jdoe@corp.acme.org`.

Base Distinguished Name (required)

The base distinguished name is the base distinguished name you'd like to perform operations on. An example base DN would be `DC=corp,DC=acme,DC=org`. If one is not defined, then Adldap will try to find it automatically by querying your server. It's recommended to include it to limit queries executed per request.

LDAP Version (required)

The LDAP protocol version to be used (2 or 3)

Port (optional)

The port option is used for authenticating and binding to your AD server. The default ports are already used for non SSL and SSL connections (389 and 636). Only insert a port if your AD server uses a unique port.

Follow Referrals (optional)

The follow referrals option is a boolean to tell active directory to follow a referral to another server on your network if the server queried knows the information you are asking for exists, but does not yet contain a copy of it locally. This option is defaulted to false.

For more information, visit: <https://technet.microsoft.com/en-us/library/cc978014.aspx>

SSL & TLS (optional)

If you need to be able to change user passwords on your server, then an SSL *or* TLS connection is required. All other operations are allowed on unsecured protocols. These options are definitely recommended if you have the ability to connect to your server securely.

LDAP Sync (optional)

The user information such as name, email, telephone is queried and synchronized. If the user's publication and group membership should also be synchronized according to certain rules, this must be specified in the mappings (array), otherwise the memberships are retained as stored in the system.

5.3 Implement Single Sign-On (SSO) using Microsoft Active Directory

5.3.1 SSO with the OAuth remote client

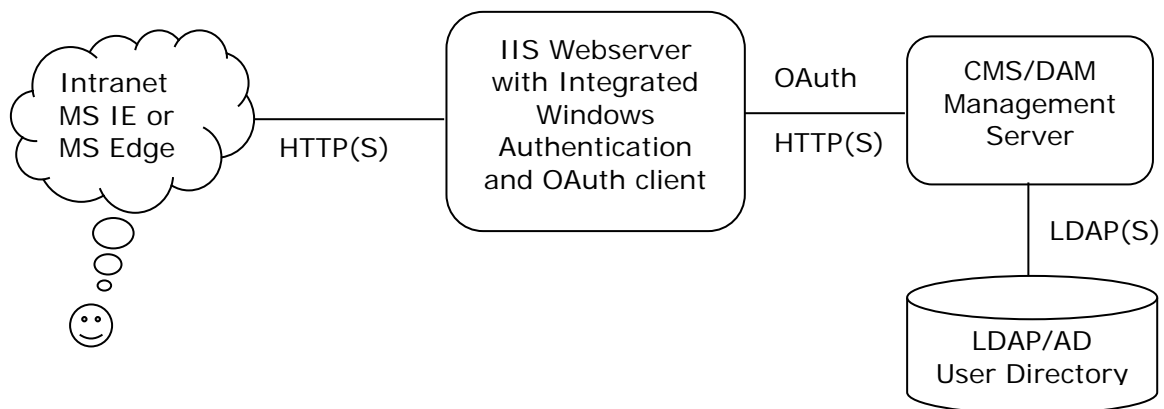
OAuth 1.0 can be used for Single Sign-On and requires the Connector module.

In order to use SSO with a Windows network, the hyper Content & Digital Asset Management Server must be operated on MS IIS or the remote webserver must be an IIS with PHP support.

The Integrated Windows authentication must be activated on the IIS webserver in order to obtain the user data for SSO with the CMS/DAM server.

It is recommended to use an internal IIS server for the authentication in the Intranet (only for internal users) and use an Apache webserver for the operation of the hyper Content & Digital Asset Management Server.

If a user accesses the remote client via HTTP(S) using MS IE or Edge browser, the remote client will receive the user ID from IIS and will forward the request to the hyper Content & Digital Asset Management Server using OAuth for the authentication of the user. The user also needs to be verified using a LDAP/AD user directory. Keep in mind that a LDAP/AD user with general read permissions is required in order to verify the user.



Requirements:

- The remote webserver must be MS IIS with Integrated Windows Authentication and PHP support.
- The OAuth remote client file in `hypercms/connector/sso/outh_remoteclient.php` need to be copied to the remote IIS server. It can be renamed.
- The users must access the remote client using MS IE or MS Edge browser.
- The OAuth configuration of the remote client must be the same as of the server, see the OAuth settings in the main configuration and the ones in the remote client file.
- A LDAP/AD user with general read permissions must be defined in the main configuration of the system for the connection to the LDAP/AD directory and the verification of users.

In order to verify the remote user, a valid LDAP/AD user account is required and need to be defined in the main configuration file in hypercms/config/config.inc.php:

```
// Define a LDAP/AD user with general read permissions
// This LDAP/AD user is only required if you want to use SSO using the OAuth
remote client
$mgmt_config['ldap_admin_username'] = "ldap_reader";
$mgmt_config['ldap_admin_password'] = "password";
```

In order to enable SSO use the following setting in the main configuration file in hypercms/config/config.inc.php:

```
// Enable (true) or disable (false) SSO
$mgmt_config['sso'] = false;

// Enable (true) or disable (false) SSO IP validation for WebDAV (do not use if
users share the same IP address)
$mgmt_config['sso_ip_validation'] = false;
```

The remote client file in hypercms/connector/sso/oauth_remoteclient.php need to be copied to the remote IIS webserver and need to use the same OAuth settings as defined in the main configuration file in hypercms/config/config.inc.php:

```
// Provide the OAuth consumer key
$mgmt_config['oauth_consumer_key'] = "hypercms";

// Provide the OAuth consumer secret
$mgmt_config['oauth_secret'] = "XMB9APEytVnxoAkLcRGqKQkxptw3rVVY";

// Define the default timezone for OAuth
$mgmt_config['oauth_timezone'] = "Europe/Vienna";
```

5.3.2 SSO with the Apache module

You need to install the mod_auth_ssapi Apache module and setup your Apache host. This solution only supports older Apache versions and does not support Apache 2.4. The mod is currently not maintained and the use of it is therefore not recommended. See: <https://sourceforge.net/projects/mod-auth-sspi/>

Sample configuration:

```
AuthType SSPI
SSPIAuth On
SSPIAuthoritative On
SSPIDomain mydomain
```

Set this to allow access with clients that do not support NTLM, or via proxy from outside. Don't forget to require SSL in this case.
For SSO the Basic Authentication must be On:

```
SSPIOfferBasic On
```

Set this if you have only one domain and don't want the MYDOMAIN\ prefix on each user name:

```
SSPIOmitDomain On
```

AD user names are case-insensitive, so use this for normalization if your application's user names are case-sensitive:

```
SSPIUsernameCase Lower
AuthName "Some text to prompt for domain credentials"
Require valid-user
```

With MS Internet Explorer or MS Edge you want to add Integrated Authentication of IE's advanced security and add your URLs to trusted zones.

And don't forget that you can also use Firefox for transparent SSO in a Windows domain: Simply go to about:config, search for network.automatic-ntlm-auth.trusted-uris, and enter the host name or FQDN of your internal application (like myserver or myserver.corp.domain.com). You can have more than one entry, it's a comma-separated list.

5.4 Workplace Integration and LDAP/AD

If the Workplace Integration is used, the digest authentication is required in order to support the Windows File Explorer and the MS Office applications, like Word, Excel, Powerpoint, and others.

Digest authentication applies MD5 cryptographic, one-way hashing with usage of nonce values to a password before sending it over the network. This option is more safe than Basic Authentication when used over HTTP. However, the usage of HTTP is not recommended and HTTPS should be used instead.

If the LDAP/AD Connector is used with the authentication of the actual user (no SSO support), the WebDAV Digest authentication can't provide the password to the LDAP/AD Connector and therefore will fall back to the last synchronized password available in the system. In case a user changes the password of his Windows account or in the LDAP directory, the password will not be synchronized automatically when performing a logon in the Windows File Explorer (WebDAV client).

In this case, the solution is to authenticate in the Web GUI in order to synchronize the user data including the password with LDAP/AD.

If Single Sign-On (SSO) is used, the authentication of a user will be performed by the Integrated Windows Authentication. SSO therefore requires an Admin LDAP/AD user for the connection to the user directory.

If an Admin LDAP/AD user has been defined for SSO, see main configuration setting `$mgmt_config['ldap_admin_username']` and `$mgmt_config['ldap_admin_password']`, the synchronization of the user data excluding the password will not be an issue when using the Windows File Explorer, since the username and password will be available for the bind of the Admin LDAP/AD user.

This however does not include the synchronization of the password of the user, since LDAP/AD does not store or provide a clear text password.

6 Assetbrowser

6.1 Introduction

The Assetbrowser enables the integration with other third-party systems, e.g. a content management system (CMS), like Wordpress, Typo3, SiteCore and any other. The CMS is the master application and uses the system as its asset repository.

The CMS editor can select images and video files stored in the hyper Content & Digital Asset Management Server while working directly in the third-party CMS.

Your organization can maintain better control over your digital assets by accessing them from the hyper Content & Digital Asset Management Server – giving other content contributors in Marketing and Sales outside the web ecosystem access to the exact same files. This means that there will be no duplication of assets and no chasing down latest versions.

Requirements: The Connector module is required.

6.2 Configuration

The Assetbrowser requires the connector module. You can configure the Assetbrowser in the module connector/assetbrowser/config.inc.php.

Here you only need to specify the callback function of your third-party application that will use the returned data of the Assetbrowser:

```
// Callback function of master application (to be called after asset has
// been selected in the assetbrowser).
// The JS function returnMedia will use the callback function to return
// the media information to the master application.
// The following parameters can be returned:
// result .... JSON encoded string with all parameters OR single
// parameters
// link .... wrapper link
// name ... asset name
// width ... width in pixel
// height ... height in pixel
// datetime ... date and time in format YYYY-MM-DD HH:MM
// filesize ... file size in KB

$assetbrowser_config['callback'] = "callback(result)";
```


6.3 Open the Assetbrowser

In order to open the Assetbrowser, you can use the REST API. The Assetbrowser will use an existing user account of the system. You need to create a user account that will only be used by the asset browser. You can also define the permissions of the users by group membership. See the Administrators Guide for more details regarding user management.

The following parameters are supported

userhash ... hashcode of the Assetbrowser user

objecthash ... hashcode of a specific object or empty, the object hash is part of the wrapper link: https://yourdomain.com/cms_dev/?wl=8pe71j28p318ogu7

filter ... apply object filter ("comp", "image", "document", "video", "audio", "flash", "compressed", "binary", "folder")

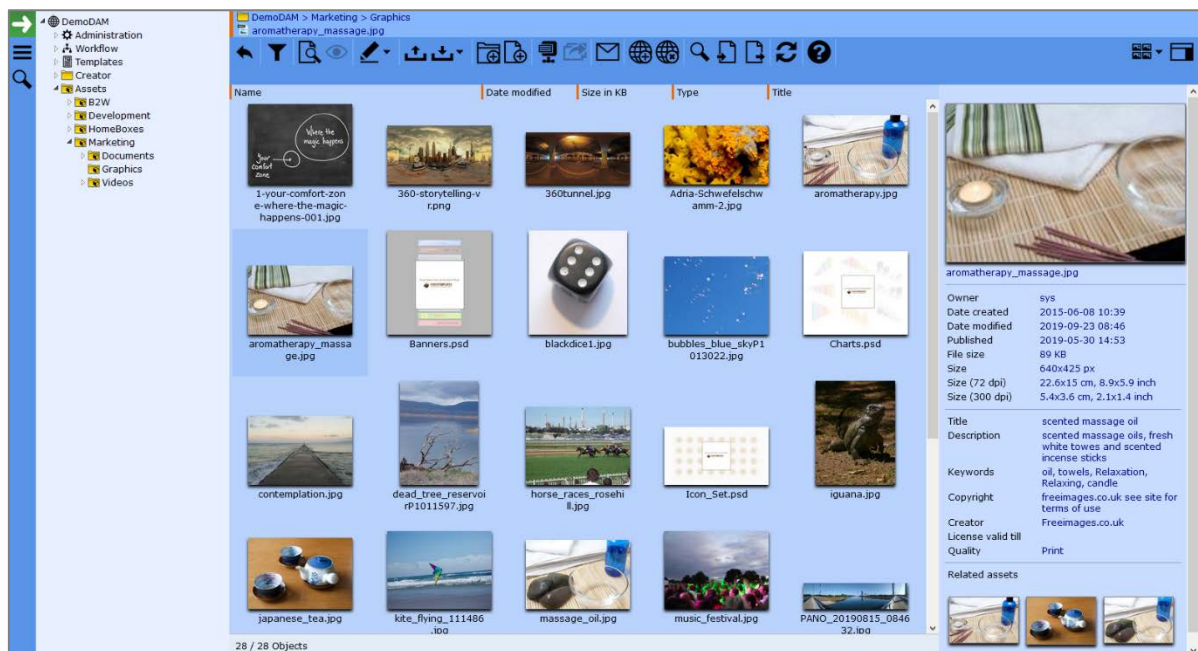
instance ... name of a specific instance, only if instances are used

Example URL to open the Assetbrowser

<https://yourdomain.com/hypercms/connector/assetbrowser/?userhash=20e634813eb278c383c0544bb098ff8&objecthash=8pe71j28p318ogu7&filter=image>

You can integrate the Assetbrowser by opening the URL in a new browser window or by using an iframe in the third-party application with the URL as source.

By pressing the select button the asset will be passed to the callback function of the third-party system.



7 YouTube Connector

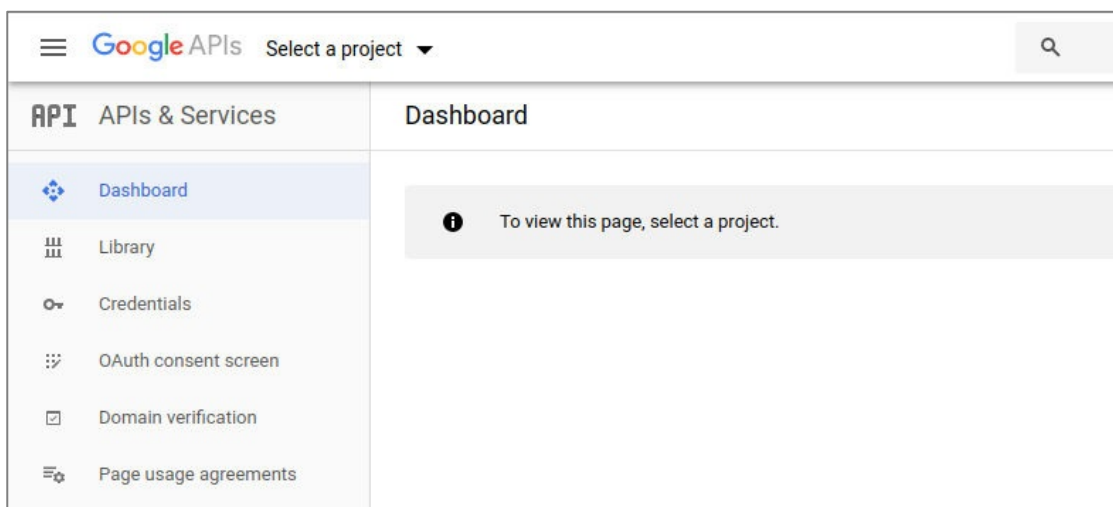
The YouTube Connector can be used to upload and publish videos stored in the system directly to a YouTube account.

User will be able to upload a video and data (title, description, tags, privacy setting, category) to their YouTube account.

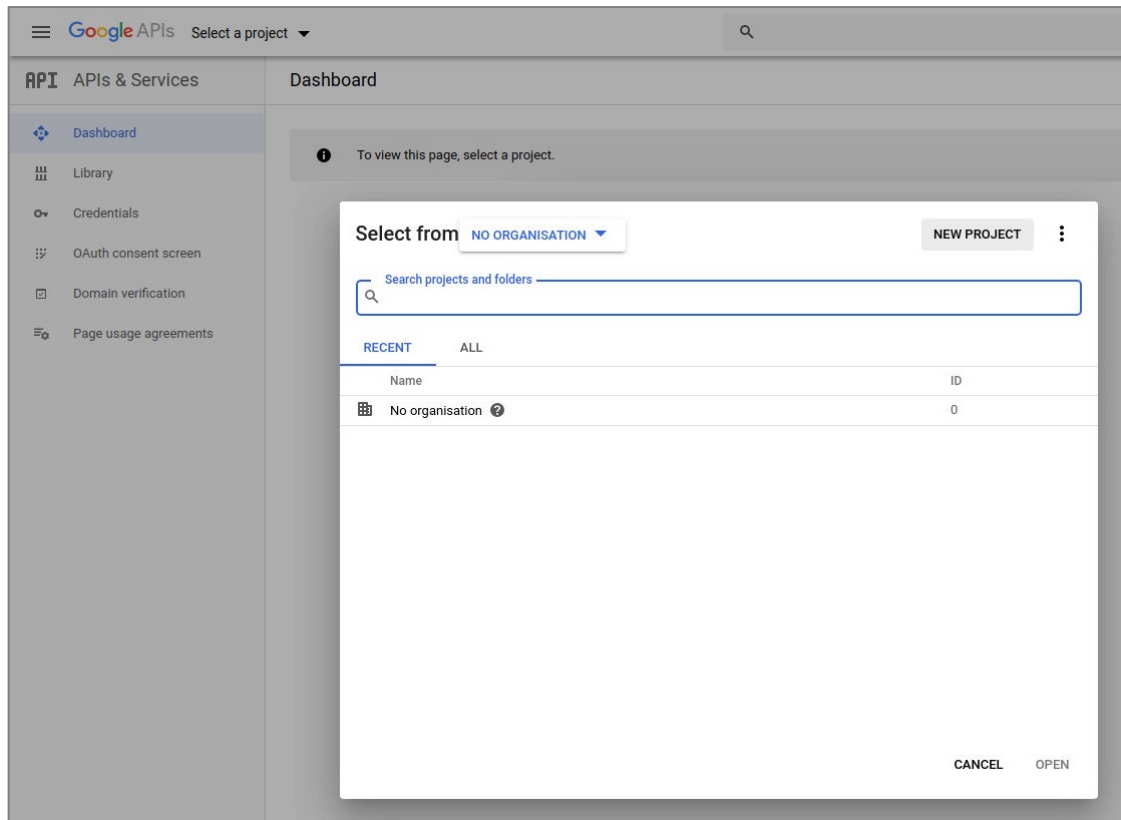
Requirements: The Connector module is required and the "Youtube upload" need to be enabled in the publication settings, see Administrators Guide.

In order to use the YouTube Connector you must have a Google account and activate the YouTube Data API v3:

Login to Google and visit your dashboard: <https://console.developers.google.com>



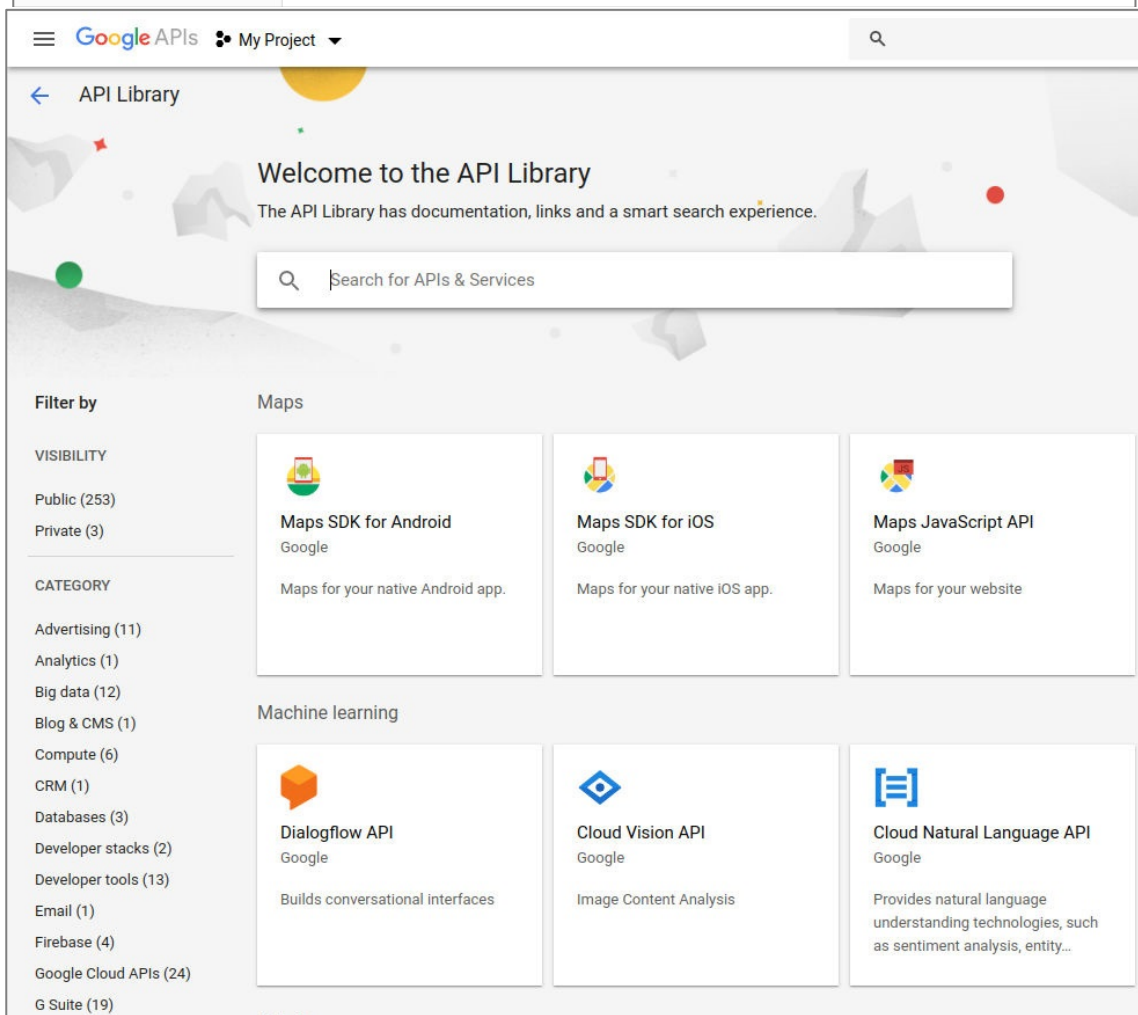
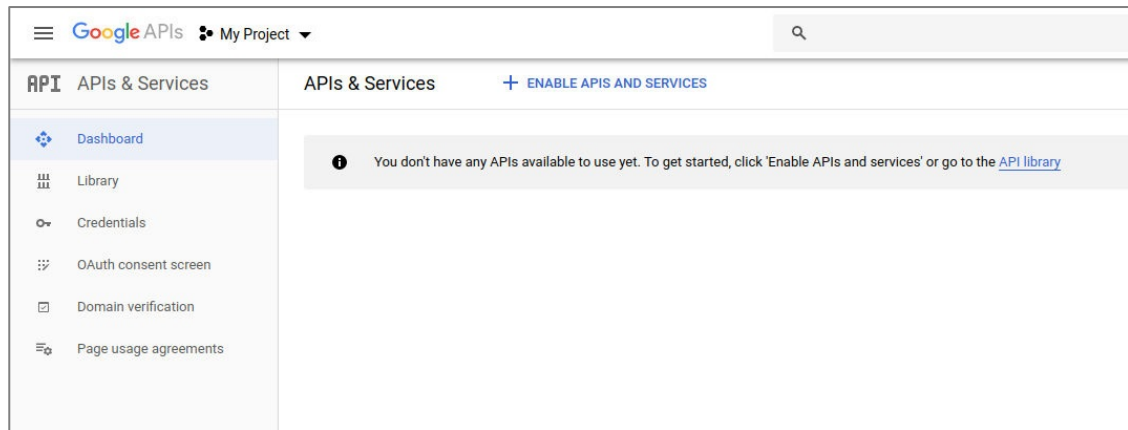
Create a new project (name and ID don't matter, they are for your reference)



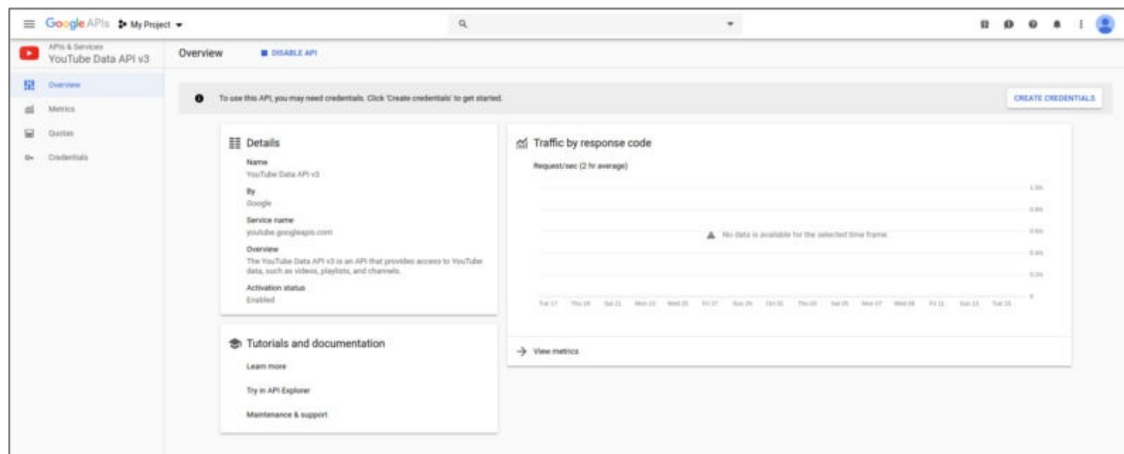
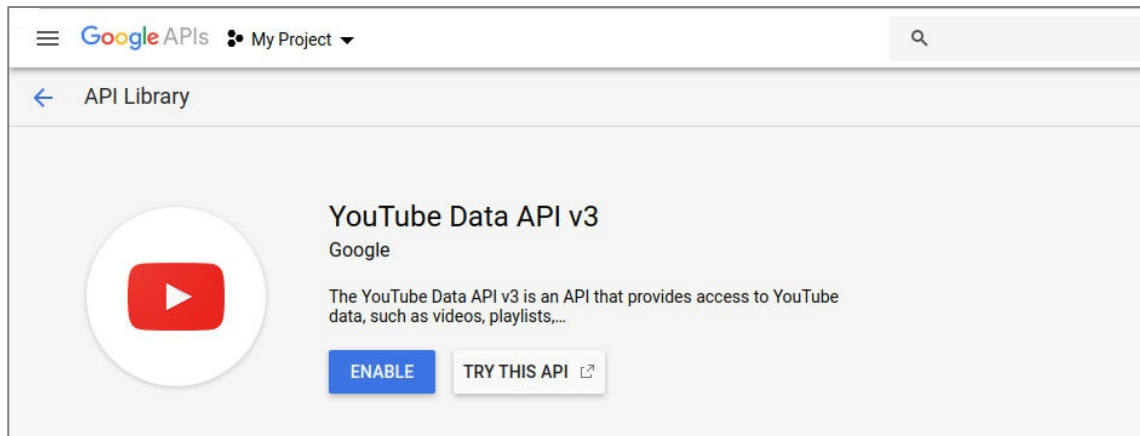
The screenshot shows the 'New Project' form in the Google APIs console. The form has the following fields and options:

- Project name ***: A text input field containing 'My Project'.
- Project ID**: A text input field containing 'confident'. Below it, a message says 'It cannot be changed later.' with an 'EDIT' link.
- Organisation**: A text input field containing 'montala.com'.
- Location ***: A text input field containing 'yourdomain.com' with a 'BROWSE' button next to it.
- Parent organisation or folder**: A label below the location field.
- Buttons**: 'CREATE' and 'CANCEL' buttons at the bottom.

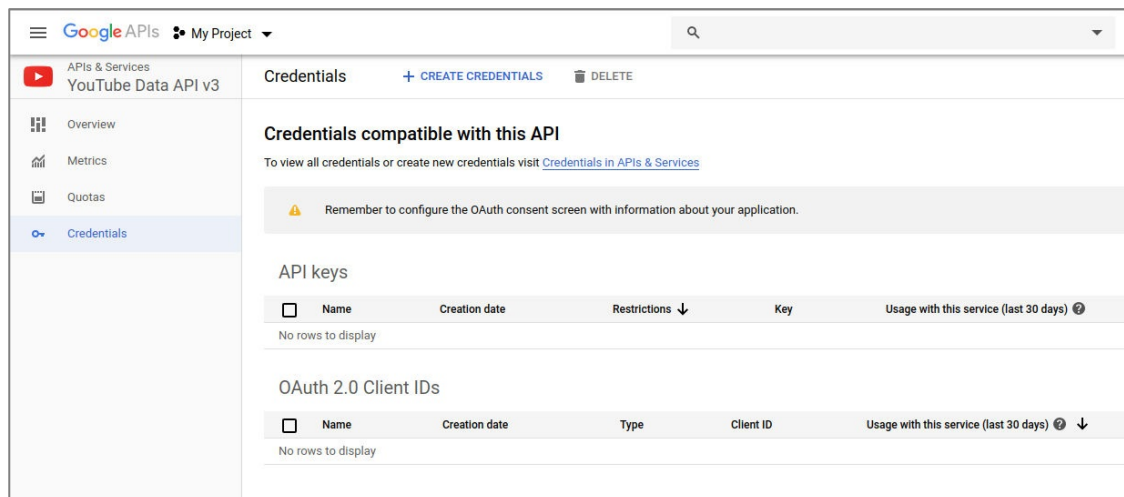
Click **“ENABLE API'S AND SERVICES”** and scroll down to **“YouTube Data API”** option



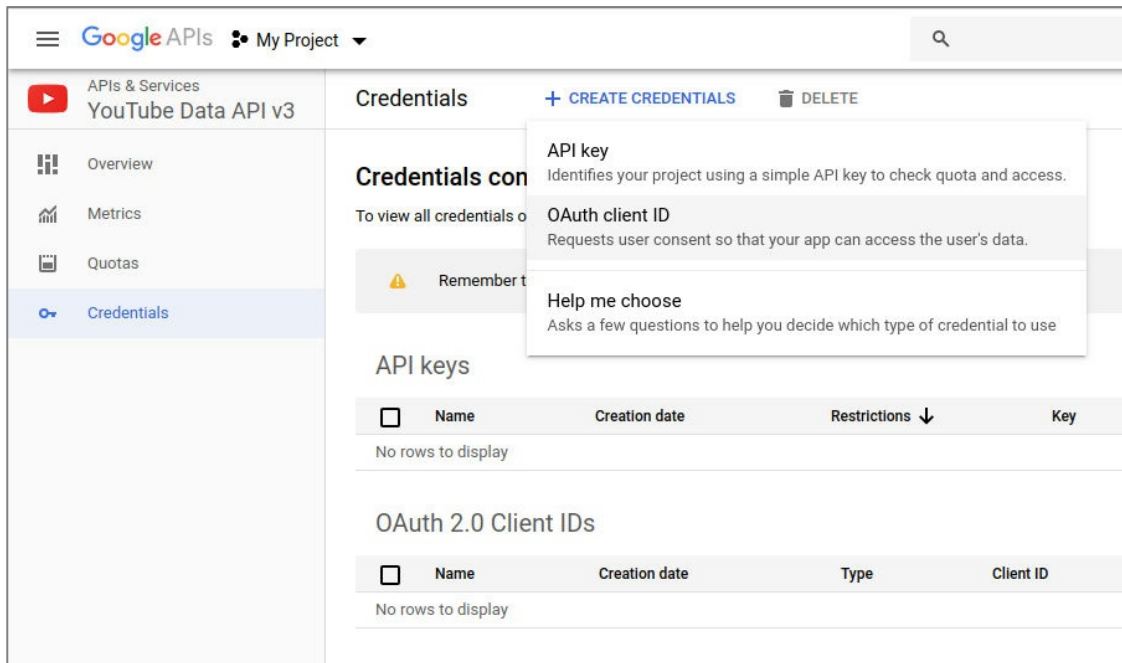
Click **"Enable"** in order to activate the API



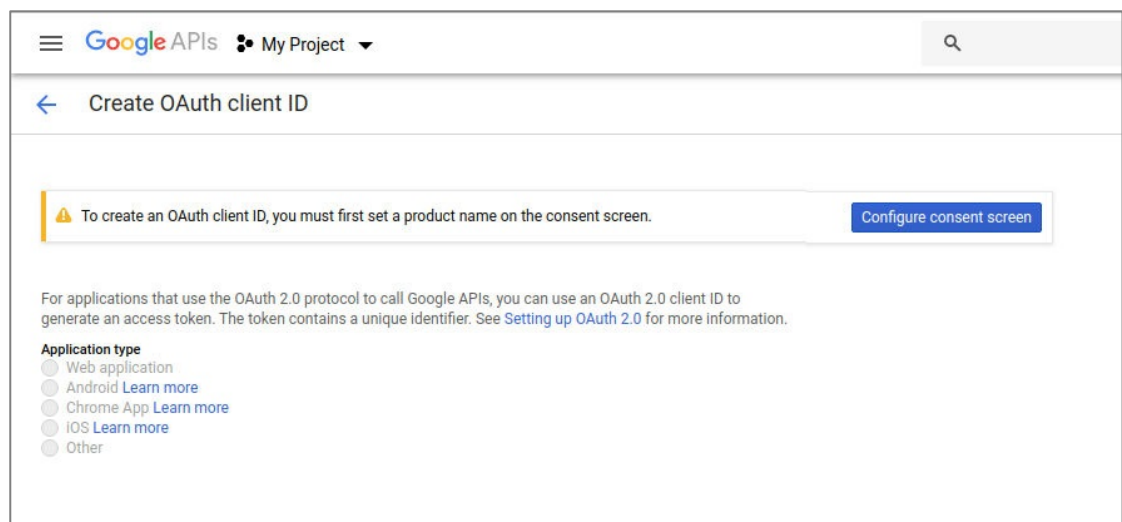
Select **"Credentials"** on the left hand side



Then click **"CREATE CREDENTIALS"** AND select **"OAuth client ID"** from the dropdown menu



You will be presented with the **“Create OAuth client ID”** page. In order to continue, you first need to click the blue button **“Configure consent screen”**.



Fill in the relevant information and save

Google APIs

My Project

API APIs & Services

Dashboard

Library

Credentials

OAuth consent screen

Domain verification

Page usage agreements

OAuth consent screen

Before your users authenticate, this consent screen will allow them to choose whether they want to grant access to their private data, as well as give them a link to your terms of service and privacy policy. This page configures the consent screen for all applications in this project.

Application type

☒ **Public**
Any Google Account can grant access to the scopes required by this app.
[Learn more about scopes](#)

☐ **Internal**
Only users with a Google Account in your organisation can grant access to the scopes requested by this app.

Verification status
Not published

Application name ?
The name of the app asking for consent

Application name

Application logo ?
An image on the consent screen that will help users recognise your app

Local file for upload

Browse

All videos uploaded via the videos.insert endpoint from unverified API projects created after 28 July 2020 will be restricted to private viewing mode by YouTube. To lift this restriction, each API project must undergo an audit to verify compliance with the Terms of Service.

In order to be able to upload videos to a YouTube account the following scopes are required:

<https://www.googleapis.com/auth/youtube.upload>

<https://www.googleapis.com/auth/youtube>

<https://www.googleapis.com/auth/youtubepartner>

<https://www.googleapis.com/auth/youtube.force-ssl>

APIs & Services | Edit app registration

OAuth consent screen — **Scopes** — Optional info

Scopes express the permissions you request users to authorize for your app and allow your project to access specific types of private user data from their Google Account. [Learn more](#)

If you add sensitive (🔒) or restricted (🔒) scopes, like scopes that let you access a user's emails or contacts, you will need to submit your app for verification. [Learn more](#)

[ADD OR REMOVE SCOPES](#)

Your non-sensitive scopes

API	Scope	User-facing description
	.../auth/userinfo.email	View your email address
	.../auth/userinfo.profile	See your personal info, including any personal info you've made publicly available
	openid	Associate you with your personal info on Google
YouTube Data API v3	.../auth/youtube.upload	Download your public YouTube videos

Update selected scopes

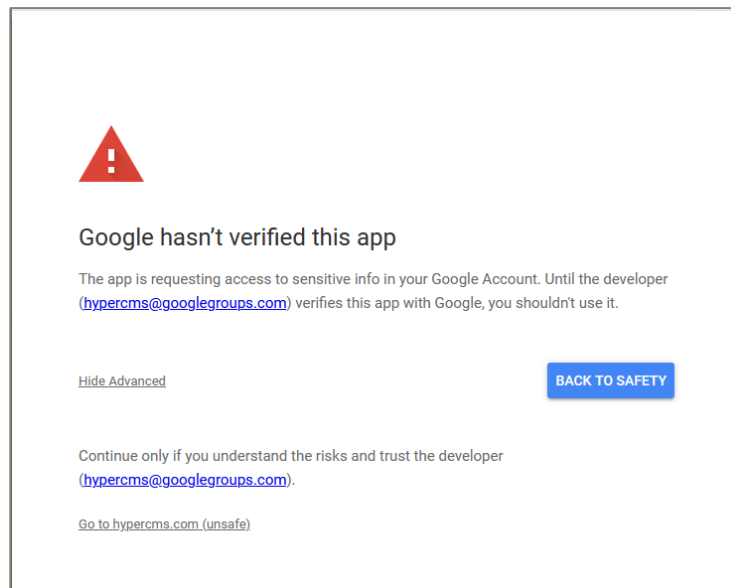
Only scopes for enabled APIs are listed below. To add a missing scope to this screen, find and enable the API in the [Google API Library](#) or use the Pasted Scopes text box below. Refresh the page to see any new APIs you enable from the Library.

Filter table

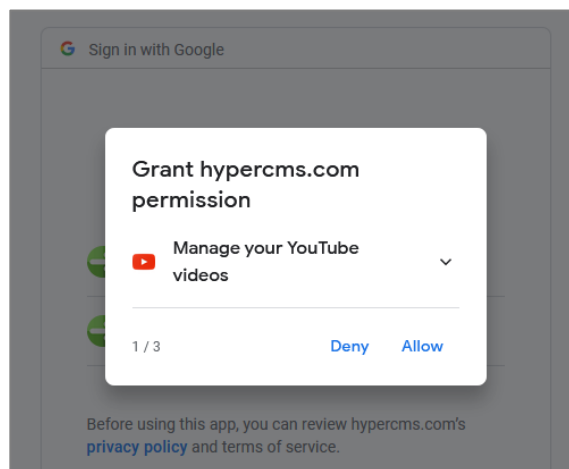
API	Scope	User-facing description
<input type="checkbox"/>	.../auth/userinfo.email	View your email address
<input type="checkbox"/>	.../auth/userinfo.profile	See your personal info, including any personal info you've made publicly available
<input type="checkbox"/>	openid	Associate you with your personal info on Google
<input checked="" type="checkbox"/>	YouTube Data API v3 .../auth/youtube.readonly	View your YouTube account
<input checked="" type="checkbox"/>	YouTube Data API v3 .../auth/youtube	Manage your YouTube account
<input checked="" type="checkbox"/>	YouTube Data API v3 .../auth/youtube.force-ssl	See, edit, and permanently delete your YouTube videos, ratings, comments and captions
<input checked="" type="checkbox"/>	YouTube Data API v3 .../auth/youtubepartner	View and manage your assets and associated content on YouTube
<input type="checkbox"/>	YouTube Data API v3 .../auth/youtubepartner-channel-audit	View private information of your YouTube channel relevant during the audit process with a YouTube partner
<input checked="" type="checkbox"/>	YouTube Data API v3 .../auth/youtube.upload	Manage your YouTube videos
<input type="checkbox"/>	YouTube Data API v3 .../auth/youtube.channel-memberships-creator	See a list of your current active channel members, their current level, and when they became a member

Rows per page: 10 | 1 - 10 of 12

Alternatively you can use the YouTube API without an audit by Google. In this case the YouTube API will be displayed as unverified. The users that want to connect it to their YouTube account need to continue with “**Advanced**” and “**Go to domain.com (unsafe)**”.



The users need to grant all required permissions in order to upload videos to YouTube.



You will be then redirected back to the “**Create OAuth client ID**” page. Select “Web application” under “**Application type**” and fill in the “**Authorized JavaScript origins**” with your system base URL (use the URL defined in \$mgmt_config['url_path_cms'] in the main configuration of your installation) and the redirect URL with the callback URL (use the URL defined in \$mgmt_config['url_path_cms'] and add /connector/youtube/) and click “**Create**”.

Google APIs

My Project

←

Create OAuth client ID

For applications that use the OAuth 2.0 protocol to call Google APIs, you can use an OAuth 2.0 client ID to generate an access token. The token contains a unique identifier. See [Setting up OAuth 2.0](#) for more information.

Application type

☒ Web application

☐ Android [Learn more](#)

☐ Chrome App [Learn more](#)

☐ iOS [Learn more](#)

☐ Other

Name ?

Web client 1

Restrictions

Enter JavaScript origins, redirect URIs or both [Learn more](#)

Origins and redirect domains must be added to the list of authorised domains in the [OAuth consent settings](#).

Authorised JavaScript origins

For use with requests from a browser. This is the origin URI of the client application. It cannot contain a wildcard (https://*.example.com) or a path (https://example.com/subdir). If you're using a non-standard port, you must include it in the origin URI.

https://~~example.com~~

https://www.example.com

Type in the domain and press Enter to add it

Authorised redirect URIs

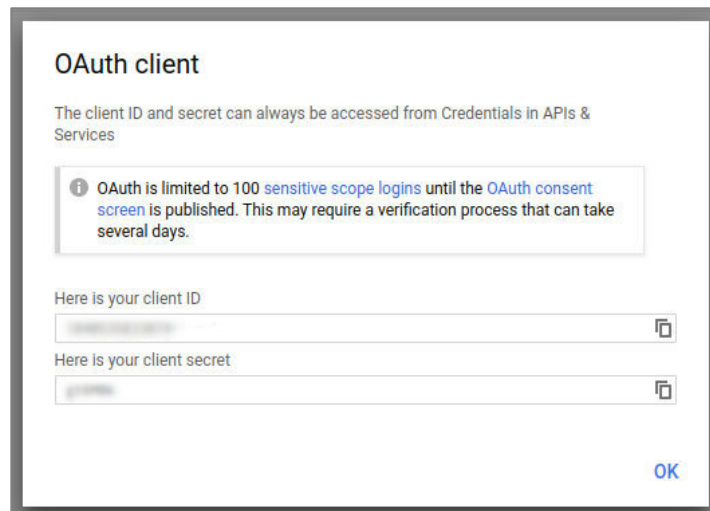
For use with requests from a web server. This is the path in your application that users are redirected to after they have authenticated with Google. The path will be appended with the authorisation code for access. Must have a protocol. Cannot contain URL fragments or relative paths. Cannot be a public IP address.

https://www.example.com

Type in the domain and press Enter to add it

CreateCancel

You will then be presented with a screen showing your newly created “**client ID**” and “**client secret**”.



The screenshot shows the 'OAuth client' creation interface. At the top, it says 'OAuth client' and 'The client ID and secret can always be accessed from Credentials in APIs & Services'. Below this is a warning box: 'i OAuth is limited to 100 sensitive scope logins until the OAuth consent screen is published. This may require a verification process that can take several days.' There are two input fields: 'Here is your client ID' and 'Here is your client secret', both with copy icons. An 'OK' button is at the bottom right.

Note down the Project ID, client ID and secret, then enter these details into the main configuration file of your installation in `hypercms/config/config.inc.php`:

```
// YouTube integration (requires Connector module)
// Please provide Google API credentials in order to upload videos to YouTube
$mgmt_config['youtube_oauth2_client_id'] = "";
$mgmt_config['youtube_oauth2_client_secret'] = "";
$mgmt_config['youtube_appname'] = "";
```

8 Adobe Creative Cloud Connector

If you want to use your assets in Adobe Creative Cloud, you can directly access your assets and add them to your Adobe content using the LinkrUI Connector.

The LinkrUI Connector is a plugin, which allows you to use assets from the hyper Content & Digital Asset Management Server in your Adobe content. The Connector can be used in InDesign, Photoshop, Illustrator, Premiere and After Effects. When your Adobe file is ready, you can use the connector to directly upload it to your system.

Using the connector is an alternative to the Workplace Integration. You can search and filter your assets, access your assets and use them all from within the connector.

Requirements: The Connector module is required and the “RESTful API” must be enabled for the publications, see Administrators Guide.

8.1 Enable the LinkrUI Connector

Contact our Customer Support if you're interested in the LinkrUI Connector:
support@hypercms.net

8.2 Download and install the Connector

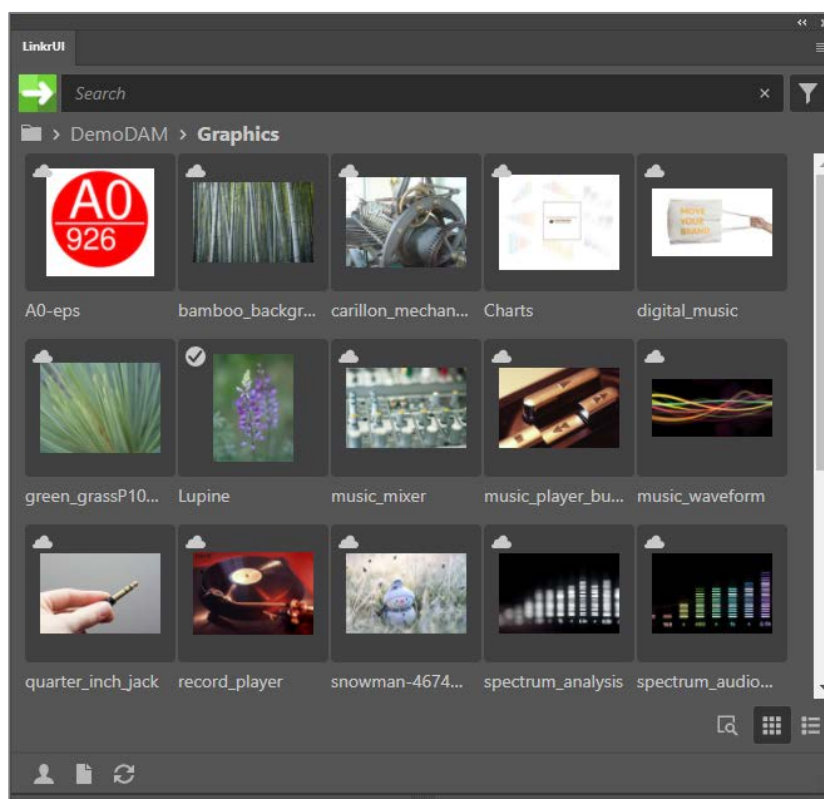
The Adobe Connector provided by LinkrUI is a paid solution.

1. Click one of the available download links below to download the Adobe CC Connector from LinkrUI.
 - For macOS:
<https://cdn.linkrui.com/installers/current/mac/hypercms/LinkrUI.dmg>
 - For Windows:
<https://cdn.linkrui.com/installers/current/windows/hypercms/LinkrUI.zip>
2. Unzip the ZIP file for Windows only.
3. Install the installation file on your machine.
4. Open your Adobe software.
5. Make sure the LinkrUI panel is opened. Go to **Window > Extensions > LinkrUI** to enable it. You'll be asked if you already have an activation code.
6. Click **Yes** to proceed.
7. Enter your name in the “**Name**” field.
8. Enter your email address in the “**Email**” field.
9. Enter the activation code that has been provided you with in the “**Activation Key**” field.
10. Click the “**Activate**” button to activate your license.

You can now start using the LinkrUI Connector.

8.3 Using the Connector

By default, the plugin shows all the assets you have access to. You can search and filter the assets to find what you're looking for.



8.3.1 Quick Start Guide

1. Make sure the LinkrUI panel is opened. Go to **Window > Extensions > LinkrUI** to enable it. The extension will ask you to log in to your hyper Content & Digital Asset Management Server account.
2. Click the **"Authenticate"** button. A window should open to authenticate. If this is not the case copy the full URL from the connector screen and visit the URL in your browser.
Log in to the hyper Content & Digital Asset Management Server the usual way. If successful, the assets folders will be loaded.
Try logging in again if the authentication was not successful.
3. Navigate, search and your assets stored in the system.
4. Drag and drop an asset to your Adobe content. Depending on the Adobe software you're using there may be additional steps to place the image into your actual content.

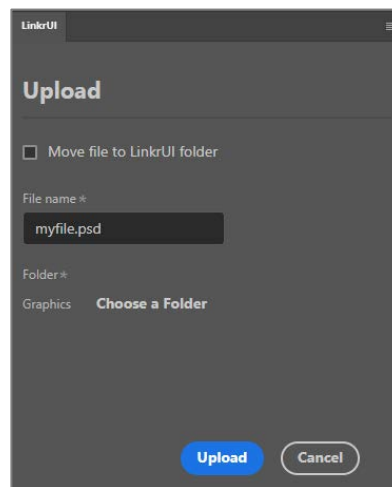
8.3.2 Add assets to your Adobe Content

1. Browse the folders in the LinkrUI panel. Click a folder to open it and navigate to the asset you want to add to your Adobe content. You can also use the available filters and/or search bar to find the asset you're looking for.
2. Drag and drop an asset to your Adobe content. Depending on the Adobe software you're using there may be additional steps to place the image into your actual content.

8.3.3 Upload your Adobe files



With the LinkrUI plugin you can add your assets to your Adobe files. You can also upload your Adobe file to your system without having to leave the Adobe suite. Follow the steps below to directly upload your Adobe file to your hyper Content & Digital Asset Management Server.

1. Make sure the Adobe file you're working on is saved.
2. Click the ☰ button and click **"Upload New Asset"**.
3. Change the name of your new asset in the **"Name"** field (optional).
4. Click **"Choose a Folder"**, select the folder you want to upload the asset to and click **"Select"**.
5. Click the **"Upload"** button to upload your Adobe file as a new asset to your hyper Content & Digital Asset Management Server.




8.3.4 Change the view

There are two types of views available in the asset overview panel in the Connector:

- **Grid View**
The grid view shows the thumbnails of the assets and a small part of the asset title. Click the  button to enable the grid view.
- **List View**
The list view shows more asset information and can show more assets compared to the grid view. Click the  button to enable the list view.

8.3.5 Search and filter Assets inside the Connector

Click the  button at the top of the panel to filter the assets. Select the appropriate values and click **"Show results"** to filter your assets.

You can also use the search bar at the top of the panel to find the asset(s) you're looking for.

8.3.6 View Asset Metadata in the Connector


Click the asset of which you want to view the basic metadata.

1. Click the  button. A **"Details"** panel will appear.

2. Scroll down in the **"Details"** panel to see all the available properties and other information of the asset
3. Select another asset in the plugin to view the asset detail data of that asset.

8.4 Other features of the LinkrUI Connector

Explore all the other options and features that are available in the LinkrUI connector below.

1. Make sure the LinkrUI panel is opened. Go to **Window > Extensions > LinkrUI** to enable it.
2. Click the  button and check one of the options below.
 - **Login**
Click **"Login"** to log in to the LinkrUI connector when you're not logged in. Enter your hyper Content & Digital Asset Management Server URL in the **"Server URL"** field and click **"Authenticate"**. A browser window should open to authenticate. Log in to your system the usual way. If successful, the assets will be loaded.
Try logging in again if the authentication was not successful.
 - **Upload New Asset**
Upload your Adobe file as a new asset in your system.
 - **Preferences**
Scroll through the preferences section to see all the available preferences.
 - **Asset size**
Drag the slider to set the display size of the thumbnails in the panel of the connector. The thumbnail preview shows you how big the thumbnail will be.
 - **Local Asset Cache Purge Interval**
This setting allows you to set after how much time your cached assets will be cleared. This means that the assets that are cached locally will be removed.
We advise using an interval that equals the time of the project you're working on in Adobe. If your project will take a week set the interval to **"After a week"**.
 - **Local Asset Cache Folder**
Click **"Choose"** to select the folder on your local machine where the cached assets can be stored.
Click **"Clear Cache"** to empty the above folder and to remove all cached assets.
 - **Link Updates**
This option can decide what happens when there's a new version of the assets that are used in your Adobe file.
For example, when you're using a Photoshop file in your InDesign file. If you upload your InDesign file to the cloud and another user open this InDesign file and modifies the Photoshop file. With the **"Link Updates"** option you can decide what happens the next time you open this InDesign file again.

Auto update - This will ensure that the latest version of the assets in your Adobe file will be retrieved automatically. In the above example the new Photoshop file will be retrieved.

Prompt - You'll be asked whether you want to retrieve the latest version of your assets used in your Adobe file. In the above example you'll be asked if you want to retrieve the latest version of the Photoshop file.

Do nothing - The latest version of the assets used in your Adobe file won't be retrieved. In the above example the latest version of the Photoshop file won't be retrieved and you won't be asked to retrieve it.

- **Log out**
Click "**Log out**" to disconnect the LinkrUI connector from your system.
- **Check Connections**
Use this option if you're experiencing issues with the LinkrUI connector to make sure the connector is connected correctly.
- **Uploads**
Click this option to see the Adobe files that are currently being uploaded to your system. If the **Upload** screen is empty there are currently no files being uploaded.
- **Refresh**
This option will refresh the panel of the LinkrUI connector. Click this option if you're not seeing the thumbnails in the connector or if you're having other issues.
- **Logging**
This option allows you to set the level of logging. Click the "**Logging**" dropdown, select one of the available options and click "**OK**" to save the option.
- **About**
Click this option to see which version of the LinkrUI connector currently is installed.
- **Contact Support**
Click this option when you need technical assistance with the LinkrUI connector.

3. Click "**Reset Preferences**" to go back to the default setup of the LinkrUI connector. This action can't be undone.

9 Microsoft Office Connector

The Microsoft Office connector allows users to search and use assets from your hyper Content & Digital Asset Management Server from within Word, Excel, and Powerpoint. By using your system as the source for all brand and creative content, your design team can focus on content creation instead of version management.

Keep in mind that the Workplace Integration is required if you want to use your assets in Outlook. The Workplace Integration enables you to insert any kind of asset as an attachment (supporting download and access links) in your e-mail message. Microsoft Outlook as well as other e-mail clients are supported.

For more information about the Workplace Integration, please take a look at the Users Guide or Installation Guide.

Requirements: The Connector module is required and the "RESTful API" must be enabled for the publications, see Administrators Guide.

9.1 About the integration

The Microsoft Office connector is compatible with Microsoft Office Suite or Microsoft 365 Online and works with Microsoft Word, Excel, and Powerpoint.

Users will be able to search for assets they have permission to, using smart filters, file types, or content.

9.2 Enable the Microsoft Office Connector

Contact our Customer Support if you're interested in the Microsoft Office Connector: support@hypercms.net

9.3 Install the Microsoft Office Connector

The Microsoft Office Connector provided by LinkrUI is a paid solution.

You can install the Microsoft add-ons from the Microsoft Store:

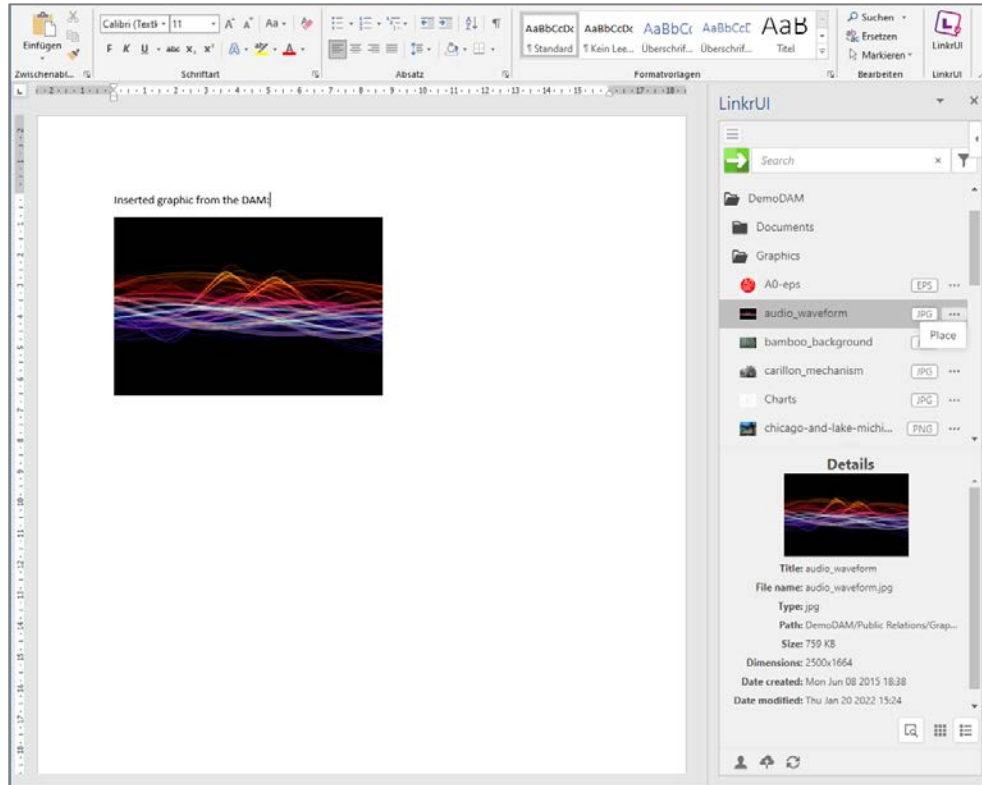
<https://appsource.microsoft.com/en-us/product/office/WA200004791?tab=Overview&exp=ubp8>

Learn more about how to manage and install Microsoft add-ons:

<https://support.microsoft.com/en-us/office/view-manage-and-install-add-ins-for-microsoft-365-programs-16278816-1948-4028-91e5-76dca5380f8d>

9.4 Using the Microsoft Office Connector

Once the connector is successfully installed and connected with your hyper Content & Digital Asset Management Server, you will be able to access your assets within your Word, Excel, and Powerpoint documents.



Once you've opened a document you will see "**LinkrUI**" appear under "**Add Ons**" in the upper right hand corner and here you can access all of your assets.

1. Open a document.
2. Click "**Add Ons**" in the upper right corner, then "**LinkrUI**".
3. Provide your details and the activation code (only once)
4. Login with your hyper Content & Digital Asset Management Server user account.
5. You should now be able to access all of your assets.
6. You can search by file name, file type, file contents/tags, and smart filters. Once you have found the asset(s) you are looking for, select the asset and place it into your document.
7. Alternatively, you can select an Office file, such as a template or existing presentation from your system and open it in another tab or window. You can make edits to the document before updating the version in the DAM.
8. When finished, you can upload the file into your system as a new asset or as an updated version. Click on the menu item and select upload. To upload as a new version, keep the filename the same as the original file in your system.

10 Legal reference / flag

10.1 Questions and suggestions

For advanced questions and suggestions, please contact the support.

hyperCMS Support:

www.hypercms.com

support@hypercms.com

10.2 Imprint

Responsible for the content:

hyperCMS

Content Management Solutions GmbH

Rembrandtstr. 35/6

A-1020 Vienna – Austria

office@hypercms.com

<http://www.hypercms.com>

10.3 Legal information

The present product information is based on the version of the program, which was available at the time the document was composed.

The maker reserves the rights of modifications and corrections of the program.
Errors and misapprehension accepted.

© 2024 by hyperCMS Content Management Solutions