



**EOSC-hub**

## CINES eTDR User guide

---



EOSC-hub receives funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No. 777536.

[www.eosc-hub.eu](http://www.eosc-hub.eu)



## Index

Transferring data using iRODS.....	4
Install the iRODS client with provided packages.....	4
User accounts.....	4
Inializing a connection.....	4
Build a submission package.....	4
Tranferring packages.....	4
Viewvng all metadata associated with an image.....	5
Monitoring the progress of the workflow.....	5
Testing the connection.....	6
Initializing a connection.....	6
Transferring a test file.....	6
Monitor deposits using HTTP REST API.....	7
REST API Interface description.....	7
Header parameters.....	7
Authentication.....	7
Internationalization.....	7
REST API parameters.....	7
List of packages by status.....	7
Search by depositId.....	9
Examples.....	10
List of packages by status.....	10
List of rejected packages since a date.....	10
Search by depositId.....	11
Searching data using HTTP REST API Version 5.....	12
REST API Interface description.....	12
REST API parameters.....	12
Request for a Full-Text search in OCR results.....	12
Request for a search in metadata.....	14
Examples of search requests.....	16
Response structure.....	18
Retrieve the indexed data associated to an image.....	26
Get the image.....	26
Examples of calls on the REST API.....	27





Full-text search in OCR results only .....	27
Full-text search in metadata only with criteria on metadata.....	27
Full-text search in metadata and OCR results .....	27
Retrieve the indexed data associated to an image .....	27
Get the image .....	27
Appendix.....	28
Create a submission information package .....	28
Create a file called sip.xml .....	28
Create a directory containing the sip.xml and a directory DEPOT .....	28
Then create the tar file .....	29



## Transferring data using iRODS

Install the iRODS client with provided packages

Go to <https://packages.irods.org>

Choose the appropriate distribution and follow the instructions.

### User accounts

an iRODS user account has been created for CERN testing purposes :

**cernftp**

### Inializing a connection

Once you have properly done the installation of the iRODS client, you will be prompted, the first time, to answer these questions after running the command **init**.

**\$ init**

One or more fields in your iRODS environment file (irods\_environment.json) are missing; please enter them.

Enter the host name (DNS) of the server to connect to: ariane.cines.fr

Enter the port number: 1247

Enter your irods user name: cernftp

Enter your irods zone: CINES

Those values will be added to your environment file (for use by other iCommands) if the login succeeds.

Enter your current irods password: *mypassword (has been sent to you on request)*

Several files under a directory called .irods should be created. One is a token containing your credentials (.irodsA), and another one contains the connection parameters with the values you entered (irods\_environment.json).

Thus, the next time you connect, you wont be prompted to enter the connection information.

### Build a submission package

see appendix

### Transferring packages

The packages must be transfered into the user home directory :

`/CINES/home/cernftp`

The transfer of files can be performed by one of the command `iput` or `irsync`.

The option `-K` (integrity check, it calculates and verifies the checksum ) must be used.



Single transfer with iput :

```
$ iput -K CERN_DATA.tar  
$ ils
```

Multiple transfer with irsync :

eg. you want to transfer all files from your local dir mydir/

mydir/

CERN\_DATA\_01.tar

CERN\_DATA\_02.tar

```
$ irsync -rK mydir i:/CINES/home/username
```

## Viewing all metadata associated with an image

```
$ imeta ls -d CERN_DATA_01.tar  
attribute: ID # eg. CERN_DATA_01  
attribute: EXTENSION # eg. tar  
attribute: OWNER # eg. cernftp  
attribute: STATUS # eg. TRANSFER_OK, ERROR  
attribute: PID # persitent identifier provided by EUDAT EPIC service  
[note that the PID will be added in the future]
```

## Monitoring the progress of the workflow

```
$ imeta ls -d CERN_DATA_01.tar STATUS  
attribute: STATUS  
value: TRANSFER_OK # possible values are TRANSFER_OK, REPLICIA_OK, ERROR
```





## Testing the connection

### Initializing a connection

Once you have properly done the installation of the iRODS client, you will be prompted, the first time, to answer these questions after running the command **init**.

```
$ init
```

```
One or more fields in your iRODS environment file (irods_environment.json) are missing; please enter them.
```

```
Enter the host name (DNS) of the server to connect to: ariane.cines.fr
```

```
Enter the port number: 1247
```

```
Enter your irods user name: eudat
```

```
Enter your irods zone: CINES
```

```
Those values will be added to your environment file (for use by other iCommands) if the login succeeds.
```

```
Enter your current irods password: eudatpwd
```

Several files under a directory called `.irods` should be created. One is a token containing your credentials (`.irodsA`), and another one contains the connection parameters with the values you entered. Thus, the next time you connect, you won't be prompted to enter the connection information.

### Transferring a test file

```
$ ils
```

```
/CINES/home/eudat
```

```
$ iput mytestFile
```

```
$ ils
```

```
/CINES/home/eudat
```

```
mytestFile
```

```
$ iget mytestFile
```

```
$ irm -f mytestFile
```



## Monitor deposits using HTTP REST API

Deposits can be monitored using a HTTP REST API. This section will describe the available parameters in requests and responses. Examples of use will also be provided at the end of this section.

### REST API Interface description

The web service is available at the following URLs:

Function	URL	Protocol
Retrieve a list of packages by status	https://herbarium.cines.fr/monitoring/rest/packages/status/{status}	HTTP GET
Search a package by its deposit identifier	https://herbarium.cines.fr/monitoring/rest/packages/{depositId}	HTTP GET

### Header parameters

#### Authentication

To use the REST API, you need a basic HTTP authentication. In consequence, you must specify the HTTP header « Authorization » in your request.

It must contain the « Basic » method and the Base64 representation of your user name and password separated by the character « : ».

For security reason, you will use a connection encrypted by SSL to call the REST API.

#### Internationalization

You can add the HTTP header « Accept-Language » to specify in which language you want to see the results. Only one language can be passed to this parameter. The monitoring is available in two languages: French and English.

By default, the selected language is French. If you use this parameter with an unavailable language, the result will be in French.

**Available values: fr | en | en-US**

### REST API parameters

#### List of packages by status

Request structure

Parameter name (in URI)	Description	Expected type	Allowed values	Location	Required
-------------------------	-------------	---------------	----------------	----------	----------

{status}	Package status	String	REJECTED ERROR	Path	true
offset	Offset position in the result set to start pagination	Integer		Query	false (default 0)
limit	Maximum number of results	Integer		Query	false (default = max = 1 000)
submissionStartDate	Starting date of the search (included)	String (format: yyyy-MM-dd)		Query	false
submissionEndDate	End date of the search (included).  Must be after the submissionStartDate.	String (format: yyyy-MM-dd)		Query	false

NB: {status} parameter is case insensitive.

### Response structure

The response contains a list of packages. Each “package” includes the following parameters:

Parameter name	Description	Expected type
depositIdentifier	Deposit identifier	String
submissionDate	Deposit date	String (format yyyy-MM-dd HH:mm:ss)
status	Package status	String
resultError	Service rejected or in error	ComplexType

Definition of “resultError”:

Parameter name	Description	Expected type
state	Service state	String
errorCode	Service error code	String
message	Service message	String
startDate	Service processing start date	String (format yyyy-MM-dd HH:mm:ss)
endDate	Service processing end date	String (format yyyy-MM-dd HH:mm:ss)

NB: Results are ordered by ascending “startDate”.

As the results are paginated, there are also three important headers to check:



Header name	Value	Description
Status Code	206 Partial Content	A status code value with « 206 » means all results are not in the response.
Accept-Range	package 1000	Indicates the response will contain a maximum of 1 000 « packages ».
Content-Range	<offset> – <limit> / <count>  (Ex: 0-999 / 4000)	Offset: index of the first element in the response  Limit: index of the last element in the response  Count : total of « packages »

### Search by depositId

#### Request structure

Parameter name	Description	Expected type	Location	Required
{depositId}	Package to find	String	Path	true

NB: {depositId} parameter is case insensitive.

#### Response structure

The response contains a “package” with a list of “deposits”:

Parameter name	Description	Expected type
depositIdentifier	Deposit identifier	String
status	Package status	String
deposits	List of deposits	Array

Definition of each element in “deposits”:

Parameter name	Description	Expected type
depositDate	Deposit date	String (format yyyy-MM-dd HH:mm:ss)
result	Result of the deposit	ComplexType

Definition of a “result”:

Parameter name	Description	Expected type
state	Service state	String

errorCode	Service error code	String (format "Exxxx" where "x" is a number [0;9])
message	Service message	String
startDate	Service start date	String (format yyyy-MM-dd HH:mm:ss)
endDate	Service end date	String (format yyyy-MM-dd HH:mm:ss)

NB: Results are sorted by ascending deposit date.

## Examples

### List of packages by status

#### Request

In this request, we want to get the list of packages in error.

#### Response

##### Headers:

This is a partial response as you can see with the "Status Code" response header.

The "Content-Range" header indicates you have 1 000 results in your response. Results are from index 0 to index 999. The total number of results is 4 182.

You will need a new request to get next results.

##### Body:

Request with paging

If you receive a partial response and you want next results, you need to use the "offset" parameter. The value of this parameter represents the id of the first result you will have.

In the previous response, the last index was 999 so the offset should be equals to 1 000.

#### Response

##### Headers:

List of rejected packages since a date

#### Request

In this request, we want to get the list of rejected packages since August 29<sup>th</sup>, 2017 included until now (September 5<sup>th</sup>, 2017).

#### Response





# **EOSC-hub**

**Headers:**

**Body:**

In this example, there is only one submitted package since September the 29<sup>th</sup>.

Search by depositId

**Request**

In this request, we want to see "P03016561" package details.

**Response**

**Header:**

**Body:**



## Searching data using HTTP REST API Version 5

CINES exposes an HTTP REST API in order to search the data associated to a deposit according to criteria on OCR results optionally combined with criteria on metadata and to retrieve an image searching by its identifier.

In this section, all the available parameters of this API will be described, and an example of how to use this web service will be provided.

### REST API Interface description

The web service is available at the following URLs:

Function	URL	Protocol
Search in OCR result and in metadata	<a href="https://opendata.cines.fr/herbadrop-api/rest/data/search">https://opendata.cines.fr/herbadrop-api/rest/data/search</a>	HTTP POST*
Retrieve the OCR result of an image	<a href="https://opendata.cines.fr/herbadrop-api/rest/data/">https://opendata.cines.fr/herbadrop-api/rest/data/</a>	HTTP GET
Get the image	<a href="https://opendata.cines.fr/herbadrop-api/rest/image/">https://opendata.cines.fr/herbadrop-api/rest/image/</a>	HTTP GET

\* This REST API uses HTTP POST protocol with JSON format for the data.

### REST API parameters

Request for a Full-Text search in OCR results

All the request parameters are transmitted using the JSON Format.

The following table contains the detailed description of all available parameters.

Category	JSON Parent	Parameter name	Description	Type	Required
Search		text	Full-text fragment you want to search in the OCR results.	String	Yes
		page	Page number of the result.	Integer	No (default 1)
		size	Number of results per page.	Integer	No (default 20)
		language	Dictionary in which the keyword will be search and which will appear in the response	String	No (default all)

Category	JSON Parent	Parameter name	Description	Type	Required
		transferringAgency	The short name of the transferring agency.	String	No
		strictCharacterSearch	A flag used to specify that Full-Text search on OCR results must match exact words.	Boolean	Yes (default false)
		searchTextInAdditionalData	A flag used to specify that the scope of the search includes the OCR results.	Boolean	Yes (default true)
		searchTextInMetadata	A flag used to specify that the scope of the search includes the metadata.	Boolean	Yes (default true)
Personalization	highlight	preTag	Start tag in HTML used to wrap highlighted text. Example : "<b>"	String	No
		postTag	End tag used to wrap highlighted text . Example : "</b>"	String	No
		fragmentSize	The size (in characters) of the highlighted fragment.	Integer	No
		fragmentsCount	The maximum number of fragments in the result.	Integer	No
Metadata criteria	metadataCriteria		List of 'criterion' objects	List	No
		field	The full path of the field. See the list below. Example : "aip.meta.note"	String	Yes
		operator	The name of the operator. See the list below. Example : "CONTAINS"	String	Yes

Category	JSON Parent	Parameter name	Description	Type	Required
		not	A flag used to specify that the condition must be inverted or not.	Boolean	Yes
		values	A text describing one or more values separated by the ' ' character. If value is a date, the ISO8601 format must be used.  Examples : - "StillImage" - "2017-01-01 2017-12-31"	String	Yes

### Request for a search in metadata

One or more criteria can be added to search for values in the metadata.

Each criterion describes:

- a path to the field from one of the list described below,
- an operator from one of the list described below,
- a flag used to specify if the condition must be reverted or not,
- one or more values to search for.

### Request parameters

The available paths for fields of the metadata are described in this table:

Path of the field	Description	Type
aip.dc.contributor	Contributor (part of the Dublin Core metadata)	String
aip.dc.coverage	Location of the collect.  Format: 'Country StateProvince County Municipality Locality VerbatimLocality'	String
aip.dc.creator	Collector name	String
aip.dc.description	Description of the specimen	String
aip.dc.endDate	Collect date (same as start date, specified using ISO8601 format)	Date
aip.dc.format	Image format or mime type  Example:	String

Path of the field	Description	Type
	'image/jpeg'	
aip.dc.identifier	The unique identifier computed during the archiving process (aka ARK)	String
aip.dc.language	Language of the specimen or "und" if undefined	String
aip.dc.publisher	Name (or acronym) of the institution in charge of the specimen	String
aip.dc.rights	License associated to the specimen, Example: 'cc-by'	String
aip.dc.source	Event number and collect number separated by ' '	String
aip.dc.startDate	Collect date (same as end date, specified using ISO8601 format)	Date
aip.dc.subject	Family of the specimen	String
aip.dc.title	Name of the specimen	String
aip.dc.type	Specimen type. Example: 'PreservedSpecimen' or 'StillImage.'	String
aip.files.format	The format of the image file (filled by the SipBuilder tool)	String
aip.files.name	The name of the image file (filled by the SipBuilder tool)	String
aip.files.originalChecksum	The checksum of the image file (computed and filled by the SipBuilder tool)	String
aip.files.originalChecksumType	The type of the checksum associated to the file (filled by the SipBuilder tool)	String
aip.meta.archivingDate	The archiving date (specified using ISO8601 format)	Date
aip.meta.filePlan	The file plan of the institution Example: 'Herbarium'	String
aip.meta.producerIdentifier	Identifier of the specimen according to the repository of the institution	String

Path of the field	Description	Type
aip.meta.transferringAgency	The full name of the institution having sent the data	String
aip.meta.version	The version of the deposit	String

Note: The contents of these fields have been discussed during one of the workshop sessions.

Hereunder is the list of supported operators:

Name	Supported type(s)	Expected number of values
EQUALS	String, Date, Number	One value
CONTAINS	String	One value
MATCHES	String	One value
MATCHES_REGEX	String	One value
STARTS_WITH	String	One value
AFTER	Date, Number	One value
BEFORE	Date, Number	One value
BETWEEN	Date, Number	Two values

Note: The 'MATCHES\_REGEX' operator uses an expression supported by the Lucene solution and is not fully compatible with the Perl expressions. To get more details, please refer to the documentation of our current indexing engine at <https://www.elastic.co/guide/en/elasticsearch/reference/5.6/query-dsl-regexp-query.html#regexp-syntax>.

### Examples of search requests

Example of a JSON query for a Full-Text search only on OCR results without specifying a language:

```
{
  "text": "Herbarium",
  "strictCharacterSearch": false,
  "searchTextInAdditionalData": true,
  "searchTextInMetadata": false,
  "page": 1,
  "size": 20
}
```





```
}
```

Example of a JSON query for a Full-Text search on OCR results and metadata using a particular language:

```
{  
  "text": "Herbarium",  
  "strictCharacterSearch": false,  
  "searchTextInAdditionalData": true,  
  "searchTextInMetadata": true,  
  "page": 1,  
  "size": 20,  
  "language": "eng"  
}
```

Example of a JSON query for a search only on metadata without specifying a language:

```
{  
  "strictCharacterSearch": false,  
  "searchTextInAdditionalData": true,  
  "searchTextInMetadata": true,  
  "page": 1,  
  "size": 20,  
  "language": "",  
  "metadataCriteria": [  
    {  
      "field": "aip.dc.type",  
      "operator": "CONTAINS",  
      "not": "false",  
      "values": [  
        "StillImage"  
      ]  
    }  
  ]  
}
```





```
]
}
```

Example of a JSON query for a search on metadata combined with Full-Text search on OCR results, still without specifying a language:

```
{
  "text": "Herbarium",
  "strictCharacterSearch": false,
  "searchTextInAdditionalData": true,
  "searchTextInMetadata": true,
  "page": 1,
  "size": 20,
  "metadataCriteria": [
    {
      "field": "aip.dc.type",
      "operator": "CONTAINS",
      "not": "false",
      "values": [
        "StillImage"
      ]
    }
  ]
}
```

## Response structure

The REST API returns a response in JSON format with the following structure:

JSON Parent	Parameter name	Description	Type
	maxScore	Indicates the max score of all the results :	Float



JSON Parent	Parameter name	Description	Type
	total	Total number of results.	Integer
Result	Score *1	Score of the result.	Float
	depositId	Identifier of the submitted image corresponding to the OCR content.	String
	transferringAgency	Name of transferring agency who has transferred the image corresponding to the OCR content	String
	transferringAgencyId	Identifier of transferring agency who has transferred the image	String
	additionalIdentifiers	Collection of additional identifiers	Object
	additionalIdentifiers,type	Type of the identifier	String
	AdditionalIdentifiers.identifier	Value associated to the identifier	String
Result/image	fileName	File name of the image corresponding to the OCR content.	String
	fileFormat	File format of the image corresponding to the OCR content.	String
Result/contentOcr	und	<p>The OCR content processed with the <b>selected</b> (und, French, Spanish, German, Latin, English) dictionary.</p> <p>«und» means the result of the OCR processing with the 5 dictionaries used together.</p> <p>Note: Values are the ISO3 codes associated to the language.</p>	String
	fra		
	spa		
	deu		
	lat		
	eng		
Result/highlight	contentOcr.fra	List of fragments containing the matching term(s) according to the fields used for the Full-Text search.	String (HTML)
	contentOcr.all		
	contentOcr.deu		

JSON Parent	Parameter name	Description	Type
	contentOcr.spa	Fields can be one (or more) of the OCR results or one (or more) of the metadata according to the search parameters.	
	contentOcr.eng		
	contentOcr.lat		
Result/metadata		The list of fields of metadata described as a pair, having a path and at least one value.  Refer to the list of fields detailed below and the following examples.	Object

\*1 This score indicate the relevance of each result. The higher the score, the more relevant document.

The max\_score value is the highest score of any document that matches the query.

The available fields of the metadata returned by this API are described in this table:

Path of the field	Description	Type
aip.dc.contributor	Contributor (part of the Dublin Core metadata)	String
aip.dc.coverage	The container describing the location of the collect per language (as ISO3) Example: { 'und' : 'Country StateProvince County Municipality Locality VerbatimLocality' }	Object
aip.dc.creator	Collector name	String
aip.dc.description	The container describing the descriptions of the specimen by language (as ISO3)	Object
aip.dc.endDate	Collect date (same as start date)	Date
aip.dc.format	The container describing the formats (or mime types) of the image per language (as ISO3) Example: { 'eng' : 'image/tiff' }	Object
aip.dc.identifier	The unique identifier computed during the archiving process (aka ARK) Example: 'ark:/87895/1.herbadrop_test=1'	String
aip.dc.language	Language of the specimen or 'und' if undefined	String
aip.dc.publisher	Name (or acronym) of the institution in charge of the specimen	String

Path of the field	Description	Type
aip.dc.relation	<i>Not used in Herbadrop at this stage</i>	String
aip.dc.rights	The container describing the licenses of the specimen per language (as ISO3) Example: { 'und' : 'cc-by' }	Object
aip.dc.source	Event number and collect number separated by ' '. The values are described in a container per language (as ISO3) where the default language code is 'und'	Object
aip.dc.startDate	Collect date (same as end date)	Date
aip.dc.subject	The container describing the family of the specimen per language (as ISO3) Example: { 'lat': 'Amaranthaceae' }	Object
aip.dc.title	The container specifying the names of the specimen per language (as ISO3)	Object
aip.dc.type	The container describing the types of the specimen per language (as ISO3) Example: { 'eng': 'PreservedSpecimen StillImage' }	Object
aip.files	The container describing a list of files objects. See lines below	Object
aip.files.compression	The information of compression	String
aip.files.format	The format of the image file (filled by the SipBuilder tool)	String
aip.files.formatVersion	The version of the format for the file (filled by the SipBuilder tool)	String
aip.files.name	The name of the image file (filled by the SipBuilder tool)	String
aip.files.note	The note associated to the image file (filled by the SipBuilder tool)	String
aip.files.checksum	The checksum of the image file (computed and filled during the archiving process)	String
aip.files.checksumType	The type of the checksum associated to the file	String

Path of the field	Description	Type
	(computed and filled during the archiving process)	
aip.files.encoding	The encoding of the file Example: 'UTF-8'	String
aip.files.id	The unique identifier of the file computed during the archiving process Example: 'ark:/87895/1.herbadrop_test=2/2'	String
aip.files.originalChecksum	The checksum of the image file (computed and filled by the SipBuilder tool)	String
aip.files.originalChecksumType	The type of the checksum associated to the file (filled by the SipBuilder tool)	String
aip.files.sizeInBytes	The size of the image file (in bytes) (filled during archiving process)	Long
aip.files.structure	The structure information associated to the image file (filled by the SipBuilder tool)	String
aip.meta.archivingDate	The archiving date (specified using ISO8601 format)	Date
aip.meta.depositIdentifier	The deposit identifier	String
aip.meta.filePlan	The container with the file plan of the institution per language (as ISO3) Example: { 'eng' : 'Herbarium' }	Object
aip.meta.finalAction	The container describing the final action per language (as ISO3), reserved for archiving purpose.	Object
aip.meta.note	The container detailing the notes per language (as ISO3)	Object
aip.meta.paclIdentifier	Internal identifier of the deposit in the archiving solution	Long
aip.meta.previousVersion	The reference on the previous version of the deposit	String
aip.meta.producerIdentifier	Identifier of the specimen according to the repository of the institution	String
aip.meta.project	The project associated to the deposit	String
aip.meta.structure	The structure information associated tot the deposit	String
aip.meta.transferringAgency	The full name of the institution having sent the data	String



Path of the field	Description	Type
aip.meta.version	The version of the deposit	String

## Example of JSON response:

```
{
  "maxScore":0.012074512,
  "total":2,
  "result":[
    {
      "metadata": {
        "aip.dc.contributor": "Some people",
        "aip.dc.coverage": {
          "und": "Yemen |||| Gov. Hadhramout. ...adi E of Alkadi al Beida. |"
        },
        "aip.dc.creator": "P. Hein",
        "aip.dc.description": {
          "und": "unavailable"
        },
        "aip.dc.endDate": "2002-09-08T00:00:00+0200",
        "aip.dc.format": {
          "eng": "Image/tiff"
        },
        "aip.dc.identifier": "ark:/87895/1.herbadrop_test=1",
        "aip.dc.language": "und",
        "aip.dc.publisher": "B",
        "aip.dc.relation": {
          "eng": "relation"
        }
      }
    }
  ]
}
```





```
},
  "aip.dc.rights": {
    "und": "cc-by"
  },
  "aip.dc.source": {
    "und": "unavailable"
  },
  "aip.dc.startDate": "2002-09-08T00:00:00+0200",
  "aip.dc.subject": {
    "lat": "Amaranthaceae"
  },
  "aip.dc.title": {
    "lat": "Chenopodiaceae"
  },
  "aip.dc.type": {
    "eng": "PreservedSpecimen|StillImage"
  },
  "aip.files": [
    {
      "checksum": "ca1748e459d7102...78dd62d044c293292",
      "checksumType": "SHA-256",
      "encoding": "UTF-8",
      "format": "TIFF",
      "formatVersion": "NA",
      "identifier": "ark:/87895/1.herbadrop_test=1/1",
      "name": "B_10_0380787_bis.tiff",
      "originalChecksum": "d1d7760ef920ae98273bf9038000a4a7",
      "originalChecksumType": "MD5",
```







# EOSC-hub

```
"sizeInBytes": 24189012
}
],
"aip.meta.archivingDate": "2017-11-13T11:08:40+0100",
"aip.meta.depositIdentifier": "B100380787BIS",
"aip.meta.filePlan": {
  "eng": "/"
},
"aip.meta.finalAction": {
  "fra": "Conservation définitive"
},
"aip.meta.note": {
  "eng": "unaivalable"
},
"aip.meta.pacIdentifier": 1,
"aip.meta.previousVersion": "1",
"aip.meta.producerIdentifier": "http://herbarium....object/B100380787",
"aip.meta.project": "herbadrop_test",
"aip.meta.transferringAgency": "Agency1",
"aip.meta.version": "2"
},
"score": 0,
"depositIdentifier": "B100380787BIS",
"transferringAgencyIdentifier": "agency1ftp",
"additonalIdentifiers": [
{
  "type": "HANDLE",
  "identifier": "APIDHANDLE1"
```





```
}  
]  
},  
{  
...  
}  
]  
}
```

Note: The fields, metadata and languages are sorted in alphabetical order when possible.

## Retrieve the indexed data associated to an image

You can retrieve the indexed data of an image by using the image identifier (depositId) and the transferring agency identifier from a HTTP GET request.

URL to use:

<https://opendata.cines.fr/herbadrop-api/rest/data/<transferringAgencyId>/<depositId>>

where <transferringAgencyId> must be replaced by the transferring agency associated to the archive,

<depositId> must be replaced by one of the deposit identifiers

## Response structure

The REST API return a response in JSON format similar to the one returned by the search query excepted that only one result is returned.

Refer to the 'Result' details of the response section of the search request feature.

## Get the image

This API provided a way to get the image as a binary content by using the image identifier (depositId) and the transferring agency identifier from a HTTP GET request.

URL to use:

<https://opendata.cines.fr/herbadrop-api/rest/image/<transferringAgencyId>/<depositId>>

where <transferringAgencyId> must be replaced by the transferring agency associated to the archive,

<depositId> must be replaced by one of the deposit identifiers



## Examples of calls on the REST API

The examples described below use the 'curl' command available on Unixes (sometimes as an additional package). For windows operative systems, a binary can be downloaded at the URL:

<https://curl.haxx.se/download.html>. Install and use of this third party at your own risk.

### Full-text search in OCR results only

```
curl --user <yourusername>:<yourpassword> -k -H "Content-Type: application/json"
-X POST -d '{ "text" : "Paris", "strictCharacterSearch" : false,
"searchTextInAdditionalData" : true, "searchTextInMetadata" : false, "page" : 1,
"size" : 3, "language" : "fra", "highlight" : { "preTag" : "<em>", "postTag" :
"</em>", "fragmentSize" : 10, "fragmentsCount" : 2 } }'
https://opendata.cines.fr/herbadrop-api/rest/data/search
```

### Full-text search in metadata only with criteria on metadata

```
curl --user <yourusername>:<yourpassword> -k -H "Content-Type: application/json"
-X POST -d '{ "searchTextInAdditionalData" : false, "searchTextInMetadata" :
true, "page" : 1, "size" : 3, "language" : "fra", "highlight" : { "preTag" :
"<em>", "postTag" : "</em>", "fragmentSize" : 10, "fragmentsCount" : 2
}, "metadataCriteria": [{ "field": "aip.dc.type", "operator": "CONTAINS",
"not": "false", "values": [ "StillImage" ] } ] }'
https://opendata.cines.fr/herbadrop-api/rest/data/search
```

### Full-text search in metadata and OCR results

```
curl --user <yourusername>:<yourpassword> -k -H "Content-Type: application/json"
-X POST -d '{ "text" : "Paris", "searchTextInAdditionalData" : true,
"searchTextInMetadata" : true, "page" : 1, "size" : 3, "language" : "fra",
"highlight" : { "preTag" : "<em>", "postTag" : "</em>", "fragmentSize" : 10,
"fragmentsCount" : 2 }, "metadataCriteria": [{ "field": "aip.dc.type",
"operator": "CONTAINS", "not": "false", "values": [ "StillImage" ] } ] }'
https://opendata.cines.fr/herbadrop-api/rest/data/search
```

### Retrieve the indexed data associated to an image

```
curl --user <yourusername>:<yourpassword> -k -H "Content-Type: application/json"
-X GET -v https://opendata.cines.fr/herbadrop-api/rest/data/agencyftp/P01742198
```

### Get the image

```
curl --user <yourusername>:<yourpassword> -k -H "Content-Type: application/json"
-X GET -v https://opendata.cines.fr/herbadrop-api/rest/image/agencyftp/P01742198
```



## Appendix

### Create a submission information package

Create a file called sip.xml

```
<?xml version="1.0" encoding="UTF-8" standalone="no" ?>
<pac xmlns="http://www.cines.fr/pac/dev/sip" xmlns:xsi="http://www.w3.org/2001/XMLSchema-
instance" xsi:schemaLocation="http://www.cines.fr/pac/dev/sip
http://www.cines.fr/pac/dev/sip.xsd">
<DocDC>
<title language="eng">MuEG primary dataset in AOD format from RunA of 2011</title>
<creator>MuEG primary dataset in AOD format from RunA of 2011</creator>
<subject language="eng">(to complete)</subject>
<description language="eng">
MuEG primary dataset in AOD format from RunA of 2011. Run period from run number 160404 to
173692
</description>
<publisher>CMS</publisher>
<date>2011/2013</date>
<type language="eng">Collision data</type>
<format language="eng">AOD</format>
<language>und</language>
<rights language="und">CC0 1.0</rights>
</DocDC>
<DocMeta>
<identifiantDocProducteur>10.7483/OPENDATA.CMS.X7AM.49VT</identifiantDocProducteur> <!-- I
chose the DOI; do you agree with that choice for the producer identifier? -->
<serviceVersant>cerntest</serviceVersant>
<planClassement language="eng">/</planClassement>
</DocMeta>
<FichMeta>
<formatFichier>ROOT</formatFichier>
<nomFichier>test1.root</nomFichier>
<empreinteOri type="MD5">b6d81b360a5672d80c27430f39153e2c</empreinteOri>
</FichMeta>
<FichMeta>
<formatFichier>ROOT</formatFichier>
<nomFichier>test2.root</nomFichier>
<empreinteOri type="MD5">2205e48de5f93c784733ffcca841d2b5</empreinteOri>
</FichMeta>
</pac>
```

Create a directory containing the sip.xml and a directory DEPOT

MyDir/

sip.xml

DEPOT/





# **EOSC-hub**

test1.root

test2.root

Create the tar file

```
tar cvf MyTarFile.tar MyDir/
```



EOSC-hub receives funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No. 777536.

[www.eosc-hub.eu](http://www.eosc-hub.eu)