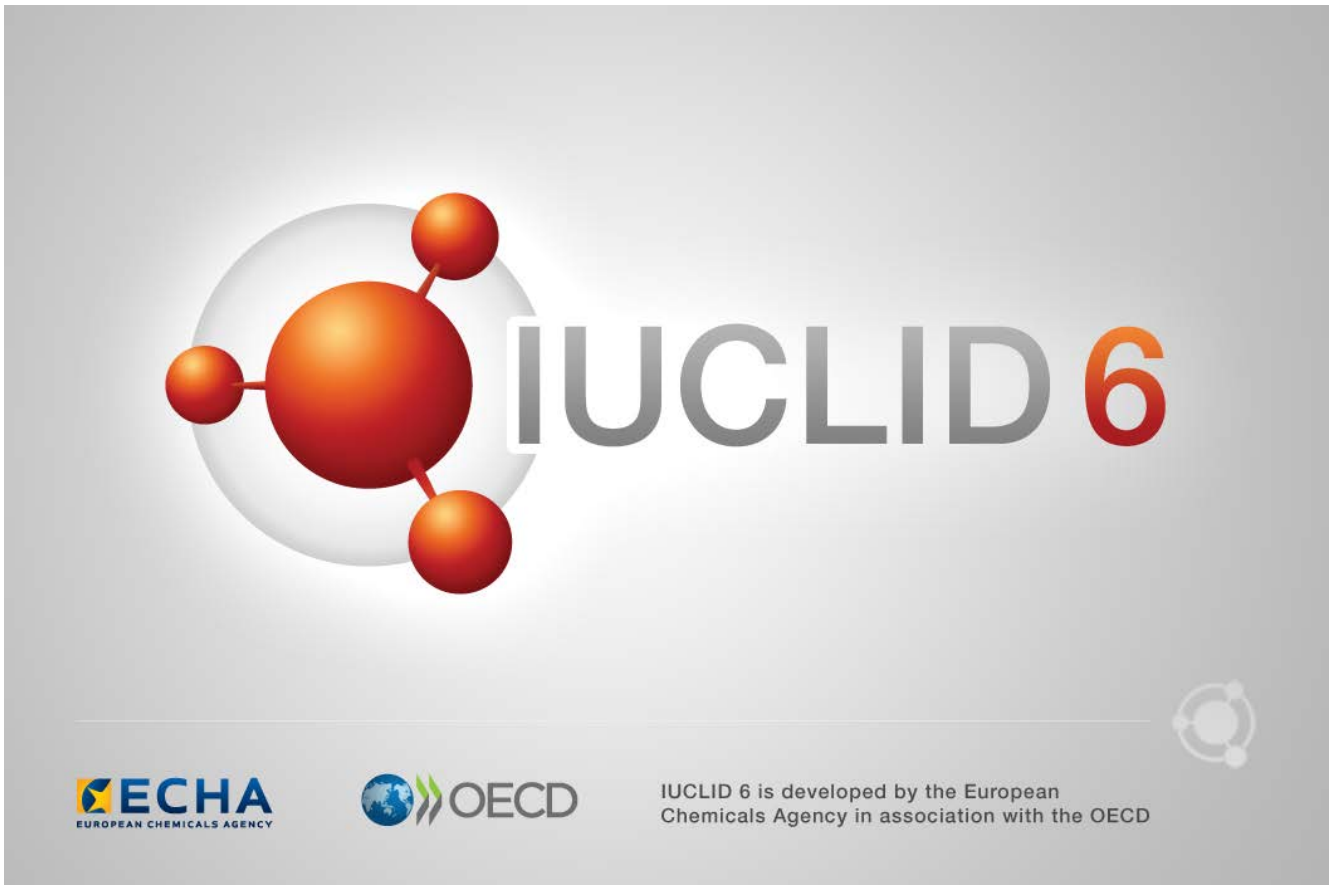
IUCLID Public REST API Documentation



Legal Notice

Title: IUCLID Public REST API Documentation

Issue date: November 2021

Language: en

IUCLID 6 is developed by the European Chemicals Agency in association with the OECD.

If you have questions or comments in relation to this document, please send them to ECHA via the information request form at the address below, quoting the reference and issue date given above:

https://echa.europa.eu/contact

European Chemicals Agency

Mailing address: P.O. Box 400, FI-00121 Helsinki, Finland

Visiting address: Telakkakatu 6, Helsinki, Finland

Changes to this document

Date	Modification
12/11/2021	Applied the IUCLID styles and Introduction section extended
28/10/2020	Describing updates of the API as exposed by IUCLID 6 5.1.2
24/10/2018	Describing updates of the API as exposed by IUCLID 6 3.1.1
15/11/2017	Describing updates of the API as exposed by IUCLID 6 2.0.0
28/04/2017	Describing updates of the API as exposed by IUCLID 6 1.3.0
31/01/2017	First version describing the API exposed by IUCLID 6 1.2.0

Table of Contents

Table of Figures

1. Introduction

The IUCLID 6 Public REST API defines a unified and uniform URI space to access and manipulate the abstract resources comprising the IUCLID system.

This means, in practice, that each resource is identified by a single URI, and that behaviour and representation are determined by standard HTTP mechanisms, such as the HTTP method and the media-type based content negotiation.

For example, the same document URI may be used to retrieve a JSON representation of a document, export it in i6d format, print it in HTML format through a simple GET with a different Accept header.

In addition, new representations and behaviour can be added to the IUCLID Public REST API through a well-defined extension mechanism. The primary extension point of the API is through the extension media type. The extension media type serves as a marker to convey alternative representations of the IUCLID data and trigger custom handlers for each of the supported resource URI.

1.1. IUCLID data structure

One of the major requirements of IUCLID application was to support the ability to add and update data structure definitions without the need to continuously change the underlying database schema that is used to persist the data. IUCLID was designed to be modular, to be able to accept the so-called *definition provider modules* that register their own structures (e.g. endpoint document types that are applicable to specific legislation), see Section 1.2.4 for more information.

To keep a general approach for the database schema, IUCLID uses several predefined structural elements that can be combined to dynamically define data structure document types. The new data structures do not introduce new tables and columns in the database schema, but instead data is stored as new records in the existing tables, containing meta-data keys that specify the type of the information.

1.1.1. IUCLID Document

The central concept of the data representation in IUCLID is the **document**. The IUCLID document is a set of field-values that store scientific and/or administrative data, that conceptually belong together and represent a logical unit.

IUCLID documents can have different types and these types can be organized into a hierarchy.

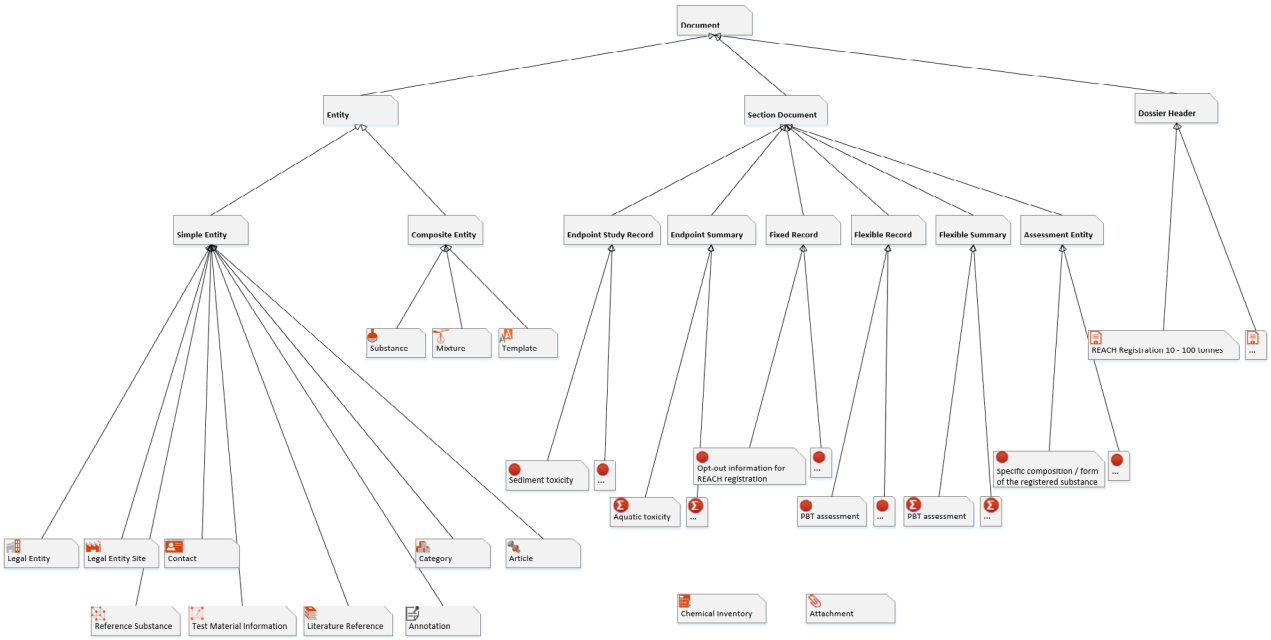


Figure 1: IUCLID Document type hierarchy

The following table describes the meaning of some of the main document types.

Document type	Description
Entity	Document that can be managed (created/updated/deleted) independently from any other entity. Entities can refer to each other via reference fields.
Simple Entity	Entity that cannot have section documents associated with it: Legal Entity, Legal Entity Site, Contact, Reference Substance, Test Material Information, Literature Reference, Chemical Inventory, Category, Annotation, Article
Composite Entity	Entity that can have section documents associated with it. A composite entity and its section documents are known collectively as a *dataset*. The composite entity is the root document of its dataset. By deleting a composite entity all its section documents are also deleted. There are 3 composite entities: Substance, Mixture, Template
Section Document	Document that can be managed (created/updated/deleted) only in the context of a composite entity. A newly created section document is associated implicitly to an existing composite entity. Section documents can make reference to entities or other section documents via reference fields.
Dossier Header	Document that represents the root document of a dossier.
Endpoint Study	Section document that is typically used to store OECD Harmonised Template

Record	(OHT) study summary data. E.g.: Sediment toxicity
Endpoint Summary	Section document that is typically used to store summarised information from multiple endpoint study record instances. E.g.: Aquatic toxicity
Fixed Record	Section document typically storing either generic core or regulatory-specific data, but not OHT study summary data. Only a single instance can be created per composite entity. E.g.: Opt-out information for REACH registration
Flexible Record	Section document typically storing either generic core or regulatory-specific data but not OHT study summary data. Multiple instances can be created per composite entity. E.g.: PBT assessment
Flexible Summary	Section document that is typically used to store summarised information from multiple flexible record instances. E.g.: PBT assessment (summary)
Assessment Entity	Section document that is used in the context of chemical safety assessment. It enables the definition of consistent sets of substance properties that are relevant to the assessment of specific compositions/forms, constituent or transformation of the substance placed on the market or generated upon use. E.g.: Specific composition / form of the registered substance

1.1.2. Datasets and dossiers

IUCLID documents can be organized into a logical collection. A composite entity together with its associated section documents constitute a **dataset** (e.g. substance dataset, mixture dataset). Documents belonging to a dataset are editable, their content can be modified and changes with time. In the IUCLID context such documents are referred to as *raw documents*.

The documents of a dataset can link to other entities via reference fields and to attachments via attachment fields.

An illustration of a mixture dataset with reference to other composite entities, a substance dataset and another mixture dataset is shown below.

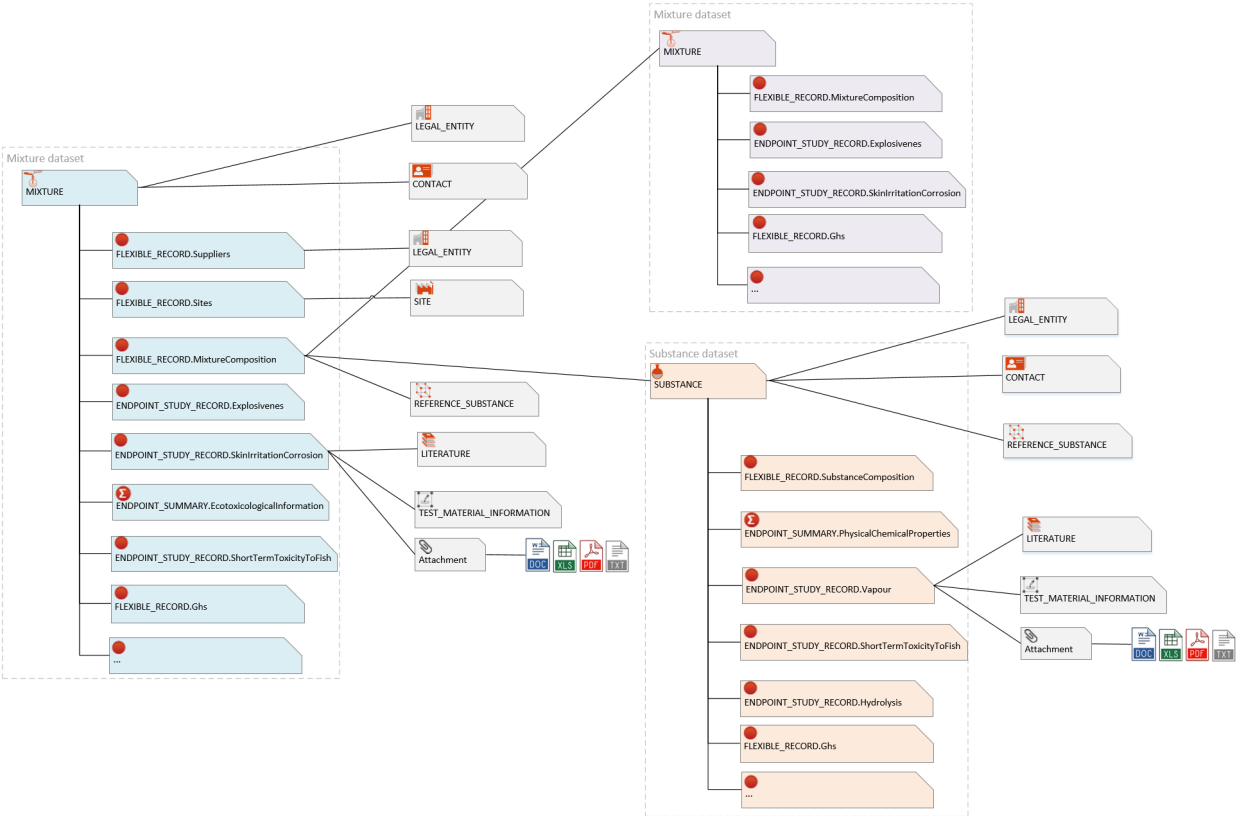


Figure 2: Mixture dataset with reference to other composite entities

A IUCLID **dossier** is created from a raw dataset with a specific purpose, typically regulatory, e.g.: REACH Registration 10 – 100 tonnes, and is a non-editable snapshot of the dataset and its referenced entities. The *subject of the dossier* is the root entity of the raw dataset for which the dossier was created (e.g. a substance or mixture). The root document of a dossier is represented by the **dossier header**. The content of the documents belonging to a dossier cannot be modified and their business meaning, or purpose, does not change with time.

The following image illustrates a “BPR Active substance application” dossier created from a mixture dataset with reference to other composite entities.

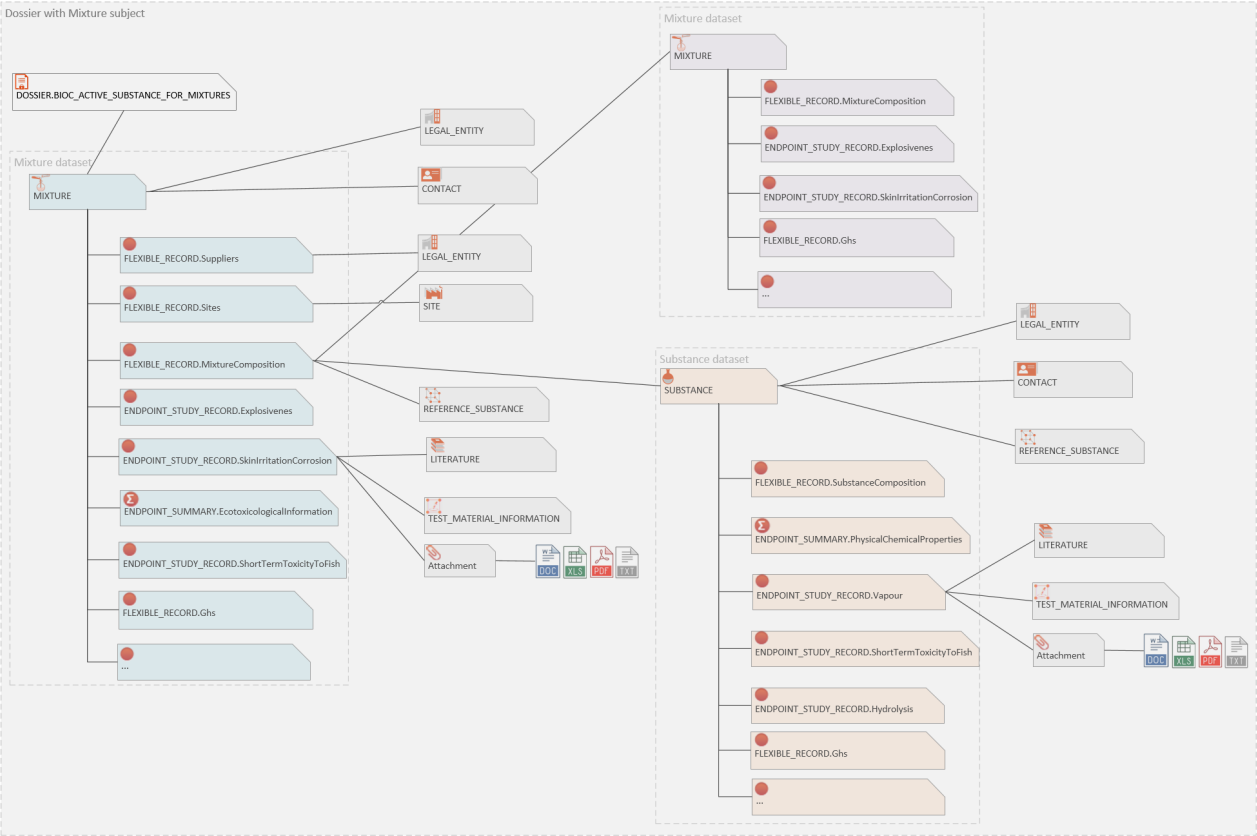


Figure 3: Dossier with mixture subject and reference to composite entities

1.1.3. Structure definitions

IUCLID is a highly extensible system, especially when it comes to the documents it can handle and how these are organized in datasets.

A **definition provider** can, through a well-defined mechanism, specify:

- document definitions, that define the structure of a particular document
- section-trees, that organize documents in *datasets* (e.g. 'Substance' dataset)
- submission types, that identify a specific section-tree, along with which dataset type applies to
- phrases to be used, for example, as values in '*picklists'* fields
- phrase-groups, that define a set of phrases that are available to a specific '*picklist*' field
- text-templates, that define texts that can be used as initial values for certain text-fields

Document definitions play a particularly important role. They consist of an ordered collection of block and field definitions that specify the structure of a given IUCLID document. Based on a document definition the IUCLID application can derive:

- a default persistence format of the document data to the database
- a default exchange format of the document data (.i6d file-format of an .i6z file, or the JSON-format of the REST web services)
- a default UI form to display and edit the content of a document

A **block definition** groups together an ordered collection of field and other block definitions, that is meant to capture information that logically belong together. There are two flavours of blocks used in IUCLID:

- *Single block*: typically, it is used for a set of fields that appear under the same header/title in the user interface displaying a document content.
- *Repeatable block*: it groups a set of fields (or blocks) that can be repeated multiple times inside a document. Typically, it is used for a set of fields, that appear as a table in the user interface, each field corresponding to a table column.

A **field definition** specifies a simple data structure in the context of a document, that has a label and a field-value type. Supported field types:

- *Text types*: Text (255 char.), Text (2,000 char.), Text (32,768 char.), Text (rich-text area), Text template
- *Numeric types*: Numeric (decimal including unit), Numeric (decimal), Numeric (integer), Numeric range (decimal, precise), Numeric range (decimal), Numeric range (decimal with picklist), Numeric range (half bounded)
- *List types*: List (picklist), List sup. (picklist with remarks), List sup. (picklist with remarks - 2,000 char.), List sup. (picklist with remarks - 32,000 char.), List multi. (multi-select list), List multi. (multi-select list with remarks), List multi. (multi-select list with remarks - 2,000 char.), List multi. (multi-select list with remarks - 32,000 char.)
- *Link types*: Link to endpoint (single), Link to endpoint (multiple), Link to entity (single), Link to entity (multiple), Link to lit. reference (multiple)
- *Other types*: Check box, Attachment (single), Attachment (multiple), Image upload, Date, Confidentiality, Endpoint reference representation

The latest definitions of the IUCLID data format are available on the following site: https://iuclid6.echa.europa.eu/format

1.1.4. UUID and document key

UUIDs (universally unique identifiers) are used at various levels of the IUCLID data structure to identify unique sets of data. The following sections describe the main usages of the UUID in the IUCLID application.

1.1.4.1. Document UUID

A **document UUID** uniquely identifies an editable/raw document.

Document UUIDs are kept during dossier creation, that is when a read-only copy is created for the document content that is be part of a dossier. As a result, documents belonging to different dossiers can share the same document UUID.

During document creation there is no need to specify the document UUID because the IUCLID application generates one.

During document update, the UUID of the document must be specified.

1.1.4.2. Repeatable block UUID

A **repeatable block UUID** uniquely identifies a repeatable block entry inside a single document.

Repeatable block UUIDs are kept during dossier creation.

During document creation or document update, if there are new repeatable block entries added there is no need to specify the repeatable block UUID as IUCLID application will take care of generating one.

During document update, if the repeatable block UUID of an entry is specified, the IUCLID application attempts to find and update the existing block entry. If the entry with the given block-UUID is not found, it will treat it as a new repeatable block entry.

1.1.4.3. Document key

IUCLID documents are uniquely identified by a **document key**, independently if they are part of an editable dataset or a read-only dossier.

Documents belonging to a dataset have a slightly different format compared to documents of a dossier:

- Document's key format belonging to a raw dataset: {document_uuid}/0. E.g.:

```
9f7c364c-8e29-4d3c-8c4a-56baea903414/0
```

- Document's key format belonging to a dossier: {document_uuid}/{dossier_uuid}. E.g.:

```
9f7c364c-8e29-4d3c-8c4a-56baea903414/3bf15315-f76f-4d61-84d0-0ac0335ad338
```

The document UUID part of the key is kept when a dossier is created from a raw dataset document.

Dossier header documents have the peculiarity that their document UUID and dossier UUID values are the same. E.g.:

```
3bf15315-f76f-4d61-84d0-0ac0335ad338/3bf15315-f76f-4d61-84d0-0ac0335ad338
```

1.2. Resource URI organization

The URI resources of the IUCLID Public REST API are organized in a hierarchical way. The complete URI space is divided in to four areas, each with its own root path.

`/raw`	Raw data resources
`/dossier`	Dossier (Snapshot) resources
`/system`	System resources
`/definition`	Definition resources

1.2.1. Raw data resources

As a general rule, each top-level IUCLID domain type has its own sub-tree of the `/raw` tree.

`/raw/LEGAL_ENTITY`	The collection of raw legal entity documents
`/raw/REFERENCE_SUBSTANCE`	The collection of raw reference substance documents
`/raw/SUBSTANCE`	The collection of raw substance datasets
`/raw/MIXTURE`	The collection of raw mixture datasets
`/raw/TEMPLATE`	The collection of raw templates datasets
`/raw/ARTICLE`	The collection of raw article datasets
`/raw/ANNOTATION`	The collection of raw annotation documents
`/raw/LITERATURE`	The collection of raw literature documents
`/raw/CATEGORY`	The collection of raw category documents
`/raw/SITE`	The collection of raw site documents
`/raw/CONTACT`	The collection of raw contact documents
`/raw/TEST_MATERIAL_INFORMATION`	The collection of raw test material information documents
`/raw/CONTACT`	The collection of raw contact documents

The IUCLID domain types can be classified in two broad categories: *documents* and *datasets*. A IUCLID dataset top-level domain type (i.e. SUBSTANCE, MIXTURE, TEMPLATE) defines an entity that contains children IUCLID documents, while a IUCLID document top-level domain type consists only of itself.

In the following diagram is depicted the sub-tree of a IUCLID dataset top-level domain type:

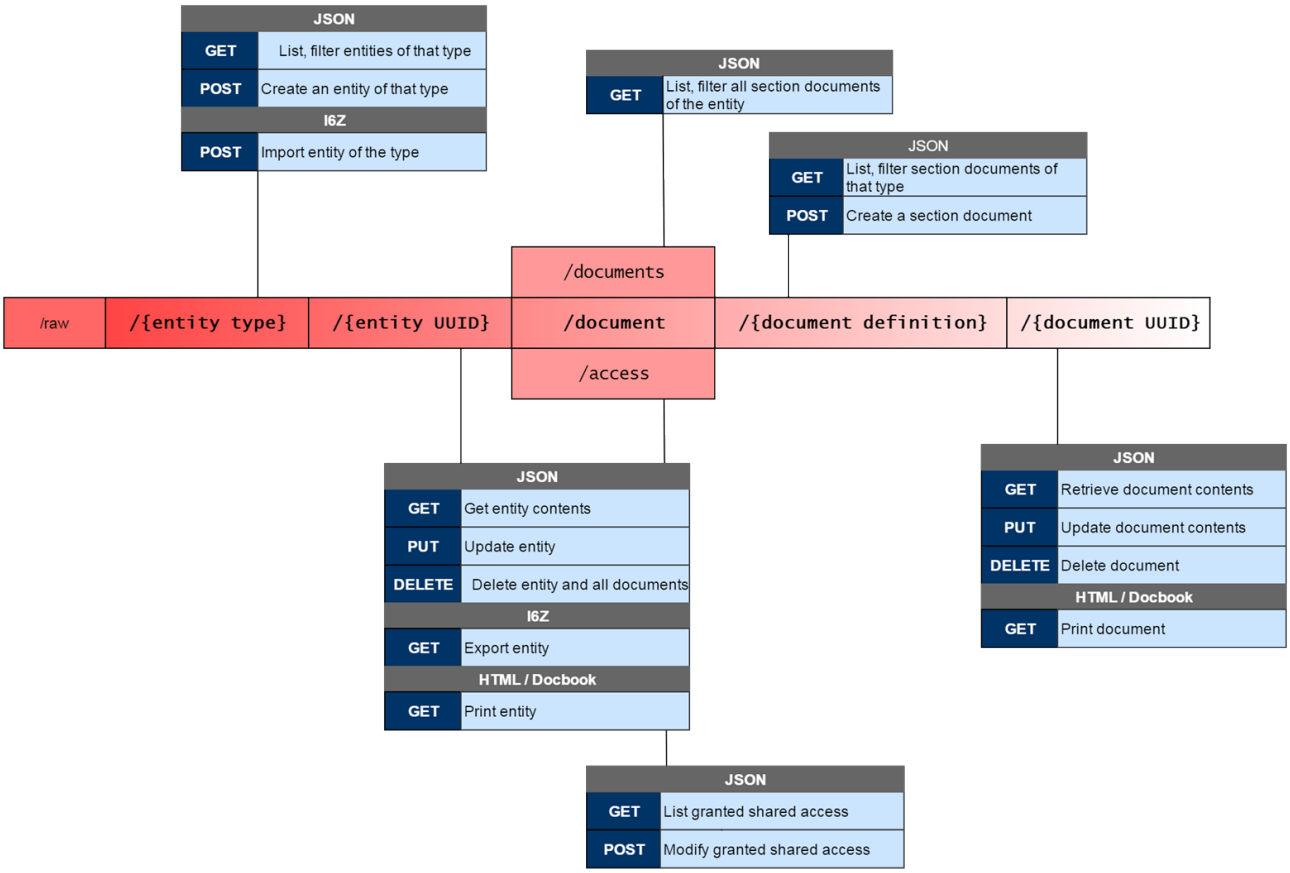


Figure 4: Raw data resource URI structure

1.2.2. Dossier resource

Dossier resources reproduce the resource paths defined for raw data resources under a path root representing a given dossier:

`/dossier/{uuid}`	Root path of a dossier with a specific snapshot UUID

Reflecting the read-only nature of dossiers, only a subset of the methods supported by raw data resources is valid. Typically, only GET is supported for each corresponding dossier path.

Additionally, the `/dossier` URI sub-tree supports resources concerning general dossier management, such as searching and import.

1.2.3. System resources

System resources cater for administration and monitoring aspects of the IUCLID server and will usually not be used directly by clients.

A notable exception is the job monitoring resource, which can be linked to by operation on raw data or dossier resources.

As this specification matures, this section may become inaccurate. Service resources such as the validation assistant and the reporting engine may also need to be hosted by this tree.

1.2.4. Definition resources

Definition resources represent the various structure schemas and metadata, contributed by the domain, core, OHT and the legislation specific definition providers, supported by a given IUCLID installation.

Nearly all definition resources can exist in more than one version, which is reflected in their path.

The special latest version number is used to indicate the latest installed version of a given definition resource and will usually result in a redirect to the actual resource path of that version.

1.2.5. I6URI

Some of the operations of IUCLID Public REST API use a special, so called I6URI (IUCLID6 URI) to reference resources of the IUCLID application.

The generic format of a I6URI is the following:

```
iuclid6:/{dossier-uuid_or_0}/{ENTITY_TYPE}/{entity-uuid}/{SUB_TYPE}/{doc-uuid}
```

Some example I6URIs can be found below:

- Dossier I6URI format: `iuclid6:/{dossier-uuid}`

```
iuclid6:/3bf15315-f76f-4d61-84d0-0ac0335ad338
```

- Substance raw entity I6URI format: `iuclid6:/0/SUBSTANCE/{substance-uuid}`

```
iuclid6:/0/SUBSTANCE/9f7c364c-8e29-4d3c-8c4a-56baea903414
```

- Substance Composition document of a raw substance entity I6URI format: `iuclid6:/0/SUBSTANCE/{substance-uuid}/FLEXIBLE_RECORD.SubstanceComposition/{composition-doc-uuid}`

```
iuclid6:/0/SUBSTANCE9f7c364c-8e29-4d3c-8c4a-56baea903414FLEXIBLE_RECORD.SubstanceComposition/3bf15315-f76f-4d61-84d0-0ac0335ad338
```

1.3. Base URL

Every REST resource of the public API is under:

Base URL	/iuclid6-ext/api/ext/v1/

The full base URL is expected to contain the server-host and port of the IUCLID instance, e.g.:

```
http://localhost:8080/iuclid6-ext/api/ext/v1/
```

On the **base URL** one can find basic information about the system and its extensions, such as the names of the various installed plugins along with their versions.

GET	/iuclid6-ext/api/ext/v1/

example response

```
{
```

```
  "type": "Framework",
  "name": "IUCEF",
  "description": "",
  "version": "1.4.0",
  "components": [
    {
      "type": "Extension",
      "name": "iuclid6",
      "description": "",
      "version": "1.0.0",
      "components": [
        {
          "type": "ArchiveManifestHandler",
          "name": "ArchiveManifest",
          "description": "",
          "version": null,
          "components": []
        },
        {
          "type": "DocumentHandler",
          "name": "DocumentSecuredRepresentation",
          "description": "",
          "version": null,
          "components": []
        },
...
```

The format of the 'about' resource's response is not stable and is subject to change.

1.4. Media Types

Media types play a central role in the IUCLID Public REST API. Making use of standard HTTP content negotiation, they affect both behaviour and representation, and support the extension mechanism of the API.

As such, the IUCLID Public REST API makes use of a wide range of media types.

Note that not all media types are available for all the resources or all the methods, because each media type may support only a subset of the API semantics.

1.4.1. IUCLID Document media types

The IUCLID Document media types correspond to the standard representation of IUCLID documents, and other associated entities.

JSON	application/vnd.iuclid6.ext+json;type=iuclid6.Document
HTML	text/html

1.5. HTTP methods and status codes

The IUCLID Public REST API is tightly integrated with the underlying HTTP protocol. This implies that the semantics of the functional operations that are supported on combinations of *resource URI + media type* are mapped to appropriate HTTP methods, and that they respect the specification of those methods https://tools.ietf.org/html/rfc7231

Similarly, the status codes used by the API are aligned with the HTTP specification and map the functional semantics of the IUCLID Public REST API execution results.

1.5.1. Semantics of HTTP methods

API function	HTTP method	Scope
list	GET	Collections of resources. Collections can be filtered using named filters and paged.
search	GET	Named query resources.
read	GET	Individual resources, however coarse or fine grained.
create / add	POST	Collections of resources. A POST maps to an addition to the collection i.e. a creation
update / edit	PUT	Individual resource. A PUT maps to an update of the resource.
delete / remove	DELETE	Individual resource. Deletion of collections is not supported.
import	POST	Collections of resources. Import is a variant of creation or update.
export	GET	Individual resource. Export is a variant of load.
print	GET	Individual resources. The print format is determined by the media type requested by the client.

1.5.2. Semantics of HTTP status codes

1.5.2.1. Success codes

HTTP Status Code	Usage
200 OK	Success of data retrieval operations such as read, list, search, export, print
200 OK	Success of update operations, such as update, edit, import
204 No Content	Success of update or deletion operations, when no entity is included in the response, e.g. delete, remove
201 Created	Success of creation operations such as create, add, import (if the import results in the creation of a new entity)
202 Accepted	POST operations may return this status code if a background job was launched as a result of the request, e.g. for import or create dossier. See below for polling.
308 Permanent Redirect	Used for generic URI that don't change, such as /dossier/{uuid}/subject
303 See Other	Result of a background job can be accessed through GET at the URL specified in the Location header.
307 Temporary Redirect	Used for generic URI that may evolve, such as /definition/document/{def_id}/latest

1.5.2.2. Error codes

HTTP Status Code	Usage
400 Bad Request	The request entity or request parameters is not parsable by the server, either because it is malformed at the protocol level, or because it is malformed at the format level (XML, JSON, number or date representation).
403 Forbidden	The caller is attempting to perform an operation on a resource it is allowed to see (otherwise 404 would be appropriate) but does not have sufficient privileges.

404 Not Found	If the resource identified by the URI path does not exist (at any depth), or a resource identified in the query string that is essential to the representation does not exist.
405 Method Not Allowed	If the method is not supported by the target resource, regardless of the privileges of the caller. E.g. update a dossier document, POST on a named query
406 Not Acceptable	If the requested media type is not supported for the resource or not known.
409 Conflict	To indicate a conflict of the request with the current state of the resource. E.g. update a document based on a version of it that is not the latest (the document has been updated in the meantime)
415 Unsupported Media Type	If the resource cannot handle the media type sent by the caller.
422 Unprocessable Entity	To indicate business validation errors on the received data. Unless those are already covered by 400, 403 or 404.
500 Internal Server Error	No particular business meaning can be associated with this code, because it is the result of an internal unexpected exception.

Usage of error status codes and response entities also take into account security considerations.

- 404 is preferred to 403 if the caller has insufficient privileges to even see the resource, i.e. perform a simple GET with any content type. In other words, for that particular user the system should behave as if the resource did not exist at all.
- The response entity in the case of an Internal Server Error (500) is kept completely general and does not include any information about the reason for the failure, because this may disclose weaknesses in the server.

1.5.2.3. Polling

Access to some resources may require asynchronous processing on the server side that is implemented via background jobs in IUCLID.

The IUCLID Public REST API does not specify exactly which calls may necessitate such processing.

Instead, it specifies a generic mechanism for informing the client, and describes the patterns of interaction that a client should be prepared to handle.

- If the server chooses to launch a background job to satisfy a given request, it **must**:
  - inform the client by returning a status code of 202 Accepted;
  - include a Location header (this part is non-standard) with the URI of the monitoring resource for the created job.
- The client should call the monitoring URI with a GET to retrieve progress information.

- When servicing the monitoring URI,
    - If the job is not completed, the server MUST
        - continue returning the 202 status code
        - include a response entity providing progress information. The format of the response entity may be adapted by extensions.
    - If the job is completed successfully, the server MUST
        - reply with a 200 if the job completed successfully and the result is included in the job status response
        - reply with a 201 if the job successfully created a new resource (e.g. import job) and include the URI of the created resource in the Location header
        - reply with a 303 if the job completed successfully and the result can be accessed from another URI (e.g. downloaded)
    - If the job failed, the server MUST
        - reply with a 422, or 500 depending if the failure was due to business conditions or technical.

Using 201 Created as the status code of a GET request is quite uncommon and seem to go against the HTTP specification. A 303 See Other may be more appropriate in all cases.

1.6. Authentication

To access most resources in the public API a client must be authenticated and authorized to do so.

The IUCLID public REST API supports two different authentication mechanisms:

- *Plain authentication*: user credentials (username and password) are expected to be specified in custom HTTP request headers (`IUCLID6-USER` and `IUCLID6-PASS`) for each HTTP request.
- *Token-based authentication*: a valid access token is expected to be specified in the `Authorization` HTTP request header for each request.

Note: the examples in this document use plain authentication.

1.6.1. Plain authentication

To authenticate, each request should contain the following HTTP headers:

Header name	Description
`IUCLID6-USER`	The username of the user, who is to be authenticated
`IUCLID6-PASS`	The password of the user

If the authentication fails, the server replies with **401 Unauthorized.**

The responses when authorization fails, depend on the nature of the REST resource, for example when a user tries to read a document to which it does not have access, **404 Not found** is returned. However, an attempt to modify a document for which the user has read access, but not write access, instead returns **403 Forbidden**.

1.6.2. Token-based authentication

The token-based authentication mechanism is considered to be more secure compared to the plain authentication. It requires that each HTTP request contains a valid access JWT token in the `Authorization` header. IUCLID has an IDP (Identity Provider) module that is in charge of issuing expirable JWT token pairs that the integrating parties could use to authenticate requests.

An API client using the token-based authentication will have to perform the following steps:

1. Issue first token pair, access and refresh tokens, by sending valid user credentials;
2. Include a valid access token in the Authorization header for each API request that requires authentication;
3. Regularly refresh / issue a new token pair by sending a valid refresh token.

By default, access tokens are valid for 10 minutes, so refresh must occur in less than 10 minutes.

1.6.2.1. Issue token pair

One can retrieve a token pair at the following resource by providing a valid username and password:

GET	/iuclid6-idp-ws/service/token/issue

The expected form-urlencoded parameters:

Param	Type	Description
`username`	String	the username of the user to login in as
`password`	String	the password of the user to login in as

The response of the request is a pair of tokens.

Param	Description
`access_token`	Contains the necessary information to access a resource URI. This token has an expiration date and is short-lived.
`refresh_token`	Contains the necessary information to get a new access token when the access_token is expired.

Example request:

```
curl --location --request POST 'http://localhost:8080/iuclid6-idp-
ws/service/token/issue' \
--header 'Content-Type: application/x-www-form-urlencoded' \
```

```
--data-urlencode 'username=user-username' \
--data-urlencode 'password=user-password'
```

And a corresponding example response:


```
{
    "access_token":
"eyJhbGciOiJIUzI1NiJ9.eyJqdGkiOiI2MjcyNTRjMi03MTg4LTRkNzUtYWE5ZS05NmQ0OTRjZWQ
wZWUiLCJpc3MiOiJpdWNsaWQ2LXRva2VuLXNlcnZlciIsInN1YiI6IlRlc3QiLCJpYXQiOjE2MzY3
MTUwNzMsImV4cCI6MTYzNjcxNTY3Mywic2NvcGVzIjpbImFjY2VzcyJdfQ.RGzEsZLemUXKTOp4dW
pKtQTvXHr62LAgb3acG1F0KCY",
    "refresh_token":
"eyJhbGciOiJIUzI1NiJ9.eyJqdGkiOiIxODk0ODJhNC0wZGZlLTRmYmUtOTQ2ZC1iOGFiYmNjNWV
mNTQiLCJpc3MiOiJpdWNsaWQ2LXRva2VuLXNlcnZlciIsImF1ZCI6Iml1Y2xpZDYtdG9rZW4tc2Vy
dmVyIiwic3ViIjoiVGVzdCIsImlhdCI6MTYzNjcxNTA3MywiZXhwIjoxNjM2NzE1OTczLCJzY29wZ
XMiOlsicmVkZWVtIl0sIm5iZiI6MTYzNjcxNTA3M30.jjocdLyn-
3Z8usLEsWetzsj7CliRODkgEnemvvVqmH4"
}
```


1.6.2.2. Redeem token pair

If an access token has expired, a new token can be obtained using the refresh token by executing a request to the following resource.

GET	/iuclid6-idp-ws/service/token/redeem

The body of the request must be the refresh token value.

Body	`{refresh_token_value}`

The response of the request is a new pair of tokens.

Example request:

```
curl --location --request POST 'http://localhost:8080/iuclid6-idp-
ws/service/token/redeem' \
--header 'Content-Type: application/jwt' \
--data-raw '{refresh_token_value}'
```

1.6.2.3. Use access token

Having obtained an access token, to execute an HTTP call to a IUCLID resources, the token must be placed in the `Authorization` header of the request in the expected format:

`Authorization: Token {access-token-value}`)

Example request:

```
curl --location --request GET 'http://localhost:8080/iuclid6-
ext/api/ext/v1/raw/SUBSTANCE?filter=web.ListQuery&formatter=iuclid6.DocumentS
ecuredRepresentation&o=0&l=-1' \
--header 'Accept: application/json, text/plain, */*' \
--header 'Authorization: Token {access-token-value}'
```

2. Definitions

The IUCLID6 Public REST API provides several resources to retrieve information about registered definition providers, **no authentication** is needed to access these resources, since they convey publicly available information.

All the following resources respond with **application/json**.

Every REST resource in this document is under /iuclid6-ext/api/ext/v1/

2.1. Definition providers

Registered definition providers, with their version and localized name, are found under:

GET	/definition/providers

The response will be a JSON array of objects, each object will contain the following fields:

Example response

```
[
  {
    "identifier": "bpr",
    "title": "EU_BPR",
    "version": "2.0"
  },
  {
    "identifier": "clp",
    "title": "EU_CLP",
    "version": "2.0"
  },
  {
    "identifier": "core",
    "title": "CORE",
    "version": "2.0"
  },
  {
    "identifier": "oecd",
    "title": "OECD",
    "version": "2.0"
  },
  {
    "identifier": "reach",
    "title": "EU_REACH",
    "version": "2.0"
  }
]
```

Name	Type	Description
identifier	string	The identification code of the definition provider.
title	string	The localized name of the provider, in the requested language via the **Accept-Language** header. If the header is omitted, or no localization exists for the requested language, English is used.
version	string	The version of the provider.

The following section on Submission types is obsolete (since v5.0.0) and replaced by the working contexts

2.2. Submission types (obsolete since v5.0.0)

Each definition provider can define its own submission types, and each submission type is accompanied by a section tree with the same identifier.

GET	/definition/submissiontypes

The resource supports query parameters for restricting the results by provider and by applicable entity type:

Parameter	Description
provider	The provider identification code
for	The entity type the submission type must be applicable to. In general, the following IUCLID6 entity types are supported: SUBSTANCE MIXTURE TEMPLATE
dossier	If true, submission types suitable for dossier creation. If false, submission types not suitable for dossier creation.
tree	If true, submission types suitable for tree visualization. If false, submission types not suitable for tree visualization.

Parameter	Description
export	If true, submission types suitable for export. If false, submission types not suitable for export.

The submission type applies to specific entity type (i.e. SUBSTANCE, MIXTURE, TEMPLATE), in case a submission type applies to more than one entity type, multiple array entries will be returned:

example response


```
[
  {
    "provider": "core",
    "identifier": "COMPLETE",
    "title": "Complete table of contents",
    "applicableFor": "MIXTURE",
    "dossier": true,
    "tree": true,
    "export": true
  },
  {
    "provider": "bpr",
    "identifier": "BIOC_ACTIVE_SUBSTANCE_FOR_MIXTURES",
    "title": "BPR Active substance application (representative product)",
    "applicableFor": "MIXTURE",
    "dossier": true,
    "tree": true,
    "export": true
  },
  {
    "provider": "bpr",
    "identifier": "BIOC_BIOCIDAL_PRODUCT",
    "title": "BPR Biocidal product authorisation",
    "applicableFor": "MIXTURE",
    "dossier": true,
    "tree": true,
    "export": true
  },
  {
    "provider": "oecd",
    "identifier": "OECD_HT",
    "title": "OECD harmonised templates",
    "applicableFor": "MIXTURE",
    "dossier": true,
```

```
    "tree": true,
    "export": true
  },
  {
    "provider": "bpr",
    "identifier": "BIOC_BASIC_INFORMATION_MIXTURE",
    "title": "BPR Basic information (mixture)",
    "applicableFor": "MIXTURE",
    "dossier": false,
    "tree": true,
    "export": true
  },
  {
    "provider": "core",
    "identifier": "COMPLETE",
    "title": "Complete table of contents",
    "applicableFor": "SUBSTANCE",
    "dossier": true,
    "tree": true,
    "export": true
  },
  {
    "provider": "reach",
    "identifier": "R_A15_REST",
    "title": "REACH Annex XV - Restriction",
    "applicableFor": "SUBSTANCE",
    "dossier": true,
    "tree": true,
    "export": true
  }
]
```


The response will be a JSON array of objects, each object will contain the following fields:

Name	Type	Description
provider	string	The identifier of the definition provider for this submission type
identifier	string	The identification code of the submission type.
title	string	The localized name of the submission type, in the requested language via the Accept-Language header. If the header is omitted, or no localization exists for the requested language, English is used.

Name	Type	Description
applicableFor	string	The entity type the submission type applies to.
dossier	boolean	If true, the submission type is suitable for dossier creation
tree	boolean	If true, the submission type is suitable for tree visualization
export	boolean	If true, the submission type is suitable for export

2.3. Working contexts (since v5.0.0)

Each definition provider can define its own working context. A working context is applicable for a single entity, in that entity is a complex entity (SUBSTANCE, MIXTURE, TEMPLATE) it defines the section tree for it. Additionally, if the working context supports dossier creation, it defines the definition of the dossier header document. Moreover, it can optionally have a "viewAsMapings" section where alternative definitions for entities are defined, and also an optional "additionalTrees" section where a provides listing of section trees to be used for additional complex entities in a dataset.

GET	/definition/workingcontext

The resource supports query parameters for restricting the results by provider and by applicable entity type:

Parameter	Description
provider	The provider identification code
for	The entity type the submission type must be applicable to. In general, the following IUCLID6 entity types are supported: SUBSTANCE MIXTURE TEMPLATE
dossier	If true, submission types suitable for dossier creation. If false, submission types not suitable for dossier creation.

The submission type applies to specific entity type (i.e. SUBSTANCE, MIXTURE, TEMPLATE), in case a submission type applies to more than one entity type, multiple array entries will be returned:

example response

```
[
    {
        "provider": "clp",
        "identifier": "CLP_ALTERNATIVE_NAME",
```

```
        "title": "CLP Alternative name request",
        "applicableFor": "SUBSTANCE",
        "dossierHeader": "DOSSIER.CLP_ALTERNATIVE_NAME",
        "mainTreeId": "CLP_ALTERNATIVE_NAME.SUBSTANCE"
    },
    {
        "provider": "clp",
        "identifier": "CLP_ALTERNATIVE_NAME",
        "title": "CLP Alternative name request",
        "applicableFor": "TEMPLATE",
        "dossierHeader": "DOSSIER.CLP_ALTERNATIVE_NAME",
        "mainTreeId": "CLP_ALTERNATIVE_NAME.TEMPLATE"
    },
    {
        "provider": "clp",
        "identifier": "CLP_NOTIF",
        "title": "CLP Notification",
        "applicableFor": "SUBSTANCE",
        "dossierHeader": "DOSSIER.CLP_NOTIF",
        "mainTreeId": "CLP_NOTIF.SUBSTANCE"
    },
    {
        "provider": "clp",
        "identifier": "CLP_NOTIF",
        "title": "CLP Notification",
        "applicableFor": "TEMPLATE",
        "dossierHeader": "DOSSIER.CLP_NOTIF",
        "mainTreeId": "CLP_NOTIF.TEMPLATE"
    },
    {
        "provider": "clp",
        "identifier": "CLP_PCN",
        "title": "CLP Poison centres notification",
        "applicableFor": "MIXTURE",
        "dossierHeader": "DOSSIER.CLP_PCN",
        "mainTreeId": "CLP_PCN.MIXTURE",
        "additionalTrees": [
            {
                "treeName": "CLP_PCN_MIM",
                "activationClauses": [
                    "MixtureInMixture"
                ]
            },
```

```
            {
                "treeName": "CLP_PCN_SUBSTANCE",
                "activationClauses": [
                    "SubstanceInMixture"
                ]
            },
            {
                "treeName": "CLP_PCN_SFC",
                "activationClauses": [
                    "StandardFormulaComponent"
                ]
            },
            {
                "treeName": "CLP_PCN_ICG",
                "activationClauses": [
                    "InterchangeableComponentGroup"
                ]
            }
        ],
        "viewAsMappings": [
            {
                "source": "CONTACT",
                "target": "CUSTOM_ENTITY.PCN_Contact_Person"
            },
            {
                "source": "MIXTURE",
                "target": "CUSTOM_ENTITY.PCN_Mixture"
            },
            {
                "source": "REFERENCE_SUBSTANCE",
                "target": "CUSTOM_ENTITY.PCN_Reference_Substance"
            },
            {
                "source": "SUBSTANCE",
                "target": "CUSTOM_ENTITY.PCN_Substance"
            }
        ]
    },
    {
        "provider": "clp",
        "identifier": "CLP_PCN",
        "title": "CLP Poison centres notification",
        "applicableFor": "TEMPLATE",
```

```
        "dossierHeader": "DOSSIER.CLP_PCN",
        "mainTreeId": "CLP_PCN.TEMPLATE",
        "additionalTrees": [
            {
                "treeName": "CLP_PCN_MIM",
                "activationClauses": [
                    "MixtureInMixture"
                ]
            },
            {
                "treeName": "CLP_PCN_SUBSTANCE",
                "activationClauses": [
                    "SubstanceInMixture"
                ]
            },
            {
                "treeName": "CLP_PCN_SFC",
                "activationClauses": [
                    "StandardFormulaComponent"
                ]
            },
            {
                "treeName": "CLP_PCN_ICG",
                "activationClauses": [
                    "InterchangeableComponentGroup"
                ]
            }
        ],
        "viewAsMappings": [
            {
                "source": "CONTACT",
                "target": "CUSTOM_ENTITY.PCN_Contact_Person"
            },
            {
                "source": "MIXTURE",
                "target": "CUSTOM_ENTITY.PCN_Mixture"
            },
            {
                "source": "REFERENCE_SUBSTANCE",
                "target": "CUSTOM_ENTITY.PCN_Reference_Substance"
            },
            {
                "source": "SUBSTANCE",
```

```
                "target": "CUSTOM_ENTITY.PCN_Substance"
            }
        ]
    },
    {
        "provider": "clp",
        "identifier": "CLP_CLH",
        "title": "CLP Regulation - CLH dossier",
        "applicableFor": "SUBSTANCE",
        "dossierHeader": "DOSSIER.CLP_CLH",
        "mainTreeId": "CLP_CLH.SUBSTANCE"
    },
    {
        "provider": "clp",
        "identifier": "CLP_CLH",
        "title": "CLP Regulation - CLH dossier",
        "applicableFor": "TEMPLATE",
        "dossierHeader": "DOSSIER.CLP_CLH",
        "mainTreeId": "CLP_CLH.TEMPLATE"
    },
    {
        "provider": "clp",
        "identifier": "CLP_CLH_CAT",
        "title": "CLP Regulation - CLH dossier - category",
        "applicableFor": "CATEGORY",
        "additionalTrees": [
            {
                "treeName": "CLP_CLH",
                "activationClauses": [
                    "Substance"
                ]
            }
        ]
    }
]
```


The response will be a JSON array of objects, each object will contain the following fields:

Name	Type	Description
provider	string	The identifier of the definition provider for this working context
identifier	string	The identification code of the working context.

Name	Type	Description
title	string	The localized name of the working context, in the requested language via the **Accept-Language** header. If the header is omitted, or no localization exists for the requested language, English is used.
applicableFor	string	The entity type the working context applies to.
dossierHeader	string	If the working context support dossier creation, the definition identifier of the dossier header (e.g. DOSSIER.CLP_ALTERNATIVE_NAME)
mainTreeId	boolean	If the applicable entity is a complex one (SUBSTANCE, MIXTURE, TEMPLATE), the identifier of the section tree (e.g. CLP_ALTERNATIVE_NAME.SUBSTANCE)
additionalTrees	array	an optional list of additional trees to be used for complex entities other than the main entity of the dataset. Each item is an object containing: the "treeName", a string of the name of one tree (the name of a tree is the first part of the tree identifier. E.g. CLP_ALTERNATIVE_NAME) "activationClauses", a list of string that enumerate the conditions an additional entity must satisfy for this tree to be used. The conditions are only referenced by name, the logic of them is specified externally: see https://pmo.trasys.be/confluence/pages/viewpage.action?spaceKey=IUCLID6&title=Activation+clauses)

2.4. Trees (since v5.0.0)

2.4.1. Listing Trees

One can retrieve a listing of all registered section trees at the following resource:

GET	/definition/tree

Every listing resource supports the following query parameters:

Param	Type	Description	Default
formatter	String	the extension name of a eu.echa.iuclid6.ext.api.SectionTreeHandler. Available extensions are: • standard.Void (it does not give any representation for the tree) • standard.SectionNode (it gets the same representation as /definition/tree/R_ABOVE_1000?for=SUBSTANCE endpoint)	standard.Void
treeName	String	if specified, it filters the list by tree name	none
for	String	if specified, it only filters the list to keep only trees applicable for the specified entity type	none

The response is of the following format:

```
{
  "limit": 102,
  "totalCount": 102,
  "results": [
    {
      "uri": "i6def:/tree/R_A15_REST.SUBSTANCE",
      "representation": null //dependents on the specified formatter
    },
    {
      "uri": "i6def:/tree/R_A15_REST.TEMPLATE",
      "representation": null //dependents on the specified formatter
    },
   ...
  ]
}
```

2.4.2. Loading a single tree

One can retrieve a section tree with a specified tree name, applicable to a specific entity type at the following resources:

GET	/definition/tree/{tree name}?for={entity_type}
GET	/definition/tree/{tree name}.{entity_type}

The response type is not **application/json** as it should be, but **application/vnd.iuclid6.ext+json**

This is something that will be corrected in a subsequent version.

For example, to fetch the tree of the "*REACH Registration above 1000 tonnes*" submission type for substances:

GET	/definition/tree/R_ABOVE_1000?for=SUBSTANCE

or equivalently:

GET	/definition/tree/R_ABOVE_1000.SUBSTANCE

will return:


```
{
  "code": "R_ABOVE_1000",
  "title": "REACH Registration above 1000 tonnes",
  "sections": [
    {
      "code": "0",
      "title": "Related information",
      "sections": [],
      "documents": []
    },
    {
      "code": "1",
      "title": "General information",
      "sections": [
        {
          "code": "1.1",
          "title": "Identification",
          "sections": [],
          "documents": [
            {
              "code": "SUBSTANCE",
              "title": "Substance Identification",
              "access": "NO_ACCESS",
              "required": false,
```

```
                "requirement": "NOT_APPLICABLE",
            "single": true,
                "permissions": []
          }
        ]
      },
      {
        "code": "1.2",
        "title": "Composition",
        "sections": [],
        "documents": [
          {
            "code": "FLEXIBLE_RECORD.SubstanceComposition",
            "title": "Composition",
            "access": "NO_ACCESS",
            "required": true,
                "requirement": "REQUIRED"
            "single": false,
                "permissions": []
          }
        ]
      },
      {
        "code": "1.3",
        "title": "Identifiers",
        "sections": [],
        "documents": [
          {
            "code": "FIXED_RECORD.Identifiers",
            "title": "Identifiers",
            "access": "NO_ACCESS",
            "required": false,
                "requirement": "NOT_REQUIRED"
            "single": true,
                "permissions": []
          }
        ]
      }
...
```


The response is a tree-like object containing the following:

Name	Type	Description
code	string	The code of the current node
title	string	The localized name of the current node, in the requested language via the Accept-Language header. If the header is omitted, or no localization exists for the requested language, English is used.
sections	array	An array of section node objects.
documents	array	An array of objects representing the allowed document types for the current node.

Each entry of the **documents** field above, contain the following attributes:

Name	Type	Description
code	string	The code of the current node

Name	Type	Description
title	string	The localized name of the current node, in the requested language via the Accept-Language header. If the header is omitted, or no localization exists for the requested language, English is used.

Name	Type	Description
access	string	If the request is authenticated, it contains the privileges of the authenticated user on the specific document type. Can be one of: • NO_ACCESS • READ_ONLY • READ_WRITE • FULL_ACCESS When the request is unauthenticated it will always contain 'NO_ACCESS'
required	boolean	True if the specific document type is required for the submission. OBSOLETE check requirement value

Name	Type	Description
requirement	string	Defines whether a document need to be included in dataset during create dossier and export operations. One of: REQUIRED, the document must be included NOT_REQUIRED, the document can be optionally included, but it must be opted in for inclusion. OPTIONAL, the document can be optionally excluded, but it must be opted in for exclusion. NOT_APPLICABLE, the document must not be included CONTROLLED_BY_REFERRER, the document can be included only if another document referring to it is included
single	boolean	True if only one document of the specific type is allowed. e.g. The "Identification" of the substance
permissions	array	An array containing the semantic permissions a user has on the node, is derived from the access using the following logic: • NO_ACCESS → empty array • READ → ["LIST"] • READ_WRITE → ["LIST", "CREATE"] • FULL_ACCESS → ["LIST", "CREATE"]

2.5. Documents

In IUCLID6, the structure of the contents of a document greatly depends on the document's type. The actual structure of a document of a particular is defined by the "document definition".

The document definition consists of "definition elements" of several types, each one of them defining a different data element in a document.

The various definition elements can be divided in two groups: the *fields* and the *block.*

Fields define data structures that can only contain values. In contrast, the block can be considered as a container of other definition elements. It defines a grouping for the contained definition elements, which can be either fields or blocks.

Each definition element has a name, that is unique within its container, and by aggregating the names of each definition element, starting from the document definition, up to a particular element, we derive the "path" to this particular element. This path can be used to uniquely identify an element within a document

2.5.1. List document definitions

This resource returns only the name of each document definition by default.

GET	/definition/documents

It accepts one optional parameter:

Parameter	Description
formatter	If set to iuclid6.Definition, we can obtain the full contents of each definition

Example response


```
{
    "limit": null,
    "totalCount": 1135,
    "results": [
        {
            "uri": null,
            "representation": "CUSTOM_SECTION.ESR_FlashPoint_ApplicantSumAndC
oncl"
        },
        {
            "uri": null,
            "representation": "ENDPOINT_SUMMARY.ToxicityToTerrestrialPlants"
        },
        {
            "uri": null,
            "representation": "ENDPOINT_STUDY_RECORD.LongTermToxicityToAquaIn
v"
        },
        {
…
```


2.5.2. Load single document definition

The document definition for a particular document type can be retrieved on the following resource:

GET	/definition/document/{document_type}

The request needs the following header to provide the contents of the document definition:

Accept	application/vnd.iuclid6.ext+json;type=iuclid6.Definition

If we request the definition for the "LEGAL_ENTITY" document

GET	/definition/document/LEGAL_ENTITY

We will get the following example response:


```
{
  "identifier": "LEGAL_ENTITY",
  "version": "2.0",
  "provider": "domain",
  "@lang": "en",
  "contents": [
    {
      "type": "block",
      "name": "GeneralInfo",
      "title": "General information",
      "contents": [
        {
          "type": "text",
          "name": "LegalEntityName",
          "title": "Legal entity name",
          "required": true,
          "mimeType": "text/plain",
          "maxLength": 255
        },
        {
          "type": "picklist",
          "name": "LegalEntityType",
          "title": "Legal entity type",
          "phrasegroup": "N01"
        },
        {
          "type": "text",
          "name": "Remarks",
          "title": "Remarks",
          "mimeType": "text/plain",
          "maxLength": 32768
        },
        {
          "type": "block",
          "name": "OtherNames",
          "title": "Other names",
          "protectedBy":
"LEGAL_ENTITY.GeneralInfo.OtherNames.DataProtection",
          "multiple": true,
```

```
          "contents": [
            {
              "type": "dataProtection",
              "name": "DataProtection",
              "title": "Flags"
            },
            {
              "type": "text",
              "name": "Name",
              "title": "Name",
              "mimeType": "text/plain",
              "maxLength": 255
            }
          ]
        }
      ]
    },
...
```

For example, the "**path**" of the "LegalEntityName" text element shown above, is: "LEGAL_ENTITY.GeneralInfo.LegalEntityName"

See below for the description of the fields included in the response object.

Name	Type	Description
identifier	string	The document type of the definition, e.g., LEGAL_ENTITY, ENDPOINT_SUMMARY.ToxicityToBirds etc.
version	string	The version of the document definition
provider	string	The identifier of the definition provider, providing this document definition
@lang	string	The locale used to localize the response
contents	array	An array of definition element objects

Name	Type	Description
type	string	The specific type of the element: Can be one of: • block • dataProtection • text • boolean • date • numeric • picklist • quantity • range • docRef • blockRef • inventoryRef • definitionRef • attachment • address
name	string	The name of the element, which uniquely identifies it, within its direct container
title	string	The localized name of the element, in the requested language via the Accept-Language header. If the header is omitted, or no localization exists for the requested language, English is used.
protectedBy	string	The path to a "dataProtection" field, that impose confidentiality claims upon the field.
multiple	boolean	Whether the data content consists of a single or multiple value.
required	boolean	Whether the data element should be populated for the document to be valid.

Depending on the type, each element can support additional properties:

Element type	Name	Type	Description
block	contents	array	An array of definition element objects, contained in this block

Element type	Name	Type	Description
text	mimeType	string	The mime-type of the text contents of the field. Can be either: **text/plain** or **text/html**
	maxLength	number	The maximum length the field support. *Omitted if unrestricted*
	textTemplates	array	The codes to relevant text-templates that can be used to populate the field
date	withTime	boolean	True, if the field contain time along with the date
numeric	numericType	string	• INTEGER • DOUBLE
	min	number	The minimum allowed value, if any
	max	number	The maximum allowed value, if any
picklist	phrasegroup	string	The code of the phrase-group, this field's allowed values can come from
	defaultPhrase	string	The default phrase, if any
	remarks	boolean	True, if additional text can be added to the field.
	remarksMaxLength	number	The maximum supported length of the additional text, if supported
quantity	unitPhrasegroup	string	The code of the phrase-group, containing the units for this field
range	numericType	string	• INTEGER • DOUBLE
	unitPhrasegroup	string	The code of the phrase-group, containing the units for this field
	isHalfBounded	boolean	True, if only one bound is supported.
	boundQualifiers	array	The allowed boundary qualifiers. Depending on whether the field is half-bounded,
	lowerBoundQualifiers		

Element type	Name	Type	Description
	upperBoundQualifiers		either only the boundQualifiers or both the lowerBoundQualifiers and upperBoundQualifiers, will be present. The allowed values are: • ca. • < • <= • >> • >=
docRef	referenceTypes	array	A list of the types of the documents, that the field can refer to.
blockRef	referenceTypes	array	A list of the relative path to the blocks, that the field can refer to
attachment	mimeType	string	The mime-type of the attachment, this field accepts. Can be one of the following: • image/* • */* •

2.6. Phrase-groups

Phrase-groups organize several phrases together in a set identified by a code. Several IUCLID fields (e.g. picklists, physical quantity, etc) use this code to indicate the phase codes they accept.

One can retrieve the phases of a particular phrase-group on the following resource:

GET	/definition/phrasegroup/{phrasegroup-code}/phrases

The response is a JSON array, the entries of which contain the **phrase** object and a boolean indicating whether the phrase is **obsolete** and it should not be used.

example response for phrase-group N16

```
[
  {
    "phrase": {
      "code": "2820",
      "text": "final",
```

```
      "description": "",
      "open": false
    },
    "obsolete": false
  },
  {
    "phrase": {
      "code": "2793",
      "text": "draft",
      "description": "",
      "open": false
    },
    "obsolete": false
  },
  {
    "phrase": {
      "code": "1342",
      "text": "other:",
      "description": "",
      "open": true
    },
    "obsolete": false
  }
]
```


The `phrase` object contains the following attributes:

Name	Type	Description
code	string	The code of the phrase
text	string	The localized name of the phrase, in the requested language via the Accept-Language header. If the header is omitted, or no localization exists for the requested language, English is used.
description	string	Explanatory notes for the phrase.
open	boolean	True if the phrase can contain additional text (a.k.a. "other text")

2.7. Phrases

The following resource, contains all registered phrases in the system, without any grouping:

GET	/definition/phrases

It returns an array of every phrase:

```
[
  {
    "code": "1",
    "text": "ISO/DIS 17556.2 (Plastics - Determination of the Ultimate
Aerobic Biodegradability in Soil by Measuring the Oxygen Demand in a
Respirometer or the Amount of Carbon Dioxide Evolved)",
    "description": "",
    "open": false
  },
  {
    "code": "10",
    "text": "#8",
    "description": "",
    "open": false
  },
  {
    "code": "100",
    "text": "Arenicola marina",
    "description": "",
    "open": false
  },...
```

2.8. Text templates

Text templates are pre-defined texts which appear in IUCLID text fields to guide users on what information to enter and act like a guidance template

2.8.1. List text templates

The following resource contains all registered text templates in the system:

GET	/definition/texts

It returns the list of text templates

Example response:

```
{
  "limit": null,
  "totalCount": 438,
  "results": [
```

```
    {
      "uri": null,
      "representation": {
        "code": "TT_1",
        "provider": "oecd",
        "header": "",
        "content": "PHYSICO-CHEMICAL PROPERTIES\n - Vapour pressure:\n -
Water solubility: \n - log Pow: \n - pKa: \n - Base or acid catalysis of test
material: \n - UV absorption:\n - Stability of test material at room
temperature: \n \n OTHER PROPERTIES (if relevant for this endpoint)"
      }
    },
    {
      "uri": null,
      "representation": {
        "code": "TT_10",
        "provider": "oecd",
        "header": "",
        "content": "PHYSICO-CHEMICAL PROPERTIES\n - Vapour pressure:\n -
Henry's Law constant: \n - log Pow: \n - pKa: \n - UV absorption:\n -
Stability of test material at room temperature: \n \n OTHER PROPERTIES (if
relevant for this endpoint)"
      }
    },
...
```


Each text template object contains the following:

Name	Type	Description
code	string	The identification code of the text template
provider	string	The identifier of the definition provider, providing this text template
header	string	A title for this text template.
content	string	The actual template content.

2.8.2. Load text templates

In addition, one can retrieve a specific text template by its identification code on the following resource:

GET	/definition/texts/{code}

3. Search

Every REST resource in this document is under /iuclid6-ext/api/ext/v1/ .

The IUCLID6 Rest API supports paged document searches by using predefined search queries, each accepting several parameters.

GET	/query/iuclid6/{query-name}

The {query-name} can get any of the following values:

- byDate
- byName
- byContact
- bySubstance
- byMixture
- byLegalEntity
- byReferenceSubstance
- byTemplate
- byLiterature
- byAnnotation
- bySite
- byAdditive
- byImpurity
- byConstituent
- byGhs
- byDsd
- byJointSubmission
- byCategory

Section 3.2 presents parameters relevant to each of the query types. Every search query supports the following query parameters:

Param	Type	Description	Default
count	boolean	include the total count of the results. If the total count is not needed, omitting it improves performance.	true
l	integer	the number of results (limit) per page.	10
o	integer	the offset of the first result in page. e.g. to fetch the 3rd page of 25 (l=25) results specify o=50 (o = l * zero_indexed_page_number)	0
formatter	string	used to indicate the format of the search response Make it "iuclid6.DocumentSecuredRepresentation" to obtain the default format response.	“”

order	string	see section 3.1.2	“touched-&modified-&uuid”

Refer to section 3.2 for parameters relevant to each of the query types.

The following query fetches the second page of legal entity documents with 20 items per page, including the total count of the legal entity documents.

```
curl --location --request GET 'http://localhost:8080/iuclid6-
ext/api/ext/v1/query/iuclid6/byType?doc.type=SUBSTANCE&l=20&o=20&count=true'
\
--header 'iuclid6-user: SuperUser' \
--header 'iuclid6-pass: root'
```

3.1. Results

The search results are wrapped inside a standard payload that contains:

- the results of the search
- plus some additional information to support paging, such as
  - the limit
  - the offset
  - and the total count of the results.

Each result comprises of two values, the URI and the representation.

For the query

```
curl --location --request GET 'http://localhost:8080/iuclid6-
ext/api/ext/v1/query/iuclid6/byType?doc.type=SUBSTANCE' \
--header 'iuclid6-user: SuperUser' \
--header 'iuclid6-pass: root'
```

IUCLID will return a response like the one below:

```
{
  "limit": 10,
  "totalCount": 102,
  "results": [
    {
      "uri": "iuclid6:/0/SUBSTANCE/f200c268-7a94-4fb6-9d1d-
6759d71f6894/SUBSTANCE/f200c268-7a94-4fb6-9d1d-6759d71f6894",
      "representation": null
    },
    {
      "uri": "iuclid6:/0/SUBSTANCE/d4e388de-aa35-4568-b25e-
39a6538363a6/SUBSTANCE/d4e388de-aa35-4568-b25e-39a6538363a6",
      "representation": null
```

```
    },
    {
      "uri": "iuclid6:/0/SUBSTANCE/c61b4b3a-ffbc-46b4-a5dc-
f3ccb512c11a/SUBSTANCE/c61b4b3a-ffbc-46b4-a5dc-f3ccb512c11a",
      "representation": null
    },
    {
      "uri": "iuclid6:/0/SUBSTANCE/db63a1af-6304-427f-aa1d-
387b1f8fefb5/SUBSTANCE/db63a1af-6304-427f-aa1d-387b1f8fefb5",
      "representation": null
    },
    {
      "uri": "iuclid6:/0/SUBSTANCE/IUC5-f0958d98-4cb3-4058-80cd-
bdc24e7d125d/SUBSTANCE/IUC5-f0958d98-4cb3-4058-80cd-bdc24e7d125d",
      "representation": null
    },
    {
      "uri": "iuclid6:/0/SUBSTANCE/40374a6f-9175-4ac8-87c9-
d220c866997a/SUBSTANCE/40374a6f-9175-4ac8-87c9-d220c866997a",
      "representation": null
    },
    {
      "uri": "iuclid6:/0/SUBSTANCE/IUC4-f696abe6-513a-3771-b2ee-
76e4110e8103/SUBSTANCE/IUC4-f696abe6-513a-3771-b2ee-76e4110e8103",
      "representation": null
    },
    {
      "uri": "iuclid6:/0/SUBSTANCE/326ab136-d74c-4088-9425-
8d9d5642b96e/SUBSTANCE/326ab136-d74c-4088-9425-8d9d5642b96e",
      "representation": null
    },
    {
      "uri": "iuclid6:/0/SUBSTANCE/1ef71d52-ddc7-4b55-9c2f-
ea72efa55ddd/SUBSTANCE/1ef71d52-ddc7-4b55-9c2f-ea72efa55ddd",
      "representation": null
    },
    {
      "uri": "iuclid6:/0/SUBSTANCE/7cec4254-65f7-4d9a-9a6a-
2a74dd1c9b1c/SUBSTANCE/7cec4254-65f7-4d9a-9a6a-2a74dd1c9b1c",
      "representation": null
    }
  ]
}
```


The above shows a successful response back from the call. In this case SUBSTANCE documents were searched for and there are 102 results. The first 10 are contained in the response.

Since no formatter was specified, the representation is null.

The query results contain the following fields:

Name	Type	Description
limit	integer	the number of the results contained in the returned page.
offset	integer	the offset of the first result in page.
totalCount	integer	the total count of records that match the search. Because of pagination, this is not necessarily the number of records in the response
results	Object[]	The actual results of the search. See below.

To retrieve the results with the default IUCLID6 representation one need to set the "formatter" query parameter to "iuclid6.DocumentSecuredRepresentation".

```
curl --location --request GET 'http://localhost:8080/iuclid6-
ext/api/ext/v1/query/iuclid6/byType?doc.type=SUBSTANCE&o=0&l=1&formatter=iucl
id6.DocumentSecuredRepresentation' \
--header 'iuclid6-user: SuperUser' \
--header 'iuclid6-pass: root'
```

Here 's an example of a response that could be potentially returned by the previous request:

QueryResults


```
{
  "limit": 1,
  "offset": 0,
  "totalCount": 102,
  "results": [
    {
      "uri": "iuclid6:/0/SUBSTANCE/f200c268-7a94-4fb6-9d1d-
6759d71f6894/SUBSTANCE/f200c268-7a94-4fb6-9d1d-6759d71f6894",
      "representation": {
        "classtype": "SubstanceSecuredRepresentation",
        "accessRight": "READ_ONLY",
        "key": "f200c268-7a94-4fb6-9d1d-6759d71f6894/0",
        "name": "SUB WITH IMAGE",
        "createdOn": "2016-09-06T16:14:59.187Z",
        "modifiedOn": "2016-09-06T15:00:51.000Z",
        "publicName": "the public name",
        "legalEntityRepresentation": {
          "classtype": "LegalEntitySecuredRepresentation",
```

```
          "accessRight": "READ_ONLY",
          "key": "4f88bc7f-395c-4d0b-997b-14e8c9aef605/0",
          "name": "Predefined Legal entity",
          "createdOn": "2016-09-06T16:14:59.187X",
          "city": null,
          "country": {
            "code": null
          },
          "definition": "LEGAL_ENTITY",
          "modifiedOn": "2016-09-01T15:40:27.000Z",
          "accessAllowed": true
        },
        "definition": "SUBSTANCE",
        "accessAllowed": true
      }
    }
  ]
}
```


Each result contains the following two properties:

Name	Type	Description
uri	string	a URI of the document.
representation	object	an object containing the representation of the result. The exact format is governed by the "formatter" query param. If omitted, null is returned.

3.1.1. Document Secured Representation

The **DocumentSecuredRepresentation** defines a hierarchy of types, that represent the different entity types of IUCLID6.

Each result is represented, according to its document type, by the appropriate sub-type.

However, every sub-type includes the following properties:

Name	Type	Description
classtype	String	the sub-type of the document representation. e.g. SubstanceSecuredRepresentation
key	String	the document key formatted as {uuid}/{snapshot}.

definition	String	the document definition identifier. e.g. ENDPOINT_STUDY_RECORD.Partition
name	String	the name of the document
accessRight	String	the access right the user has on the document. Possible values are: • NO_ACCESS • READ_ONLY • READ_WRITE • FULL_ACCESS
createdOn	String	the instant the document was created, in ISO 8601 format e.g. 2016-09-06T13:14:59.187Z
modifiedOn	String	the instant the document was last modified, in ISO 8601 format e.g. 2016-09-01T07:20:11Z
remarks	String	the optional, remarks of the document

3.1.2. Ordering

The results are ordered by default:

- by descending touched date
- then by descending modification date
- and finally by ascending uuid.

Touched date

As touched date is considered the latest of the modification and creation date.

In case of import, the modification date can precede the creation date of a document, since as creation is the date when the import takes place, while the modification date is the respective field in the imported archive.

One can change the result ordering by specifying the `**order**` parameter. The available values for the **order** parameter are:

Parameter value	Description
touched	the latest of the modification and creation date
modified	the modification date
created	the creation date

uuid	the uuid of the document

To reverse order on a value, append the minus sign "-" to the end of the name.

For example, the default ordering, would be specified as:

GET	/query/iuclid6/byType?doc.type=SUBSTANCE&order=touched-&modified-&uuid

3.2. Available queries

The parameter values can be of the following types:

- **String**
- **Picklist**, formatted as `code[:other_text]` *e.g. 55446 or 1342:other_text_value*
- **Date**, formatted as `yyyy-MM-dd` *e.g. 2016-07-24*
- **Boolean**, true/false

The default type of each parameter is **String**, unless explicitly specified.

Every query supports the following parameters:

Parameter	Description
doc.type	The type of the document. In general, the following IUCLID6 document types are supported: • DOSSIER • SUBSTANCE • MIXTURE • TEMPLATE • CATEGORY • ANNOTATION • LEGAL_ENTITY • SITE • REFERENCE_SUBSTANCE • CONTACT • LITERATURE • TEST_MATERIAL_INFORMATION However, some queries might restrict the accepted values.
group	The name of group that the document should be shared to.
owner	The owner of the document. Can be one of: • ME • OTHER

The following sub-chapters describe the available queries.

3.2.1. byDate

Find entities by creation/modification date.

GET	/query/iuclid6/byDate

Supported query specific parameters:

Parameter name	Description
created.after	Date. The document should be created from the specified date and afterwards, e.g. "2016-06-27"
created.before	Date. The document should be created until before the specified date. e.g. "2016-07-27"
modified.after	Date. The document should be last modified from the specified date and afterwards, e.g. "2016-06-27"
modified.before	Date. The document should be last modified until before the specified date. e.g. "2016-07-27"

3.2.2. byName

Find entities of specific name.

GET	/query/iuclid6/byName

Supported query specific parameters:

Parameter name	Description
doc.name	The name of the document

3.2.3. byContact

Find contacts by various criteria.

GET	/query/iuclid6/byContact

Note: The doc.type parameter should be omitted.

Supported query specific parameters:

Parameter name	Description
contact.first_name	The first name of the contact
contact.last_name	The last name of the contact
contact.organisation	The organization of the contact

3.2.4. bySubstance

Find entities, being themselves or referring to, 'Substances' satisfying certain criteria.

GET	/query/iuclid6/bySubstance

Note: the doc.type parameter can be one of:

- SUBSTANCE
- MIXTURE
- DOSSIER

Depending on the value of doc.type parameter, the following criteria are applied:

- in case of SUBSTANCE, directly to the substance document
- in case of MIXTURE, to the linked substance of the MixtureComposition of a mixture
- in case of DOSSIER, to any dossier component of type SUBSTANCE

Supported query specific parameters:

Parameter name	Description
sub.chemical	The substance chemical name
sub.owner	The name of the legal entity owner
sub.other	The other names
ref_sub.name	The name of the reference substance that is linked through field: SUBSTANCE.ReferenceSubstance.ReferenceSubstance.
ref_sub.inv.cas_number	The CAS number in the chemical inventory of the linked reference substance.
ref_sub.inv.name	The name in the chemical inventory of the linked reference substance.

ref_sub.inv.number	The number in the chemical inventory of the linked reference substance.
ref_sub.cas_number	The CAS number of the linked reference substance.
ref_sub.cas_name	The CAS name of the linked reference substance.
ref_sub.iupac_name	The IUPAC name of the linked reference substance.
ref_sub.smiles_notation	The smiles notation of the linked reference substance.
ref_sub.mol_formula	The molecular formula of the linked reference substance.
ref_sub.in_chl	The inChl of the linked reference substance.
sub.reg_prog_id	The regulatory programme identifier of the FIXED_RECORD.Identifiers child of the SUBSTANCE.
sub.reg_prog	**Picklist**. The regulatory programme of the FIXED_RECORD.Identifiers child of the SUBSTANCE. Should be one of the phase codes in Phrase-group **N12**
sub.it_system_id	The IT System Identifier of the FIXED_RECORD.Identifiers child of the SUBSTANCE.

3.2.5. byMixture

Find entities, being themselves or referring to, 'Mixtures' satisfying certain criteria.

GET	/query/iuclid6/byMixture

Note: the doc.type parameter can be one of:

- MIXTURE
- DOSSIER

Depending on the value of doc.type parameter, the following criteria are applied:

- in case of MIXTURE, directly to the mixture document
- in case of DOSSIER, to any dossier component of type MIXTURE

Supported query specific parameters:

Parameter name	Description
mix.name	The mixture name.
mix.other_name	The name in the 'Other_names'
sub.reg_prog_id	The regulatory programme identifier of the FIXED_RECORD.Identifiers child of the MIXTURE.
sub.reg_prog	**Picklist**. The regulatory programme of the FIXED_RECORD.Identifiers child of the MIXTURE. Should be one of the phase codes in Phrase-group **N12**
sub.it_system_id	The IT System Identifier of the FIXED_RECORD.Identifiers child of the MIXTURE.

3.2.6. byLegalEntity

Find entities, being themselves or referring to, 'legal entities' satisfying certain criteria.

GET	/query/iuclid6/byLegalEntity

Note: the doc.type parameter can be one of:

- LEGAL_ENTITY
- SUBSTANCE
- MIXTURE
- TEMPLATE
- CATEGORY
- DOSSIER

Depending on the value of doc.type parameter, the following criteria are applied:

- in case of LEGAL_ENTITY, directly to the legal entity document
- in case of DOSSIER, to the submitting legal entity of the dossier.
- the owner legal entity, for the rest.

Supported query specific parameters:

Parameter name	Description
le.name	The legal entity name.
le.country	**Picklist.** The country of the legal entity. Should be one of the phase codes in Phrase-group **N03**

le.town	The town of the legal entity.

3.2.7. byReferenceSubstance

Find entities, being themselves or referring to, 'Reference substances' satisfying certain criteria.

GET	/query/iuclid6/byReferenceSubstance

Note: the doc.type parameter can be one of:

- REFERENCE_SUBSTANCE
- SUBSTANCE
- DOSSIER

Depending on the value of doc.type parameter, the following criteria are applied:

- in case of REFERENCE_SUBSTANCE, directly to the reference substance document
- in case of DOSSIER, to the reference substance linked to the dossier subject.
- in case of SUBSTANCE, the reference substance linked to the substance

Supported query specific parameters:

Parameter name	Description
ref_sub.name	The name of the reference substance.
ref_sub.inv.cas_number	The cas number in the chemical inventory.
ref_sub.inv.name	The name in the chemical inventory.
ref_sub.inv.number	The number in the chemical inventory.
ref_sub.cas_number	The CAS number.
ref_sub.cas_name	The CAS name.
ref_sub.iupac_name	The IUPAC name.
ref_sub.smiles_notation	The smiles notation.
ref_sub.mol_formula	The molecular formula.
ref_sub.in_chl	The inChl of the linked reference substance.

3.2.8. byTemplate

Find 'Templates' satisfying certain criteria.

GET	/query/iuclid6/byTemplate

Note: the doc.type parameter should be omitted.

Supported query specific parameters:

Parameter name	Description
doc.name	The template name.
le.name	The legal entity name.
le.country	**Picklist**. The country of the legal entity. Should be one of the phase codes in Phrase-group **N03**
le.town	The town of the legal entity.

3.2.9. byLiterature

Find 'Literature references' satisfying certain criteria.

GET	/query/iuclid6/byLiterature

Note: the doc.type parameter should be omitted.

Supported query specific parameters:

Parameter name	Description
lit.title	The literature title.
lit.type	**Picklist**. The reference type of the literature. Should be one of the phase codes in Phrase-group **Z31**
lit.author	The author of the literature.
lit.ref_year	The year of the literature.
lit.source	The bibliographic source.
lit.test_lab	The testing laboratory

lit.owner	The company owner.
lit.owner_study_no	The company study no.
lit.report_no	The report no.
lit.report.after	Date. The report should be from the specified date and afterwards, e.g. "2016-06-27"
lit.report.before	Date. The document should be created until before the specified date. e.g. "2016-07-27"

3.2.10. byAnnotation

Find annotations by various criteria.

GET	/query/iuclid6/byAnnotation

Note: the doc.type parameter should be omitted.

Supported query specific parameters:

Parameter name	Description
annotation.status	**Picklist**. The status of the annotation Should be one of the phase codes in Phrase-group **N16**
annotation.authority	The name of the authority/organisation of the annotation.
annotation.agreement	Boolean, Is the agreement with the applicant's summary checked.
annotation.data_waiver	**Picklist**. The data waiver acceptable of the literature. Should be one of the phase codes in Phrase-group **N19**
annotation.reliability	**Picklist**. The reliability of the literature. Should be one of the phase codes in Phrase-group **N17**

3.2.11. bySite

Find entities, being themselves or referring to, 'Sites' satisfying certain criteria.

GET	/query/iuclid6/bySite

Note: the doc.type parameter can be one of:

- SITE
- SUBSTANCE
- DOSSIER

Depending on the value of doc.type parameter, the following criteria are applied:

- in case of SITE, directly to the site document
- in case of SUBSTANCE, the SITE linked to a FLEXIBLE_RECORD.Sites of a substance
- in case of DOSSIER, to any dossier component of type SITE.

Supported query specific parameters:

Parameter name	Description
site.name	The name of the site.
site.country	**Picklist**. The country of the site. Should be one of the phase codes in Phrase-group **N03**
site.town	The town of the site.
site.owner	The name of the legal entity owner of the site.

3.2.12. byAdditive

Find entities having specific reference substances as additives.

GET	/query/iuclid6/byAdditive

Note: the doc.type parameter can be one of:

- SUBSTANCE
- MIXTURE
- DOSSIER

Depending on the value of doc.type parameter, the following criteria are applied:

- in case of SUBSTANCE, the linked reference substance through the additives field of a SubstanceComposition of a substance
- in case of MIXTURE, the linked reference substance through the additives field of a MixtureComposition of a mixture
- in case of DOSSIER, the linked reference substance through the additives field of a SubstanceComposition or MixtureComposition of the dossier's subject

Supported query specific parameters:

Parameter name	Description
ref_sub.name	The name of the reference substance that is linked through field: SUBSTANCE.ReferenceSubstance.ReferenceSubstance.
ref_sub.inv.cas_number	The cas number in the chemical inventory of the linked reference substance.
ref_sub.inv.name	The name in the chemical inventory of the linked reference substance.
ref_sub.inv.number	The number in the chemical inventory of the linked reference substance.
ref_sub.cas_number	The CAS number of the linked reference substance.
ref_sub.cas_name	The CAS name of the linked reference substance.
ref_sub.iupac_name	The IUPAC name of the linked reference substance.
ref_sub.smiles_notation	The smiles notation of the linked reference substance.
ref_sub.mol_formula	The molecular formula of the linked reference substance.
ref_sub.in_chl	The inChl of the linked reference substance.

3.2.13. byImpurity

Find entities having specific reference substances as impurities.

GET	/query/iuclid6/byImpurity

Note: the doc.type parameter can be one of:

- SUBSTANCE
- MIXTURE
- DOSSIER

Depending on the value of doc.type parameter, the following criteria are applied:

- in case of SUBSTANCE, the linked reference substance through the impurities field of a SubstanceComposition of a substance
- in case of MIXTURE, the linked reference substance through the impurities field of a MixtureComposition of a mixture
- in case of DOSSIER, the linked reference substance through the impurities field of a SubstanceComposition or MixtureComposition of the dossier's subject

Supported query specific parameters:

Parameter name	Description
ref_sub.name	The name of the reference substance that is linked through field: SUBSTANCE.ReferenceSubstance.ReferenceSubstance.
ref_sub.inv.cas_number	The cas number in the chemical inventory of the linked reference substance.
ref_sub.inv.name	The name in the chemical inventory of the linked reference substance.
ref_sub.inv.number	The number in the chemical inventory of the linked reference substance.
ref_sub.cas_number	The CAS number of the linked reference substance.
ref_sub.cas_name	The CAS name of the linked reference substance.
ref_sub.iupac_name	The IUPAC name of the linked reference substance.
ref_sub.smiles_notation	The smiles notation of the linked reference substance.
ref_sub.mol_formula	The molecular formula of the linked reference substance.
ref_sub.in_chl	The inChl of the linked reference substance.

3.2.14. byConstituent

Find entities having specific reference substances as constituent.

GET	/query/iuclid6/byConstituent

Note: the doc.type parameter can be one of:

- SUBSTANCE
- MIXTURE
- DOSSIER

Depending on the value of doc.type parameter, the following criteria are applied:

- in case of SUBSTANCE, the linked reference substance through the constituents field of a SubstanceComposition of a substance
- in case of MIXTURE, the linked reference substance through the components field of a MixtureComposition of a mixture
- in case of DOSSIER, the linked reference substance through the constituents field of a SubstanceComposition or the **components** field MixtureComposition of the dossier's subject

Supported query specific parameters:

Parameter name	Description
ref_sub.name	The name of the reference substance that is linked through field: SUBSTANCE.ReferenceSubstance.ReferenceSubstance.
ref_sub.inv.cas_number	The cas number in the chemical inventory of the linked reference substance.
ref_sub.inv.name	The name in the chemical inventory of the linked reference substance.
ref_sub.inv.number	The number in the chemical inventory of the linked reference substance.
ref_sub.cas_number	The CAS number of the linked reference substance.
ref_sub.cas_name	The CAS name of the linked reference substance.
ref_sub.iupac_name	The IUPAC name of the linked reference substance.
ref_sub.smiles_notation	The smiles notation of the linked reference substance.
ref_sub.mol_formula	The molecular formula of the linked reference substance.
ref_sub.in_chl	The inChl of the linked reference substance.

3.2.15. byGhs

Find entities containing GHS sections with certain criteria.

GET	/query/iuclid6/byGhs

Note: the doc.type parameter can be one of:

- SUBSTANCE
- MIXTURE
- DOSSIER

Depending on the value of doc.type parameter, the following criteria are applied:

- in case of SUBSTANCE or MIXTURE, to the FLEXIBLE_RECORD.Ghs of a substance
- in case of DOSSIER, to the FLEXIBLE_RECORD.Ghs of the dossier's subject

Supported query specific parameters:

Parameter name	Description
ghs.not_classified	**Boolean**. Is the 'Not classified' field checked?
ghs.hazard_statement	**Picklist**. The hazard statement (*path: FLEXIBLE_RECORD.Ghs.Labelling.HazardStatementsBlock.HazardStatements.HazardStatement*) Should be one of the phase codes in Phrase-group **GHS65**
ghs.hazard_pictogram	**Picklist**. The hazard pictogram (*path: FLEXIBLE_RECORD.Ghs.Labelling.HazardPictogramBlock.HazardPictogram.Code*) Should be one of the phase codes in Phrase-group **DM02**
ghs.precaution_statement	**Picklist**. The precautionary statement (*path: FLEXIBLE_RECORD.Ghs.Labelling.PrecautionaryStatementsBlock.PrecautionaryStatements.PrecautionaryStatement*) Should be one of the phase codes in Phrase-group **GHS66**

3.2.16. byDsd

Find entities containing DSD sections with certain criteria.

GET	/query/iuclid6/byDsd

Note: the doc.type parameter can be one of:

- SUBSTANCE
- MIXTURE
- DOSSIER

Depending on the value of doc.type parameter, the following criteria are applied:

- in case of SUBSTANCE or MIXTURE, to the FLEXIBLE_RECORD.DsdDpd of a substance
- in case of DOSSIER, to the FLEXIBLE_RECORD.DsdDpd of the dossier's subject

Supported query specific parameters:

Parameter name	Description
dsd.not_classified	**Boolean**. Is the 'Not classified' field checked?

dsd.risk	**Picklist**. The risk phrases (*path: FLEXIBLE_RECORD.DsdDpd.Labelling.RiskPhrases.Risks*) Should be one of the phase codes in Phrase-group **N29**
dsd.danger	**Picklist**. The indication of danger (*path: FLEXIBLE_RECORD.DsdDpd.Labelling.IndicationsOfDangerBlock.IndicationsOfDanger*) Should be one of the phase codes in Phrase-group **N30**
dsd.safety	**Picklist**. The safety phrases (*path: FLEXIBLE_RECORD.DsdDpd.Labelling.SafetyPhrases.Safeties.Code*) Should be one of the phase codes in Phrase-group **N31**

3.2.17. byJointSubmission

Find entities containing 'Joint submission' sections with certain criteria.

GET	/query/iuclid6/byJointSubmission

Note: the doc.type parameter can be one of:

- SUBSTANCE
- MIXTURE
- DOSSIER

Depending on the value of doc.type parameter, the following criteria are applied:

- in case of SUBSTANCE or MIXTURE, to the FLEXIBLE_RECORD.JointSubmission of a substance
- in case of DOSSIER, to the FLEXIBLE_RECORD.JointSubmission of the dossier's subject

Supported query specific parameters:

Parameter name	Description
joint_submission.name	The name of the joint submission
joint_submission.leader	The name of the leader legal entity.
joint_submission.member	The name of the member legal entity.

3.2.18. byCategory (since v2.0.0)

Find CATEGORY entities by various criteria.

GET	/query/iuclid6/byCategory

Note: The doc.type parameter should be omitted.

Supported query specific parameters:

Parameter name	Description
doc.name	The category name
cat.public_name	The public name of the category
sub.uuid	The uuid of the substance category member
le.name	The legal entity name.
le.country	**Picklist**. The country of the legal entity. Should be one of the phase codes in Phrase-group **N03**
le.town	The town of the legal entity.

3.2.19. SubstanceRelatedCategories

Find CATEGORY entities that are related to a Substance. A Category is considered related to a given Substance, if its 'Substances' field references the Substance.

GET	/query/iuclid6/SubstanceRelatedCategories

Supported query specific parameters:

Parameter name	Description
key	The document key of the substance

3.2.20. SubstanceRelatedMixtures

Find MIXTURE entities that are related to a Substance. A Mixture is considered related to a given Substance, if it has a Mixture Composition document that has the Substance as component, impurity or additive.

GET	/query/iuclid6/SubstanceRelatedMixtures

Supported query specific parameters:

Parameter name	Description
key	The document key of the substance

3.2.21. ByUuid (since v6.0.0)

Find an entity by its UUID.

GET	/query/iuclid6/ByUuid

Supported query specific parameters:

Parameter name	Description
uuid	The entity's UUID

3.2.22. ByDossierUuid (since v6.0.0)

Find a dossier by its UUID. Similar to `byUuid`, but returns only dossiers.

GET	/query/iuclid6/ByDossierUuid

Supported query specific parameters:

Parameter name	Description
uuid	The dossier's UUID

3.2.23. ByDossierSubjectUuid (since v6.0.0)

Find a dossier by its subject's UUID.

GET	/query/iuclid6/ByDossierSubjectUuid

Supported query specific parameters:

Parameter name	Description
uuid	The dossier's subject UUID

3.2.24. ByDefinitionIdentifier (since v6.0.0)

Returns all the documents of an entity of a specific definition id.

GET	/query/iuclid6/ByDefinitionIdentifier

Supported query specific parameters:

Parameter name	Description
def_id	The document definition identifier. If this parameter is not set, all the entity's documents will be returned.
from	The UUID of the entity to search in.

4. Document handling

Every REST resource in this document is under /iuclid6-ext/api/ext/v1/

4.1. Datatypes

4.1.1. JsonDocumentEnvelope

A IUCLID6 document is represented by a JsonDocumentEnvelope object, that is a JSON array of two elements, the **JsonDocumentHeader** being the first, followed by the **JsonDocumentContent**.

Media type

The media type for the JsonDocumentEnvelope object is **application/vnd.iuclid6.ext+json;type=iuclid6.Document**

To give an example, the following snippet represents a SubstanceComposition section document:

JsonDocumentEnvelope

```
[
  {
    "key": "7bcaf4db-9f2c-403f-aa79-c2cadef74a0e/0",
    "definition": "FLEXIBLE_RECORD.SubstanceComposition",
    "parentKey": "9915ba13-ffa7-405b-b024-c06296b43820/0",
    "parentDefinition": "SUBSTANCE",
   "order": 1,
    "name": "Composition.001",
   "attachments": [
      "e1c921f6-5c85-4e0a-a5c5-33fabade0a2f/0",
      "cd166edf-1d7f-49eb-b1c0-d7424c28de1f/0"
    ],
    "createdOn": "2016-09-13T17:41:43.220Z",
    "modifiedOn": "2016-09-13T17:42:25.609Z"
  },
  {
    "GeneralInformation": {
      "Name": "substance-composition-1",
      "TypeOfComposition": {
        "code": "61423"
      }
    },
    "Constituents": {
      "Constituents": [
        {
          "uuid": "e67542c8-094b-422e-8bbb-d65d6bd66d22",
```

```
          "ReferenceSubstance": "b0c168d2-c97c-4cc7-83d0-9b2716759030/0",
          "ProportionTypical": {
            "lowerQualifier": ">",
            "lowerValue": 1,
            "unit": {
              "code": "2098"
            }
          }
        }
      ]
    }
  }
]
```


Note, that to make the response more compact, not every field of the SubstanceComposition document is included, only the ones containing data.

4.1.1.1. JsonDocumentHeader

The **JsonDocumentHeader** contains **metadata about the document**, such as the document key, the definition identifier, etc.

In detail it includes the following fields:

Name	Type	Description
key	String	The document key, formatted as `{uuid}/{snapshot}`
definition	String	The document definition identifier. e.g. *FLEXIBLE_RECORD.SubstanceComposition*
parentKey	String	**Only if the document is a section.** The document key of the parent, formatted as `{uuid}/{snapshot}`
parentDefinition	String	**Only if the document is a section.** The document definition identifier of the parent. e.g. SUBSTANCE
order	Number	Only if the document is a section. The order of the document within the group of documents of the same definition.
name	String	The document name, up to 255 characters.
attachments	Array	An array of attachment keys formatted as `{uuid}/{snapshot}`

Name	Type	Description
createdOn	Date	The instant the document was created, in ISO 8601 format. e.g. *2016-09-06T13:14:59.187Z*
modifiedOn	Date	The instant the document was last modified, in ISO 8601 format. e.g. *2016-09-06T13:14:59.187Z*
templates	Array	An array of template keys formatted as `{uuid}/{snapshot}`
creationTool	String	The name of the application that created the entity, e.g. 'IUC6'
snapshotCreationTool	String	The name of the application that created the dossier, e.g. 'IUC6'

4.1.1.2. JsonDocumentContent

The `JsonDocumentContent` contains the actual data of the document, the structure of this object varies according to the structure definition of each document, which can be found in I.TE.M. (IUCLID Template Manager)

In general, it follows a tree-like structure where each node can either be a container or a leaf node.

Container nodes can contain leaf nodes and other container nodes, while leaf nodes contain field values.

In the lists below you can find how each datatype of I.TE.M. is represented in the json format of the `JsonDocumentContent`.

4.1.1.3. Container types

I.TE.M Type	Type	Description

Header	Object	It defines a grouping of fields and other container elements
		A JSON object with properties for every contained element.
		e.g. The "GeneralInformation" header of the "SubstanceComposition" document
		`"GeneralInformation": {`
		`"Name": "substance-composition-1",`
		`"TypeOfComposition": {`
		`"code": "61423"`
		`}`
		`}`

Repeatable Set	Array	It defines a grouping of fields and other container elements, that can be repeated. Both varieties are converted to the same json representation A JSON array containing JSON objects for every entry of the repeatable set/list. Each object, in addition to its other contents, will include the uuid for that particular entry. e.g. The "Constituents" repeatable set of the "SsubstanceComposition" document `"Constituents": [ { "uuid": "e67542c8-094b-422e-8bbb-d65d6bd66d22", "ReferenceSubstance": "b0c168d2-c97c-4cc7-83d0-9b2716759030/0", "ProportionTypical": { "lowerQualifier": ">", "lowerValue": 1, "unit": { "code": "2098" } } } ]`
Repeatable List		

4.1.1.4. Field types

I.TE.M Type	Type	Description
Date	string	A date in ISO 8601 format. e.g. 2016-09-06
TextField255	string	Text up to 255 characters.
MultiLineText2000	string	Text up to 2000 characters.
TextArea32768	string	Text up to 32786 characters.
TextTemplate	string	Text up to 32786 characters.
RichTextArea	string	Text up to 32786 characters. Can contain html
CheckBox	boolean	A true/false value

Integer	number	An integer value
Decimal	number	A decimal value
PickList	object	An object containing the 'code' string field, and optionally an 'other' string field if the phrase is 'open' The allowed values of the 'code' field vary with the 'phrase-group' defined for this element in I.TE.M e.g. `{ "code": "1342", "other": "other" }`
PickListWithRemarks	object	An object containing the 'code' string field, and optionally an 'other' string field if the phrase is open and a 'remarks' string field. e.g. `{ "code": "1342", "other": "other", "remarks": "remarks" }`
PickListWithRemarks2000		
PickListWithRemarks32000		
UnitMeasure	object	An object containing a 'value' number and a 'unit' object that is a picklist e.g. `{ "value": 1.3, "unit":{ "other": "2493" } }`

<table>
<tr><td>RangeDecimalWithPickList</td><td>object</td><td>An object containing the following:
<table>
<tr><td>field</td><td>type</td><td>description</td></tr>
<tr><td>lowerQualifier</td><td>string</td><td>one of:
• ca.
• >
• >=</td></tr>
<tr><td>lowerValue</td><td>number</td><td>the lower numeric boundary</td></tr>
<tr><td>upperQualifier</td><td>string</td><td>one of:
• ca.
• <
• <=</td></tr>
<tr><td>upperValue</td><td>number</td><td>the upper numeric boundary</td></tr>
<tr><td>unit</td><td>object</td><td>a picklist object</td></tr>
</table></td></tr>
<tr><td>RangeDecimal</td><td>object</td><td>same as 'RangeDecimalWithPickList' without the 'unit' field.</td></tr>
<tr><td>HalfBounded</td><td>object</td><td>An object containing the following:
<table>
<tr><td>field</td><td>type</td><td>description</td></tr>
<tr><td>lowerQualifier</td><td>string</td><td>one of:
• ca.
• <
• <=
• >
• >=</td></tr>
<tr><td>lowerValue</td><td>number</td><td>the lower numeric boundary</td></tr>
<tr><td>unit</td><td>object</td><td>a picklist object</td></tr>
</table></td></tr>
</table>

<table>
<tr><td>Confidentiality</td><td>object</td><td>An object containing the following:
<table>
<tr><th>field</th><th>type</th><th>description</th></tr>
<tr><td>confidentiality</td><td>object</td><td>a picklist object with codes from phase-group N64:
2732 (confidential business information)
2859 (intellectual property)
3441 (not publicly available)</td></tr>
<tr><td>legislations</td><td>array</td><td>an array of picklist objects. The phrasegroup of each picklist is N78</td></tr>
<tr><td>justification</td><td>string</td><td>text up to 32786 characters. Can be populated with text template TT_501</td></tr>
</table></td></tr>
<tr><td>MultiSelect</td><td rowspan="4">array</td><td rowspan="4">An array of picklist objects with/without remarks.
e.g.
<pre>[
  {
    "code": "60224",
    "remarks": "remarks-1"
  },
  {
    "code": "60225",
    "remarks": "remarks-2"
  },
  {
    "code": "60230"
  }
]</pre></td></tr>
<tr><td>MultiSelectWithRemarks</td></tr>
<tr><td>MultiSelectWithRemarks2000</td></tr>
<tr><td>MultiSelectWithRemarks32000</td></tr>
<tr><td>EntityReferenceField</td><td rowspan="2">string</td><td rowspan="2">a document key, formatted as <code>{uuid}/{snapshot}</code>
e.g.
<pre>"4f88bc7f-395c-4d0b-997b-14e8c9aef605/0"</pre></td></tr>
<tr><td>EndpointReferenceField</td></tr>
<tr><td>SingleFileAttachment</td><td rowspan="2">string</td><td rowspan="2">an attachment key, formatted as <code>{uuid}/{snapshot}</code>
e.g.
<pre>"4f88bc7f-395c-4d0b-997b-14e8c9aef605/0"</pre></td></tr>
<tr><td>Image</td></tr>
</table>

EntityReferenceList	array	an array of strings, that represent keys, either of documents or attachments. The keys are formatted as `{uuid}/{snapshot}` e.g. `[ "77748886-6911-4a41-af93-7d5a86fc075f/0", "f9d571cd-d0ed-403d-b944-21e8d26c7c26/0" ]`
EndpointReferenceList		
AttachmentsList		
LiteratureReferenceList		

4.1.2. Response types

4.1.2.1. Links

The **Links** object is returned from resources having create semantics. Typically, these resources will respond with **HTTP CODE 201 (**created) and will include the Links object in the response payload.

It contains the following fields:

Name	Type	Description
source	string	The URI of the created entity/document.
links	object	For future use. Currently empty.

4.1.2.2. Error

The **Error** object is returned when something goes wrong, and it contains more detailed information about the source of the problem.

It contains the following fields:

Name	Type	Description
requestId	string	**Optional**. The request identifier. As sent by the client or generated by the server if not sent.
uri	string	The I6URI of the request resource.
subjectKey	string	**Optional**. The key of the entity that is related with the error.

Name	Type	Description
code	string	The specific code of the error. *Not to be confused with the HTTP code.*
message	string	**Optional**. The error message.
info	object	**Optional**. An object that varies with the specific error and contains detailed information about the error.

4.2. Semantics of HTTP status codes

4.2.1. Success codes

HTTP Status Code	Usage
200 OK	Success of data retrieval operations such as *read, list, search, export, print, ...*
200 OK	Success of update operations, such as *update, edit, import*
204 No Content	Success of update or deletion operations, when no entity is included in the response, e.g. *delete, remove*
201 Created	Success of creation operations
308 Permanent Redirect	Used for generic URI that don't change, such as /dossier/{uuid}/subject

4.2.2. Error codes

HTTP Status Code	Usage
400 Bad Request	The request entity or request parameters is not parsable by the server, either because it is malformed at the protocol level, or because it is malformed at the format level (XML, JSON, number or date representation).
403 Forbidden	The caller is attempting to perform an operation on a resource he is allowed to see (otherwise 404 would be appropriate) but hasn't sufficient privileges.

HTTP Status Code	Usage
404 Not Found	If the resource identified by the URI path does not exist (at any depth), or a resource identified in the query string that is essential to the representation does not exist.
405 Method Not Allowed	If the method is not supported by the target resource, regardless of the privileges of the caller. E.g. update a dossier document, create a dossier for an unsupported entity type
406 Not Acceptable	If the requested media type is not supported for the resource or not known.
415 Unsupported Media Type	If the resource cannot handle the media type sent by the caller.
422 Unprocessable Entity	To indicate business validation errors on the received data. Unless those are already covered by 400, 403 or 404.
500 Internal Server Error	No particular business meaning can be associated to this code, as it is the result of an internal unexpected exception.

4.3. Load

Loading of IUCLID6 documents is supported by a HTTP GET call on the appropriate REST resources, and requesting the JsonDocumentEnvelope via the `Accept` header:

Accept	application/vnd.iuclid6.ext+json;type=iuclid6.Document

4.3.1. Raw data

Raw documents are the documents that their content can be modified. These documents can be either **entities**, or **sections**.

4.3.1.1. Entities

Entities in IUCLID6 are the top-level documents (without a parent), that may or may not contain children.

They are categorized by their entity-type:

- SUBSTANCE
- MIXTURE
- TEMPLATE

- CATEGORY
- ANNOTATION
- LEGAL_ENTITY
- SITE
- REFERENCE_SUBSTANCE
- CONTACT
- LITERATURE
- TEST_MATERIAL_INFORMATION
- ARTICLE

More entity-types might be added in the future

To load an entity, two pieces of information are needed, its **entity-type** and its **uuid**:

GET	/raw/{entity_type}/{uuid}

So, for example, to load the legal entity with uuid 4f88bc7f-395c-4d0b-997b-14e8c9aef605 execute the following:

```
curl --request GET \
  --url http://localhost:8080/iuclid6-
ext/api/ext/v1/raw/LEGAL_ENTITY/4f88bc7f-395c-4d0b-997b-14e8c9aef605 \
  --header 'accept: application/vnd.iuclid6.ext+json; type=iuclid6.Document'
\
  --header 'iuclid6-user: SuperUser' \
  --header 'iuclid6-pass: ********'
```

4.3.1.2. Sections

Section documents are identified by a UUID and a document definition, same as entities, however they exist in the context of their parent entity, therefore they can only be accessed through the sub-tree of their parent.

4.3.1.2.1. List sections

The following, obtains a listing of every section contained in a given entity:

GET	/raw/{entity_type}/{uuid}/documents

4.3.1.2.2. Load the sections of a definition

To get only the sections of a specific definition:

GET	/raw/{entity_type}/{uuid}/document/{doc_def}

To call any of the previous two resources you need to specify the `Accept` header

Accept	application/vnd.iuclid6.ext+json

These resources return a wrapper object containing the URIs of the sections, see the searching documentation for more details.

Notice the change from plural `documents` to singular `document` in the previous resources.

By default, the above two resources do not list inherited documents, however this can be altered by setting the query param `inherited` to true

4.3.1.2.3. Load a specific section of a composite entity

To retrieve a specific section:

GET	/raw/{entity_type}/{uuid}/document/{doc_def}/{doc_uuid}

For example, to load the `SubstanceComposition` of a `Substance`:

```
curl --request GET \
  --url http://localhost:8080/iuclid6-ext/api/ext/v1/raw/SUBSTANCE/9915ba13-ffa7-405b-b024-c06296b43820/document/FLEXIBLE_RECORD.SubstanceComposition/1aca8956-fbfe-4f2e-80f7-54182bc72454 \
  --header 'accept: application/vnd.iuclid6.ext+json; type=iuclid6.Document' \
  --header 'iuclid6-user: SuperUser' \
  --header 'iuclid6-pass: ********'
```

4.3.1.3. Attachments

You can download the content of an attachment on the following resource:

GET	/raw/attachment/{attachment_uuid}

The attachment content will be contained in the response body and information about the attachment such as size, mime type and md5 hash will be in the relevant standard HTTP headers:

Header	Description
Content-Type	the mime type of the attachment content

Content-Length	The size in bytes of the attachment content
Content-MD5	The md5 hash of the attachment content
Content-Disposition	The attachment filename will be contained in the 'filename' parameter

4.3.2. Dossier

The IUCLID6 **dossier**, is a structured set of **non-modifiable** documents related with each other.

Every dossier includes:

- a **header document**
- a **dossier subject** entity along with its section documents
- a **submitting legal entity**
- a list of other entities referenced by the subject entity or its sections.

4.3.2.1. Header document

You access the contents of the header document by:

GET	/dossier/{dossier_uuid}/header

4.3.2.2. Subject

The dossier subject is available on:

GET	/dossier/{dossier_uuid}/subject

4.3.2.3. Sections of subject

The section documents of the dossier's subject entity can be accessed in a similar fashion as in the raw sections:

4.3.2.3.1. List sections

GET	/dossier/{dossier_uuid}/subject/documents

4.3.2.3.2. Load a specific section of a composite entity

To access a specific document included in a dossier's entity:

GET	/dossier/{uuid}/{entity_type}/{entity_uuid}/document/{doc_def}/{doc_uuid}

You can replace {entity_uuid} with “subject” to access a specific document included in the dossier’s subject entity.

4.3.2.4. Submitter

The submitting legal entity of the dossier is accessible through:

GET	/dossier/{dossier_uuid}/submitter

4.3.2.5. Dossier components

Dossier components are considered all the entities in a dossier, **including** the **subject** and **submitter**, **except** the dossier header.

Listing every component of a dossier, without specifying its type, is not supported in this version of the API

You can access any of them via the following resource:

GET	/dossier/{uuid}/{entity_type}/{entity_uuid}

As with the section documents of the subject entity, the sections of a dossier component can be accessed in a similar fashion as in the raw sections.

4.3.2.6. Dossier attachments

Similarly, with raw attachments, you can access the content of a dossier attachment on the following resource:

GET	/dossier/{uuid}/attachment/{attachment_uuid}

The attachment content will be contained in the response body and information about the attachment such as size, mime type and md5 hash will be in the relevant standard HTTP headers:

Header	**Description**
Content-Type	the mime type of the attachment content
Content-Length	The size in bytes of the attachment content
Content-MD5	The md5 hash of the attachment content
Content-Disposition	The attachment filename will be contained in the 'filename' parameter

4.4. Write raw data

Writing operations, namely:

- creation
- modification
- deletion

of documents, are supported by using the relevant HTTP verbs POST, PUT and DELETE respectively. Wherever content needs to be sent, its type is expected to be:

Content-Type	application/vnd.iuclid6.ext+json;type=iuclid6.Document

4.4.1. Entities

4.4.1.1. Create

The creation of entities is supported by posting the JsonDocumentEnvelope to create, to the relative entity's collection resource:

POST	/raw/{entity_type}

For instance, the following HTTP request creates a new literature reference document:

```
curl -X POST -H "IUCLID6-USER: SuperUser" -H "IUCLID6-PASS: *******" -H
"Content-Type: application/vnd.iuclid6.ext+json; type=iuclid6.Document"-d '[
  {
    "definition": "LITERATURE",
    "name": "A demo literature reference"
  },
  {
    "GeneralInfo": {
      "LiteratureType": {
        "code": "1586"
      },
      "Name": "A demo literature reference",
      "Author": "Unknown",
      "ReferenceYear": 2016,
      "Source": "unknown source",
      "TestLab": "unknown lab",
      "ReportNo": "xx-xxxx-x",
      "CompanyOwner": "unknown owner",
      "CompanyOwnerStudyNo": "xx-xxxx-x",
```

```
      "ReportDate": "2016-09-20",
      "Remarks": "nothing remarkable"
    }
  }
]' "http://localhost:8080/iuclid6-ext/api/ext/v1/raw/LITERATURE"
```

It is possible to create a new document with a specific key, by including the **key** property in the posted JsonDocumentHeader, be prepared however to handle the **409** (conflict) HTTP error code, in case the specified key already exists.

If the entity is successfully created, 201 (Created) is returned along with the Links object.

Error responses

HTTP Status code	Payload	Reason

409	e.g.
	`{`
	`"requestId": "f1017ae7-ab4e-426e-a597-c699d6819a34",`
	`"uri": "iuclid6:/0/SUBSTANCE",`
	`"subjectKey": "a97d1b21-1f58-44ae-a2dc-58781ebb4abb/0",`
	`"code": "DOC409",`
	`"message": "Duplicate document key: a97d1b21-1f58-44ae-a2dc-58781ebb4abb/0"`
	`}`
		A document with the same key already exists.

422	e.g.
	`{`
	`"requestId": "015737f7-8b5c-410f-9218-fb6be90dddac",`
	`"uri": "iuclid6:/0/SUBSTANCE",`
	`"code": "SYS400",`
	`"message": "Field 'SUBSTANCE.TypeOfSubstance.Composition': Phrase '2915123' is not a valid value for phrasegroup N08."`
	`}`
		The document content is invalid.
		This can be either due to incorrect structure,
		or value content.

4.4.1.2. Update

You can modify the contents of an entity, by using PUT on the entity's resource:

PUT	/raw/{entity_type}/{entity_uuid}

To change, for example, the author on the previously created literature:

```
curl -X PUT -H "IUCLID6-USER: SuperUser" -H "IUCLID6-PASS: ********" -H
"Content-Type: application/vnd.iuclid6.ext+json; type=iuclid6.Document" -d '[
  {
    "definition": "LITERATURE"
  },
  {
    "GeneralInfo": {
      "LiteratureType": {
        "code": "1586"
      },
      "Name": "A demo literature reference",
      "Author": "John Doe",
      "ReferenceYear": 2016,
      "Source": "unknown source",
      "TestLab": "unknown lab",
      "ReportNo": "xx-xxxx-x",
      "CompanyOwner": "unknown owner",
      "CompanyOwnerStudyNo": "xx-xxxx-x",
      "ReportDate": "2016-09-20",
      "Remarks": "nothing remarkable"
    }
  }
]' "http://localhost:8080/iuclid6-ext/api/ext/v1/raw/LITERATURE/fe9dbc97-
1c00-4f4f-875f-906c7b864443"
```

If the request succeeds, HTTP CODE 204 will be returned.

Error responses

Same as in create scenario.

4.4.1.3. Delete

Entity deletions can be achieved using the DELETE verb on the entity resource:

DELETE	/raw/{entity_type}/{entity_uuid}

If the request succeeds, HTTP CODE 204 will be returned.

Error responses

HTTP Status code	Payload	Reason

409	e.g.
	`{`
	`"requestId": "c53935bb-0002-4669-9030-2c5340419074",`
	`"uri": "iuclid6:/0/LEGAL_ENTITY/4f88bc7f-395c-4d0b-997b-14e8c9aef605",`
	`"subjectKey": "4f88bc7f-395c-4d0b-997b-14e8c9aef605/0",`
	`"code": "DOC502",`
	`"message": "Document is referred by other: 4f88bc7f-395c-4d0b-997b-14e8c9aef605/0 ",`
	`"info": {`
	`"referralKeys": []`
	`}`
	`}`
		There are documents that refer to the entity you are trying to delete.

4.4.2. Composite entities

4.4.2.1. Sections reordering (since v3.0.0)

To **reorder** section documents of an entity, one can perform the following call:

PUT	/raw/{ENTITY_TYPE}/{uuid}

where ENTITY_TYPE is the type of the entity (e.g., SUBSTANCE) and uuid is the identifier of the entity

The headers need to be set as follows:

Content-Type	application/vnd.iuclid6.ext+json;type=standard.reorder
Accept	application/vnd.iuclid6.ext+json

The caller needs to specify the following instructions:

- the entity which contains the section documents to be reordered
- a list of "DocumentMoves", with each one describing the movement of a single section document from one position to another. Reordering of multiple sections at the same time is supported. This means that one may pass a list of `DocumentMove`s that refer to documents belonging to different sections.

•

A DocumentMove is an object composed of the following properties:

Name	Type	Description
uri	string	The I6URI of the section document to be reordered
from	integer	The current position of the section document (1-based). Taken only into account in case of validation
to	integer	The new desired position of the section document (1-based)

To specify the section documents to be reordered as well as the respective moves from one position to the other a list of DocumentMoves need to be provided in the HTTP request body

Body	`[{ "uri": "iuclid6:/0/{ENTITY_TYPE}/{uuid}/{SUB_TYPE}/{{doc-uuid}}", "from": {from_position}, "to": {from_position} }, ... ]`

The following optional query parameter can also be used to control the reordering operation:

Parameter	Type	Description	Default value
validate	boolean	Whether to skip validation on the current position of sections as provided through the 'from' property	true

Validations performed

- The URI given in each DocumentMove is valid, in that the respective document exists
- The 'from' position given in each DocumentMove is the same as the actual position of the record to be moved

For example, to reorder two section documents in one invocation, the following call can be performed:

```
curl --request PUT \
  --url http://localhost:8080/iuclid6-ext/api/ext/v1/raw/SUBSTANCE/sub-uuid \
```

```
  --header 'content-type:
application/vnd.iuclid6.ext+json;type=standard.reorder' \
  --header 'accept: application/vnd.iuclid6.ext+json' \
  --header 'iuclid6-user: username' \
  --header 'iuclid6-pass: ********' \
  --data '[{ \
"uri": "iuclid6:/0/SUBSTANCE/sub-uuid/ENDPOINT_STUDY_RECORD.Hydrolysis/doc-
uuid-1", \
"from": 1, \
"to": 2 \
}, { \
"uri": "iuclid6:/0/SUBSTANCE/sub-uuid/ENDPOINT_STUDY_RECORD.Hydrolysis/doc-
uuid-2", \
"from": 2, \
"to": 1 \
}]'
```

4.4.2.2. Copy sections to target entity (since v3.0.0)

To copy a template into a dataset, one can perform the following call:

POST	/system/script

To specify the template to be copied as well as the dataset to which it will be copied, the following information needs to be provided in the HTTP request body:

Body	`{ "sourceEntity":"iuclid6:/0/TEMPLATE/{templ-uuid}", "targetEntity":"iuclid6:/0/{ENTITY_TYPE}/{e-uuid}" }`

where "templ-uuid" is the identifier of the template, while "ENTITY_TYPE" and "e-uuid" are the entity type and its identifier of the dataset respectively.

The `Content-Type` header needs to be set as follows:

Content-Type	application/vnd.iuclid6.ext+json;type=iuclid6.copy

The `Accept` header needs to be set as follows:

Accept	application/vnd.iuclid6.ext+json;type=iuclid6.CopyResult

The result of this call is an object which contains a list of document I6URIs, which correspond to the newly created documents that have been copied from the template.

```
{
"documents": [
"iuclid6:/0/{ENTITY_TYPE}/{e-uuid}/{doc-def-id}/{doc-uuid}",
...
```

```
]
}
```

For example, to copy a template with uuid "templ-uuid" into a substance dataset with uuid "sub-uuid" we may perform the following call:

```
curl --request POST \
  --url http://localhost:8080/iuclid6-ext/api/ext/v1/system/script \
  --header 'content-type: application/vnd.iuclid6.ext+json;type=iuclid6.copy'
\
  --header 'accept: application/vnd.iuclid6.ext+json;type=iuclid6.CopyResult'
\
  --header 'iuclid6-user: username' \
  --header 'iuclid6-pass: ********' \
  --data '{"sourceEntity":"iuclid6:/0/TEMPLATE/templ-
uuid","targetEntity":"iuclid6:/0/SUBSTANCE/sub-uuid"}'
```

4.4.3. Sections

The modification operations on section documents are very similar to those on entities. The only difference is the resources to which the request is made.

4.4.3.1. Create

POST	/raw/{entity_type}/{entity_uuid}/document/{doc_def}

4.4.3.2. Update

PUT	/raw/{entity_type}/{entity_uuid}/document/{doc_def}/{doc_uuid}

4.4.3.3. Delete

DELETE	/raw/{entity_type}/{entity_uuid}/document/{doc_def}/{doc_uuid}

4.4.4. Attachments

You can upload binary content that will be attached to IUCLID documents by posting the content to the following resource:

POST	/raw/attachment

You can specify a filename for the uploaded content by the Content-Disposition request header

Content-Disposition:	attachment;filename*=utf-8''{filename}

Currently omitting the Content-Disposition header leads to 500 error, this will be fixed in a subsequent release.

If the attachment upload is successful, the server will respond with **201 Created** and the response body will contain information about the created attachment:

```
{
  "snapshotUuid": "0",
  "uuid": "a4fdfe60-9852-4a38-929a-50415b8bebd1",
  "filename": "thor.jpg",
  "md5": "8a3aca8c73ecb6f9338d88d73b57a841",
  "size": 62208,
  "mediaType": "image/jpeg"
}
```

Name	Type	Description
snapshotUuid	string	The snapshot uuid of the created attachment. Will always be '0'
uuid	string	The uuid of the created attachment
filename	string	The filename of the attachment
md5	string	The md5 hash of the attachment content
size	string	The size in bytes of the attachment content
mediaType	object	The mime type of the attachment content

4.5. Create dossier

To create a dossier of a particular subject you can POST a JsonDocumentEnvelope, see section 4.1.1, of the header document of the dossier to be created on the following resource:

POST	/raw/{entity_type}/{entity_uuid}/dossiers/{submission_type}

And using the Accept header

Content-Type:	application/vnd.iuclid6.ext+json; type=iuclid6.Document

For example, to create a **REACH Inquiry** dossier of a specific substance:

```
curl -X POST -H "IUCLID6-USER: SuperUser" -H "IUCLID6-PASS: ********" -H
"Content-Type: application/vnd.iuclid6.ext+json; type=iuclid6.Document" -d '[
```

```
  {
    "definition": "DOSSIER.R_INQUIRY",
    "name": "demo dossier"
  },
  {
    "DossierSpecificInformation": {
      "Remarks": "demo remarks"
    }
  }
]' "http://localhost:8080/iuclid6-ext/api/ext/v1/raw/SUBSTANCE/5307a86a-afd4-4da9-9c01-4ac12b1bc9c0/dossiers/R_INQUIRY"
```

The above POST request will create a dossier with the default settings, that will include every document and field that is present in the subject entity dataset.

4.5.1. Advanced dossier creation

To alter the default behaviour, and specify exclusions, instead of sending only the header document, you can send a `DossierCreationDTO` object that, apart of the dossier header, can contain configuration parameters for the dossier creation process.

The resource is the same

POST	/raw/{entity_type}/{entity_uuid}/dossiers/{submission_type}

But the Accept header changes

Content-Type:	application/vnd.iuclid6.ext+json; type=iuclid6.DossierCreationDTO

Example DossierCreationDTO

```
{
  "excludeLegalEntity": true,
  "excludeNotRequiredDocuments": true,
  "excludeAnnotations": false,
  "excludeConfidentialFields": false,
  "excludeDetailFields": false,
  "filterCategoryMemberChildren": false,
  "excludeConfidentialityRestrictions":[],
  "excludeRegulationRestrictions":[],
  "excludeDocuments":[],
  "header": [
    {
      "definition": "DOSSIER.R_INQUIRY",
      "name": "demo dossier"
```

```
    },
    {
      "DossierSpecificInformation": {
        "Remarks": "demo remarks"
      }
    }
  ]
}
```

The **DossierCreationDTO** object contains the following properties:

Name	Type	Description	Default value
excludeLegalEntity	boolean	Exclude the legal entity of the subject.	true
excludeNotRequiredDocuments	boolean	Exclude not required sections	true
excludeAnnotations	boolean	Exclude annotations	false
excludeConfidentialFields	boolean	Exclude fields marked as confidential	false
excludeDetailFields	boolean	Exclude fields marked as detailed	false
filterCategoryMemberChildren	boolean	Exclude the section documents of category member substances	false
excludeConfidentialityRestrictions	array	Exclude fields with the specified confidentiality claims	empty array
excludeRegulationRestrictions	array	Exclude fields with the specified regulatory programme	empty array
excludeDocuments	array	Exclude specific documents by document key.	empty array
header	array	The header for the dossier as a JsonDocumentEnvelope	

For example, to create a **REACH Inquiry** dossier of a specific substance:

```
curl -X POST -H "IUCLID6-USER: SuperUser" -H "IUCLID6-PASS: ********" -H
"Content-Type: application/vnd.iuclid6.ext+json;
type=iuclid6.DossierCreationDTO" -d '
{
  "excludeLegalEntity": true,
  "excludeNotRequiredDocuments": true,
```

```
  "excludeAnnotations": false,
  "excludeConfidentialFields": false,
  "excludeDetailFields": false,
  "filterCategoryMemberChildren": false,
  "excludeConfidentialityRestrictions":[],
  "excludeRegulationRestrictions":[],
  "excludeDocuments":[],
  "header": [
  {
    "definition": "DOSSIER.R_INQUIRY",
    "name": "demo dossier"
  },
  {
    "DossierSpecificInformation": {
      "Remarks": "demo remarks"
    }
  }
 ]
}' "http://localhost:8080/iuclid6-ext/api/ext/v1/raw/SUBSTANCE/5307a86a-afd4-
4da9-9c01-4ac12b1bc9c0/dossiers/R_INQUIRY"
```

Note that each submission type has a corresponding definition for the header document of the dossier, trying to create a dossier with submission type and header definition that not match, will fail.

By convention, the header document definition identifiers are in the form: "DOSSIER.{submission_type}"

Error responses

HTTP Status code	Payload	Reason

405	e.g.
	`{`
	`"requestId": "8394333f-9748-49fd-b0f0-af800a1ce953",`
	`"uri": "iuclid6:/0/LEGAL_ENTITY/4f88bc7f-395c-4d0b-997b-14e8c9aef605",`
	`"subjectKey": "LEGAL_ENTITY:4f88bc7f-395c-4d0b-997b-14e8c9aef605/0",`
	`"code": "DOS405",`
	`"message": "Invalid subject for dossier: LEGAL_ENTITY 4f88bc7f-395c-4d0b-997b-14e8c9aef605/0"`
	`}`
		You are trying to create a dossier for an entity,
		that cannot be a subject of a dossier.

422	e.g.
	`{`
	`"requestId": "530d1d96-0a0a-416d-9903-10e6204bc495",`
	`"uri": "iuclid6:/0/SUBSTANCE/2f4c245c-bcd9-43c2-add1-cfe199e7cc20",`
	`"subjectKey": "iuclid6.Document: DOSSIER.COMPLETE.foreign-field",`
	`"code": "EXT422",`
	`"message": "Validation failure for: DOSSIER.COMPLETE.foreign-field ",`
	`"info": {`
	`"componentType": "Module",`
	`"extensionName": "iuclid6",`
	`"componentName": "Document",`
	`"errorDetails": {}`
	`}`
		The document content is invalid.
		This can be either due to incorrect structure,
		or value content.

4.6. Delete Dossier

Dossier deletions can be achieved using the **DELETE** verb on the dossier resource:

DELETE	/dossier/{dossier_uuid}

If the request succeeds, **HTTP CODE 204** will be returned.

4.7. Links (since v3.0.0)

4.7.1. Annotations

4.7.1.1. Link annotation to document

To link an annotation to a document we may perform the following call:

POST	/raw/ANNOTATION/{ann-uuid}/links/annotates

where ann-uuid corresponds to the identifier of the annotation entity.

The target document can be specified by the corresponding I6URI and passed through the body of the HTTP request:

Body	{I6URI}

The **Content-Type** header needs to be set as follows:

Content-Type	application/json

For example, to annotate a dossier with uuid "dossier-uuid" with the annotation with uuid "ann-uuid" we may perform the following call:

```
curl --request POST \
  --url http://localhost:8080/iuclid6-ext/api/ext/v1/raw/ANNOTATION/ann-
uuid/links/annotates \
  --header 'content-type: application/json' \
  --header 'iuclid6-user: username' \
  --header 'iuclid6-pass: ********' \
  --data '"iuclid6:/dossier-uuid"'
```

Valid resources that can be annotated are entities, dossiers or single documents within a composite entity. Some example I6URIs can be found below:

- Dossier: "iuclid6:/<dossier-uuid>"
- Substance: "iuclid6:/0/SUBSTANCE/<substance-uuid>"
- Substance Composition document: "iuclid6:/0/SUBSTANCE/<substance-uuid>/FLEXIBLE_RECORD.SubstanceComposition/<composition-doc-uuid>"

4.7.1.2. Unlink annotation from document

To unlink an annotation from a document we may perform the following call:

DELETE	/raw/ANNOTATION/{ann-uuid}/links/annotates/{doc-uri}

where "ann-uuid" corresponds to the identifier of the annotation entity, and "doc-uri" to the "reduced" I6URI (without the prefix "iuclid6:") of the annotated document .

The **Content-Type** header needs to be set as follows:

Content-Type	application/json

For example, to unlink the annotation with uuid "ann-uuid" from the raw substance with uuid "sub-uuid" we may perform the following call:

```
curl --request DELETE \
  --url http://localhost:8080/iuclid6-ext/api/ext/v1/raw/ANNOTATION/ann-
uuid/links/annotates/0/SUBSTANCE/sub-uuid \
  --header 'content-type: application/json' \
  --header 'iuclid6-user: username' \
  --header 'iuclid6-pass: ********'
```

Note: Unlinking an annotation from a document does only affect the relation between the two resources. Neither the annotation nor the document are deleted after this operation

4.7.1.3. List documents linked to an annotation

To list the documents that are annotated by an annotation we may perform the following call:

GET	/raw/ANNOTATION/{ann-uuid}/links/annotates

where "ann-uuid" corresponds to the identifier of the annotation entity.

A formatter can be optionally provided as a query parameter for the returned documents to have a specific representation format.

For example, to retrieve the documents annotated by the annotation with uuid "ann-uuid" we may perform the following call:

```
curl --request GET \
  --url http://localhost:8080/iuclid6-ext/api/ext/v1/raw/ANNOTATION/ann-
uuid/links/annotates \
  --header 'iuclid6-user: username' \
  --header 'iuclid6-pass: ********'
```

4.7.1.4. List annotations linked to a document

To list the annotations that are linked to a document we may perform any of the following calls, depending on the type of the document:

Annotations of any document within a dataset

GET	/raw/{ENTITY_TYPE}/{e-uuid}/links/annotatedBy

Annotations of a specific document within a dataset

GET	/raw/{ENTITY_TYPE}/{e-uuid}/document/{doc-def-id}/{doc-uuid}/links/annotatedBy

Annotations of any document within a dossier

GET	/dossier/{dossier-uuid}/links/annotatedBy

Annotations of the header of a dossier

GET	/dossier/header/{dossier-uuid}/links/annotatedBy

Annotations of any document within a dataset of a dossier

GET	/dossier/header/{dossier-uuid}/{ENTITY_TYPE}/{e-uuid}/links/annotatedBy

Annotations of a specific document within a dataset of a dossier

GET	/dossier/header/{dossier-uuid}/{ENTITY_TYPE}/{e-uuid}/document/{doc-def-id}/{doc-uuid}/links/annotatedBy

where "ENTITY-TYPE" corresponds to the entity type of a dataset (e.g., SUBSTANCE), "e-uuid" to the identifier of the dataset, "doc-uuid" to the identifier of a document with the dataset and "dossier-uuid" to the identifier of a dossier.

A formatter can be optionally provided as a query parameter for the returned documents to have a specific representation format.

For example, to retrieve the annotations related to the raw substance with uuid "sub-uuid" we may perform the following call:

```
curl --request GET \
  --url http://localhost:8080/iuclid6-ext/api/ext/v1/raw/SUBSTANCE/sub-
uuid/links/annotatedBy \
  --header 'iuclid6-user: username' \
  --header 'iuclid6-pass: ********'
```

4.7.2. Templates

4.7.2.1. Link template to dataset (inherit)

To link a template to a dataset we may perform the following call:

POST	/raw/{ENTITY_TYPE}/{e-uuid}/links/inherits

where "ENTITY_TYPE" is the entity type of the dataset (e.g., SUBSTANCE) and "e-uuid" corresponds to the identifier of the dataset.

The template to be inherited can be specified by the corresponding I6URI and passed through the body of the HTTP request:

Body	iuclid6:/0/TEMPLATE/{templ-uuid}

where "templ-uuid" is the identifier of the template.

The **Content-Type** header needs to be set as follows:

Content-Type	application/json

For example, for a SUBSTANCE with uuid "sub-uuid" to inherit a template with uuid "templ-uuid" we may perform the following call:

```
curl --request POST \
  --url http://localhost:8080/iuclid6-ext/api/ext/v1/raw/SUBSTANCE/sub-
uuid/links/inherits \
  --header 'content-type: application/json' \
  --header 'iuclid6-user: username' \
  --header 'iuclid6-pass: ********' \
  --data '"iuclid6:/0/TEMPLATE/templ-uuid"'
```

4.7.2.2. Unlink template from dataset

To unlink a template from a dataset we may perform the following call:

DELETE	/raw/{ENTITY_TYPE}/{e-uuid}/links/inherits/{templ-uri}

where "ENTITY_TYPE" is the entity type of the dataset (e.g., SUBSTANCE), "e-uuid" the identifier of the dataset, and "templ-uri" the I6URI of the inherited template.

The **Content-Type** header needs to be set as follows:

Content-Type	application/json

For example, to unlink the template with uuid "templ-uuid" from the raw substance with uuid "sub-uuid" we may perform the following call:

```
curl --request DELETE \
  --url http://localhost:8080/iuclid6-ext/api/ext/v1/raw/SUBSTANCE/sub-
uuid/links/inherits/iuclid6:/0/TEMPLATE/templ-uuid/TEMPLATE/templ-uuid \
  --header 'content-type: application/json' \
  --header 'iuclid6-user: username' \
  --header 'iuclid6-pass: ********'
```

Note: Unlinking a template from a dataset does only affect the relation between the two resources. Neither the template nor the dataset are deleted after this operation.

4.7.2.3. List datasets linked to a template (inheriting template)

To list the documents that are annotated by an annotation we may perform the following call:

GET	/raw/TEMPLATE/{templ-uuid}/links/inheritedBy

where "templ-uuid" corresponds to the identifier of the template entity.

A formatter can be optionally provided as a query parameter for the returned documents to have a specific representation format.

For example, to retrieve the datasets which inherit the template with uuid "templ-uuid" we may perform the following call:

```
curl --request GET \
  --url http://localhost:8080/iuclid6-ext/api/ext/v1/raw/TEMPLATE/templ-
uuid/links/inheritedBy \
  --header 'iuclid6-user: username' \
  --header 'iuclid6-pass: ********'
```

4.7.2.4. List templates linked to a dataset (inherited by dataset)

To list the templates that are inherited by a dataset we may perform the following call, depending on the type of the document:

Templates inherited by a raw a dataset

GET	/raw/{ENTITY_TYPE}/{e-uuid}/links/inherits

Templates inherited by a dataset in a dossier

GET	/dossier/header/{dossier-uuid}/{ENTITY_TYPE}/{e-uuid}/links/inherits

where "ENTITY-TYPE" corresponds to the entity type of a dataset (e.g., SUBSTANCE), "e-uuid" to the identifier of the dataset, and "dossier-uuid" to the identifier of a dossier.

A formatter can be optionally provided as a query parameter for the returned documents to have a specific representation format.

For example, to retrieve the templates inherited by the raw substance with uuid "sub-uuid" we may perform the following call:

```
curl --request GET \
  --url http://localhost:8080/iuclid6-ext/api/ext/v1/raw/SUBSTANCE/sub-
uuid/links/inherits \
  --header 'iuclid6-user: username' \
  --header 'iuclid6-pass: ********'
```

5. Entity access control

Every REST resource in this document is under /iuclid6-ext/api/ext/v1/

5.1. Introduction

IUCLID 6 can enforce access control at the entity level, if the installation is configured to do so.

A user "owns" the entities she creates, and by default, no other user can access these.

It is desirable, however, for other users to be able to access data created by others. To achieve this, in a controlled fashion, two concepts emerge: "groups" and "access rights":

- A **"group"** is nothing more than a name to identify a set of particular users
- while **"access right"** is the level of privileges someone has upon an entity.

Four such levels are defined:

- **"read only"** allowing read but not modification
- **"read/write"** allowing both read and modification, but not deletion
- **"full access"** allowing deletion on top of read and modification
- and finally, **"no access"** which is effectively the absence of any privilege.

Please refer to the IUCLID6 server installation manual for a more throughout description of these concepts.

A user can control who and with what privileges, will be able to access an entity she owns, by sharing that entity with a "group" specifying the "access right" for this group. It is possible, of course, for an entity to be shared across multiple groups.

The public REST API provide a means to both retrieve the sharing information of an entity and change it.

5.2. Datatypes

The standard Accept header for exchanging data access information is

Accept	application/vnd.iuclid6.ext+json;type=standard.access

The response is a json array, of simple objects containing to properties: "**group**" and "**access**":

```
[
  {
    "group": "group A",
    "access": "READ_ONLY"
  },
  {
    "group": "Common",
    "access": "NO_ACCESS"
  }
```

```
]
```

Name	Type	Description
group	String	The name of the group
access	String	The access right, can be one of: • NO_ACCESS • READ_ONLY • READ_WRITE • FULL_ACCESS

Note that both the **group** and **access** are case sensitive

5.3. Semantics of HTTP status codes

5.3.1. Success Codes

HTTP Status Code	Usage
200 OK	Success of data retrieval operations
204 No Content	Success of update or deletion operations, when no entity is included in the response, e.g. delete, remove

5.3.2. Error codes

HTTP Status Code	Usage
403 Forbidden	The caller is not allowed to share the entity
404 Not Found	If the resource identified by the URI path does not exist (at any depth).
422 Unprocessable Entity	To indicate business validation errors on the received data.

5.4. Retrieve data access

You can retrieve the groups an **entity** is currently shared with, in addition to the groups that can be shared with, on:

GET	/raw/{entity_type}/{entity_uuid}/access

or, in the case of **dossiers**:

GET	/dossier/{uuid}/access

For both of the above resources you should specify the **Accept** header:

Accept	application/vnd.iuclid6.ext+json;type=standard.access

For example, to see the groups that the dossier with uuid: 'uuid-1' is shared to:

```
curl --request GET\
  --url http://localhost:8080/iuclid6-ext/api/ext/v1/dossier/uuid-1/access \
  --header 'accept: application/vnd.iuclid6.ext+json;type=standard.access' \
  --header 'iuclid6-user: username' \
  --header 'iuclid6-pass: *********'
```

5.5. Modify data access

You can share, or "un-share", an entity by posting the changes to:

POST	/raw/{entity_type}/{entity_uuid}/access

or, in the case of dossiers:

POST	/dossier/{uuid}/access

For both of the above resources you should specify the `Content-Type` header:

Content-Type	application/vnd.iuclid6.ext+json;type=standard.access

For example, let’s say that the substance with uuid "uuid-1" is currently shared to both "group-A" and "group-B" with full access, and we wish to lower the access privilege for "group-B" to "READ_ONLY"

```
curl --request POST \
  --url http://localhost:8080/iuclid6-ext/api/ext/v1/raw/SUBSTANCE/uuid-
1/access \
  --header 'content-type:
application/vnd.iuclid6.ext+json;type=standard.access' \
  --header 'iuclid6-user: username' \
  --header 'iuclid6-pass: *********' \
```

```
  --data '[{"group":"group-B","access":"READ_ONLY"}]'
```

6. Engines

Every REST resource in this document is under /iuclid6-ext/api/ext/v1/

6.1. Supported Types

6.1.1. Entities

The IUCLID 6 Public API supports **both**

- **raw** entities export
- and **dossier** export capabilities.

All raw entity types (both composite types and simple types) are supported for export and can be one of the following:

- SUBSTANCE
- MIXTURE
- TEMPLATE
- REFERENCE_SUBSTANCE
- TEST_MATERIAL_INFORMATION
- CATEGORY
- LEGAL_ENTITY
- LITERATURE
- SITE
- CONTACT
- ANNOTATION
- ATTACHMENT
- ARTICLE

Export of section documents on their own (providing the uuid of a section document for export) is currently not supported from the public API.

6.1.2. Dossiers

The IUCLID6 dossier, is a structured set of non-modifiable (snapshot), related documents. Apart from the dossier header, the dossier includes a dossier subject and a list of documents with their sections.

It also may or may not contain a submitting legal entity, according to the dossier creation process (submitting legal entity is not included by default in the created dossier, unless the user selects to include it).

6.2. Export

6.2.1. Initiate basic export operation

Export of IUCLID6 entities / dossiers is supported by an HTTP POST call on the appropriate REST resources (Entity Export Resource / Dossier Export Resource) by providing as content-type the FullExport type and an empty body "{}" in the http request:

Content-Type	application/vnd.iuclid6.ext+json;type=iuclid6.FullExport

The above request, will create the proper BackgroundJob for the export process, which will be persisted on the IUCLID 6 database with (among other) the following information:

- **job id**
- **status**
- **name** (the UUID of the exported entity)

and the export result in the file system.

6.2.2. Customization of the export operation

The export operation can be controlled in terms of the contents of the exported archive. Specifically, it is possible to choose whether stylesheet files are included in the archive, as well as whether empty elements (elements with no value) are included in the XML files contained in the archive.

A raw dataset can be exported through the following call:

POST	/raw/{ENTITY_TYPE}/{e-uuid}/export

where "ENTITY-TYPE" is the entity type of the dataset (e.g., SUBSTANCE) and "e-uuid" the identifier of the dataset.

A dossier can be similarly exported as follows:

POST	/dossier/{dossier-uuid}/export

The `Content-Type` header needs to be set as follows:

Content-Type	application/vnd.iuclid6.ext+json;type=iuclid6.FullExport

The body of the request should be an object with the following fields:

Name	Type	Default Value	Description
excludeXslStylesheets	boolean	false	If true, stylesheets are not included in the exported archive
excludeEmptyElements	boolean	true	If true, empty document elements are not included in the exported archive
Body	`{ "excludeXslStylesheets": true/false, "excludeEmptyElements": true/false }`		

Note: an empty object ({ }) is acceptable, in which case the default values are applied

For example, to export a dossier with uuid "dossier-uuid" without including the stylesheets, we may perform the following call:

```
curl --request POST \
  --url http://localhost:8080/iuclid6-ext/api/ext/v1/dossier/dossier-
uuid/export \
  --header 'content-type:
application/vnd.iuclid6.ext+json;type=iuclid6.FullExport' \
  --header 'iuclid6-user: username' \
  --header 'iuclid6-pass: ********' \
  --data '{"excludeXslStylesheets": true, "excludeEmptyElements": true}'
```

6.2.3. Get export operation status

An HTTP GET request can then be applied on the corresponding Job REST resource by providing the job id which has been created as a result of the previous HTTP POST request.

GET	/system/job/{job_uuid}

The Accept header needs to be modified.

ACCEPT	application/vnd.iuclid6.ext+json;type=iuclid6.Iuclid6Job

An example of the response after applying the above request on uuid 7652670e-6f28-4867-a084-065bd25203ad corresponding to a SUBSTANCE entity is shown below:

```
curl --request GET
   --url 'http://localhost:8080/iuclid6-ext/api/ext/v1/system/job/7652670e-
6f28-4867-a084-065bd25203ad'
   --header 'accept:
application/vnd.iuclid6.ext+json;type=iuclid6.Iuclid6Job'
   --header 'IUCLID6-USER: SuperUser'
   --header 'IUCLID6-PASS: root'
```

JobInfo


```
{
   "id":"7652670e-6f28-4867-a084-065bd25203ad",
   "status":"SUCCEEDED",
   "uri":"/system/jobs/7652670e-6f28-4867-a084-065bd25203ad",
   "representation": {
                     "classtype":"Iuclid6Job",
                           "group":null,
                           "name":"IUC5-e44ee572-3f11-4ad9-bdeb-
52160e442055",
                           "type":"EXPORT",
                           "documentUri":null,
                           "logs":     [
                                            {
                                                        "code":null,
                                                        "level":"INFO",
                                                        "value":"Export
process succeeded",
                                                        "documentUri":null,
                                                        "params":[]
                                                        }
                                                  ]
                                 }
}
```


The **status** of the Background export Job can have either of the following values:

- `QUEUED`: Job is queued for execution => Non-final state
- `IN_PROGRESS`: Job is being processed by IUCLID => Non-final state
- `SUCCEEDED`: Job completed successfully => Final state
- `FAILED`: Job failed => Final state
- `CANCELED`: Job is cancelled => Final state

The status of the background export job can be provided by applying an HTTP GET request on the corresponding job resource by providing only the UUID of the background job and setting the following `Accept` header:

ACCEPT	text/plain

6.2.4. Retrieve export operation result

The binary output result of the export job, can be downloaded in binary format from the user by providing the uuid of the background job in the following HTTP GET request:

GET	/system/job/{job_uuid}/result

6.2.5. Export entity

To create the export background job **for an entity (raw data)**, two pieces of information are needed:

- its **entity-type**
- and **uuid**.

GET	/raw/{entity_type}/{uuid}/export

So, for example, to export the substance with uuid 7a96f12c-f7b3-445d-a19f-e096f9562646 the following request is required:

```
curl --request POST \
  --url http://localhost:8080/iuclid6-ext/api/ext/v1/raw/SUBSTANCE/7a96f12c-
f7b3-445d-a19f-e096f9562646/export \
  --header 'accept: application/vnd.iuclid6.ext+json;
type=iuclid6.Iuclid6Job' \
  --header 'content-type: application/vnd.iuclid6.ext+json;
type=iuclid6.FullExport' \
  --header 'iuclid6-user: SuperUser' \
  --header 'iuclid6-pass: ********'
  --data   '{}'
```

To get the **status** of the job created from the previous request with job_uuid=0b30ad7c-decc-40ac-9edb-4e845879cc99:

```
curl --request GET \
  --url http://localhost:8080/iuclid6-ext/api/ext/v1/system/job/0b30ad7c-
decc-40ac-9edb-4e845879cc99/status \
```

```
  --header 'accept: text/plain' \
  --header 'iuclid6-user: SuperUser' \
  --header 'iuclid6-pass: ********'
```

In order to download the binary output of the export procedure (i6z file) the following GET request could be applied:

```
curl --request GET \
  --url http://localhost:8080/iuclid6-ext/api/ext/v1/system/job/0b30ad7c-
decc-40ac-9edb-4e845879cc99/result \
  --header 'iuclid6-user: SuperUser' \
  --header 'iuclid6-pass: ********'
```

6.2.6. Export dossier

To create the export background job **for a dossier**, the dossier **uuid** is required as shown in the below request:

GET	/dossier/{dossier_uuid}/export

So, for example, to export the dossier with uuid 1dc1301c-64eb-4984-8446-e73182e08764 the following request is required:

```
curl --request POST \
  --url http://localhost:8080/iuclid6-ext/api/ext/v1/dossier/1dc1301c-64eb-
4984-8446-e73182e08764/export \
  --header 'content-type: application/vnd.iuclid6.ext+json;
type=iuclid6.FullExport' \
  --header 'accept: application/vnd.iuclid6.ext+json;
type=iuclid6.Iuclid6Job' \
  --header 'iuclid6-user: SuperUser' \
  --header 'iuclid6-pass: ********'
  --data   '{}'
```

The result of the above request would be the following:

JobInfo

```
{
   "id": "e83f86f6-e6f8-4823-8725-1750dfa82e5e",
   "status": "QUEUED",
   "uri": "/job/e83f86f6-e6f8-4823-8725-1750dfa82e5e",
   "representation": null
}
```

To get the **status** of the dossier export job created from the previous request with job_uuid=e83f86f6-e6f8-4823-8725-1750dfa82e5e:

```
curl --request GET \
  --url http://localhost:8080/iuclid6-ext/api/ext/v1/system/job/e83f86f6-
e6f8-4823-8725-1750dfa82e5e/status \
  --header 'accept: text/plain' \
  --header 'iuclid6-user: SuperUser' \
  --header 'iuclid6-pass: ********'
```

and the text/plain result of the above request could be:

```
SUCCEEDED
```

To download the binary output of the dossier export procedure (i6z file) the following GET request could be applied:

```
curl --request GET \
  --url http://localhost:8080/iuclid6-ext/api/ext/v1/system/job/e83f86f6-
e6f8-4823-8725-1750dfa82e5e/result \
  --header 'iuclid6-user: SuperUser' \
  --header 'iuclid6-pass: ********'
```

6.3. Import

You can import **both** iuclid**6** archives (***.i6z**) and iuclid**5** (***.i5z**) using the public REST API.

To do so, you issue a POST request to **/system/archives** resource, the binary content of the archive to be imported should be included in the body of the request. The **Content-Type** header determines the IUCLID archive version.

Content-Type	“application/vnd.iuclid6.archive” or “application/vnd.iuclid5.archive“ depending on the IUCLID archive version to be imported

The overwrite mode can be set with the **overwrite** query parameter

Param	Type	Description	Default
overwrite	string	It can take the following values: **if_newer** If the archive contains documents that are already present in the system, then by default they will only be updated if they are more recently modified than the existing version. **never** Do not update existing documents. **always** Update existing documents even if they are more recently modified **replace** In case the archive is of an entity with section documents (e.g. SUBSTANCE), then replace the existing dataset with the archive. That is, if a child section exists in the system, but is not contained in the archive it will be removed.	If_newer

An example of an import request using curl:

```
curl --request POST \
 --url 'http://localhost:8080/iuclid6-
ext/api/ext/v1/system/archives?overwrite=if_newer' \--data-binary
<path_to_file>
 --header 'content-type: application/vnd.iuclid5.archive' \
 --header 'iuclid6-user: SuperUser'--header 'iuclid6-pass: *********'
```

By using the returned job uuid, e.g., eb685af2-b9bc-45a0-a2b1-b470a0283bd8, we may fetch more information about the status of the job as we did with the export operation.

GET	/system/job/{job_uuid}

The Accept header needs to be modified.

ACCEPT	application/vnd.iuclid6.ext+json;type=iuclid6.Iuclid6Job

```
curl --request GET
   --url 'http://localhost:8080/iuclid6-ext/api/ext/v1/system/job/eb685af2-
b9bc-45a0-a2b1-b470a0283bd8'
   --header 'accept:
application/vnd.iuclid6.ext+json;type=iuclid6.Iuclid6Job'
   --header 'IUCLID6-USER: SuperUser'
   --header 'IUCLID6-PASS: *****'
```

JobInfo


```
{
  "id": "eb685af2-b9bc-45a0-a2b1-b470a0283bd8",
  "status": "SUCCEEDED",
  "uri": "/system/jobs/eb685af2-b9bc-45a0-a2b1-b470a0283bd8",
  "representation": {
    "classtype": "Iuclid6Job",
    "group": null,
    "name": "344446c5-4c9e-4bed-ab0c-9bf6a257a6cd_unnamed.i5z",
    "type": "IMPORT",
    "documentUri": "iuclid6:/IUC5-e44ee572-3f11-4ad9-bdeb-52160e442055",
    "logs": [
      {
        "code": null,
        "level": "INFO",
        "value": "Started import of file: 344446c5-4c9e-4bed-ab0c-
9bf6a257a6cd_unnamed.i5z",
        "documentUri": null,
        "params": [
          "",
          "",
          "",
          "",
          ""
        ]
      },
      {
        "code": null,
        "level": "INFO",
        "value": "Overwrite mode: IF_NEWER_THAN_EXISTING",
        "documentUri": null,
        "params": [
          "",
          "",
          "",
          "",
          ""
        ]
      },
      ...
      {
        "code": null,
        "level": "INFO",
```

```
        "value": "Import process succeeded",
        "documentUri": null,
        "params": []
      }
    ]
  }
}
```

To fetch information on the imported entity only, use the 'application/vnd.iuclid6.ext+json;type=standard.URI' accept header

```
curl --request GET
   --url 'http://localhost:8080/iuclid6-ext/api/ext/v1/system/job/eb685af2-b9bc-45a0-a2b1-b470a0283bd8'
   --header 'accept: application/vnd.iuclid6.ext+json;type=standard.URI'
   --header 'IUCLID6-USER: SuperUser'
   --header 'IUCLID6-PASS: root'
```

JobInfo

```
{
   "id":"eb685af2-b9bc-45a0-a2b1-b470a0283bd8",
   "status":"SUCCEEDED",
   "uri":"/system/jobs/eb685af2-b9bc-45a0-a2b1-b470a0283bd8",
   "representation":"iuclid6:/IUC5-e44ee572-3f11-4ad9-bdeb-52160e442055"
}
```

6.4. Validation assistant

The validation assistant checks if a IUCLID6 dossier contains errors, that would reject its submission, and it produces a report containing information about these errors, if any.

The system supports the execution of the validation assistant, both on created dossiers, and on raw datasets.

However, since the validation assistant operates on dossiers, to validate raw datasets, you need to specify the additional data that would create a dossier from this dataset, i.e. the submission type and the dossier header.

6.4.1. Raw entities

To execute the validation assistant on a raw dataset you can **POST** a **DossierCreationDTO** object to the following resource:

POST	/raw/{entity_type}/{entity_uuid}/dossiers/{submission_type}/validation

For example, to validate a **REACH Inquiry** dossier of a specific substance:

```
curl -X POST -H "IUCLID6-USER: SuperUser" -H "IUCLID6-PASS: ********" -H
"Content-Type: application/vnd.iuclid6.ext+json;
type=iuclid6.DossierCreationDTO" -H "Accept:
application/vnd.iuclid6.ext+json; type=iuclid6.ValidationReport" -d '
{
  "excludeLegalEntity": true,
  "excludeNotRequiredDocuments": true,
  "excludeAnnotations": false,
  "excludeConfidentialFields": false,
  "excludeDetailFields": false,
  "filterCategoryMemberChildren": false,
  "excludeConfidentialityRestrictions":[],
  "excludeRegulationRestrictions":[],
  "excludeDocuments":[],
  "header":[
  {
    "definition": "DOSSIER.R_INQUIRY",
    "name": "demo dossier"
  },
  {
    "DossierSpecificInformation": {
      "Remarks": "demo remarks"
    }
  }
 ]
}' "http://localhost:8080/iuclid6-ext/api/ext/v1/raw/SUBSTANCE/5307a86a-afd4-
4da9-9c01-4ac12b1bc9c0/dossiers/R_INQUIRY/validation"
```

The above request will respond with the validation report:

ValidationReport

```
{
  "subjectUuid": "9f7c364c-8e29-4d3c-8c4a-56baea903414",
  "snapshotUuid": "3bf15315-f76f-4d61-84d0-0ac0335ad338",
  "rulesetId": "SC0005",
  "rulesetName": "SC0005 - Inquiry",
  "ruleResults": [
    {
      "ruleId": "BR018",
      "ruleType": "BR",
      "ruleStatus": "FAIL",
      "checks": [
        {
```

```
          "ruleCheckId": "BR018",
          "level": "FAILURE",
          "element": {
            "uri": "iuclid6:/3bf15315-f76f-4d61-84d0-
0ac0335ad338/SUBSTANCE/9f7c364c-8e29-4d3c-8c4a-
56baea903414;section=R_INQUIRY:1.1/SUBSTANCE/9f7c364c-8e29-4d3c-8c4a-
56baea903414#ReferenceSubstance.ReferenceSubstance",
            "description": "",
            "documentName": "test1",
            "sectionName": "Identification"
          },
          "message": "A reference substance must be linked in IUCLID section
1.1."
        }
      ]
    },
    ...
  ]
}
```


The **ValidationReport** contains the following properties

Name	Type	Description
subjectUuid	string	The uuid of the dossier subject
snapshotUuid	string	The uuid of the dossier
rulesetId	string	The identification of the ruleset
rulesetName	string	The title of the ruleset
ruleResults	array	An array of RuleResult objects

The **RuleResult** contains the following properties

Name	Type	Description
ruleId	string	The identification of the rule

Name	Type	Description
ruleType	string	The type of the rule. Can be one of: • TCC • BR • QLT • SID
ruleStatus	string	The status of the rule. Can be on of: • PASS • FAIL • EXCEPTION
checks	array	An array of RuleCheckResult objects

The **RuleCheckResult** contains the following properties

Name	Type	Description
ruleCheckId	string	The uuid of the dossier subject
level	string	The severity of the check. Can be one of: • REMINDER • WARNING • FAILURE
message	string	A localized description of the check.
element	object	The **RuleCheckElement** the check applies to.
relatedElements	array	An array of related **RuleCheckElement** objects, that the check might apply to
conditions	array	An array of **ReportCondition** objects. A check might be relevant only if some additional conditions are met.

The **RuleCheckElement** contains the following properties

Name	Type	Description
uri	string	The I6URI of the element

Name	Type	Description
description	string	An optional description for the element
documentName	string	The name of the document the element is contained in.
sectionName	string	The name of the section in the tree of the document containing the element.

The **ReportCondition** contains the following properties

Name	Type	Description
conditionKey	string	The key of the condition
conditionValue	string	The value of the condition
localizedCondition	string	A localized title of the condition

6.4.2. Dossiers

To execute the validation assistant on created dossier, you can make a POST request without any payload, to the following resource:

POST	dossier/{dossier_uuid}/validation

For example, the following request executes the validation assistant for a dossier with UUID: 3bf15315-f76f-4d61-84d0-0ac0335ad338

```
curl -X POST -H "IUCLID6-USER: SuperUser" -H "IUCLID6-PASS: ********" -H
"Accept: application/vnd.iuclid6.ext+json; type=iuclid6.ValidationReport"
"http://localhost:8080/iuclid6-ext/api/ext/v1/dossier/3bf15315-f76f-4d61-
84d0-0ac0335ad338/validation"
```

6.5. Dossier comparison (since v3.0.0)

You can compare two dossiers and receive a comparison report in HTML format by performing the following call:

POST	/dossier/{uuid1}/compare

where uuid1 corresponds to the identifier of the first dossier.

The uuid of the second dossier is passed through the body of the HTTP request:

Body	{uuid2}

The `Content-Type` header needs to be set as follows:

Content-Type	application/json

The Accept header needs to be set as follows:

Accept	text/html

For example, to compare a dossier with uuid "uuid-1" to a dossier with uuid "uuid-2" we may perform the following call:

```
curl --request POST \
  --url http://localhost:8080/iuclid6-ext/api/ext/v1/dossier/uuid-1/compare \
  --header 'content-type: application/json' \
  --header 'accept: text/html' \
  --header 'iuclid6-user: username' \
  --header 'iuclid6-pass: ********' \
  --data 'uuid-2'
```

6.6. Report generation (since v3.0.0)

6.6.1. Document Printing

To **print** a dataset or dossier to PDF format, one can perform the following call:

POST	/system/script

To specify the resource to be printed, a valid I6URI needs to be provided in the HTTP request body

Body	"{I6URI}"

The `Content-Type` header needs to be set as follows:

Content-Type	application/vnd.iuclid6.ext+json;type=iuclid6.print

The `Accept` header needs to be set as follows:

Accept	application/vnd.iuclid6.ext+pdf;type=iuclid6.pdf

The following optional query parameters can also be used to control the printing operation:

Query parameter name	Default value	Description
skipEmpty	true	Whether to ignore sections and fields that have no data
skipConfidential	true	Whether to skip fields that are declared as confidential (as per their definition, not any related data protection)
showCover	true	Whether to include a cover page
showAnnotations	false	Whether to include annotations of the documents that are printed

For example, to print a dossier with UUID "dossier-uuid" we may perform the following call:

```
curl --request POST \
  --url http://localhost:8080/iuclid6-ext/api/ext/v1/system/script \
  --header 'content-type:
application/vnd.iuclid6.ext+json;type=iuclid6.print' \
  --header 'accept: application/vnd.iuclid6.ext+pdf;type=iuclid6.pdf' \
  --header 'iuclid6-user: username' \
  --header 'iuclid6-pass: ********' \
  --data '"iuclid6:/dossier-uuid"'
```

Valid resources for printing are entities, dossiers or single documents within a composite entity. Some example I6URIs can be found below:

- Dossier: "iuclid6:/<dossier-uuid>"
- Substance: "iuclid6:/0/SUBSTANCE/<substance-uuid>"
- Substance Composition document: "iuclid6:/0/SUBSTANCE/<substance-uuid>/FLEXIBLE_RECORD.SubstanceComposition/<composition-doc-uuid>"

6.6.2. CSR Report

To generate the CSR report for a dataset or dossier to RTF format, one can perform the following call:

POST	/system/script

To specify the resource to be printed, a valid I6URI needs to be provided in the HTTP request body

Body	"{I6URI}"

The `Content-Type` header needs to be set as follows:

Content-Type	application/vnd.iuclid6.ext+json;type=web.csr

The `Accept` header needs to be set as follows:

Accept	application/vnd.iuclid6.ext+json;type=web.rtf

For example, to produce the CSR a dossier with uuid "dossier-uuid" we may perform the following call:

```
curl --request POST \
  --url http://localhost:8080/iuclid6-ext/api/ext/v1/system/script \
  --header 'content-type: application/vnd.iuclid6.ext+json;type=web.csr' \
  --header 'accept: application/vnd.iuclid6.ext+json;type=web.rtf' \
  --header 'iuclid6-user: username' \
  --header 'iuclid6-pass: ********' \
  --data '"iuclid6:/dossier-uuid"'
```

6.7. Dossier filtering (since v3.0.0)

One may filter a dossier by performing the following call:

POST	/dossier/{dossier-uuid}/filter

The `Content-Type` header needs to be set as follows:

Content-Type	application/vnd.iuclid6.ext+json;type=web.FilterRequest

The `Accept` header needs to be set as follows:

Accept	application/vnd.iuclid6.ext+json;type=web.FilterReport

The body of the HTTP request should be an empty object:

Body	{ }

The filtering ruleset configuration used is the built-in configuration in the IUCLID application.

The result of this call is a filtering report comprised of a list of rows. Each row represents an event emitted by the filtering engine and has the following structure:

Name	Type	Description
entity	string	The entity related to the filtering event. Possible values: • "Document", • "Dossier", • "Substance", • "Mixture/Product", • "Category", • "Template", • "Reference substance", • "Legal entity", • "Legal entity site", • "Contact", • "Literature Reference", • "Endpoint Study Record", • "Endpoint Summary", • "Flexible Summary", • "Flexible Record", • "Fixed Record", • "Assessment entity", • "Annotation", • "Test materials"
sectionName	string	The section name where the relevant entity belongs to (e.g., "1.2 Composition")
documentName	string	The name of the affected document (e.g., "Boundary composition.001")
field	string [ ]	The localized path of the affected field as an array of localized path segments (e.g., ["Composition", "General Information", "Name"])

Name	Type	Description
outcome	string	The filtering outcome. Possible values: • "Published", • "Not published", • "Flagged confidential – Not published", • "Not published (ineligible for dissemination)", • "Filtered dossier contained confidentiality claim on tonnage band – Not published", • "Filtered dossier contained confidentiality claim on the registration number – Not published", • "Removed open picklist 'Other' field", • "Removed picklist remark", • "Filtered section 0.4 Assessment entities", • "Removed constituent", • "Filtered category member dataset", • "Filtered category member dataset - Substance identity claimed confidential"
sourceDocumentKey	string	The document key of the affected document e.g. `"164eb2f1-6b40-493b-95a6-851cc072d47b/59749183-d14c-40f1-975b-39d1024dcc4f"`
path	string	The path of the affected entity (e.g., *FLEXIBLE_RECORD/SubstanceComposition.GeneralInformation.Name*)

The JSON structure of the response looks like below:

```
{
"rows": [
...
{
"entity": "Flexible Record",
"sectionName": "1.2 Composition",
"documentName": "Boundary composition.001",
"field": [
"Composition",
"General Information",
"Justification for deviations"
],
"outcome": "Not published",
"sourceDocumentKey": "164eb2f1-6b40-493b-95a6-851cc072d47b/59749183-d14c-40f1-975b-39d1024dcc4f",
"path":
```

```
"FLEXIBLE_RECORD/SubstanceComposition.GeneralInformation.JustificationForDevi
ations"
},
{
"entity": "Flexible Record",
"sectionName": "1.2 Composition",
"documentName": "Boundary composition.001",
"field": [
"Composition",
"General Information",
"Related composition(s)",
"Related composition"
],
"outcome": "Published",
"sourceDocumentKey": "164eb2f1-6b40-493b-95a6-851cc072d47b/59749183-d14c-
40f1-975b-39d1024dcc4f",
"path":
"FLEXIBLE_RECORD/SubstanceComposition.GeneralInformation.RelatedCompositions.
RelatedComposition"
},
...
]
}
```

For example, to filter the dossier with uuid "dossier-uuid" we may perform the following call:

```
curl --request POST \
  --url http://localhost:8080/iuclid6-ext/api/ext/v1/dossier/dossier-
uuid/filter \
  --header 'content-type:
application/vnd.iuclid6.ext+json;type=web.FilterRequest' \
  --header 'accept: application/vnd.iuclid6.ext+json;type=web.FilterReport' \
  --header 'iuclid6-user: username' \
  --header 'iuclid6-pass: ********' \
  --data '{}'
```

7. Inventories

Every REST resource in this document is under /iuclid6-ext/api/ext/v1/

The IUCLID6 Rest API supports paged inventory entry listing and searching. This is an unsecured resource. In other words, **no authentication** is needed to access this resource

GET	/inventory

The above resource supports the following query parameters:

Param	Type	Description	Default
count	boolean	include the total count of the results. If the total count is not needed, omitting it improves performance.	true
l	integer	the number of results (limit) per page.	10
o	integer	the offset of the first result in page. e.g. to fetch the 3rd page of 25 (l=25) results specify o=50 (o = l * zero_indexed_page_number)	0
code	string	matches the code of the inventory. E.g. 'EC'	
number	string	matches the number of the inventory entry. E.g. '500-001*'	
name	string	matches the name of the inventory entry. E.g. '*Methyl*'	
cas	string	matches the CAS number of the inventory entry. E.g. '50-*'	
formula	string	matches the molecular formula of the inventory entry. E.g. 'C11H15Cl2NO.ClH'	
q	string	matches the specified value in either of the following inventory entry fields: CAS number molecular formula number name	

7.1. Results

The inventory entry searching results are wrapped inside a standard payload that contains:

- the results of the search
- plus some additional information to support paging, such as

- the limit
- the offset
- and the total count of the results.

Each result comprises of two values, the URI and the representation:


```
{
   "limit": 15,
   "totalCount": 407,
   "results": [
         {
               "uri": "iuclid6:/EC/200-057-3",
               "representation": {
                     "inventoryCode": "EC",
                     "inventoryTitle": "EC Inventory",
                     "inventoryVersion": "1.3",
                     "number": "200-057-3",
                     "status": "ACTIVE",
                     "casNumber": "50-66-8",
                     "name": "6-(methylthio)purine",
                     "molecularFormula": "C6H6N4S",
                     "createdOn": "1990-06-15T00:00:00Z",
                     "modifiedOn": "2015-02-19T22:00:00Z"
               }
         },
...
```

8. Users (since v5.12.0)

The API supports listing users, loading, editing, and deleting a user.

8.1. Listing

The resource lists users that are accessible to the *actor.*

GET	/system/users

The above resource supports the following query parameters

Query Params	Type	Description	Default
l	number	the limit for the results	10
o	number	the offset for the results	0
permissions	boolean	whether to load the permissions that the **actor** has for each user	false

A successful response (200 HTTP status code) is a JSON object representing a paged result list, each item corresponding to a user-entity.

Example of a request:

```
curl --request GET 'http://industry-release.iuclid6.trasys.dev.gr/iuclid6-ext/api/ext/v1/system/users \
--header 'IUCLID6-USER: SuperUser' \
--header 'IUCLID6-PASS: *********'
```

And its response:

```
{
    "limit": 10,
    "totalCount": 2,
    "results": [{
            "uri": "iuclid6:/user/SuperUser",
            "representation": "SuperUser",
            "permissions": []
        }, {
            "uri": "iuclid6:/user/pub-efsa-user",
            "representation": "pub-efsa-user",
            "permissions": []
        }
    ]
}
```

8.2. Load

Loads the data for a given user-entity identified by {username}.

Note: Standard implementation for this resource is pending. As a temporary solution the implementation done for the web-UI client can be used, but with the disclaimer that the payload of the request and response might change with newer IUCLID versions.

GET	/system/user/{username}

The response of the resource is extensible and is controlled by the media type of the `Accept` header.

Accept	application/vnd.iuclid6.ext+json;type=web.UserManagement

A successful response (200 HTTP status code) is a JSON object representing a user-entity with all its data.

Example of a request:

```
curl --location --request GET 'http://localhost:8080/iuclid6-
ext/api/ext/v1/system/user/SuperUser' \
--header 'IUCLID6-USER: SuperUser' \
--header 'IUCLID6-PASS: root' \
--header 'Accept: application/vnd.iuclid6.ext+json;type=web.UserManagement'
```

And its response:

```
{
    "username": "SuperUser",
    "password": null,
    "firstname": "User",
    "lastname": "Super",
    "email": "superuser@iuclid6.com",
    "remarks": "This is the super user",
    "accessAll": true,
    "groups": [
        "Common"
    ],
    "roles": [
        "Read-only",
        "Full access",
        "System administrator"
    ],
    ...
}
```

8.3. Create user

Creates a new user-entity.

Note: Standard implementation for this resource is pending. As a temporary solution the implementation done for the web-UI client can be used, but with the disclaimer that the payload of the request and response might change with newer IUCLID versions.

POST	/system/user/{username}

The `Content-Type` header needs to be set as follows:

Content-Type	application/vnd.iuclid6.ext+json;type=web.UserManagement

Body	{
	"password": "Test3Password",
	"username": "test3",
	"firstname": "Tester3",
	"lastname": "Test",
	"email": "test@homail.com",
	…
	}

A successful response (201 HTTP status code) is a JSON object with a source property.

Example of a request:

```
curl --location --request POST 'http://localhost:8080/iuclid6-
ext/api/ext/v1/system/user/test3' \
--header 'IUCLID6-USER: SuperUser' \
--header 'IUCLID6-PASS: root' \
--header 'Content-Type:
application/vnd.iuclid6.ext+json;type=web.UserManagement' \
--data-raw '{
  "password": "Test3Password",
  "username": "test3",
  "firstname": "Tester3",
  "lastname": "Test",
  "email": "test@homail.com",
  "contactKey": "",
  "locked": false,
  "suspended": false,
  "expired": false,
  "lastLoginDate": "",
  "remarks": "",
```

```
  "accessAll": false,
  "roles": [
    "Read-only"
  ],
  ...
}'
```

And its response:

```
{
    "source": "test3",
    "links": []
}
```

8.4. Edit user

Updates the properties of an existing user-entity.

Note: Standard implementation for this resource is pending. As a temporary solution the implementation done for the web-UI client can be used, but with the disclaimer that the payload of the request and response might change with newer IUCLID versions.

PUT	/system/user/{username}

The `Content-Type` header needs to be set as follows:

Content-Type	application/vnd.iuclid6.ext+json;type=web.UserManagement

Body	{
	"username": "test3",
	"firstname": "Tester3 updated",
	"lastname": "Test updated",
	"email": "test@homail.com",
	…
	}

A successful response (200 HTTP status code) is a JSON object with a source property.

Example of a request:

```
curl --location --request PUT 'http://localhost:8080/iuclid6-
ext/api/ext/v1/system/user/test3' \
--header 'IUCLID6-USER: SuperUser' \
```

```
--header 'IUCLID6-PASS: root' \
--header 'Content-Type:
application/vnd.iuclid6.ext+json;type=web.UserManagement' \
--data-raw '{
  "username": "test3",
  "firstname": "Tester3 updated",
  "lastname": "Test updated",
  "email": "test@homail.com",
  "contactKey": "",
  "locked": false,
  "suspended": false,
  "expired": false,
  "lastLoginDate": "",
  "remarks": "",
  "accessAll": false,
  "roles": [
    "Read-only"
  ],
  ...
}'
```

And its response:

```
{
    "source": "test3",
    "links": []
}
```

8.4.1. Change Password for user

Updates the password of an existing user-entity.

PUT	/system/user/{username}

The `Content-Type` header needs to be set as follows:

Content-Type	application/vnd.iuclid6.ext+json;type=standard.password

The body of the HTTP request should be the new password:

Body	`{new-password}`

A successful response (200 HTTP status code) is a JSON object with a source property.

Example of a request:

```
curl --location --request PUT 'http://industry-
release.iuclid6.trasys.dev.gr/iuclid6-ext/api/ext/v1/system/user/pub-efsa-
user' \
--header 'IUCLID6-USER: SuperUser' \
--header 'IUCLID6-PASS: *********' \
--header 'Content-Type:
application/vnd.iuclid6.ext+json;type=standard.password' \
--data-raw '"my-new-password"'
```

And its response:

```
{
    "source": "test3",
    "links": []
}
```

8.5. Delete user

Deletes an existing user-entity.

DELETE	/system/user/{username}

The body of the HTTP request should be empty.

A successful response (204 HTTP status code) has no content.

Example of a request:

```
curl --location --request DELETE 'http://industry-
release.iuclid6.trasys.dev.gr/iuclid6-ext/api/ext/v1/system/user/pub-efsa-
user' \
--header 'IUCLID6-USER: SuperUser' \
--header 'IUCLID6-PASS: root'
```

9. Roles (since v5.12.0)

The API supports listing roles, loading, editing, and deleting a role.

9.1. Listing

The resource lists roles that are accessible to the *actor.*

GET	/system/roles

The above resource supports the following query parameters

Query Params	Type	Description	Default
l	number	the limit for the results	10
o	number	the offset for the results	0
permissions	boolean	whether to load the permissions that the **actor** has for each user	false

A successful response (200 HTTP status code) is a JSON object representing a paged result list, each item corresponding to a role.

Example of a request:

```
curl --request GET 'http://industry-release.iuclid6.trasys.dev.gr/iuclid6-ext/api/ext/v1/system/roles' \
--header 'IUCLID6-USER: SuperUser' \
--header 'IUCLID6-PASS: *********'
```

And its response:

```
{
    "limit": 10,
    "totalCount": 5,
    "results": [{
            "uri": "i6sys:/role?id=1",
            "representation": "System administrator",
            "permissions": []
        }, {
            "uri": "i6sys:/role?id=2",
            "representation": "User manager",
            "permissions": []
        }, {
            "uri": "i6sys:/role?id=3",
            "representation": "Group manager",
```

```
            "permissions": []
        }, {
            "uri": "i6sys:/role?id=4",
            "representation": "Full access",
            "permissions": []
        }, {
            "uri": "i6sys:/role?id=5",
            "representation": "Read-only",
            "permissions": []
        }
    ]
}
```

9.2. Load

Loads the data for an existing role identified by {rolename}.

Note: Standard implementation for this resource is pending. As a temporary solution the implementation done for the web-UI client can be used, but with the disclaimer that the payload of the request and response might change with newer IUCLID versions.

GET	/system/role/{rolename}

The response of the resource is extensible and is controlled by the media type of the `Accept` header.

Accept	application/vnd.iuclid6.ext+json;type=web.RoleManagement

A successful response (200 HTTP status code) is a JSON object representing a role with its properties.

Example of a request:

```
curl --location --request GET 'http://localhost:8080/iuclid6-
ext/api/ext/v1/system/role/Full%20access ' \
--header 'IUCLID6-USER: SuperUser' \
--header 'IUCLID6-PASS: root' \
--header 'Accept: application/vnd.iuclid6.ext+json;type=web.RoleManagement'
```

And its response:

```
{
    "name": "Full access",
    "description": "Predefined role for Full access",
    "systemConfiguration": false,
    "securityManagement": false,
    "managePrivateGroups": false,
    "manageRoles": null,
    "manageUsers": null,
```

```
    "manageGroups": null,
    "assignLeToUsers": false,
    "assignRolesToUsers": false,
    "removeRolesFromUsers": false,
    "assignUsersToGroups": false,
    "opPrint": true,
    "opExport": true,
    "opImport": true,
    "opValidationAssistant": true,
    "opGenerateReport": true,
    ...
}
```

9.3. Create role

Creates a new role.

Note: Standard implementation for this resource is pending. As a temporary solution the implementation done for the web-UI client can be used, but with the disclaimer that the payload of the request and response might change with newer IUCLID versions.

POST	/system/role/{rolename}

The `Content-Type` header needs to be set as follows:

Content-Type	application/vnd.iuclid6.ext+json;type=web.RoleManagement

Body	{ "name": "MyRole", "description": "The description of my role", "opPrint": true, "opExport": true, "opImport": true, "opValidationAssistant": false, "opGenerateReport": false, "opManageReport": false, … }

A successful response (201 HTTP status code) is a JSON object with a source property.

Example of a request:

```
curl --location --request POST 'http://localhost:8080/iuclid6-
ext/api/ext/v1/system/role/MyRole' \
--header 'IUCLID6-USER: SuperUser' \
--header 'IUCLID6-PASS: root' \
--header 'Content-Type:
application/vnd.iuclid6.ext+json;type=web.RoleManagement' \
--data-raw '{
  "name": "MyRole",
  "description": "The description of my role",
  "opPrint": true,
  "opExport": true,
  "opImport": true,
  ...
}'
```

And its response:

```
{
    "source": "i6sys:/role?name=MyRole",
    "links": []
}
```

9.4. Edit role

Updates the properties of an existing role.

Note: Standard implementation for this resource is pending. As a temporary solution the implementation done for the web-UI client can be used, but with the disclaimer that the payload of the request and response might change with newer IUCLID versions.

PUT	/system/role/{rolename}

The `Content-Type` header needs to be set as follows:

Content-Type	application/vnd.iuclid6.ext+json;type=web.RoleManagement

Body	{
	"name": "MyRole",
	"description": "The description of my role updated",
	"opPrint": true,
	"opExport": true,
	"opImport": true,
	"opValidationAssistant": true,
	"opGenerateReport": true,
	"opManageReport": false,
	…
	}

A successful response (200 HTTP status code) is a JSON object with a source property.

Example of a request:

```
curl --location --request PUT 'http://localhost:8080/iuclid6-
ext/api/ext/v1/system/role/MyRole' \
--header 'IUCLID6-USER: SuperUser' \
--header 'IUCLID6-PASS: root' \
--header 'Content-Type:
application/vnd.iuclid6.ext+json;type=web.RoleManagement' \
--data-raw '{
  "name": "MyRole",
  "description": "The description of my role updated",
  "opPrint": true,
  "opExport": true,
  "opImport": true,
  ...
}'
```

And its response:

```
{
    "source": "i6sys:/role?name=MyRole",
    "links": []
}
```

9.5. Delete role

Deletes an existing role.

DELETE	/system/role/{rolename}

The body of the HTTP request should be empty.

A successful response (204 HTTP status code) has no content.

Example of a request:

```
curl --location --request DELETE 'http://industry-
release.iuclid6.trasys.dev.gr/iuclid6-ext/api/ext/v1/system/role/MyRole' \
--header 'IUCLID6-USER: SuperUser' \
--header 'IUCLID6-PASS: root'
```

10. Groups (since v5.12.0)

The API supports listing groups, loading, editing, and deleting a group.

10.1. Listing

The resource lists groups that are accessible to the *actor*.

GET	/system/groups

The above resource supports the following query parameters

Query Params	Type	Description	Default
l	number	the limit for the results	10
o	number	the offset for the results	0
permissions	boolean	whether to load the permissions that the **actor** has for each user	false

A successful response (200 HTTP status code) is a JSON object representing a paged result list, each item corresponding to a group.

Example of a request:

```
curl --request GET 'http://industry-release.iuclid6.trasys.dev.gr/iuclid6-
ext/api/ext/v1/system/groups\ --header 'IUCLID6-USER: SuperUser' \
--header 'IUCLID6-PASS: *********'
```

And its response:

```
{
    "limit": 10,
    "totalCount": 1,
    "results": [{
            "uri": "i6sys:/group?id=1",
            "representation": "Common",
            "permissions": []
        }
    ]
}
```

10.2. Load

Loads the data for an existing group identified by {groupname}.

Note: Standard implementation for this resource is pending. As a temporary solution the implementation done for the web-UI client can be used, but with the disclaimer that the payload of the request and response might change with newer IUCLID versions.

GET	/system/role/{groupname}

The response of the resource is extensible and is controlled by the media type of the `Accept` header.

Accept	application/vnd.iuclid6.ext+json;type=web.GroupManagement

A successful response (200 HTTP status code) is a JSON object representing a group with its properties.

Example of a request:

```
curl --location --request GET 'http://localhost:8080/iuclid6-
ext/api/ext/v1/system/group/Common' \
--header 'IUCLID6-USER: SuperUser' \
--header 'IUCLID6-PASS: root' \
--header 'Accept: application/vnd.iuclid6.ext+json;type=web.GroupManagement'
```

And its response:

```
{
    "name": "Common",
    "description": "Predefined common group",
    "memberUsers": [
        {
            "username": "SuperUser",
            "editable": false,
            "isManager": true
        },
        {
            "username": "test",
            "editable": true,
            "isManager": false
        },
        {
            "username": "test2",
            "editable": true,
            "isManager": false
        }
    ]
}
```

10.3. Create group

Creates a new group.

Note: Standard implementation for this resource is pending. As a temporary solution the implementation done for the web-UI client can be used, but with the disclaimer that the payload of the request and response might change with newer IUCLID versions.

POST	/system/role/{groupname}

The `Content-Type` header needs to be set as follows:

Content-Type	application/vnd.iuclid6.ext+json;type=web.GroupManagement

Body	<pre>{
	"name": "MyGroup",
	"description": "The description of my group",
	"memberUsers": [
	{
	"username": "SuperUser",
	"editable": false,
	"isManager": true
	},
	{
	"username": "test",
	"editable": true,
	"isManager": false
	}
	]
	}</pre>

A successful response (201 HTTP status code) is a JSON object with a source property.

Example of a request:

```
curl --location --request POST 'http://localhost:8080/iuclid6-
ext/api/ext/v1/system/group/MyRole' \
--header 'IUCLID6-USER: SuperUser' \
--header 'IUCLID6-PASS: root' \
--header 'Content-Type:
application/vnd.iuclid6.ext+json;type=web.GroupManagement' \
```

```
--data-raw '{
    "name": "MyGroup",
    "description": "The description of my group",
    "memberUsers": [
        {
            "username": "SuperUser",
            "editable": false,
            "isManager": true
        },
        {
            "username": "test",
            "editable": true,
            "isManager": false
        }
    ]
}'
```

And its response:

```
{
    "source": "i6sys:/group?name=MyGroup",
    "links": []
}
```

10.4. Edit group

Updates the properties of an existing role.

Note: Standard implementation for this resource is pending. As a temporary solution the implementation done for the web-UI client can be used, but with the disclaimer that the payload of the request and response might change with newer IUCLID versions.

PUT	/system/role/{rolename}

The `Content-Type` header needs to be set as follows:

Content-Type	application/vnd.iuclid6.ext+json;type=web.RoleManagement

Body	{
	"name": "MyGroup",
	"description": "The description of my group updated",
	"memberUsers": [
	{
	"username": "SuperUser",
	"editable": false,
	"isManager": true
	},
	{
	"username": "test2",
	"editable": true,
	"isManager": false
	}
	]
	}

A successful response (200 HTTP status code) is a JSON object with a source property.

Example of a request:

```
curl --location --request PUT 'http://localhost:8080/iuclid6-
ext/api/ext/v1/system/group/MyGroup' \
--header 'IUCLID6-USER: SuperUser' \
--header 'IUCLID6-PASS: root' \
--header 'Content-Type:
application/vnd.iuclid6.ext+json;type=web.GroupManagement' \
--data-raw '{
    "name": "MyGroup",
    "description": "The description of my group updated",
    "memberUsers": [
        {
            "username": "SuperUser",
            "editable": false,
            "isManager": true
        },
        {
            "username": "test2",
            "editable": true,
            "isManager": false
        }
```

```
    ]
}'
```

And its response:

```
{
    "source": "i6sys:/group?name=MyGroup",
    "links": []
}
```

10.5. Delete group

Deletes an existing group.

DELETE	/system/group/{groupname}

The body of the HTTP request should be empty.

A successful response (204 HTTP status code) has no content.

Example of a request:

```
curl --location --request DELETE 'http://industry-
release.iuclid6.trasys.dev.gr/iuclid6-ext/api/ext/v1/system/group/MyGroup' \
--header 'IUCLID6-USER: SuperUser' \
--header 'IUCLID6-PASS: root'
```