# IUCLID6 Public REST API

# Definitions

## Introduction

IUCLID6 is a highly extensible system, especially when it comes to the documents it can handle and how these are organized in datasets.

A legislation provider can, through a well defined mechanism, specify:

- document templates, 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,

The IUCLID6 Public REST API provide several resources to retrieve information about registered legislation 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/`

## Legislation providers

Registered legislation 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 legislation provider. |
| title | string | The localized name of the provider, in the requested language via the **Accept-Language** header. <br> If the header is omitted, or no localization exists for the requested language, English is used. |
| version | string | The version of the provider. |

## Submission types

Each legislation 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: |
| | <ul><li>SUBSTANCE</li><li>MIXTURE</li><li>TEMPLATE</li></ul> |

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"
  },
  {
    "provider": "bpr",
    "identifier": "BIOC_ACTIVE_SUBSTANCE_FOR_MIXTURES",
    "title": "BPR Active substance application (representative product)",
    "applicableFor": "MIXTURE"
  },
  {
    "provider": "bpr",
    "identifier": "BIOC_BIOCIDAL_PRODUCT",
    "title": "BPR Biocidal product authorisation",
    "applicableFor": "MIXTURE"
  },
  {
    "provider": "oecd",
    "identifier": "OECD_HT",
    "title": "OECD harmonised templates",
    "applicableFor": "MIXTURE"
  },
  {
    "provider": "bpr",
    "identifier": "BIOC_BASIC_INFORMATION_MIXTURE",
    "title": "BPR Basic information (mixture)",
    "applicableFor": "MIXTURE"
  },
  {
    "provider": "core",
    "identifier": "COMPLETE",
    "title": "Complete table of contents",
    "applicableFor": "SUBSTANCE"
  },
  {
    "provider": "reach",
    "identifier": "R_A15_REST",
    "title": "REACH Annex XV - Restriction",
    "applicableFor": "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 legislation 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. |

| | | |
|---|---|---|
| `applicableFor` | string | The entity type the submission type applies to. |

## Trees

You can retrieve the section tree of a submission type, applicable to a specific entity type at the following resource:

| GET | `/definition/tree/{submission_id}?for={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 the to fetch the tree of the "*REACH Registration above 1000 tonnes*" submission type for substances:

| GET | `/definition/tree/R_ABOVE_1000?for=SUBSTANCE` |
|---|---|

will return:
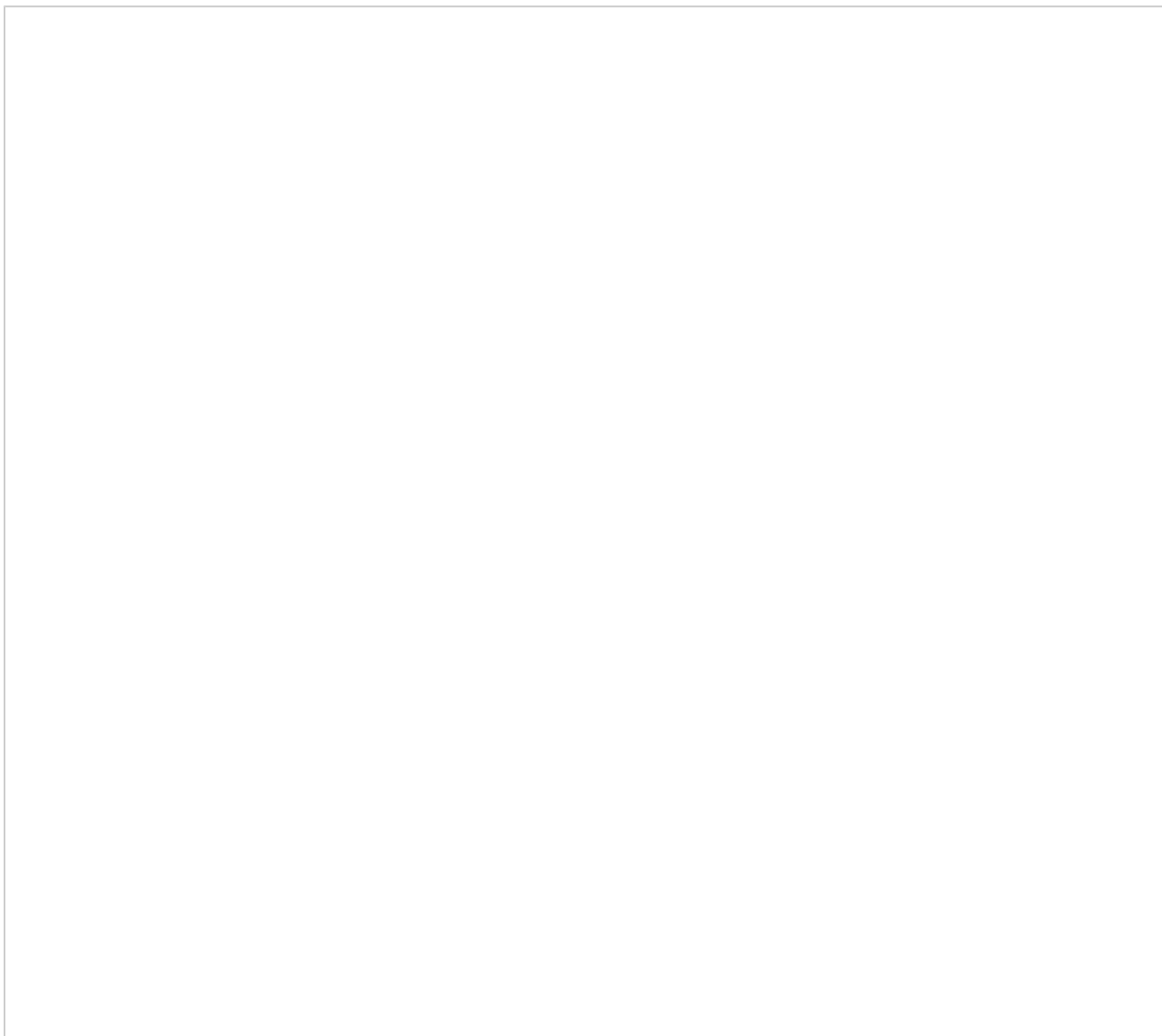
```json
{
  "code": "R_ABOVE_1000",
  "title": "REACH Registration above 1000 tonnes",
  "sections": [
    {
      "code": "0",
      "title": "Related information",
      "sections": [],
      "documents": []
    },
    {
      "code": "1",
      "title": "General information",
      "sections": [
        {
          "code": "1.1",
          "title": "Identification",
          "sections": [],
          "documents": [
            {
              "code": "SUBSTANCE",
              "title": "Substance Identification",
              "access": "NO_ACCESS",
              "required": false,
              "single": true
            }
          ]
        },
        {
          "code": "1.2",
          "title": "Composition",
          "sections": [],
          "documents": [
            {
              "code": "FLEXIBLE_RECORD.SubstanceComposition",
              "title": "Composition",
              "access": "NO_ACCESS",
              "required": true,
              "single": false
            }
          ]
        },
        {
          "code": "1.3",
          "title": "Identifiers",
          "sections": [],
          "documents": [
            {
              "code": "FIXED_RECORD.Identifiers",
              "title": "Identifiers",
              "access": "NO_ACCESS",
              "required": false,
              "single": true
            }
          ]
        }
...
```

The response is an 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 |
| 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. |
| access | string | If the request is authenticated, it contains the privileges of the authenticated user on the specific document type. Can be one of: <ul><li>NO_ACCESS</li><li>READ_ONLY</li><li>READ_WRITE</li><li>FULL_ACCESS</li></ul> When the request is unauthenticated it will always contain 'NO_ACCESS' |
| required | boolean | True if the specific document type is required for the submission. |
| single | boolean | True if only one document of the specific type is allowed. e.g. The "Identification" of the substance |

## 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.

To help visualizing the aforementioned structure, follows an abstract of the definition of the "LEGAL_ENTITY" document:

```json
{
  "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 show above, is: "LEGAL_ENTITY.GeneralInfo.LegalEntityName"

| 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 legislation 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:<br><br>Can be one of:<br><br>- block<br>- dataProtection<br>- text<br>- boolean<br>- date<br>- numeric<br>- picklist<br>- quantity<br>- range<br>- docRef<br>- blockRef<br>- inventoryRef<br>- definitionRef<br>- attachment<br>- 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.<br><br>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 values. |
| `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 |
| text | mimeType | string | The mime-type of the text contents of the field.<br><br>Can be either: **text/plain** or **text/html** |
|  | `maxLength` | number | The maximum length the field support.<br><br>*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<br>• 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 |
| | 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<br>• 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<br>lowerBoundQualifiers<br>upperBoundQualifiers | array | The allowed boundary qualifiers. Depending on whether the field is half-bounded,<br><br>either only the boundQualifiers or both of the lowerBoundQualifiers and upperBoundQualifiers, will be present.<br><br>The allowed values are:<br><br>• ca.<br>• <<br>• <=<br>• ><br>• >= |
| 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.<br><br>Can be one of the following:<br><br>• image/*<br>• */* |

**Media type**
The media type for the document definition object is **application/vnd.iuclid6.ext+json;type=iuclid6.Definition**

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

**GET** `/definition/document/{document_type}`

A listing of every registered document definition can be obtained by the following resource:

**GET** `/definition/documents`

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

However you can obtain the full contents of each definition by setting the "formatter" query parameter to "iuclid6.Definition": `/definition/documents?formatter=iuclid6.Definition`

# 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.

You 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"`) |

## 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
  },...
```

## Text templates

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

**GET** `/definition/texts`

```
{
  "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 legislation provider, providing this text template |
| header | string | A title for this text template. |
| content | string | The actual template content. |

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

**GET** `/definition/texts/{code}`

# 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 obtain any of the following values:

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

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, omiting 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 |

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

```
GET   /query/iuclid6/byType?doc.type=SUBSTANCE&l=20&o=20&count=true
```

## 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.

```
GET   /query/iuclid6/byType?doc.type=SUBSTANCE
```

```
{
  "limit": 10,
  "offeset": 0,
  "totalCount": 102,
  "results": [
    {
      "uri":
"iuclid6:/0/SUBSTANCE/f200c268-7a94-4fb6-9d1d-6759d71f6894/SUBSTANCE/f200c268-7a94-4fb
6-9d1d-6759d71f6894",
      "representation": null
    },
    {
      "uri":
"iuclid6:/0/SUBSTANCE/d4e388de-aa35-4568-b25e-39a6538363a6/SUBSTANCE/d4e388de-aa35-456
8-b25e-39a6538363a6",
      "representation": null
    },
    {
      "uri":
"iuclid6:/0/SUBSTANCE/c61b4b3a-ffbc-46b4-a5dc-f3ccb512c11a/SUBSTANCE/c61b4b3a-ffbc-46b
4-a5dc-f3ccb512c11a",
      "representation": null
    },
    {
      "uri":
"iuclid6:/0/SUBSTANCE/db63a1af-6304-427f-aa1d-387b1f8fefb5/SUBSTANCE/db63a1af-6304-427
f-aa1d-387b1f8fefb5",
      "representation": null
    },
    {
      "uri":
"iuclid6:/0/SUBSTANCE/IUC5-f0958d98-4cb3-4058-80cd-bdc24e7d125d/SUBSTANCE/IUC5-f0958d9
8-4cb3-4058-80cd-bdc24e7d125d",
      "representation": null
    },
    {
      "uri":
"iuclid6:/0/SUBSTANCE/40374a6f-9175-4ac8-87c9-d220c866997a/SUBSTANCE/40374a6f-9175-4ac
```

8-87c9-d220c866997a",
      "representation": null
    },
    {
      "uri":
"iuclid6:/0/SUBSTANCE/IUC4-f696abe6-513a-3771-b2ee-76e4110e8103/SUBSTANCE/IUC4-f696abe
6-513a-3771-b2ee-76e4110e8103",
      "representation": null
    },
    {
      "uri":
"iuclid6:/0/SUBSTANCE/326ab136-d74c-4088-9425-8d9d5642b96e/SUBSTANCE/326ab136-d74c-408
8-9425-8d9d5642b96e",
      "representation": null
    },
    {
      "uri":
"iuclid6:/0/SUBSTANCE/1ef71d52-ddc7-4b55-9c2f-ea72efa55ddd/SUBSTANCE/1ef71d52-ddc7-4b5
5-9c2f-ea72efa55ddd",
      "representation": null
    },
    {
      "uri":
"iuclid6:/0/SUBSTANCE/7cec4254-65f7-4d9a-9a6a-2a74dd1c9b1c/SUBSTANCE/7cec4254-65f7-4d9
a-9a6a-2a74dd1c9b1c",
      "representation": null

```
        }
    ]
}
```

The above shows a successful response back from the call. In this case SUBSTANCE documents where 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.<br><br>*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 you need to set the "formatter" query parameter to "iuclid6.DocumentSecuredRepresentation".

| **GET** | `/query/iuclid6/byType?doc.type=SUBSTANCE?o=0&l=1&formatter=iuclid6.DocumentSecuredRepresentation` |
|---------|---------------------------------------------------------------------------------------------------|

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

**QueryResults**

```json
{
  "limit": 1,
  "offset": 0,
  "totalCount": 102,
  "results": [
    {
      "uri":
"iuclid6:/0/SUBSTANCE/f200c268-7a94-4fb6-9d1d-6759d71f6894/SUBSTANCE/f200c268-7a94-4fb
6-9d1d-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. |

## 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 incude the following properties:

| Name | Type | Description |
|---|---|---|
| classtype | String | the sub-type of the document representation.<br><br>*e.g. SubstanceSecuredRepresentation* |
| key | String | the document key formatted as {uuid}/{snapshot}. |
| definition | String | the document definition identifier.<br><br>*e.g. ENDPOINT_STUDY_RECORD.Partition* |
| name | String | the name of the document |
| accessRight | String | the access right the user has on the document.<br><br>Possible values are:<br><br><ul><li>NO_ACCESS</li><li>READ_ONLY</li><li>READ_WRITE</li><li>FULL_ACCESS</li></ul> |
| createdOn | String | the instant the document was created, in ISO 8601 format<br><br>*e.g. 2016-09-06T13:14:59.187Z* |
| modifiedOn | String | the instant the document was last modified, in ISO 8601 format<br><br>*e.g. 2016-09-01T07:20:11Z* |
| remarks | String | the optional, remarks of the document |

## Ordering

The results are ordered by default:

- by descending touched date ℹ️
- 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.

You 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` |
|---|---|

## 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.<br><br>In general, the following IUCLID6 document types are supported:<br><br>• DOSSIER<br>• SUBSTANCE<br>• MIXTURE<br>• TEMPLATE<br>• CATEGORY<br>• ANNOTATION<br>• LEGAL_ENTITY<br>• SITE<br>• REFERENCE_SUBSTANCE<br>• CONTACT<br>• LITERATURE<br>• TEST_MATERIAL_INFORMATION<br><br>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:<br><br>• ME<br>• OTHER |

**The following queries are available:**

## byDate

| byDate | |
|---|---|
| Find entities by creation/modification date. | |
| **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" |

## byName

| byName | |
|---|---|
| Find entities of specific name. | |
| **Parameter name** | **Description** |
| doc.name | The name of the document |

## byContact

| byContact | |
|---|---|
| Find contacts by various criteria. | |
| **Note: The doc.type parameter should be omitted** | |
| **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 |

## bySubstance

## bySubstance

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

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

| 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.<br><br>Should be one of the phase codes in Phasegroup **N12** |
| sub.it_system_id | The IT System Identifier of the FIXED_RECORD.Identifiers child of the SUBSTANCE. |

## byMixture

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

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

| 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.<br><br>Should be one of the phase codes in Phasegroup **N12** |
| sub.it_system_id | The IT System Identifier of the FIXED_RECORD.Identifiers child of the MIXTURE. |

**byLegalEntity**

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

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.

| Parameter name | Description |
|---|---|
| le.name | The legal entity name. |
| le.country | **Picklist.** The country of the legal entity.<br><br>Should be one of the phase codes in Phasegroup **N03** |
| le.town | The town of the legal entity. |

**byReferenceSubstance**

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

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

| 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. |

**byTemplate**

| byTemplate | |
|---|---|
| Find 'Templates' satisfying certain criteria. | |
| **Note: the doc.type parameter should be omitted.** | |

| 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 Phasegroup **N03** |
| le.town | The town of the legal entity. |

**byLiterature**

<table>
<tr><td colspan="2" align="center">**byLiterature**</td></tr>
<tr><td colspan="2">`Find 'Literature references' satisfying certain criteria.`<br><br>**Note: the doc.type parameter should be omitted.**</td></tr>
<tr><td>**Parameter name**</td><td>**Description**</td></tr>
<tr><td>`lit.title`</td><td>The literature title.</td></tr>
<tr><td>`lit.type`</td><td>**Picklist.** The reference type of the literature.<br><br>Should be one of the phase codes in Phasegroup **Z31**</td></tr>
<tr><td>`lit.author`</td><td>The author of the literature.</td></tr>
<tr><td>`lit.ref_year`</td><td>The year of the literature.</td></tr>
<tr><td>`lit.source`</td><td>The bibliographic source.</td></tr>
<tr><td>`lit.test_lab`</td><td>The testing laboratory</td></tr>
<tr><td>`lit.owner`</td><td>The company owner.</td></tr>
<tr><td>`lit.owner_study_no`</td><td>The company study no.</td></tr>
<tr><td>`lit.report_no`</td><td>The report no.</td></tr>
<tr><td>`lit.report.after`</td><td>**Date.** The report should be from the specified date and afterwards, e.g. "2016-06-27"</td></tr>
<tr><td>`lit.report.before`</td><td>**Date.** The document should be created until before the specified date. e.g. "2016-07-27"</td></tr>
</table>

**byAnnotation**

| byAnnotation | |
|---|---|
| Find annotations by various criteria. | |
| **Note: the doc.type parameter should be omitted.** | |
| **Parameter name** | **Description** |
| annotation.status | **Picklist.** The status of the annotation |
| | Should be one of the phase codes in Phasegroup **N16** |
| annotation.authority | The name of the authority/organisation of the annotation. |
| annotation.agreement | **Boolean.** Is the aggreement 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 Phasegroup **N19** |
| annotation.reliability | **Picklist.** The reliability of the literature. |
| | Should be one of the phase codes in Phasegroup **N17** |

**bySite**

| bySite | |
|---|---|
| Find entities, being themselves or referring to, 'Sites' satisfying certain criteria.<br><br>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.

| **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 Phasegroup **N03** |
| site.town | The town of the site. |
| site.owner | The name of the legal entity onwer of the site. |

**byAdditive**

| byAdditive |
|---|
| Find entities having specific reference substances as additives. |

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

| 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. |

**byImpurity**

| byImpurity |
|---|
| Find entities having specific reference substances as impurities. |

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

| 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. |

**byConstituent**

| byConstituent |
|---|

Find entities having specific reference substances as constituent.

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

| 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. |

**byGhs**

## byGhs

Find entities containing GHS sections with certain criteria.

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

| 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.HazardSt* Should be one of the phase codes in Phasegroup **GHS65** |
| ghs.hazard_pictogram | **Picklist.** The hazard pictogram *(path: FLEXIBLE_RECORD.Ghs.Labelling.HazardPictogramBlock.HazardPict* Should be one of the phase codes in Phasegroup **DM02** |
| ghs.precaution_statement | **Picklist.** The precautionary statement *(path: FLEXIBLE_RECORD.Ghs.Labelling.PrecautionaryStatementsBlock.Pre* Should be one of the phase codes in Phasegroup **GHS66** |

**byDsd**

| byDsd | |
|---|---|
| Find entities containing DSD sections with certain criteria.<br><br>Note: the doc.type parameter can be one of:<br><br>    • SUBSTANCE<br>    • MIXTURE<br>    • DOSSIER<br><br>Depending on the value of doc.type parameter, the following criteria are applied:<br><br>    • in case of SUBSTANCE or MIXTURE, to the FLEXIBLE_RECORD.DsdDpd of a substance<br>    • in case of DOSSIER, to the FLEXIBLE_RECORD.DsdDpd of the dossier's subject | |

| 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 Phasegroup **N29** |
| dsd.danger | **Picklist.** The indication of danger *(path: FLEXIBLE_RECORD.DsdDpd.Labelling.IndicationsOfDangerBlock.Indications* Should be one of the phase codes in Phasegroup **N30** |
| dsd.safety | **Picklist.** The safety phrases *(path: FLEXIBLE_RECORD.DsdDpd.Labelling.SafetyPhrases.Safeties.Code)* Should be one of the phase codes in Phasegroup **N31** |

**byJointSubmission**

| byJointSubmission | |
|---|---|
| Find entities containing 'Joint submission' sections with certain criteria.<br><br>Note: the doc.type parameter can be one of:<br><br>    • SUBSTANCE<br>    • MIXTURE<br>    • DOSSIER<br><br>Depending on the value of doc.type parameter, the following criteria are applied:<br><br>    • in case of SUBSTANCE or MIXTURE, to the FLEXIBLE_RECORD.JointSubmission of a substance<br>    • in case of DOSSIER, to the FLEXIBLE_RECORD.JointSubmission of the dossier's subject | |

| 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. |

# Document handling

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

## Datatypes

## `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**

```json
[
  {
    "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.

## 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}` |
| 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* |

## 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`.

### 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": {`<br>`    "Name": "substance-composition-1",`<br>`    "TypeOfComposition": {`<br>`      "code": "61423"`<br>`    }`<br>`  }` |
| 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 "SubstanceComposition"** |

| Repeatable List | | |
| --- | --- | --- |

**document**

```
"Constituents": [
       {
          "uuid": "e67542c8-094b-422e-8bbb-d65d6bd66d22",
          "ReferenceSubstance":
"b0c168d2-c97c-4cc7-83d0-9b2716759030/0",
          "ProportionTypical": {
            "lowerQualifier": ">",
            "lowerValue": 1,
            "unit": {
               "code": "2098"
            }
          }
       }
     ]
```

**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 | A integer value |
| Decimal | number | A decimal value |
| PickList | object | An object containing the '**code**' string field, and optionally a **'other'** string field if the phrase is *'open'*<br><br>The allowed values of the 'code' field vary with the '**phrase-group**' defined for this element in I.TE.M<br><br>e.g.<br><br>```{\n  "code": "1342",\n  "other": "other"\n}``` |
| PickListWithRemarks<br><br>PickListWithRemarks2000 | object | An object containing the **'code'** string field, and optionally a '**other**' string field if the phrase is *open* and a '**remarks**' string field.<br><br>e.g.<br><br>```{\n  "code": "1342",``` |

| PickListWithRemarks32000 | | |
| --- | --- | --- |

```
      "other": "other",
      "remarks": "remarks"
    }
```

| UnitMeasure | object | An object containing a '**value**' number and a '**unit**' object that is a **picklist** e.g. |
| --- | --- | --- |

```
{
  "value": 1.3,
  "unit":{
   "other": "2493"
     }
}
```

| RangeDecimalWithPickList | object | An object containing the following: |
| --- | --- | --- |

| field | type | description |
| --- | --- | --- |
| lowerQualifier | string | one of:<br><br>• ca.<br>• ><br>• >= |
| lowerValue | number | the lower numeric boundary |
| upperQualifier | string | one of:<br><br>• ca.<br>• <<br>• <= |
| upperValue | number | the upper numeric boundary |
| unit | object | a picklist object. |

| RangeDecimal | object | same as 'RangeDecimalWithPickList' whithout the **'unit'** field. |
| --- | --- | --- |
| HalfBounded | object | An object containing the following: |

| field | type | description |
| --- | --- | --- |
| lowerQualifier | string | one of:<br><br>• ca.<br>• <<br>• <=<br>• ><br>• >= |
| lowerValue | number | the lower numeric boundary |
| unit | object | a picklist object. |

| Confidentiality | object | 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 one of the following codes:<ul><li>2732 (*confidential business information*)</li><li>2859 (*intellectual property*)</li><li>3441 (*not publicly available*)</li></ul></td></tr><tr><td>legislations</td><td>array</td><td>an array of picklist objects</td></tr><tr><td>justification</td><td>object</td><td>a picklist object. Allowed codes from phase-group **N78**</td></tr></table> |

| MultiSelect | array | An array of picklist objects with/without remarks. |
| --- | --- | --- |
| MultiSelectWithRemarks | | e.g. |
| MultiSelectWithRemarks2000 | | |
| MultiSelectWithRemarks32000 | | ```[
  {
      "code": "60224",
      "remarks": "remarks-1"
  },
  {
      "code": "60225",
      "remarks": "remarks-2"
  },
  {
      "code": "60230"
  }
]``` |

| EntityReferenceField | string | a document key, formatted as `{uuid}/{snapshot}` |
| --- | --- | --- |
| EndpointReferenceField | | e.g. "4f88bc7f-395c-4d0b-997b-14e8c9aef605/0" |
| SingleFileAttachment | string | an attachment key, formatted as `{uuid}/{snapshot}` |
| Image | | e.g. "4f88bc7f-395c-4d0b-997b-14e8c9aef605/0" |
| EntityReferenceList | array | an array of strings, that represent keys, either of documents or attachments. |
| EndpointReferenceList | | The keys are formatted as `{uuid}/{snapshot}` |
| AttachmentsList | | e.g. |
| LiteratureReferenceList | | ```[
    "77748886-6911-4a41-af93-7d5a86fc075f/0",
    "f9d571cd-d0ed-403d-b944-21e8d26c7c26/0"
]``` |

## Response types

### 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.**<br>Currently empty. |


## 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. |
| code | string | The specific code of the error.<br>*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. |


# Semantics of HTTP status codes

## 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` |


## 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. |
| 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. |

# 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` |
| --- | --- |

## Raw data

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

### 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

> 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: ********'
```

## 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.

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

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

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 `application/vnd.iuclid6.ext+json` as the `Accept` header. 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.

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-c06
296b43820/document/FLEXIBLE_RECORD.SubstanceComposition/1aca8956-fbfe-4f2e-80f7-54182b
c72454 \
   --header 'accept: application/vnd.iuclid6.ext+json; type=iuclid6.Document' \
   --header 'iuclid6-user: SuperUser' \
   --header 'iuclid6-pass: ********'
```

### 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 |

## 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.

### Header document

You access the contents of the header document by:

**GET** `/dossier/{dossier_uuid}/header`

### Subject

The dossier subject is available on:

**GET** `/dossier/{dossier_uuid}/subject`

### Sections of subject

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

**GET** `/dossier/{dossier_uuid}/subject/documents`

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

**GET** `/dossier/{dossier_uuid}/subject/document/{doc_def}/{document_uuid}`

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

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

### Submitter

The submitting legal entity of the dossier is accessible through:

**GET** `/dossier/{dossier_uuid}/submitter`

### 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**.

## 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 |

# Write raw data

Writing operations, namely:

- **creation**
- **modification**
- and **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** |
|------------------|-----------------------------------------------------------|

## Entities

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

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

## 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-9
06c7b864443"
```

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

## Error responses

Same as in create scenario.

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

## Sections

The modification operations on section documents are very similar to those on entities. The only difference is the resources that you make the requests.

Briefly:

| Create | POST | /raw/{entity_type}/{entity_uuid}/document/{doc_def} |
|---|---|---|
| Update | PUT | /raw/{entity_type}/{entity_uuid}/document/{doc_def}/{doc_uuid} |
| Delete | DELETE | /raw/{entity_type}/{entity_uuid}/document/{doc_def}/{doc_uuid} |

The modification operations on section documents are nearly identical to those on entities. The only difference is the resources that you make the requests.

## 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, for example:

Content-Disposition: attachment;filename*=utf-8''my-attachment-filename.txt

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 |

## Create Dossier

To create a dossier of a particular subject you can **POST** a `JsonDocumentEnvelope` of the header document of the dossier to be created on the following resource:

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

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-afd4-4da9-9c01-4a
c12b1bc9c0/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.<br><br>```{<br>  "requestId": "8394333f-9748-49fd-b0f0-af800a1ce953",<br>  "uri": "iuclid6:/0/LEGAL_ENTITY/4f88bc7f-395c-4d0b-997b-14e8c9aef605",<br>  "subjectKey": "LEGAL_ENTITY:4f88bc7f-395c-4d0b-997b-14e8c9aef605/0",<br>  "code": "DOS405",<br>  "message": "Invalid subject for dossier: LEGAL_ENTITY 4f88bc7f-395c-4d0b-997b-14e8c9aef605/0"<br>}``` | You are trying to create a dossier for an entity, <br><br>that can not be a subject of a dossier. |

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

## 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.

# Sharing (data access across security groups)

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

## 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. In order 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.

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.

## Datatypes

The standard media type for exchanging data access information is: **`application/vnd.iuclid6.ext+json;type=standard.access`**

It 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:<br>• NO_ACCESS<br>• READ_ONLY<br>• READ_WRITE<br>• FULL_ACCESS |

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

## Semantics of HTTP status codes

### 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* |

### 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. |

## 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: *********'
```

## 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, lets 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"}]'
```

# Import / Export

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

## Supported Types

### 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

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

### 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).

## Export

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:

- a **job id**;
- **status**;
- **name** (the uuid of the exported entity);
- and the **export result** in binary format (blob).

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.

An example of the response after applying the above request on uuid bfd3c521-9a89-4b25-a982-1ac533e727bf corresponding to a SUBSTANCE entity is shown below:

**JobInfo**

```
{
 "id": "bfd3c521-9a89-4b25-a982-1ac533e727bf",
    "status": "SUCCEEDED",
 "uri": "/system/jobs/bfd3c521-9a89-4b25-a982-1ac533e727bf",
 "representation": {
        "classtype": "Iuclid6Job",
        "group": null,
        "name": "43643b46-6c7b-338f-930b-40e256702c9b",
        "type": "EXPORT",
        "logs": [
          {
            "code": null,
            "level": "INFO",
            "value": "Export process succeeded",
            "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 canceled                              => 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` |
|---|---|

Apparently, the binary output result of the export job, can also 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` |
|---|---|

## 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-e09
6f9562646/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-4e8458
79cc99/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-4e8458
79cc99/result \
  --header 'iuclid6-user: SuperUser' \
  --header 'iuclid6-pass: ********'
```

## 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-e73182e08
764/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-e6f8-4823-8725-1750df
a82e5e/status \
  --header 'accept: text/plain' \
  --header 'iuclid6-user: SuperUser' \
  --header 'iuclid6-pass: ********'
```

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

```
SUCCEEDED
```

In order 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-e6f8-4823-8725-1750df
a82e5e/result \
  --header 'iuclid6-user: SuperUser' \
  --header 'iuclid6-pass: ********'
```

## 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, and the `Content-Type` header can be one of the following:

| IUCLID6 | application/vnd.iuclid6.archive |
| --- | --- |

| IUCLID5 | `application/vnd.iuclid5.archive` |
|---------|----------------------------------|

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.

To change the default overwrite mode set the **overwrite** query parameter with one of the following values:

| Value | Description |
|-------|-------------|
| 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 exist in the system, but is not contained in the archive it will be removed. |

An example of an import request using curl:

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

The same response types as in **EXPORT are used**.