Culture-Hub API Documentation

Release 0.16.1

Sjoerd Siebinga

November 30, 2015

Contents

The initial release of this documentation is focused on making the API documentation available to third-party developers.

Contents:

CHAPTER 1

Introduction Delving Culture-Hub API documentation

This document describes the delving Culture-Hub APIs that are available on the default deployments. The APIs are always constrained to the information of a single organization.

The URL structure that we use is:

```
`http://{baseUrl}:{portNumber}/api/{apiType}`
```

- **baseUrl** = is the basic ip or domain where the hub is hosted.
- **portNumber** = is the port at which the hub is listening for requests. (**default**: 80)
- **apiType** = is the main type of the API. Currently, there are the following main API types:
  - search
  - statistics
  - proxies
  - OAI-PMH harvesting

An example of a full URL is the Norvegiana Culture-Hub:

```
`http://kulturnett2.delving.org:80/api/search?query=norge`
```

The Culture-Hub API has as its core design principle that all the state the application has must be available to the API consumer. This means that computations made on the server should not have to re-computed by the client that is consuming the API. This is also why the response are so elaborate. To reduce the verboseness it is possible to specify which main elements should be returned via the *verbose* and *strict* parameters.

This documentation refers to API version **0.16** and above. The version number of the API can be found as an *@version* attribute in the main element of the API response.

CHAPTER 2

Search API

The API base URL for all search actions is:

```
`http://{baseUrl}:{portNumber}/api/search`
```

The following sections describe the parameters and usage of the three main search modes:

- summary view
- full view
- explain view

2.1 Search Result Mode (summary view)

The *summary view* is the basic search response provided by the hub. In the following sections we will describe the output and the parameters that are available to the API developer.

2.1.1 General parameters

There are some general parameters that apply to all modes and they are described below.

- **wskey** = the APIs can be configured to only be available with an API key. This can be configured in the `production.conf`. By default all APIs are open. When they are protected the `wskey` parameter needs to be specified in the URL as a request parameter.
- **lang** = here the language can be specified. It takes valid ISO two letter codes. When the language is not supported the default *lang* **en** is returned. All the **i18n** elements of the API response are rendered depending on the *lang* that is specified

2.1.2 Basic query parameters

- **query** = is the search term you want to search on. It accepts simple searches and fielded searches with fields that are available in the 'explain response' you must use the fields as they are represented in the `@search` attribute. It accepts all the valid Solr/Lucene query syntax, see https://wiki.apache.org/solr/SolrQuerySyntax and http://lucene.apache.org/core/3_6_0/queryparsersyntax.html
- **start** = is the integer for start page of the results. Mostly you just take the value from the `/results/pagination/` block.
- **rows** = is the integer for number of records that you want to have returned. Default is **20**.

- **format** = is the response format you want the API response to be returned it. The *summary view* mode supports the following enumerated options:
  - *xml* = return the results as XML response. (**default**)
  - *json* = return the results as JSON response.
  - *jsonp* = return the results as padded JSON. The default callback is *delvingCallback*, but you can specify your own with the `callback` parameter.
  - *kml* = return the results in the KML format. This format can be loaded directly in Google Maps or Google Earth for rendering. It filters out all the results that don't have a valid geospatial reference in the record.
  - **custom formats**: there are also a number of custom output formats that have been created for specific projects but don't have a great general re-use. Th e following is a non-exhaustive list of them:
    - *simile* = return the results in the JSON format that can be loaded into SIMILE widgets directly
    - *similep* = same as *simile*, but now padded with a callback, see also *jsonp*.
    - kml-abc = is a KML flavour that can be used to load the result onto 'erfgoed op de kaart' projects
- **callback** = to be used with the padded formats of *json* and *simile*. Here you can specify your own callback string for integration into your system.
- **fl** = a comma separated list with the metadata fields you want to have returned in the response. You can find the valid fields in the *explain response*.
- **cache** = accepts boolean values to use the Culture-Hub cache for thumbnails. Currently, the following fields are supported: , , . It works by prepending the Culture-Hub cache url to the thumbnails url and url-encodes it. The default value is **false** and it renders the thumbnail URLs without any modifications.
- **sortBy** = the valid sort field you want to sort the records on. For valid sort field see the 'explain response'. The default sort order is by relevance as returned by the search engine.
- **sortOrder** = It has the following enumerated options:
  - *asc* = sort records ascending
  - *desc* = sort records descending
- **group.field** = (planned for API version 0.17)

examples

- basic query = http://kulturnett2.delving.org:80/api/search?query=Gruppe,%20Vik%201920-1925
- fielded query = http://kulturnett2.delving.org:80/api/search?query=dc_title_text:Gruppe,%20Vik%201920-1925

2.1.3 Facets parameters

Depending on the configuration in the `production.conf` a number of facets is returned with each *summary view*. With the following parameters this behavior can be be changed.

- **qf** = the basic query filter. It expects a valid *facet field* - see explain response - with its value separated with a ':', for example `europeana_dataProvider_facet:Fylkesarkivet%20i%20Sogn%20og%20Fjordane`. This field can be repeated to add more query filters
- **hqf** = the same functionality as the *qf* but now it is treated as a hidden constraint. So the filter is applied to treat the output as if no records outside the filtered results exist in the index. This is useful for dynamically creating custom APIs.

- **facet.field** = for adding additional facets to the output that are not specified in the `production.conf`.
- **facet.limit** = the number of facet links returned per facet. The default is **100**
- **facet.boolType** = is the boolean type that specifies if multiple facet links are selected are treated as 'OR' or 'AND'. The enumerated options are:
  - *OR* = is the default. Matched records that have either of the filter queries specified.
  - *AND* matches only records that match all the filter queries.

Examples

- facet.field = http://kulturnett2.delving.org:80/api/search?query=Gruppe,%20Vik%201920-1925&facet.field=europeana_dataProvider_facet
- fq = http://kulturnett2.delving.org:80/api/search?query=Gruppe,%20Vik%201920-1925&facet.field=europeana_dataProvider_facet&qf[]=europeana_dataProvider_facet:Fylkesarkivet%20i%20Sogn%20og%20Fj
- breadcrumbs = http://kulturnett2.delving.org:80/api/search?query=Gruppe,%20Vik%201920-1925&facet.field=europeana_dataProvider_facet&qf[]=europeana_dataProvider_facet:Fylkesarkivet+i+Sogn+og+Fjordane&qf[]=
- hidden query filters = http://kulturnett2.delving.org:80/api/search?query=Gruppe,%20Vik%201920-1925&facet.field=europeana_dataProvider_facet&hqf[]=europeana_dataProvider_facet:Fylkesarkivet+i+Sogn+og+Fjordane&qf[

2.1.4 GeoSpatial search parameters

GeoSpatial search is implemented using the Solr SOLR2155 extension, see also https://wiki.apache.org/solr/SpatialSearch.

- **pt** = the center point of the query. It expects a lat,long pair separated by a comma
- **d** = an integer specifying the distance in kilometers from the centre point.
- **sfield** = the field you want to perform the geospatial search on. It can be on any field that has the `_geohash` field extension/type. The default field that is being used is field
- **geoType** = is the type of geoSpatial search that you want to perform. The enumerated options are:
  - **geofilt** = (is the **default** type when nothing is specified). Is the distance filter function from the center point.
  - **bbox** = creates a bounding box query of the size specified in `d` from the center point specified in `pt`

2.1.5 Description API response components

The output of the *summary view* request is structured as follows:

```
<?xml version='1.0' encoding='utf-8' ?>
<results
xmlns:abm="http://to_be_decided/abm/" xmlns:itin="http://www.itin.nl/namespace" xmlns:drup="http://ww
     <query numFound="13154">
          <terms>{queryTerm}</terms>
          <breadCrumbs>
               <breadcrumb value="{breadcrump value}" field="{medataField used}" href="{the query par
          </breadCrumbs>
     </query>
     <pagination>
          <start>{start record}</start>
          <rows>{number of records returned}</rows>
          <numFound>{total number of records found}</numFound>
```

```
          <nextPage>{next page if it has a next page}</nextPage>
          <lastPage>{last page nr}</lastPage>
          <currentPage>{current page nr}</currentPage>
          <links>
               <link isLinked="{}" start="{page nr}"> {page nr} </link>
               {... up to 10 links ...}
          </links>
     </pagination>
     <layout>
          <fields>
               <field>
                    <name>{metadata field name}</name>
                    <i18n>{translated field name based on the lang specfied}</i18n>
               </field>
         </fields>
     </layout>
     <items>
          <item>
               <fields>
               {... metadata fields ... }
              </fields>
               <highlights>
                {... highlighted fields ...}
               </highlights>
          </item>
          {... more items ..}
     </items>
     <facets>
          <facet isSelected="{boolean if the facet is selected with a qf}" name="{name of the facet}"
               <link isSelected="{boolean if link is selected}" url="{query parameters to be appended
          {... more facet links}
          </facet>

          {... more facets ...}
     </facets>
</results>
```

- **result** = The surrounding wrapper of the whole API response, i.e. the root of the response
  - `@numFound` = is the total number of records found
- **query** = the query block. It return the query terms and breadcrumbs. This information is used to render the user query and to provide a bread-crumb trail with the facets clicked.
  - **terms** = returns the raw query string as entered by the user.
  - **breadcrumbs** = contains a list of all the breadcrumbs based on the user query and facets in the order they were selected
    - **breadcrumb** = is the entry with the user readable query. In the attributes in contains elements that can be used for a variety of display purposes
      - `@value` = the value that was search for
      - `@field` = the field that was searched in. This is empty for the user query.
      - `@href` = the URL parameters that need to be appended to the base URL to get back to this point in the breadcrumb trail
      - `@i18n` = the translation of the `@field` as defined by the language specified in `lang` parameter
- **pagination** = is the wrapper of all elements that are needed to build pagination for the search results

  - **start** = the number of the first record on the current page.
  - **rows** = the number of records returned per page. The default number is 20, but this can be overridden by using the `rows` parameter in the request
  - **numFound** = is the total number of records found
  - **currentPage** = the page number of the current page
  - **nextPage** = the page number of the next page, if the current page is not the last page. In that case this element is not displayed.
  - **previousPage** = the pageNumber of the previous page if the current page is not the first page. In that case this element is not displayed.
  - **links** = the links can be used to build the link navigation for a result pager. When the selected page is more than 4 links removed from the start page, the selected page link will be centered among the linked pages.
    * **link** = each link represents a page.
      · `@islinked` = is a boolean to determine which page your are on. *true* for this page, *false* for other page.
      · `@start` = contains an int for the `start` parameter if you want to jump to this page.

- **layout** = the layout block that can be used to localize the metadata fields based on the language specified in the `lang` parameter
  - **fields** = list of fields with i18n translations
    * **field** = the wrapper for the field values * **name** = the name of the metadata field as it is used in the API response, but then with the ':' separator replaced with an '_'. * **i18n** = the translated value of the metadata field specified in *name*

- **items** = list of metadata records returned
  - **item** = wrapper of the actual metadata record
    * **fields** = wrapper of the metadata fields as they are stored in the Search Engine
    * **highlights** = contains a list of highlighted fields that contain a match for the query. This is useful when the records also contain large blocks of text, such as from text-extraction of PDFs. The highlighted fields can be configured in the `production.conf`

- facets = wrapper of all facets that are returned in the response
  - facet = contains a list of all facets until the `face.limit` for this facet field. By default this list is reverse sorted by frequency.
    * `@isSelected` = contains a boolean that describes if any of facet links are selected by the user. This can be used to expand or collapse the facet display.
    * `@name` = the metadata field for this facet that is used
    * `@missingDocs` = the number of records that don't have this metadata field with a value
    * `@i18n` = the translation of the `@name` into the language specified by `@lang`
    * link = has all the information
      · `@isSelected` = contains a boolean that describes if this facet link is selected by the user
      · `@url` = contains the parameters that need to be attached to the URL in order to select this facet
      · `@count` = the frequency of the number of records this value found in as string in field `facet/@name`

- `@value` = the string value the `@count` refers to.

2.2 Full View Mode

The Full View mode is activated by passing a valid identifier to the **id** parameter on the search API base-URL, see

```
`http://{baseUrl}:{portNumber}/api/search?id={id}`
```

The API responses from *summary view* are retrieved directly from the search engine. The *full view* however retrieves the mapped version from the metadata storage. By default you will get the same schema that is used for indexing. In the output of the *summary view*, you have the `delving:allSchemas` and `delving:currentSchema` fields. The `allSchemas` field contains all the mapped and publicly available fields. Via the `schema` parameter in the api call you can specify which of the publicly available schemas you want to have returned.

The *full view* mode accepts the following parameters:

- **id** = the identifier of the record you wish to retrieve.
- **idType** = the type of identifier you wish to retrieve. It has the following enumerated options:
  - *hubId* = is the default and is retrieved from field
  - *legacy* = is the record identifier used by the legacy portal system and is retrieved from the field
  - *pmhId* = is identifier used in the OAI-PMH output to identify records and is retrieved from the field
- **format** = the response format you want to have your API request returned in. The enumerated options are:
  - *json* = JSON output
  - *xml* = XML output (**default**)
- **lang** = the language into which the layout field blocks will be translated. It accepts two letter ISO language codes, like for example 'en', 'no, 'nl'
- **schema** = the metadata schema you want to have your record returned in. The default schema is the same that was used for indexing.
- **mlt** = is a boolean operator that triggers the 'more-like-this' functionality that is configured in the `production.conf` file. The enumerated options are:
  - *true*
  - *false* (**default**) You can configure the following options in the configuration file for the `mlt` functionality. For more information on them, see https://wiki.apache.org/solr/MoreLikeThis.
  - *fieldList* = list of fields to be returned. Can be taken from the search attributes in the explain response. Default: delving_creator, delving_title, delving_description
  - *minimumTermFrequency* = integer, default: 1
  - *minimumDocumentFrequency* = integer, default: 2
  - *minWordLength* = integer, default: 0
  - *maxWordLength* = integer, default: 0
  - *maxQueryTerms* = integer, default: 25
  - *maxNumToken* = integer, default: 5000
  - *boost* = boolean, default: false
  - *queryFields* = list of query fields, see also fieldList.

The output of the *full view* request is structured as follows:

```
<result xmlns:abc="http://www.ab-c.nl/" xmlns:delving="http://www.delving.eu/schemas/" xmlns:tib="htt
    <layout>
        <fields>
            <field>
                <name>abm_municipality</name>
                <i18n>Municipality</i18n>
            </field>
            {.. more fields ...}
        </fields>
    </layout>
    <item>
        <fields>
            <dc:creator>Fosse, Ole Pedersen</dc:creator>
            <dc:title>Gruppe, Hang 1920-1925</dc:title>
            {.. more fields ...}
        </fields>
    </item>
    <relatedItems>
        <item>
            <fields>
                <dc:title>Gruppe, Hang 1920-1925</dc:title>
                {.. more fields ...}
            </fields>
        </item>
        {.. more items ...}
    </relatedItems>
</result>
```

- **result** = the surrounding wrapper
  - **layout** = the layout block that can be used to translate the metadata fields based on the `@lang` specified. The default *lang* is **en**.
    - **fields** = the list of fields
      - **field** = the wrapper for the field values
      - **name** = the name of the metadata field as it is used in the API response, but then with the ':' separator replaced with an '_'.
      - **i18n** = the translated value of the metadata field specified in *name*
  - **item** = The actual full view item that was requested via the `id` parameter
    - **fields** =
      - metadata fields as returned by the schema defined in the `schema` parameter
  - **relatedItems** = this is an optional block that is only displayed when the parameter `mlt=true` is specified. It returns a list of items
    - **item** = contains the metadata fields of the related item
      - **fields** =
      - metadata fields as returned by the schema defined in the `schema` parameter. They are basically the same as `/result/item/fields/`.

examples

- basic = http://kulturnett2.delving.org:80/api/search?id=kulturnett_Foto-SF_SFFf-1987001.0027

- related items = http://kulturnett2.delving.org:80/api/search?id=kulturnett_Foto-SF_SFFf-1987001.0027&mlt=true
- format http://kulturnett2.delving.org:80/api/search?id=kulturnett_Foto-SF_SFFf-1987001.0027&mlt=true&format=json

2.3 Explain Mode

The **explain** API's main function is to give an overview of the API options, the search fields, the facet fields, and the sort fields. The data is directly generated from the search index.

The **explain** mode has two main functions:

1. Explain the search API
    - light
    - all
2. Provide facet field based autocompletion for fields. This is mostly used to provide basic autocomplete functionality for advanced search fields.
    - fieldExplain

Since the output of both modes is very different, they will be explained in separate sub-sections.

2.3.1 Basic Explain

The functionality is requested by adding the `explain={light|all}` to the base search API url, see

```
`http://{baseUrl}:{portNumber}/api/search?explain={light|all}`
```

It supports the following additional parameter:

format = the response format you want to have your API request returned in. The enumerated options are: * *json* = JSON output * *xml* = XML output (**default**)

The output of the *fieldValue* request is structured as follows:

```
<results>
    <api>
        <parameters>
            <element>
                <label> query </label>
                <options>
                    <option> any string </option>
                    </options>
                <description> Will output a summary result set. Any valid Lucene or Solr Query syntax
            </element>
            <element>
                <label> format </label>
                <options>
                    <option> xml </option>
                    <option> json </option>
                    <option> jsonp </option>
                    <option> simile </option>
                    <option> similep </option>
                    <option> kml     </option>
```

```
                    <option> kml-abc </option>
                </options>
            </element>
        </parameters>
        <solr-dynamic>
            <fields>
                <field fieldType="text_general" docs="0" xml="dc:title" distinct="537693" search="dc_
                {... more fields ...}
            </fields>
            <facets>
                <facet fieldType="string" docs="0" xml="dc:date" distinct="131856" search="dc_date_fa
                {... more facets ...}
            </facets>
            <sort-fields>
                <sort-field fieldType="string" docs="0" xml="all:delving_hasDigitalObject" distinct="
                {... more sort fields ...}
            </sort-fields>
        </solr-dynamic>
    </api>
</results>
```

- **parameters** = contains a list of all the parameters (as `<element>`) the search API accepts. Each API parameter listed here in the API response is also listed above in the **Search API** section
  - **element** = the block describing API parameter
    - **label** = the actual label that should be used in the API
    - **options** = contains either a list or a description of the values the api parameter accepts
    - **description** = the optional description of the usage of the API parameter
- **solr-dynamic** = contains a list of all valid search/metadata fields that are present in the index. They are split up into three different types: *fields*, *facets*, *sort-fields*
  - **fields** =
    - **field** = contains a number of indicators describing the field in the xml attributes.
      - `@search` = contains the full field name as it in indexed with the field type suffix and how it should be used in search (for fielded searches) and how it should be used in the `fl` parameter to specify which fields must be returned in the *summary view* response. Currently, the following field types that are used as suffixes are supported: string, facet, location,int,single,text,date,link,s,lowercase,geohash. When a field does not contain any of these suffixes, it means that it is a system field that already configured with the correct type in the search engine `schema.xml` configuration file.
      - `@fieldtype` = the index field type. This type is appended as a suffix to the metadata fieldname at indexing time and stripped during rendering. The types that are rendered here are the field types as they are know to the search engine. This type is determined by the suffix you can see in the `@search` attribute.
      - `@xml` = contains the raw format of the field name as it was seen in mapping and how it will be rendered in the API output.
      - `@docs` = the number of documents/records in the index that contain this field.
      - `@distinct` = the number of distinct values that are indexed in this field
  - **facets** = contains a number of indicators describing the field in the xml attributes. Same attributes as fields. The facet field `@search` value can be used in `query` for fielded search, `qf` and `hqf` for filtering,

`facet.field` for listing additional facets outside the `production.conf` configuration, and as fields in the **Statistics API**.

  - **sort-fields** = contains a number of indicators describing the field in the xml attributes. Same attributes as fields. The sort field can be used in the `sortBy` parameter.

examples = http://kulturnett2.delving.org:80/api/search?explain=light

2.3.2 Field Explain

The field explain functionality was developed to drive the advanced search autocompletion for the Delving Drupal module. This module consumed the Culture-Hub search APIs represent slices of the total index for regional and institutional portals.

The functionality is requested by adding the `explain=fieldValue` to the base search API url, see

```
`http://{baseUrl}:{portNumber}/api/search?explain=fieldValue`
```

It supports the following parameters:

- **field** = the facetable field that you want to have autocompletion for
- **value** = the optional prefix that you want to constrain your results to. For example, when you give *M* will give back all values starting with M. When you give *mo* it will give the values starting with *mo*, etc. The default is nothing and then it uses reverse sort by frequency of occurrence.
- **format** = the response format you want to have your API request returned in. The enumerated options are:
  - *json* = JSON output
  - *xml* = XML output (**default**)
- **rows** = integer of the number of values you want to have returned in the response. Default is **10**

The output of the *fieldValue* request is structured as follows:

JSON

```
{
         results:
     [
          {
                  value: "Midtbyen",
                  count: 15036
              }
    ]
}
```

XML

```
<results>
    <item count="15036">Midtbyen</item>
</results>
```

- **results** = is the list of response returned
  - **item** = is the actual fieldValue response pair with
    - **value** = is the facet value returned
    - **count** = the number of occurrences in the full index

examples = http://kulturnett2.delving.org:80/api/search?explain=fieldValue&field=abm_namedPlace_facet&value=M&format=json&rc

CHAPTER 3

Statistics API

The statistics API is a JSON API that provides statistics on any facetable field. See the **explain** response for a list of facetable fields.

The API base URL for all statistics actions is:

```
`http://{baseUrl}:{portNumber}/api/statistics`
```

The statistics API accepts the following parameters:

- **facet.field** = a repeatable field with facetable metadata fields you want to have returned. Note that you must use the full search field as specified in the explain response.
- **facet.limit** = must contain an integer for the number of statistics entries for each field specified in *facet.field* parameter. The default value is **100**
- **filter** = provide any valid query to constrain the set for which statistis are being returned. For example, constrain the statistics per *region* or *material type*.
- **lang** = the language in which you want the *i18n* tags to be returned. The default value is **en**

The output of the **statistics API** is a list of statistics objects structured as follows:

```
{
    statistics: {
        totalRecords: 3304080,
        totalRecordsWithDigitalObjects: 2561466,
        totalRecordsWithLandingPages: 3262741,
        facetCounts: {
            icn_technique_facet: 100,
            icn_material_facet: 100
        },
        facets: [{
            name: "icn_technique_facet",
            i18n: "icn_technique_facet",
            entries: [{
                name: "zwart-wit foto",
                total: 18699,
                digitalObjects: 18258,
                digitalObjectsPercentage: 1,
                noDigitalObjects: 3285822,
                noDigitalObjectsPercentage: 99,
                landingPages: 18699,
                landingPagesPercentage: 1,
                nolandingPages: 3285381,
```

```
                nolandingPagesPercentage: 99
            },
            {... more entries ...}]
        }, {
            name: "icn_material_facet",
            i18n: "icn_material_facet",
            entries: [{
                name: "aardewerk",
                total: 27187,
                digitalObjects: 23093,
                digitalObjectsPercentage: 1,
                noDigitalObjects: 3280987,
                noDigitalObjectsPercentage: 99,
                landingPages: 27186,
                landingPagesPercentage: 1,
                nolandingPages: 3276894,
                nolandingPagesPercentage: 99
            },
            {... more entries ...}
            }]
        }
    }
```

- **statistics**
  - **totalRecords** = is the total number of records in the index
  - **totalRecordsWithDigitalObjects** = is the total number of records in the index with Digital objects. The definition of digital object is that it either has a link to the source object or a link to a thumbnail representing the object described in the metadata.
  - **totalRecordsWithLandingPages** = is the total number of records in the index with Digital objects. The definition of landingPage is the page at the dataProviders website where this object is described.
  - **facetCounts** = returns a map with the names of the statistics fields returned and how many entries are returned in the response
  - **facets**
    - **name** = is the name of the metadata field whose entries are listed. This field is specified in the *facet.field* parameter
    - **i18n** = if a translation of the metadata field is found this is returned based on the value of the *lang* parameter. If no translation is found the name of the field is returned
    - **entries** = is a map of statistics per unique value in the facet. This reverse sorted by the frequency in which it occurs in the index. The names of the keys should be self-explanatory.

examples = http://www.dimcon.nl:80/api/statistics?facet.field=icn_technique_facet&facet.field=icn_material_facet&facet.limit=1&lang

CHAPTER 4

Proxies API

The proxy API is an XML API that has been implemented as convenience to group together various remote resources with the same output wrapping as all the other Culture-Hub search APIs. The configuration of these proxies is done in the organization `production.conf`.

The API base URL for all proxy actions is:

```
`http://{baseUrl}:{portNumber}/api/proxy`
```

note: The **proxy** API is an XML only API.

4.1 List all proxies

This commands list all the proxies thats have been configured for this organization.

The **list** command is given via a REST command appended to the proxy base URL:

```
`http://{baseUrl}:{portNumber}/api/proxy/list`
```

The output of the **list** command is a list of all available proxies structured as follows:

```
<explain>
    <item>
        <id>europeana</id>
        <url>http://api.europeana.eu/api/opensearch.rss</url>
    </item>
</explain>
```

- `<id>` = is the identifier that can be used for the proxy search
- `<url>` = is the url that is used by the proxy

In the proxy configuration some hidden parameters like api-keys are already included.

examples = http://kulturnett2.delving.org:80/proxy/list

4.2 Search a specific proxy

The **search** command is given via a REST command appended to the proxy base URL:

```
`http://{baseUrl}:{portNumber}/api/proxy/{proxyId}/search`
```

The **proxyId** is the `<id>` from the output of the proxy **list** command.

The **search** command accepts the following url query parameters:

- **query** = any search query supported by the service that is proxied.
- **start** = any integer that is less than the total records return and starts at 1. Services which are zero based will be remapped to 1 based paging.

The output of the **search** command is a list of records structured as follows:

```
<results xmlns:europeana="http://www.europeana.eu" xmlns:atom="http://www.w3.org/2005/Atom" xmlns:ope
    <pagination>
        <numFound>1</numFound>
        <start>1</start>
        <rows>1</rows>
    </pagination>
    <items>
        <item>
            <id>{itemId}</id>
            <fields>
                {all metadata fields as returned by the proxied service}
            </fields>
        </item>
    </items>
</results>
```

- `<pagination>` = if the proxied service supports returning paging information the pagination block will be returned in the response
  - `<numFound>` = the total numbers of records found (int)
  - `<start>` = the start number of the first record of the returned page (int)
  - `<rows>` = to number records - i.e. items - returned on the page (int)
- `<item>` = This wraps each record returned by the proxied service
  - `<id>` = is the identifier that can be used to return the *full-view* in the **item** service, i.e. *itemId*. The **id** field is only shown if the proxied service supports the request of a single record with all metadata fields.
  - `<fields>` = has as its children each metadata field returned by the proxied service

examples = http://kulturnett2.delving.org:80/proxy/wikipedia.en/search?query=bard

4.3 Request full-view item from proxy

The **item** command is given via a REST command appended to the proxy base URL:

```
`http://{baseUrl}:{portNumber}/api/proxy/item/{itemId}`
```

The **itemId** can be any of the *ids* specified in the `/items/item/id/` path of the **search** response.

The **item** command has no query parameters.

The output of the **item** command is a verbose rendering of the return of the proxy service.

Contents

CHAPTER 5

Planned functionality for the API

5.1 access statistics API

Overview envisioned functionality:

- statistics
  - index fields and usages
    - which fields are indexed
    - which types
      - these are the dynamic type suffixes
      - based on the record definition
    - which can be used as facets
    - access to individual histograms
      - gathered in the Sip-Creator
  - access statistics
    - origin
      - unique users
      - return visitors
      - unique areas
      - reverse ip lookups
    - information accessed
      - per
      - municipality
      - county
      - country
      - language
      - provider
      - dataprovider

        - record type
        - in
        - search result page view
        - used as facet
        - objects viewed
        - nr of outgoing ‘ <>‘__links clicked
    * From
        - API consumer
        - instant website
        - Drupal module
        - other
        - which named-slice or API is used
        - Hub-Website
- quantitative indicators

5.2 Grouping / Clustering API

The grouping API is designed to group together the search results, based on the value of a field. You could, for example, group the search results of your query by country or language and then show under each header the first 5 results. This functionality is nice for home pages where you want to show the variety of the collection you have gathered by provider, dataProvider, etc.

CHAPTER 6

Indices and tables

- genindex
- modindex
- search