search

Description

The method `search` is providing an interface to the search engine of the DDB. This method provides response data only as `application/json`. It is a read-only service and must be accessed with a HTTP-GET-request. dd

```
GET /search?<QueryParameters> HTTP/1.1
```

The resulting JSON object is made up of several parts. Each part will be described separately. The described parts are properties of the search result object and can be joined to form a complete result object.

First the number of results is in the property `numberOfResults`. This gives the total number of found results independent of the rows returned.

numberOfResults Expand source

```
{
   "numberOfResults":1,
   "results":[ ... ],
}
```

Another part is the `hightlightedTerms`. This contains an array of all terms that should be highlighted. In the preview they are already highlighted. But this array can be transferred to the detail view where the terms could be highlighted as well.

highlightedTerms Expand source

```
{
   ...
   "highlightedTerms":[
      "Johann",
      "Wolfgang",
      "Goethe"
 ],
   ...
}
```

If the random sorting is activated this property will contain a value that can be used as subsequent sort-parameter. The value should be used as is. If the random sorting is deactivated or the query was not a get-all-objects-query (`*` or `*:*`) the property will be empty. See below for a set property:

randomSeed Expand source

```
{
   ...
   "randomSeed":"random_01234567",
   ...
}
```

If the query is mistyped, i.e. there are more hits for an alternative spelling of one of the query terms a corrected query will be returned with the search result. This property `correctedQuery` contains the entered query with all terms corrected. The corresponding JSON property is noted below:

correctedQuery › Expand source

```
{
   ...
   "correctedQuery":"Johann Wolfgang Goethe",
   ...
}
```

One lengthier part are the `facets` and their values. Each facet defined to have a `displayType` of `SEARCH` will always be returned with the search results. If no value is given for the query parameter facet the facets will look like this:

facets › Expand source

```
{
   ...
   "facets":[
      {
         "field":"affiliate_fct",
         "numberOfFacets":0,
         "facetValues":[

         ]
      },
      {
         "field":"type_fct",
         "numberOfFacets":0,
         "facetValues":[

         ]
      },
   ...
}
```

The property `field` will contain the name of the facet. The `numberOfFacets` is the number of distinct facet values found for this facet. If the `facet.limit` parameter is set in the query the number of facets will be equal to this number. The `facetValues` array will contain all found values and the number of documents having this facet if this facet is asked for. As in the example given below:

numberOfFacets

Expand source

```
...
{
  field:"place_fct",
  facetValues:[
    {
      value: "DE - L152",
      count: 81
    },
    {
      value: "University Library Felix Mendelssohn Bartholdy, Leipzig, Germany",
      count: 67
    }
  ],
  numberOfFacets:2
}
],
...
```

At last the search results matching the query are returned. The actual results are again made up of several parts. The results property contains an array of clusters. Each cluster is again a object containing a name and an array of documents related to this cluster. In the case of an unclustered search, there will only be one cluster named `single`. Each document contains its ID, a label with a short description of the item, a preview that contains the HTML snippet that should be used for the list view of the item. The link to the image in the thumbnail div can be a absolute link or a link to a binary on the backend. In this case it is relative to the backend host. The category property gives the type of the item. The two possible values are “Institution” and “Kultur”. This is a repetition of the data-type attribute in the first div of the preview. So it could be removed if no longer needed. At last the location as longitude and latitude of this item are given. For most items the values will be `null`. This value is used in the institution map to locate the institutions. The last property of a cluster is the `numberOfDocs` this cluster contains.

Request Header

Name	Value(s)	Required	Repeatable	Default value	Description
Accept	`application/json` `*/*`	yes	no	`application/json`	Specifies the format of the accepted data. Every request needs to be sent with a valid Accept Header defining the requested response format. Otherwise the response to the request will be a `406 - Not acceptable` status code. If the request accepts every kind of data (`Accept: */*`) JSON will be returned.
Authorization	`OAuth oauth_consumer_key="<API key>"`	yes	no	–	An API key is mandatory for data access. You may enclose it by either sending it as Query Parameter or (as mentioned here) in the Request Header.

`Host`	`api.deutsche-digitale-bibliothek.de`	yes	no	–	The host is mandatory.

Parameters

Query

Parameter	Value(s)	Required	Repeatable	Default value	Description
`facet`	`<facet name>`	no	yes	–	This is the name of a facet which will be taken into account. Only facets which are supplied via a query parameter will be included into the result.
`<facet_name>`	`<query string>`	no	yes	–	The value of a query parameter named like a valid, existing facet defines a search query on the given facet name. It allows to reduce the result by certain values of facets.
`facet.limit`	`<number>`	no	no	–	Limits the number of values of a facet to the given amount.
`minDocs`	`<number>`	no	no	–	The amount of documents a facet must exceed to be included in the result set.
`oauth_consumer_key`	`<API key>`	yes	no	–	An API key is mandatory for data access. You may enclose it by either sending it as Query Parameter or (as mentioned here) in the Request Header.
`offset`	`<number>`	no	no	–	This is the number of the first entry of the result.
`query`	`<query string>`	yes	no	`*:*`	Term(s) to be searched. Must be URL-encoded and compliant to the SOLR Query Syntax.
`rows`	`<number>`	no	no	–	This is the count of result entries to be shown in total.

`sort`	`<restricted values>`	no	no	–	Defines the sorting order, which can be one of the following values. It requires the query parameter, otherwise the sort is always random_<seed>. • `ALPHA_ASC` • `ALPHA_DESC` • `RANDOM_<seed>` • `RELEVANCE`

Authorization

This method needs an API key for authentication (Who are you?) and authorization (What you are authorized to do?). For more information please read the Authorization How-To.

This method is offered over HTTP and HTTP Secure.

Errors

Code	Text	Description
`404`	`Not Found`	Either the item does not exist or has no data.
`406`	`Not Acceptable`	The Request Header is not acceptable. Please see section Request Header.
`500`	`Server Error`	Something went terribly wrong. An error message will provide a meaningful description.

Samples

Request 1

Search for all data in the DDB which contain “Johann Wolfgang Goethe” using query authorization.

The search result is limited by 1000 entries for each search result. To get all entries it is necessary to use the Query Parameters `offset`, `rows` and `sort` (see Request 2).

```
GET /search?oauth_consumer_key=abcdefgh12345678&query=Johann+Wolfgang+Goethe HTTP/1.1
Host: api.deutsche-digitale-bibliothek.de
Accept: application/json
```

Request 2

Search over all information in the data sets which contain the term “Johann+Wolfgang+G*”, but give me all documents after the 21st most relevant results and only up to two results at all. Authorization over HTTP Request Header is used.

```
GET /search?query=Johann+Wolfgang+G*&offset=21&rows=2&sort=RELEVANCE HTTP/1.1
Host: api.deutsche-digitale-bibliothek.de
Accept: */*
Authorization: OAuth oauth_consumer_key="abcdefgh12345678"
```

Request 3

Search for all documents which contain the term “Romanorum et Germanorum” in any field and include the facet `place_fct` in the result retaining all values of the facets which are part of at least one documents or more. Authorization over HTTP Request Header is used.

```
GET /search?query=Romanorum+et+Germanorum&facet=place_fct&minDocs=1 HTTP/1.1
Host: api.deutsche-digitale-bibliothek.de
Accept: */*
Authorization: OAuth oauth_consumer_key="abcdefgh12345678"
```

Request 4

Search for all documents which contain the term “Romanorum et Germanorum” in any field and contribute to the facet `type_fct` with the mediatype value “Text”. The result contains the values of `type_fct`. Authorization over HTTP Request Header is used.

```
GET /search?query=Romanorum+et+Germanorum&facet=type_fct&type_fct=mediatype_003
HTTP/1.1
Host: api.deutsche-digitale-bibliothek.de
Accept: */*
Authorization: OAuth oauth_consumer_key="abcdefgh12345678"
```

Request 5

Search for all documents which are located in Berlin. The result will be sorted ascending by their label.

```
GET /search?query=Haus&facet=place_fct&place_fct=Berlin&sort=ALPHA_ASC HTTP/1.1
Host: api.deutsche-digitale-bibliothek.de
Accept: */*
Authorization: OAuth oauth_consumer_key="abcdefgh12345678"
```