

RemBench, a Digital Workbench for Rembrandt Research

– User guidelines

This document contains some guidelines for the use of RemBench. RemBench combines the content of four different databases behind one search interface:

- [RKDArtists and RKDimages](#), two databases maintained by the Netherlands Institute for Art History (RKD)
- [RemDoc](#), a collection of original documents related to Rembrandt van Rijn from the period between 1475 to circa 1750
- [RUQuest](#), a library system that provides access to full text articles, as well as the complete collection of (e-)books and journals from the Radboud University Library Catalogue

RemBench does not influence the content of these databases. For questions or comments about the information in the RKD databases, RemDoc or RUQuest, we refer to their own websites.

For any remaining questions about RemBench, you can contact Suzan Verberne, s.verberne@let.ru.nl

1. General introduction

For a short introduction to RemBench see this instructional video:

2. Features of RemBench

Free text search



The image shows a search interface with a light gray background. At the top left, there is a 'Search' label followed by a blue square icon and the word 'Fuzzy'. Below this is a white rectangular input field for text.

You can use this field for free text search. Note that:

- The information you are looking for might be in English or Dutch. If you don't get results in one language you can try the same query in the other language (if you speak Dutch).
- The words in the query are interpreted separately: The query 'saskia brothers' gives results about either Saskia, brothers, or both.
- If you want all results to be about brothers then change your query to 'saskia +brothers'.
- You can also use literal phrases: "two brothers"
- You can also use wildcards for flexible search: 'amsterd*'
- Select the option 'Fuzzy' if you also want to find results for spelling variants of your query. For example, the query 'tsjouderville' with the fuzzy option, finds results for 'Jouderville'
- You can either use the return key or click the Search button to search.

Filtering

You can filter your results for Date, Location, Author/artist name, Artist/author type and Content type.

- Click on 'More' to see more values for a filter type. In the 'More' screen you can sort the values by name or frequency, and you can sort in the values.


Location 

- The meaning of the filters is as follows:
 - Date: for works of art, the creation date (or date range); for artists: the years between birth and death; for primary and secondary sources: the publication date. Note that not for all records an exact date is known, so it is possible that results are found that are not strictly within the date range that you provided.
 - Location: for works of art and primary sources, the location (place) where the work/source physically resides; for artists: places that are mentioned in his record (birth place, place of death, places of activity); for secondary sources: place where it was published.
 - Author/artist name: for words of art, the name of the artist; for primary and secondary sources: the author. No results for artists.
 - Artist/author type: similarly as Author/artist name.
 - Content type: By selecting 'works of art', all works of art are found. By selecting 'publications', all secondary sources are found. In addition, all works of art have a subtype (painting, drawing, etc.); the same holds for primary sources all have a specific type (court records, notarial acts, etc.) and secondary sources (newspaper article, books/e-books etc.); no results for artists;

View results

After entering a query and/or selecting filter values, results are shown for the four different databases. In each set of results, you can click 'More' to browse through all retrieved results. You can click on a result to see the page with selected details about the results. Each detail page also contains the link ('Permalink') to the original location of the data in the RKD database, RemDoc or RUQuest.

Clear everything

Click on the refresh button  to clear all fields and start a new search



Technical Documentation RemBench

10 September 2014

1. Back End

JAVA

The RemBench backend was written in JAVA, and can roughly be divided in two parts: the Harvester and the Search Server.

Maven2 (<http://maven.apache.org/>) was used as build/dependency system.

Harvester

The harvester queries the four external databases (RemDoc, RuQuest, RKDArtists, RKDImages) periodically, converts the results to the RemBench format, and indexes these subsequently.

SearchServer

The index that was made by the Harvester can be accessed through the Search Server.

The Search Server was engineered as a REST Service:

http://en.wikipedia.org/wiki/Representational_state_transfer.

SOLR (<http://lucene.apache.org/solr/>) was used for the Full Text Search and the Faceted Search.

2. REST API

GET /application.wadl

Returns the (Jersey-generated) WADL document

(http://en.wikipedia.org/wiki/Web_Application_Description_Language) for the REST API

GET /details/{recordid}

Returns the details for the RemDoc record with the given recordid

POST /search

Executes a search, creates a searchresults object, returning the url of this searchresult in the Location: header

parameters (mediaType="application/json; charset=utf-8"):

term (string): text to fulltext search for

fuzzy (boolean): set to true if you want the search to be fuzzy: slight variations of the term will also be used in the search.

facetValues (array of name/values objects): the required values for selected facets (there is an OR relation between the values per facet, and an AND relation between the different facets) so for a search where facet "location" has values "Amsterdam" and "Rotterdam", and facet "artist_name" has values "Rembrandt" and "Bol, Ferdinand", the search will be for records where "location" is "Amsterdam" or "Rotterdam" and "artist_name" is "Rembrandt" or "Bol, Ferdinand"

example:

```
{
  "term":"saskia",
  "fuzzy":true,

  "facetValues":[
    {
      "name":"location",
      "values":["Amsterdam"]
    }
  ],
}
```

GET /search/{searchid}

Optional parameters:

- database - return only the results from this database (remdoc,ruquest,rkdimages,rkdartists)
- rows - return this amount of results
- start - start at this result

Returns a JSON map with:

- **numFound** : total number of records found
- **facets** : the facet values and counts for the RemBench facet fields
- **results** : the search result info, grouped per database

GET /version

Returns a JSON representation of some version information for the backend webapp.

3. Front End

Preprocessors

RemBench was written using preprocessors. Instead of writing direct HTML, CSS and JavaScript we used:

- Jade (<http://jade-lang.com>),
- Stylus (<http://learnboost.github.io/stylus>),
- CoffeeScript (<http://coffeescript.org>).

Gulp

In order to avoid repetitive acts, a task manager was used. In the case of RemBench Gulp (<http://gulpjs.com>) was chosen. Gulp ensures that preprocessing is automated and that a local Web server is started.

Browserify

Using Browserify (<http://browserify.org>) has two advantages in the development of the Front End:

1. We can use Back End dependencies (Node);
2. All code is bundled.

Backbone

RemBench is a so-called single-page application. That means that only one page is loaded and that from that one page the whole app(lication) is available. For the structure of the app the Backbone framework (<http://backbonejs.org>) is used.

Faceted Search

The most important library that is used in RemBench is the Faceted Search that was developed by Huygens ING. (<https://github.com/HuygensING/faceted-search>). The Faceted Search enables the user to filter the search results.