SPARQList: Markdown-based highly configurable REST API hosting server for SPARQL

Toshiaki Katayama[1] , Shuichi Kawashima[1]

[1] Database Center for Life Science, Research Organization of Information and Systems,
178-4-4 Wakashiba, Kashiwa-shi, Chiba 277-0871, Japan
ktym@dbcls.jp, kwsm@dbcls.jp

Abstract. SPARQList is a REST API server which executes a SPARQL query, transform the result into formatted data if defined, and then send it back to web client application. In SPARQList, the configuration of API is written in the Markdown format in which parameters of the API, SPARQL endpoints and SPARQL queries, and JavaScript functions for data transformation are defined along with a free text documentation. Each SPARQList server instance can host multiple API configurations, therefore, each service can also be considered as a repository of reusable SPARQL queries with documentation. The source code of the SPARQList is freely available at https://github.com/dbcls/sparqlist and can be easily deployed as it is implemented in the Node.js.

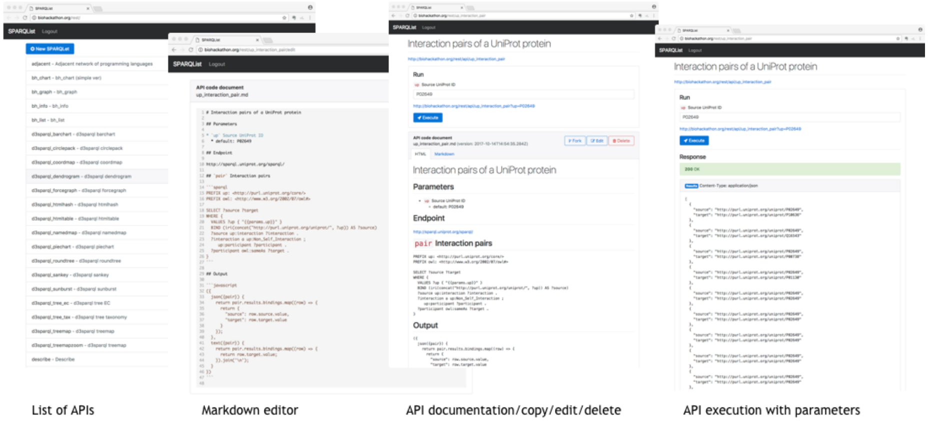
Keywords: Semantic Web, SPARQL, JavaScript, Markdown, REST API

Introduction

SPARQL endpoint allows users to make arbitrary queries, however, it is also demanded to provide REST APIs for frequently used typical queries or complex queries to be executed instantly so that web applications can easily make use of the semantic data. SPARQList is designed to provide such REST APIs with ease by (1) introducing the Markdown format for describing the API, (2) provide a GUI Web interface to create/copy/edit/delete/execute the API configuration, (3) transform the SPARQL results into other format according to content negotiation by JavaScript.

Results

We developed a REST API server namely SPARQList which has a Web interface to list, create, copy, edit, delete and execute “SPARQLet”, a definition of SPARQL queries and data transformations described in a Markdown text. **Fig 1** shows the Web interface of SPARQList which provides (1) a list of SPARQLets, (2) a Markdown editor for API configuration, (3) a formatted documentation and buttons to copy, edit, delete the SPARQLet, (4) an interface to specify parameters and to execute the SPARQLet which comes with a trace for debugging.



List of APIs

- adjacent - Adjacent network of programming languages
- bh_chart - bh_chart (simple ver)
- bh_graph - bh_graph
- bh_info - bh_info
- bh_list - bh_list
- d3sparql_barchart - d3sparql barchart
- d3sparql_circlepack - d3sparql circlepack
- d3sparql_coordmap - d3sparql coordmap
- d3sparql_dendrogram - d3sparql dendrogram
- d3sparql_forcegraph - d3sparql forcegraph
- d3sparql_htmlhash - d3sparql htmlhash
- d3sparql_htmltable - d3sparql htmltable
- d3sparql_namedmap - d3sparql namedmap
- d3sparql_piechart - d3sparql piechart
- d3sparql_roundtree - d3sparql roundtree
- d3sparql_sankey - d3sparql sankey
- d3sparql_sunburst - d3sparql sunburst
- d3sparql_tree_ec - d3sparql tree EC
- d3sparql_tree_tax - d3sparql tree taxonomy
- d3sparql_treemap - d3sparql treemap
- d3sparql_treemapzoom - d3sparql treemap
- describe - Describe

Markdown editor

API code document
up_interaction_pair.md

~~~
# Interaction pairs of a UniProt protein

## Parameters

* `up` Source UniProt ID
  * default: P02649

## Endpoint

http://sparql.uniprot.org/sparql/

## `pair` Interaction pairs

```sparql
PREFIX up: <http://purl.uniprot.org/core/>
PREFIX owl: <http://www.w3.org/2002/07/owl#>

SELECT ?source ?target
WHERE {
  VALUES ?up { "{{params.up}}" }
  BIND (iri(concat("http://purl.uniprot.org/uniprot/", ?up)) AS ?source)
  ?source up:interaction ?interaction .
  ?interaction a up:Non_Self_Interaction ;
     up:participant ?participant .
  ?participant owl:sameAs ?target .
}
```


## Output

```javascript
({
  json({pair}) {
    return pair.results.bindings.map((row) => {
      return {
        "source": row.source.value,
        "target": row.target.value
      }
    });
  },
  text({pair}) {
    return pair.results.bindings.map((row) => {
      return row.target.value;
    }).join("\n");
  }
})
```

~~~

API documentation/copy/edit/delete

Interaction pairs of a UniProt protein

http://biohackathon.org/rest/api/up_interaction_pair

Run

up Source UniProt ID

P02649

http://biohackathon.org/rest/api/up_interaction_pair?up=P02649

API code document
up_interaction_pair.md (version: 2017-10-14T14:54:35.284Z)

Interaction pairs of a UniProt protein

Parameters

- up Source UniProt ID
  - default: P02649

Endpoint

http://sparql.uniprot.org/sparql/

pair Interaction pairs

Output

API execution with parameters

Interaction pairs of a UniProt protein

http://biohackathon.org/rest/api/up_interaction_pair

up Source UniProt ID

P02649

http://biohackathon.org/rest/api/up_interaction_pair?up=P02649

200 OK

Results Content-Type: application/json

```
[
  {
    "source": "http://purl.uniprot.org/uniprot/P02649",
    "target": "http://purl.uniprot.org/uniprot/P10636"
  },
  {
    "source": "http://purl.uniprot.org/uniprot/P02649",
    "target": "http://purl.uniprot.org/uniprot/Q16543"
  },
  {
    "source": "http://purl.uniprot.org/uniprot/P02649",
    "target": "http://purl.uniprot.org/uniprot/P00738"
  },
  {
    "source": "http://purl.uniprot.org/uniprot/P02649",
    "target": "http://purl.uniprot.org/uniprot/P01130"
  },
  {
    "source": "http://purl.uniprot.org/uniprot/P02649",
    "target": "http://purl.uniprot.org/uniprot/P02649"
  },
  {
    "source": "http://purl.uniprot.org/uniprot/P02649",
    "target": "http://purl.uniprot.org/uniprot/P02649"
  },
  {
    "source": "http://purl.uniprot.org/uniprot/P02649",
    "target": "http://purl.uniprot.org/uniprot/P02649"
  },
  {
    "source": "http://purl.uniprot.org/uniprot/P02649",
    "target": "http://purl.uniprot.org/uniprot/P02649"
  },
  {
    "source": "http://purl.uniprot.org/uniprot/P02649",
    "target": "http://purl.uniprot.org/uniprot/P02649"
  },
```

Fig 1. Web interface of the SPARQList

In the SPARQLet configuration (formatted in the Markdown), a service provider can specify a title and a description of the API, arbitrary pairs of a SPARQL endpoint and a SPARQL query template which can embed API parameters, and JavaScirpt snippets for the purpose described below.

Even though most SPARQL endpoints can return the results in the SPARQL 1.1 Query Results JSON Format (application/sparql-results+json), it is often preferable for the Web client applications to consume formatted JSON (application/json) data or other formats like a text or HTML. Therefore, a SPARQLet can store the SPARQL results as variables and the JavaScript snippets embedded in the SPARQLet can make use of them in combination to transform into any other format according to the specified mime-type for content negotiation.

```
% curl –H “Accept: application/json” http://ex.org/api/test?p=foo
% curl –H “Accept: text/plain” http://ex.org/api/test?p=foo
```

To make SPARQList APIs compatible with the smartAPI (http://smart-api.info/) and promoting the FAIR principles [1], supporting JSON-LD (application/ld+json) is planned. Note that, our separate open source product SPARQL-proxy, a cache server for any SPARQL endpoint, can be used in combination to improve the response time of repeated queries.

References

1. Wilkinson M.D. *et al*.: The FAIR Guiding Principles for scientific data management and stewardship. Sci. Data. 3, 160018 (2016).