

Creating a semantic portal easily using Sampo-UI

Annastiina Ahola, Heikki Rantala

Sampo-UI framework

- **Framework for developing user interfaces for semantic portals**
 - multiple application perspectives to the data
 - two-step usage cycle: filter & analyze
 - has been used both in the Sampo series of portals as well as external projects
- **Available on GitHub under the MIT license:**
<https://github.com/SemanticComputing/sampo-ui>

Philosophy

- **Faceted searching done with SPARQL**
 - Sampo-UI doesn't require your triple store to implement a certain kind of text indexing to work
 - *Text indexing can be utilized for efficient text-based search*
- **Configuration primarily done through JSON configuration and SPARQL query files without having to touch JavaScript**

Sampo-UI “standard” UI model

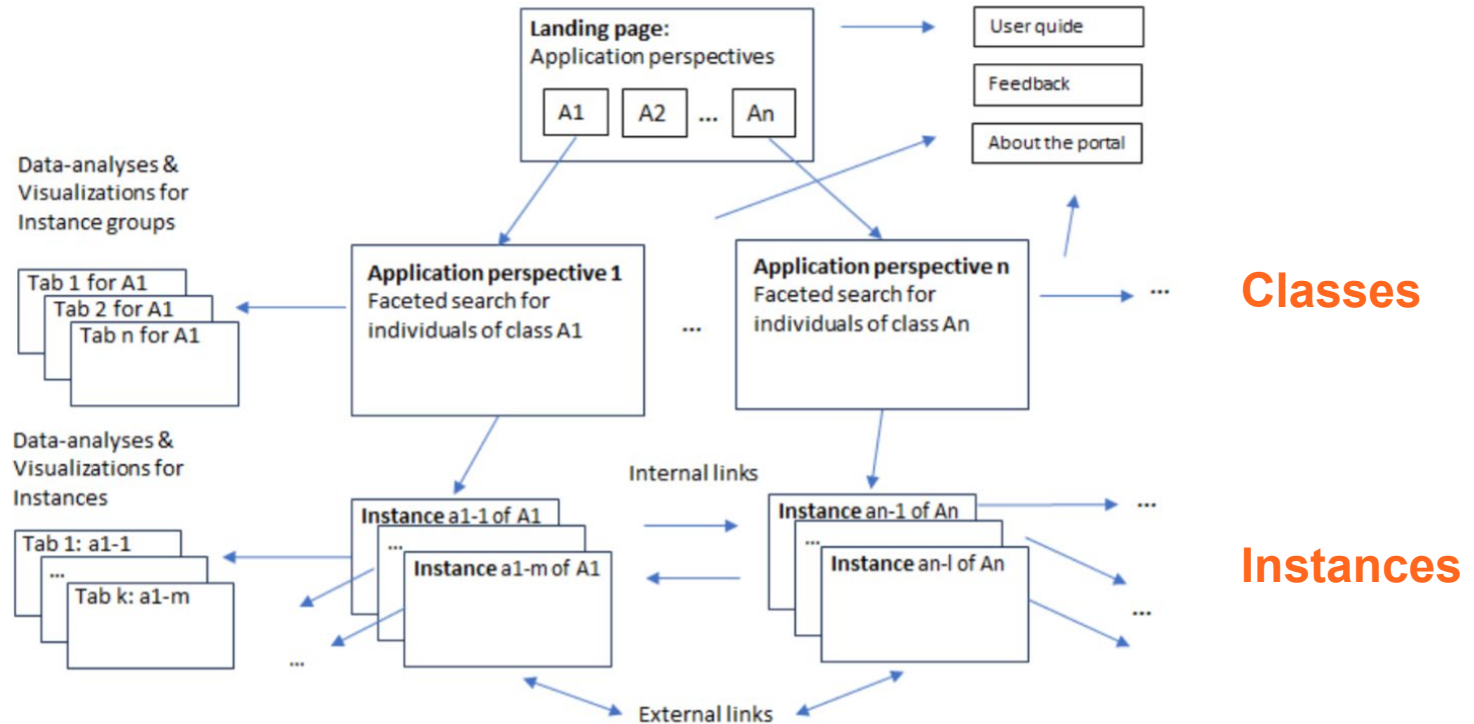


Figure 1: Navigational page structure of a portal based on Sampo-UI.

Sampo-UI framework: Structure

The screenshot displays the DHNB Sampo-UI Example interface. At the top, there is a dark navigation bar with the DHNB logo, a search bar labeled "Search all content", and a menu with links: PERFORMANCES, PEOPLE, COMPOSITIONS, ROLE CHARACTERS, PERFORMANCE VENUES, FEEDBACK, INFO, and INSTRUCTIONS. The main header area features a background image of old books with the title "DHNB Sampo-UI Example" and a quote: "Here to forge for us the Sampo, Hammer us the lid in colors". Below this, a prompt reads "Select a perspective to search and browse the knowledge graph:". Five buttons are arranged in a grid, each with a title, a description, and a representative image: "Performances" (opera house), "People" (woman's portrait), "Compositions" (musical score), "Role characters" (stage scene), and "Performance venues" (theater building). The footer contains logos for Aalto University School of Science, the University of Helsinki, and HELDIG.

DHNB Sampo-UI Example

Search all content

PERFORMANCES PEOPLE COMPOSITIONS ROLE CHARACTERS PERFORMANCE VENUES | FEEDBACK INFO INSTRUCTIONS

DHNB Sampo-UI Example

"Here to forge for us the Sampo, Hammer us the lid in colors"

Select a perspective to search and browse the knowledge graph:

Performances
Perspective for searching for performances.

People
Perspective for searching for people related to opera (composers, performers, etc.).

Compositions
Perspective for searching for compositions.

Role characters
Perspective for searching for role characters from compositions.

Performance venues
Perspective for searching for performance venues.

Aalto University School of Science

UNIVERSITY OF HELSINKI

HELDIG
Helsinki Centre for Digital Humanities

Multiple application perspectives to the data

Sampo-UI framework: Structure

Visualization tabs for visualizing the data

Facet menu for filtering the data

Results

DHNB Sampo-UI Example

Search all content

PERFORMANCES PEOPLE COMPOSITIONS ROLE CHARACTERS PERFORMANCE VENUES

PERFORMANCES ⓘ

Results: 37 performance(s)

Narrow down by:

Name ⓘ

Composition ⓘ

Composer ⓘ

Librettist ⓘ

Producer ⓘ

Performance venue ⓘ

Performers ⓘ

Director ⓘ

Conductor ⓘ

TABLE

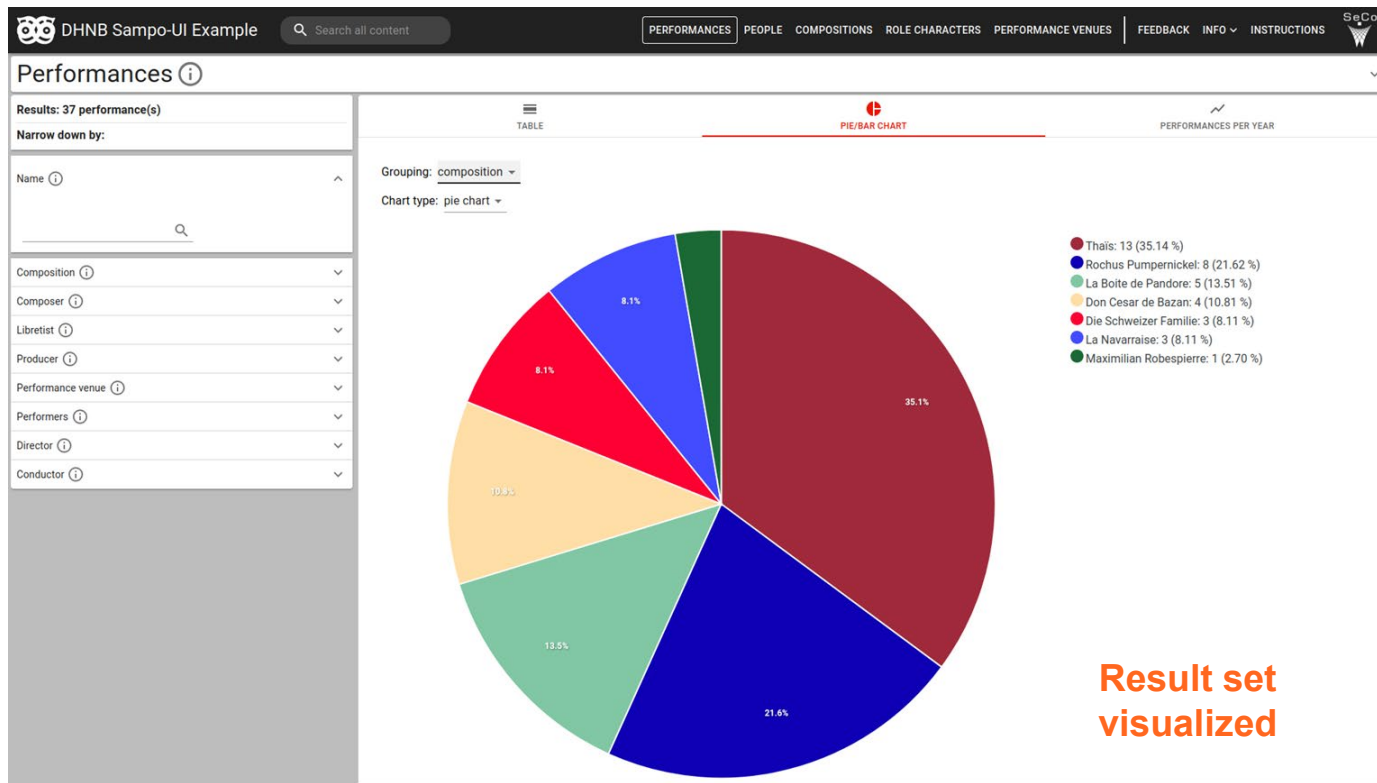
Rows per page 15 1-15 of 37

PIE/BAR CHART

PERFORMANCES PER YEAR

Name ⓘ	Composition ⓘ	Composer ⓘ	Librettist ⓘ	Producer ⓘ	Time of performance ⓘ
Die Schweizer-Familie (1833-12-20)	Die Schweizer Familie	Weigl, Joseph	Castelli, Ignaz Franz	Deutsches Schauspiel-Gesellschaft	1833-12-20
Thaïs (1910-04-13)	Thaïs	Massenet, Jules	Gallet, Louis	Aino Ackté & Edvard Fazer	1910-04-13
Thaïs (1910-04-15)	Thaïs	Massenet, Jules	Gallet, Louis	Aino Ackté & Edvard Fazer	1910-04-15
Thaïs (1910-04-16)	Thaïs	Massenet, Jules	Gallet, Louis	Aino Ackté & Edvard Fazer	1910-04-16
Thaïs (1910-04-18)	Thaïs	Massenet, Jules	Gallet, Louis	Aino Ackté & Edvard Fazer	1910-04-18
Thaïs (1910-04-20)	Thaïs	Massenet, Jules	Gallet, Louis	Aino Ackté & Edvard Fazer	1910-04-20
Thaïs (1910-04-21)	Thaïs	Massenet, Jules	Gallet, Louis	Aino Ackté & Edvard Fazer	1910-04-21
Thaïs (1919-11-03)	Thaïs	Massenet, Jules	Gallet, Louis	Aino Ackté	1919-11-03
Thaïs (1919-11-05)	Thaïs	Massenet, Jules	Gallet, Louis	Aino Ackté	1919-11-05
Thaïs (1919-11-07)	Thaïs	Massenet, Jules	Gallet, Louis	Aino Ackté	1919-11-07
Thaïs (1919-11-10)	Thaïs	Massenet, Jules	Gallet, Louis	Aino Ackté	1919-11-10
Thaïs (1919-11-12)	Thaïs	Massenet, Jules	Gallet, Louis	Aino Ackté	1919-11-12
Kuningas ja katulaulajat (1886-10-22)	Don Cesar de Bazan	Massenet, Jules	Dumanoir / Pinel, Philippe Francois ...	Suomalainen Teatteri	1886-10-22
Kuningas ja katulaulajat (1886-02-18)	Don Cesar de Bazan	Massenet, Jules	Dumanoir / Pinel, Philippe Francois ...	Suomalainen Teatteri	1886-02-18
Rochus Pumpnickel (1868-12-30)	Rochus Pumpnickel	Seyfried, Ignaz	Steinmayer, Matthäus	-	1868-12-30

Sampo-UI framework: Structure



Written & video tutorial

- **Tutorial showing the steps of setting up Sampo-UI using DBpedia's endpoint/data**
 - In video format:
<https://vimeo.com/817028609>
 - Written tutorial:
<https://seco.cs.aalto.fi/tools/sampo-ui/Sampo-UI-tutorial.pdf>
 - Code from the tutorial available on GitHub:
<https://github.com/SemanticComputing/dbpedia-sampo-ui-demo>

Tutorial Sampo-UI setup

- **Sampo-UI setup for running it with the data from “Creating LOD from databases” (subset of OperaSampo data)**
 - 5 search perspectives: performances, people, compositions, role characters and performance venues
 - 1 perspective with only instance pages: producers
- **Data and scripts as well as a portal setup available on GitHub:**
<https://github.com/SemanticComputing/sampo-lod-tutorial>
 - Sampo-UI setup in the ‘set-up-sampo-ui’ folder:
<https://github.com/SemanticComputing/sampo-lod-tutorial/tree/main/set-up-sampo-ui>

Requirements

- **Node.js (<https://nodejs.org/en/>) version 16.13.0**
 - Recommended: Node Version Manager (nvm) for managing different versions of Node.js installed on your computer
- **Nodemon (<https://nodemon.io/>)**
- **Optional: Docker**
 - You can also run Sampo-UI inside a Docker container on your computer, but this is not required for development

Getting started

- Fork or download source code from GitHub

The screenshot shows the GitHub repository page for `SemanticComputing/sampo-ui`. The repository is public and has 11 branches and 2 tags. The commit history shows a recent commit by `annasahola` 5 months ago. The repository details on the right show the repository description, license (MIT), and release information. Two orange arrows point to the `Fork` button in the top right and the `Code` button in the center.

Product Solutions Resources Open Source Enterprise Pricing

Search or jump to... Sign in Sign up

SemanticComputing/sampo-ui Public

Notifications Fork 8 Star 33

<> Code Issues 13 Pull requests 8 Actions Projects Wiki Security Insights

master 11 Branches 2 Tags

Go to file

Code

annasahola Fix top bar info items with external links on smaller screens 56a17e7 · 5 months ago 1,865 Commits

src	Fix top bar info items with external links on smaller screens	5 months ago
.dockerignore	Dockerize	6 years ago
.editorconfig	Use redux-observable	6 years ago
.gitignore	Add ds_Store to gitignore	last year
Dockerfile	Add new build arg	3 years ago
LICENSE	Update configs	3 years ago
README.md	Small addition to readme	last year
babel.config.js	@babel/preset-env: target node 16	3 years ago
package-lock.json	Merge branch 'dependabot/npm_and_yarn/qs-and-express-...	last year
package.json	Merge branch 'dependabot/npm_and_yarn/qs-and-express-...	last year
webpack.client.common.js	Add Fontsource Roboto node module	last year
webpack.client.dev.js	Upgrade webpack-dev-server and webpack-cli	3 years ago
webpack.client.prod.js	Gzip main js file and serve it with express-static-gzip	3 years ago

README MIT license

code style standard

Sampo-UI

A framework for building user interfaces for semantic portals.

About

Sampo-UI -- A framework for building user interfaces for semantic portals

seco.cs.salt.pools/sampo-ui

Readme MIT license Activity Custom properties 33 stars 12 watching 8 forks Report repository

Releases 2

v2.0.0 (Latest) on May 12, 2020 + 1 release

Packages

No packages published

Contributors 8

Languages

JavaScript 99.7% Other 0.3%

Configuration

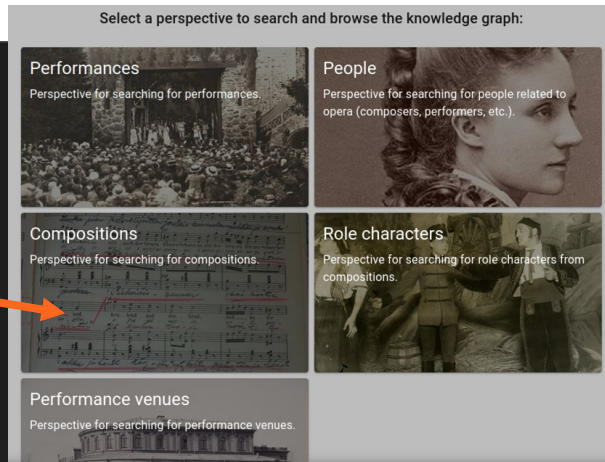
- **Configuration done through JSON configuration files**
 - a file for portal-wide configuration
 - one file for each configured perspective
- **Queries with SPARQL**
 - one file for each perspective with the perspective-specific queries for getting all wanted properties
 - many general queries already exist (e.g., getting facet values), you just need to pass the correct predicates in configuration

Portal configuration

```
{
  "portalID": "sampo",
  "rootUrl": "",
  "perspectives": {
    "searchPerspectives": [
      "performances",
      "people",
      "compositions",
      "roles",
      "places"
    ],
    "onlyInstancePages": [
      "producers"
    ]
  },
  "localeConfig": {
    "defaultLocale": "en",
    "readTranslationsFromGoogleSheets": false,
    "availableLocales": [
      {
        "id": "en",
        "label": "English",
        "filename": "localeEN.json"
      }
    ]
  },
  "sitemapConfig": {
    "baseUrl": "https://sampo-ui.demo.seco.cs.aalto.fi",
    "langPrimary": "en",
    "langSecondary": "fi",
    "outputDir": "./src/server/sitemap_generator",
    "sitemapUrl": "https://sampo-ui.demo.seco.cs.aalto.fi/sitemap",
    "sitemapInstancePageQuery": "sitemapInstancePageQuery"
  }
}
```

Configure available perspectives by ID

Configure available locales and files for getting the required translations



Perspective configuration: Basics

```
"id": "performances", Perspective ID (referenced in portal configuration)
"endpoint": {
  "url": "http://localhost:3048/ds/sparql", Configuration for the endpoint used in the perspective
  "useAuth": false, and what prefixes should be used in queries
  "prefixesFile": "SparqlQueriesPrefixes.js"
},
"sparqlQueriesFile": "SparqlQueriesPerformances.js", Name of file containing the SPARQL queries for the perspective
"baseURI": "http://ldf.fi/operasampo/",
"URITemplate": "<BASE_URI><LOCAL_ID>",
"facetClass": "scop:Performance", Class of the instances included in the perspective
"frontPageImage": "main_page/performances-card.jpg",
"searchMode": "faceted-search",
"defaultActiveFacets": [
  "prefLabel"
],
"defaultTab": "table",
"defaultInstancePageTab": "table",
"resultClasses": {
  "performances": {
    "paginatedResultsConfig": {
      "tabID": 0,
      "component": "ResultTable",
      "tabPath": "table",
      "tabIcon": "CalendarViewDay",
      "propertiesQueryBlock": "performanceProperties", Variable name for result table query in query file
      "pagesize": 15,
      "sortBy": null,
      "sortDirection": null
    }
  }
}
```

Perspective configuration: Queries

```
const perspectiveID = 'performances'
```

```
export const performanceProperties = `
```

```
{
  ?id skos:prefLabel ?prefLabel_id .
  BIND(?prefLabel_id AS ?prefLabel_prefLabel)
  BIND(CONCAT("/${perspectiveID}/page/", REPLACE(STR(?id), "^..*\\\\\\/(.+)", "$1")) AS ?prefLabel_dataProviderUrl)
  BIND(?id as ?uri_id)
  BIND(?id as ?uri_dataProviderUrl)
  BIND(?id as ?uri_prefLabel)
}
```

Queries basic label information and forms
the link for the instance page

```
UNION
```

```
{
  ?id scop:composition ?composition_id .
  ?composition_id skos:prefLabel ?composition_prefLabel .
  FILTER(LANG(?composition_prefLabel) = 'fi')
  BIND(CONCAT("/compositions/page/", REPLACE(STR(?composition_id), "^..*\\\\\\/(.+)", "$1")) AS ?composition_dataProviderUrl)
}
```

Queries the performance's original
composition and its label in Finnish

```
UNION
```

```
{
  ?id scop:composition/scop:composedBy ?composer_id .
  ?composer_id skos:prefLabel ?composer_prefLabel .
  BIND(CONCAT("/people/page/", REPLACE(STR(?composer_id), "^..*\\\\\\/(.+)", "$1")) AS ?composer_dataProviderUrl)
}
```

Queries the composer of the original
composition and that person's label

```
UNION
```

```
{
  ?id scop:composition/scop:libretist ?libretist_id .
  ?libretist_id skos:prefLabel ?libretist_prefLabel .
  BIND(CONCAT("/people/page/", REPLACE(STR(?libretist_id), "^..*\\\\\\/(.+)", "$1")) AS ?libretist_dataProviderUrl)
}
```

Queries the libretist of the original
composition and that person's label

```
UNION
```

Perspective configuration: Properties

```
properties": [  
  {  
    "id": "uri",  
    "valueType": "object",  
    "makeLink": true,  
    "externalLink": true,  
    "sortValues": true,  
    "numberedList": false,  
    "onlyOnInstancePage": true  
  },  
  {  
    "id": "prefLabel",  
    "valueType": "object",  
    "makeLink": true,  
    "externalLink": false,  
    "sortValues": true,  
    "numberedList": false,  
    "minWidth": 200  
  },  
  {  
    "id": "composition",  
    "valueType": "object",  
    "makeLink": true,  
    "externalLink": false,  
    "minWidth": 200  
  }  
]
```

Included properties (columns in the table) and their configuration are configured in the “properties” list

ID should match the name used in the SPARQL queries

Object type properties should at least have an ID (e.g., composition__id) and a label (composition__prefLabel) from the queries

Name ⓘ	Composition ⓘ	Composer ⓘ	Librettist ⓘ	Producer ⓘ
▼ Die Schweizer-Familie (1833-12-20)	Die Schweizer Familie	Weigl Joseph	Castelli Ignaz Franz	Deutschen Schauspieler-Gesellschaft
▼ Thaïs (1910-04-13)	Thaïs	Massenet Jules	Gallet Louis	Aino Ackté & Edvard Fazer
▼ Thaïs (1910-04-15)	Thaïs	Massenet Jules	Gallet Louis	Aino Ackté & Edvard Fazer
▼ Thaïs (1910-04-16)	Thaïs	Massenet Jules	Gallet Louis	Aino Ackté & Edvard Fazer

Perspective configuration: Facets

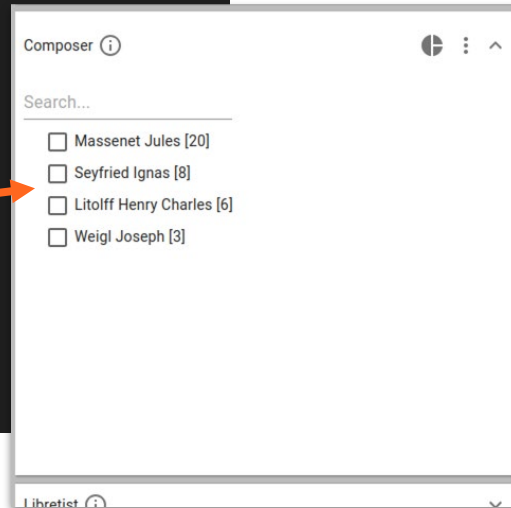
```
"facets": {  
  "prefLabel": {  
    "containerClass": "one",  
    "facetType": "text",  
    "filterType": "textFilter",  
    "sortByPredicate": "skos:prefLabel",  
    "textQueryProperty": "skos:prefLabel"  
  },  
}
```

Type for facet (e.g., checkbox, integer range, ...)

Configuration for getting the labels for values

Predicate for getting label values from instances

```
"composition": {  
  "containerClass": "ten",  
  "facetType": "list",  
  "facetLabelFilter": "FILTER(LANG(?prefLabel_) = 'fi')",  
  "filterType": "uriFilter",  
  "predicate": "scop:composition",  
  "searchField": true,  
  "sortBy": "instanceCount",  
  "sortByPredicate": "scop:composition/skos:prefLabel",  
  "sortDirection": "desc",  
  "pieChartButton": true  
},  
"composer": {  
  "containerClass": "ten",  
  "facetType": "list",  
  "filterType": "uriFilter",  
  "predicate": "scop:composition/scop:composedBy",  
  "searchField": true,  
  "sortBy": "instanceCount",  
  "sortByPredicate": "scop:composition/scop:composedBy/skos:prefLabel",  
  "sortDirection": "desc",  
  "pieChartButton": true  
},  
}
```



Perspective configuration: Visualizations

```
"resultClasses": {
  "performancesByProperty": {
    "resultClasses": {
      "performancesByPerformancePlace": {
        }
      }
    },
    "performanceLineChart": {
      "tabID": 2,
      "component": "ApexCharts",
      "tabPath": "performances_per_year",
      "tabIcon": "ShowChart",
      "sparqlQuery": "performancesByYearQuery",
      "facetClass": "performances",
      "filterTarget": "performance",
      "resultMapper": "mapLineChart",
      "resultMapperConfig": {
        "fillEmptyValues": true
      },
      "createChartData": "createSingleLineChartData",
      "title": "Vuositittaiset esitykset",
      "xaxisTitle": "Vuosi",
      "xaxisType": "category",
      "xaxisTickAmount": 30,
      "yaxisTitle": "Lukumäärä",
      "seriesTitle": "Lukumäärä",
      "stroke": {
        "width": 2
      }
    }
  }
}
```

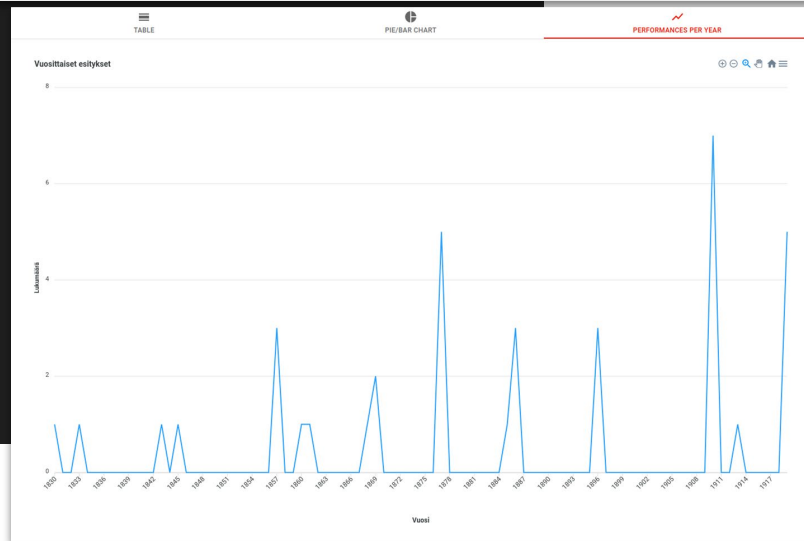
Component used for visualizing

Mapper for mapping the data from SPARQL bindings

Function taking the mapped data and outputting the data with the necessary configuration and format for the visualization component (e.g., adding chart options)

```
SELECT ?category (COUNT (DISTINCT ?performance) as
?count) WHERE {
  <FILTER>
  ?performance a scop:Performance ;
                scop:performanceDate ?date .
  BIND (YEAR(xsd:date(?date)) as ?category)
}
GROUP BY ?category
ORDER BY ?category
```

SPARQL query for the visualization data



Other tools and frameworks for creating web applications for LOD

- **Linked Data Reactor (LD-R)**
- **Metaphactory**
- **(VRTI-KG Explorer)**
- ...

Conclusions

- **After learning the basic process, setting up a new portal using Sampo-UI is easy and can be done in a few hours**
- **Sampo-UI includes many ready-to-use components that only require editing JSON and writing SPARQL to configure**
 - With basic use there is no need to touch the underlying JavaScript code or have deep understand of how it works
 - Examples of different kinds of facets and visualizations come with the framework and can be used as a base
 - *For more advanced use, Sampo-UI can be extended with new components, mappers, chart data creation functions to allow for more customization*

Links

- Tool page: <https://seco.cs.aalto.fi/tools/sampo-ui/>
- GitHub: <https://github.com/SemanticComputing/sampo-ui>
- Demo: <https://sampo-ui.demo.seco.cs.aalto.fi/>
- Tutorial:
 - Video: <https://vimeo.com/817028609>
 - Written tutorial: <https://seco.cs.aalto.fi/tools/sampo-ui/Sampo-UI-tutorial.pdf>