SHACL Playground

A constraint validator for the Shapes Constraint Language, written in JavaScript. **See also Zazuko SHACL Playground for a newer implementation**

Shapes Graph

```
@prefix sh: <http://www.w3.org/ns/shacl#> .
@prefix ns1: <https://makg.org/property/> .
@prefix ns2: <http://purl.org/spar/cito/> .
@prefix ns3: <http://www.w3.org/ns/org#> .
@prefix ns4: <http://purl.org/dc/terms/> .
@prefix ns5: <http://purl.org/spar/fabio/> .
@prefix ns6:
<http://lsdis.cs.uga.edu/projects/semdis/opus#> .
@prefix mag: <http://MAGexample.org/type/> .

mag:PaperShape
    a sh:NodeShape ;
    sh:targetClass mag:paper ;
    sh:property [
    sh:path  ns1:appearsInConferenceSeries ;
    sh:minCount 1 ;
] .
```

Update Format: Turtle

Always included: shacl.ttl dash.ttl

Parsing took 49 ms. Preparing the shapes took 3 ms. Validation the data took 10 ms.

Data Graph

Example Data in Turtle Format

```
@prefix ns1: <https://makg.org/property/>
.
@prefix ns2: <http://purl.org/spar/cito/>
.
@prefix ns3: <http://www.w3.org/ns/org#> .
@prefix ns4: <http://purl.org/dc/terms/> .
@prefix ns5: <http://purl.org/spar/fabio/>
.
@prefix ns6:
<http://lsdis.cs.uga.edu/projects/semdis/o
pus#> .

<http://MAGexample.org/00114AFB> a
<http://MAGexample.org/type/paper> ;
    ns1:appearsInConferenceSeries
<http://MAGexample.org/465F7C62> .

<http://MAGexample.org/019B9317> a
<http://MAGexample.org/type/paper> ;
    ns4:creator
<http://MAGexample.org/47C39427> .

<http://MAGexample.org/0286DD28> a
<http://MAGexample.org/type/paper> ;
    ns4:creator
```

Update Format: Turtle

TypeError: Cannot read properties of undefined (reading 'getShapesWithTarget')

Show function call sequence

Shapes Graph Structure

Shapes with Target (8)
Constraint Components (38)

Validation Report (32 results)

```
[
        a sh:ValidationResult ;
        sh:resultSeverity sh:Violation ;
        sh:sourceConstraintComponent
sh:MinCountConstraintComponent ;
        sh:sourceShape _:n12 ;
        sh:focusNode
<http://MAGexample.org/019B9317> ;
        sh:resultPath
<https://makg.org/property/appearsInConfer
enceSeries> ;
        sh:resultMessage "Less than 1
values" ;
] .
[
        a sh:ValidationResult ;
        sh:resultSeverity sh:Violation ;
        sh:sourceConstraintComponent
sh:MinCountConstraintComponent ;
        sh:sourceShape _:n12 ;
        sh:focusNode
<http://MAGexample.org/0286DD28> ;
        sh:resultPath
<https://makg.org/property/appearsInConfer
enceSeries> ;
        sh:resultMessage "Less than 1
values" ;
```

About SHACL

SHACL has been defined by a W3C Working Group and has become a widely used standard for describing structural constraints on data graphs and validate RDF instance data against those.

In a nutshell, a SHACL processor takes a *shapes graph* and a *data graph* as input. Both can be entered here using data formats such as Turtle or JSON-LD. The shapes graph defines so-called *shapes* which are a collection of *constraints*. A shape also tells the engine for which nodes in the data graph it applies to (using *targets*). Constraints are of a type, called *constraint components*.

This Implementation

This service has been implemented in JavaScript by Holger Knublauch. The service is free to use from this web site and a version of the source code can be found at SHACL-JS GitHub project. Last updated: 2017-05-01 and no longer actively maintained.

The design follows the SHACL-JS extension mechanism for SHACL. Each constraint component (such as `sh:minCount`) is backed by a JavaScript function. The linkage from constraint components to these JavaScript functions is entirely declarative via RDF triples in the shapes graph. The built-in SHACL Core components are implemented in the same way as any user-defined extension. As a result, SHACL is extensible for any kind of constraint that can be expressed in JavaScript, while the shape and constraint definitions can be distributed using linked data standards.

The parsing services and in-memory triple store used by this service are provided, with many thanks, by rdflib.js.

Known Issues

- Not yet systematically tested via test case library
- sh:path: Not all SHACL path constructs are supported
- sh:lessThan: Comparison of nodes via < etc does not work for all datatypes
- sh:datatype: Well-formedness of literals in sh:datatype is not always tested correctly
- sh:resultPath is not created as a deep copy if the path is a blank node
- No SHACL-SPARQL support
- No dynamic loading of JavaScript libraries and sh:js