JSON-LD Update

State of JSON-LD in 2019
TPAC 2019 – Fukuoka

Gregg Kellogg
#json-ld (W3C)

JSON-LD 1.1

- More generalized object indexing
- Properties can implicitly name anonymous graphs (“`@container`”: “`@graph`”).
- Property nesting.
- Recursive Lists (finally).
- Scoped Contexts (property or type) with control of propagation.
- Protected Term Definitions.
- Imported contexts.
- Framing on `@id`, more uniform frame matching semantics
- JSON Literals
- Deprecate Blank Node properties.
- Document-relative vocabulary mapping
- Normative HTML data-block extraction.
- Included Nodes.
- Poor-man’s content-negotiation (rel=alternate)
- Abstract from JSON-itself, allowing for YAML, CBOR and other LD representations.

Version Announcement

- For backwards compatibility, version 1.1 must be specified to use new 1.1 features (may be through API).

```
{
  "@context":
  {
     "@version": 1.1,
     "schema": "http://schema.org/",
     "name": "schema:name",
     "body": "schema:articleBody",
     "words": "schema:wordCount",
     "post": {
       "@id": "schema:blogPost",
       "@container": "@id"
     }
  },
  "@id": "http://example.com/",
  "@type": "schema:Blog",
  "name": "World Financial News",
  …
}
```

@id Maps

```
{
  "@context":
  {
     "@version": 1.1,
     "schema": "http://schema.org/",
     "name": "schema:name",
     "body": "schema:articleBody",
     "words": "schema:wordCount",
     "post": {
       "@id": "schema:blogPost",
       "@container": "@id"
     }
  },
  "@id": "http://example.com/",
  "@type": "schema:Blog",
  "name": "World Financial News",
  "post": {
     "http://example.com/posts/1/en": {
       "body": "World commodities were up today with heavy trading of crude oil...",
       "words": 1539
     },
     "http://example.com/posts/1/de": {
       "body": "Die Werte an Warenbörsen stiegen im Sog eines starken Handels von Rohöl...",
       "words": 1204
     }
  }
}
```

- Also @type maps and @graph/@id maps.
- Keys may include @none.

Graph Containers

- Property values can reference anonymously/ implicitly named graphs:

```
{
  "@context": {
    "@version": 1.1,
    "@base": "http://dbpedia.org/resource/",
    "said": "http://example.com/said",
    "wrote": {"@id": "http://example.com/wrote", "@container": "@graph"}
  },
  "@id": "William_Shakespeare",
  "wrote": {
    "@id": "Richard_III_of_England",
    "said": "My kingdom for a horse"
  }
}
```

Nested Properties

```
{
  "@context": {
    "@version": 1.1,
    "skos": "http://www.w3.org/2004/02/skos/core#",
    "labels": "@nest",
    "main_label": {"@id": "skos:prefLabel"},
    "other_label": {"@id": "skos:altLabel"},
    "homepage": {"@id": "http://schema.org/description", "@type": "@id"}
  },
  "@id": "http://example.org/myresource",
  "homepage": "http://example.org",
  "labels": {
     "main_label": "This is the main label for my resource",
     "other_label": "This is the other label"
  }
}
```

- Also allows nesting to be created when compacting.

Document-relative Vocabulary

- Better alternative to (deprecated) blank-node properties.

Recursive Lists

```
{
  "@context": {
    "@vocab": "https://purl.org/geojson/vocab#",
    "type": "@type",
    "bbox": {"@container": "@list"},
    "coordinates": {"@container": "@list"}
  },
  "type": "Feature",
  "bbox": [-10.0, -10.0, 10.0, 10.0],
  "geometry": {
    "type": "Polygon",
    "coordinates": [
        [
            [-10.0, -10.0],
            [10.0, -10.0],
            [10.0, 10.0],
            [-10.0, -10.0]
        ]
    ]
  }
  //...
}
```

```
@prefix geojson: <https://purl.org/geojson/vocab#>.

[
  a geojson:Feature ;
  geojson:bbox (-1.0E1 -1.0E1 1.0E1 1.0E1) ;
  geojson:geometry [
    a geojson:Polygon ;
    geojson:coordinates (
      (
        (-1.0E1 -1.0E1)
        (1.0E1 -1.0E1)
        (1.0E1 1.0E1)
        (-1.0E1 -1.0E1)
      )
    )
  ]
] .
```

Scoped Contexts

```
{
  “@context": {
    "@version": 1.1,
    "name": "http://schema.org/name",
    "interest": {
      "@id":"http://xmlns.com/foaf/0.1/interest",
      "@context": {"@vocab": "http://xmlns.com/foaf/0.1/"}
    }
  },
  "name": "Manu Sporny",
  "interest": {
    "@id": "https://www.w3.org/TR/json-ld/",
    "name": "JSON-LD",
    "topic": "Linking Data"
  }
}
```

- Terms can also be used as values of @type, with similar behavior.

Protected Term Definitions

```
{
  "@context": [
    {
      "@version": 1.1,
      "Person": "http://xmlns.com/foaf/0.1/Person",
      "knows": "http://xmlns.com/foaf/0.1/knows",
      "name": {
        "@id": "http://xmlns.com/foaf/0.1/name",
        "@protected": true
      }
    },
    {
      - this attempt will fail with an error
      "name": "http://schema.org/name"
    }
  ],
  "@type": "Person",
  "name": "Manu Sporny",
  "knows": {
    "@context": [
      - this attempt would also fail with an error
      null,
      "http://schema.org/"
    ],
    "name": "Gregg Kellogg"
  }
}
```

- Prevents accidental overriding of term definitions by downstream contexts.
  - Can be overridden in a property-scoped context.

Imported Contexts

```
{
  "@context": {
    "@version": 1.1,
    "MyType": {
      "@id": "http://example.com/vocab#MyType",
      "@context": {
        "@version": 1.1,
        "@import": "https://json-ld.org/contexts/remote-context.jsonld",
        "@propagate": true
      }
    }
  }
}
```

- Useful for adding 1.1 behavior to 1.0 contexts
- In the future, may allow for metadata such as sub-resource integrity.

JSON Literals

```
{
  "@context": {
    "@version": 1.1,
    "e": {"@id": "http://example.com/vocab/json", "@type": "@json"}
  },
  "e": [
    56.0,
    {
      "d": true,
      "10": null,
      "1": [ ]
    }
  ]
}
```

- Equivalent to rdf:JSON datatype in N-Triples/Turtle.
- Defines a canonical lexical form for value representation, whitespace and ordering (based on JCS[1]).

[1] JCS: JSON Canonicalization Scheme https://tools.ietf.org/html/draft-rundgren-json-canonicalization-scheme-05

Included Nodes

```
{
  "@context": {
    "@version": 1.1,
    "@vocab": "http://example.org/",
    "classification": {"@type": "@vocab"},
    "service": {"@type": "@vocab"}
  },
  "@id": "http://example.org/base/1",
  "@type": "Thing-with-Items",
  "items": [{
    "@id":"http://example.org/base/2",
    "classification": "enum#c6",
    "service": "enum#s2"
  }, {
    "@id": "http://example.org/base/3",
    "classification": "enum#c6"
  }],
  "@included": [{
    "@id": "http://example.org/enum#c6",
    "@type": "Type",
    "label": "Classification 6"
  }, {
    "@id": "http://example.org/enum#s2",
    "@type": "Service",
    "label": "Login Service"
  }]
}
```

- Useful as an alternative to @graph when objects are not directly related by a property, but should appear together.

Framing Improvements (embedding)

- `@embed: @last` => `@embed: @once (@never & @always`

Framing Improvements (Named Graphs)

```
{
  "@context": {"@vocab": "http://example.org/"},
  "@type": "Library",
  "contains": {
    "@id": "http://example.org/graphs/books",
    "@graph": {
      "@type": "Book"
    }
  }
}
```

- Outer is merged graph, unless frame includes @graph.

Framing Improvements (@id matching)

```
{
  "@context": {"ex": "http://example.org/"},
  "@id": ["ex:Sub1", "ex:Sub2"]
}
```

```
{
  "@context": {
    "@base": "http://example.org/",
    "@vocab": "http://example.org/vocab#",
    "@version": 1.1
  },
  "characters": {
    "format": {
      "char": {
        "@id": {},
        "states": {
          "@default": {}
        }
      }
    }
  }
}
```

- Selects any node with the selected identifier (or wildcard ({}).

Framing Improvements (@included)

```
{
  "@context": {
    "@version": 1.1,
    "@vocab": "http://example.org/"
  },
  "@requireAll": true,
  "foo": "bar",
  "prop": "value",
  "@included": [{
    "@requireAll": true,
    "foo": "bar",
    "prop": "value2"
  }]
}
```

- For adding nodes without a direct property relationship.

Framing Improvements (Value Patterns)

Exact Value Match

```
{
  "@context": {"ex": "http://example.org/"},
  "@id": "ex:Sub1",
  "ex:p": "P",
  "ex:q": {"@value": "Q", "@type": "ex:q"},
  "ex:r": {"@value": "R", "@language": "r"}
}
```

Match on exact type/ language but wildcard value

```
{
  "@context": {"ex": "http://example.org/"},
  "ex:p": {"@value": {}},
  "ex:q": {"@value": {}, "@type": "ex:q"},
  "ex:r": {"@value": {}, "@language": "r"}
}
```

Match on exact value wildcard type/language

```
{
  "@context": {"ex": "http://example.org/"},
  "ex:q": {"@value": "Q", "@type": {}}
}
```

Match on no value

```
{
  "@context": {"ex": "http://example.org/"},
  "ex:p": {"@value": {}, "@type": []},
  "ex:q": {"@value": {}, "@type": "ex:q"},
  "ex:r": {"@value": {}, "@language": "r"}
}
```

- Also match on any selected value/type/language with […]

Framing Improvements (@included)

```
{
  "@context": {
    "@version": 1.1,
    "@vocab": "http://example.org/"
  },
  "@requireAll": true,
  "foo": "bar",
  "prop": "value",
  "@included": [{
    "@requireAll": true,
    "foo": "bar",
    "prop": "value2"
  }]
}
```

HTML data-block extraction

```
<script type="application/ld+json">
{
  "@context": "https://json-ld.org/contexts/person.jsonld",
  "@id": "http://dbpedia.org/resource/John_Lennon",
  "name": "John Lennon",
  "born": "1940-10-09",
  "spouse": "http://dbpedia.org/resource/Cynthia_Lennon"
}
</script>
```

- It's been in practice since schema.org, now normalized.
- Also describe how to manage multiple script elements, using document base and fragment identifier.

Poor-man's content-negotiation

```
GET /index.html HTTP/1.1
Host: example.com
Accept: application/ld+json,application/json,*/*;q=0.1

====================================

HTTP/1.1 200 OK
...
Content-Type: text/html
Link: <alternate.jsonld>; rel="alternate"; type="application/ld+json"

<html>
  <head>
    <title>Primary Entrypoint</title>
  </head>
  <body>
    <p>This is the primary entrypoint for a vocabulary</p>
  </body>
</html>
```

- For non-JSON formats, rel=alternate with type=application/ld points to an alternate version to be used by a processor.

YAML-LD

```
Example 044: Defining an @context within a term definition
---
"@context":
  "@version": 1.1
  name: http://schema.org/name
  interest:
    "@id": http://xmlns.com/foaf/0.1/interest
    "@context":
      "@vocab": http://xmlns.com/foaf/0.1/
name: Manu Sporny
interest:
  "@id": https://www.w3.org/TR/json-ld11/
  name: JSON-LD
  topic: Linking Data
```

```
{
  "@context": {
    "@version": 1.1,
    "name": "http://schema.org/name",
    "interest": {
      "@id": "http://xmlns.com/foaf/0.1/interest",
      "@context": {"@vocab": "http://xmlns.com/foaf/0.1/"}
    }
  },
  "name": "Manu Sporny",
  "interest": {
    "@id": "https://www.w3.org/TR/json-ld11/",
    "name": "JSON-LD",
    "topic": "Linking Data"
  }
}
```

- Can be extended to other formats that map to JSON

JSON-LD 1.1 Timeline

- Working Group started June 2018 – Completes in June 2020 [5]
  - CR expected Q4 2019
- Ruby implementation tracks Working Drafts.
  - Live at Ruby RDF Distiller [6].

[5] https://w3.org/2018/json-ld-wg
[6] http://rdf.greggkellogg.net/distiller

More Information

w3.org/2018/json-ld-wg/

json-ld.org

Gregg Kellogg

gregg@greggkellogg.net

@gkellogg