

Future Internet

ILARIA TORRE

Ilaria.torre@unige.it

Serialization Formats for Web Pages



RDF Graph Serializations for Web Pages

- Microdata
- RDFa
- JSON-LD

RDFa, Microdata, Json-LD and Microformats



RDFa, Microdata, Json-LD and Microformats are progressively evolving the Web towards a machine-processable semantic web.

All four formalisms allow you to add annotations to web pages using HTML tags.

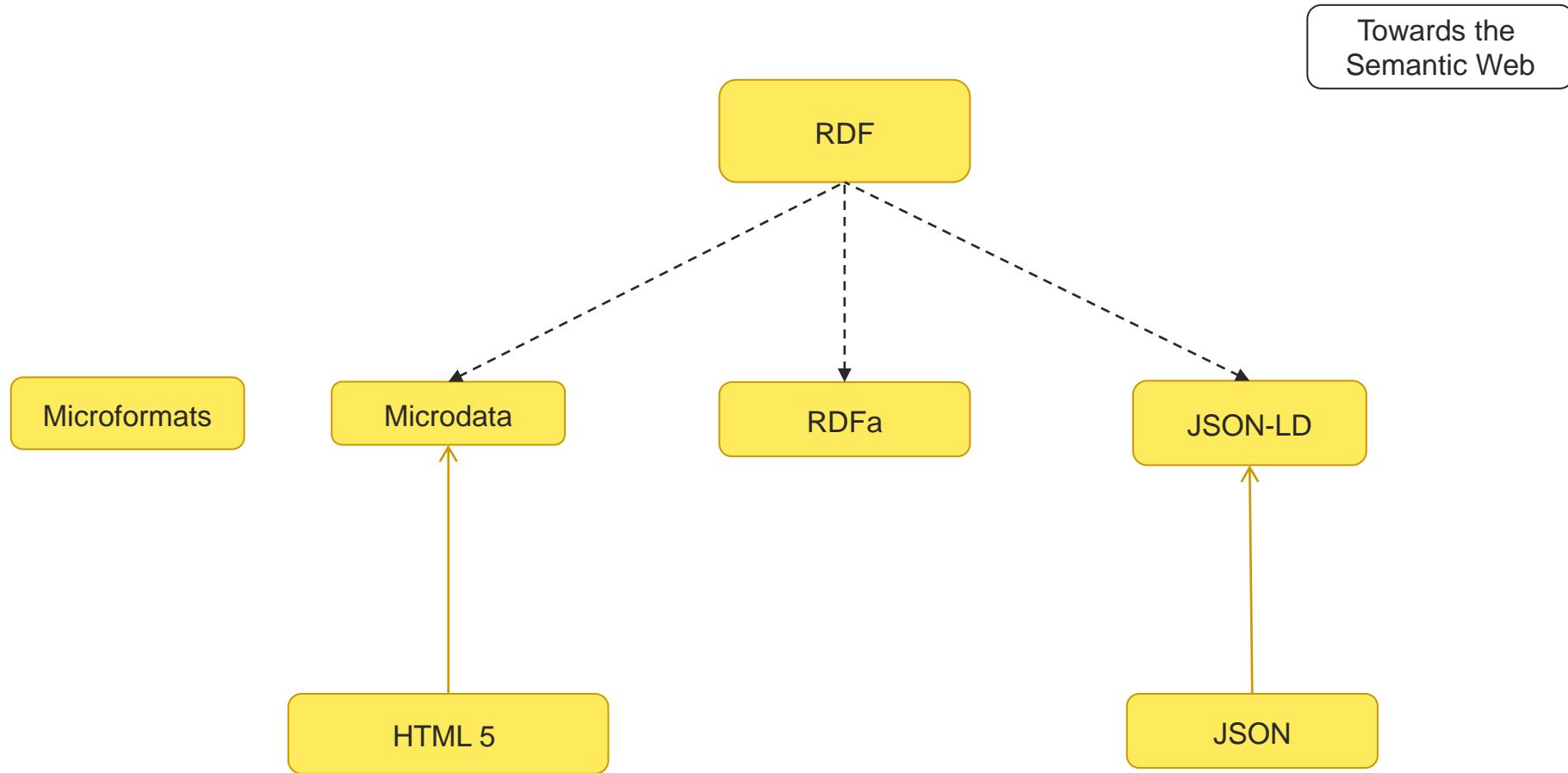
- RDFa, Microdata, Json-LD are serializations of RDF datasets
- RDFa, Microdata introduce new HTML attributes
- Json-LD uses the Json format inserted in the SCRIPT tag of the HTML code
- Microformats was the first attempt to use vocabularies to annotate web page data, which is now obsolete

Google defines this metadata formats as **STRUCTURED DATA** and uses them, for example, to obtain **rich snippets**.

A **snippet** is the informational content that Google adds to your search results: in addition to Page Title and Page Link, it can include:

Description | Images | Reviews, etc.

Structured data



JSON-LD



JSON-LD (*JSON-Linked Data*) is a JSON-based format to serialize Linked Data.

It is currently Google's recommended format for annotating web pages for SEO purposes

JSON-LD



JSON-LD (*JSON-Linked Data*) is a JSON-based format to serialize Linked Data.

JSON (JavaScript Object Notation): is an open standard file format, and data interchange format, that uses human-readable text to store and transmit data objects consisting of attribute–value pairs

Goals:

- easily integrate semantic annotation into deployed systems that already use JSON,
- store Linked Data in JSON-based storage engines

JSON-LD



The latest version released is

JSON-LD 1.1 - Third Edition,
W3C Recommendation 16 July 2020

<https://www.w3.org/TR/json-ld11/>

Unlike microdata and RDFa,

JSON-LD structured are placed in the SCRIPT tag

JSON-LD – HTML page



```
<html>
```

```
<head>
```

```
<title>Generative Artificial Intelligence </title>
```

```
<script type="application/ld+json">
```

```
{
```

```
  "@context": {
```

```
    "rdf": "http://www.w3.org/1999/02/22-rdf-syntax-ns#",
```

```
    "rdfs": "http://www.w3.org/2000/01/rdf-schema#",
```

```
    ..... .
```

```
}
```

```
</script>
```

```
</head>
```


JSON-LD



Basic concepts:

- a universal identifier mechanism for JSON objects via the use of IRIs
- a node is identified using the **@id** keyword
- vocabularies can be declared using **@context** keyword
More generally, @context can be used to map terms (i.e., strings) to IRIs and use them as shortcuts throughout a JSON-LD document
- default vocabulary can be set using **@vocab** keyword
- the type (i.e., class) of a graph node is specified using **@type** keyword
- multiple nodes can be specified using **@graph** keyword

JSON-LD



```
{
  "@context": {
    "rdf": "http://www.w3.org/1999/02/22-rdf-syntax-ns#",
    "rdfs": "http://www.w3.org/2000/01/rdf-schema#",
    "xsd": "http://www.w3.org/2001/XMLSchema#",
    "ex": "http://www.exampleOntol.org#",
    "@vocab": "http://schema.org/"
  },
  .....
}
```

In the case of a single context, you can simply indicate "@context": "IRI"

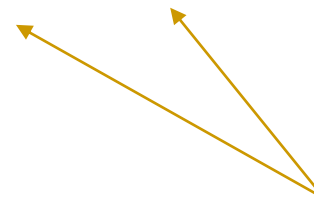
```
"@context": "https://schema.org",
.....
```

JSON-LD



```
{
  "@context": {
    "rdf": "http://www.w3.org/1999/02/22-rdf-syntax-ns#",
    "rdfs": "http://www.w3.org/2000/01/rdf-schema#",
    "xsd": "http://www.w3.org/2001/XMLSchema#",
    "ex": "http://www.exampleOntol.org#",
    "@vocab": "http://schema.org/"
  },
  "@id": "Event#0005",
  "@type": "http://www.exampleOntol.org/Event",
  "ex:name": "Generative Artificial Intelligence",
  "ex:place": "Genoa"
}
```

or "@type": "ex:Event"



Data properties

JSON-LD



```
{
  "@context": {
    "rdf": "http://www.w3.org/1999/02/22-rdf-syntax-ns#",
    "rdfs": "http://www.w3.org/2000/01/rdf-schema#",
    "xsd": "http://www.w3.org/2001/XMLSchema#",
    "ex": "http://www.exampleOntol.org#",
    "@vocab": "http://schema.org/"
  },
  "@id": "Event#0005",
  "@type": "http://www.exampleOntol.org/Event",
  "ex:name": "Generative Artificial Intelligence",
  "ex:place": "Genoa",

  "ex:hasSpeaker": {
    "@id": "Speaker#1",
    "@type": "Person"
    .....
  }
}
```




Object property

JSON-LD

```
{
  "@context": {
    "rdf": "http://www.w3.org/1999/02/22-rdf-syntax-ns#",
    "rdfs": "http://www.w3.org/2000/01/rdf-schema#",
    "xsd": "http://www.w3.org/2001/XMLSchema#",
    "ex": "http://www.exampleOntol.org#",
    "@vocab": "http://schema.org/"
  },
  "@id": "Event#0005",
  "@type": "ex:Event",
  "ex:name": "Generative Artificial Intelligence",
  "ex:place": "Genoa",

  "ex:hasSpeaker": {
    "@id": "Speaker#1",
    "@type": "Person",
    "url": ["http://homepage.com", "http://office.com"]
  }
}
```

multiple values: array -
list of values [.... ,]



Testing



To test the JSON-LD code, use:

JSON-LD Playground: <https://json-ld.org/playground>

JSON-LD

<https://json-ld.org/playground>

JSON-LD Playground

Compacted view

The screenshot displays the JSON-LD Playground interface. At the top, there are tabs for 'Examples:' (selected), 'Person', 'Event', 'Place', 'Product', 'Recipe', and 'Library'. Below these are 'JSON-LD Input' and 'Options' buttons, and a 'Document URL' input field. The main text area contains a JSON-LD document in expanded form, which is then rendered in a compacted view below. The compacted view shows the document as a single JSON object with keys for '@id', '@type', and property URIs. The 'Expanded' button is selected in the bottom toolbar.

```

{
  "@id": "Event#0005",
  "@type": "http://www.exampleOntol.org#Event",
  "http://www.exampleOntol.org#hasSpeaker": {
    "@id": "Speaker#1",
    "@type": "http://schema.org/Person",
    "http://schema.org/url": [
      "http://homepage.com",
      "http://office.com"
    ]
  },
  "http://www.exampleOntol.org#name": "Generative Artificial Intelligence",
  "http://www.exampleOntol.org#place": "Genoa"
}

```

JSON-LD

<https://json-ld.org/playground>

JSON-LD Playground

Expanded view

```
{
  "@context": {
    "rdf": "http://www.w3.org/1999/02/22-rdf-syntax-ns#",
    "rdfs": "http://www.w3.org/2000/01/rdf-schema#",
    "xsd": "http://www.w3.org/2001/XMLSchema#",
    "ex": "http://www.exampleOntol.org#",
    "@vocab": "http://schema.org/"
  },
  "@id": "Event#0005",
  "@type": "ex:Event",
  "ex:name": "Generative Artificial Intelligence",
  "ex:place": "Genoa",

  "ex:hasSpeaker": {
    "@id": "Speaker#1",
    "@type": "Person",
    "url": ["http://homepage.com", "http://office.com"]
  }
}
```

Expanded Compacted Flattened Framed N-Quads Canonicalize

```
{
  "@id": "Event#0005",
  "@type": [
    "http://www.exampleOntol.org#Event"
  ],
  "http://www.exampleOntol.org#hasSpeaker": [
    {
      "@id": "Speaker#1",
      "@type": [
        "http://schema.org/Person"
      ],
      "http://schema.org/url": [
        {
          "@value": "http://homepage.com"
        },
        {
          "@value": "http://office.com"
        }
      ]
    }
  ],
  "http://www.exampleOntol.org#name": [
    {
      "@value": "Generative Artificial Intelligence"
    }
  ],
  "http://www.exampleOntol.org#place": [
    {
      "@value": "Genoa"
    }
  ]
}
```


JSON-LD



```
{
  "@context": {
    "rdf": "http://www.w3.org/1999/02/22-rdf-syntax-ns#",
    "rdfs": "http://www.w3.org/2000/01/rdf-schema#",
    "xsd": "http://www.w3.org/2001/XMLSchema#",
    "ex": "http://www.exampleOntol.org#",
    "@vocab": "http://schema.org/"
  },
  "@graph": [{
    "@id": "Event#0005",
    "@type": "http://www.exampleOntol.org/Event",
    "ex:name": "Generative Artificial Intelligence",
    "ex:place": "Genoa",
    "ex:hasSpeaker": "Speaker#1"
  }, {
    "@id": "Speaker#1",
    "@type": "Person",
    "url": "[http://homepage.com, http://office.com]"
  }
]
```



@graph
to represent a graph
with more nodes

JSON-LD

<https://json-ld.org/playground>

JSON-LD Playground

The screenshot shows the JSON-LD Playground interface. At the top, there are tabs for 'JSON-LD Input', 'Options', and 'Document URL'. The 'JSON-LD Input' tab is active, displaying a JSON-LD document. Below the input area, there are buttons for 'Expanded', 'Compacted', 'Flattened', 'Framed', 'N-Quads', and 'Canonized'. The 'Expanded' button is selected, and the expanded JSON-LD document is shown in the output area.

```
{
  "@context": {
    "rdf": "http://www.w3.org/1999/02/22-rdf-syntax-ns#",
    "rdfs": "http://www.w3.org/2000/01/rdf-schema#",
    "xsd": "http://www.w3.org/2001/XMLSchema#",
    "ex": "http://www.exampleOntol.org#",
    "@vocab": "http://schema.org/"
  },
  "@graph": [
    {
      "@id": "Event#0005",
      "@type": "http://www.exampleOntol.org/Event",
      "ex:name": "Generative Artificial Intelligence",
      "ex:place": "Genoa",
      "ex:hasSpeaker": "Speaker#1"
    },
    {
      "@id": "Speaker#1",
      "@type": "Person",
      "url": "[http://homepage.com, http://office.com]"
    }
  ]
}
```

The expanded JSON-LD document is shown below:

```
{
  "@graph": [
    {
      "@id": "Event#0005",
      "@type": "http://www.exampleOntol.org/Event",
      "http://www.exampleOntol.org#hasSpeaker": "Speaker#1",
      "http://www.exampleOntol.org#name": "Generative Artificial Intelligence",
      "http://www.exampleOntol.org#place": "Genoa"
    },
    {
      "@id": "Speaker#1",
      "@type": "http://schema.org/Person",
      "http://schema.org/url": "[http://homepage.com, http://office.com]"
    }
  ]
}
```

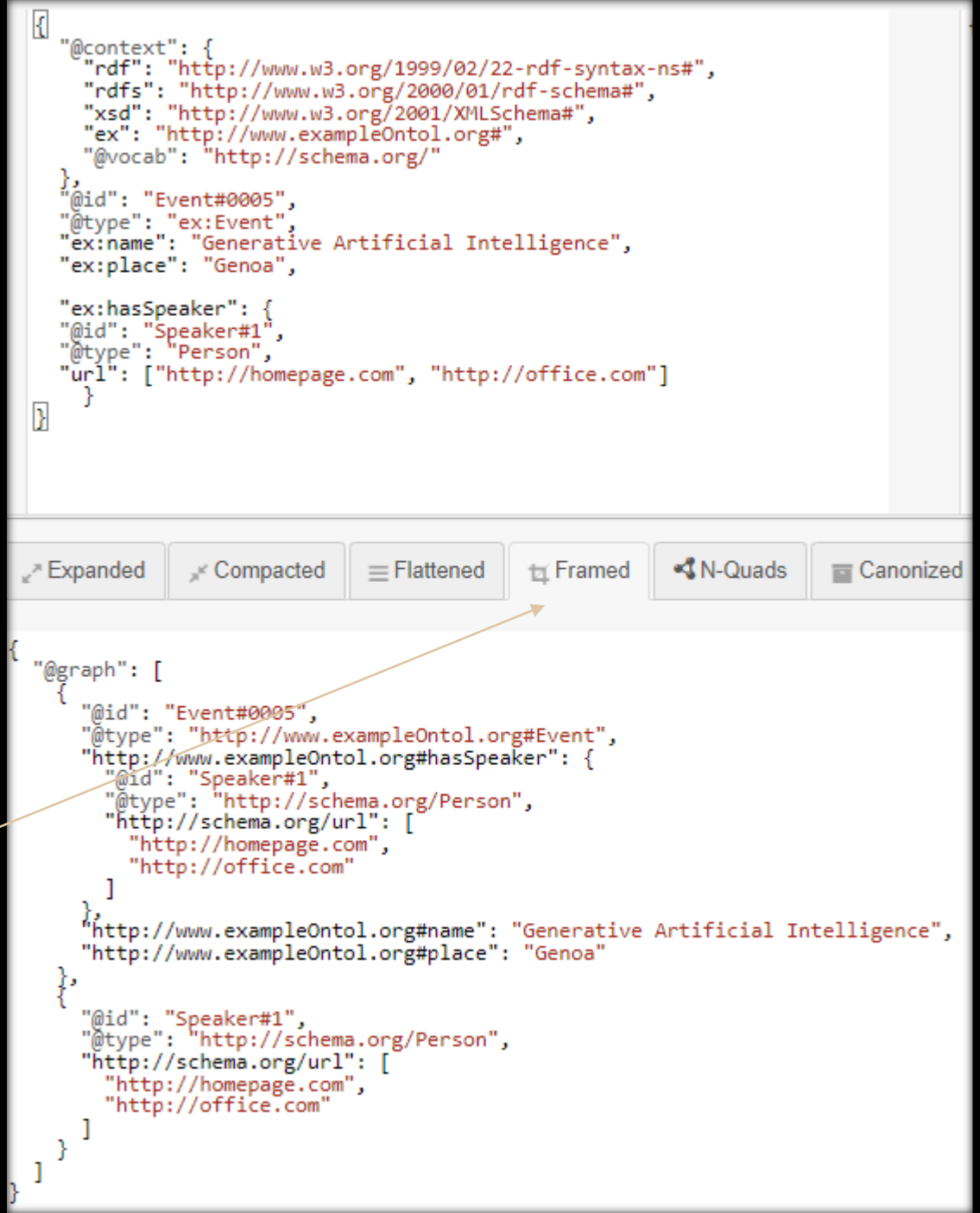
JSON-LD

<https://json-ld.org/playground>

JSON-LD Playground

Framed view for the previous example:

Note that it uses the `@graph` keyword even though it was not in our code (it is taken from the object property `hasSpeaker`)



The screenshot shows the JSON-LD Playground interface. The top panel displays the input JSON-LD code in the 'Expanded' view. The bottom panel shows the output in the 'Framed' view, which is selected in the navigation bar. An orange arrow points from the text 'it is taken from the object property hasSpeaker' to the `hasSpeaker` property in the input code. Another orange arrow points from the `@graph` keyword in the output code to the 'Framed' button in the navigation bar.

```
{
  "@context": {
    "rdf": "http://www.w3.org/1999/02/22-rdf-syntax-ns#",
    "rdfs": "http://www.w3.org/2000/01/rdf-schema#",
    "xsd": "http://www.w3.org/2001/XMLSchema#",
    "ex": "http://www.exampleOntol.org#",
    "@vocab": "http://schema.org/"
  },
  "@id": "Event#0005",
  "@type": "ex:Event",
  "ex:name": "Generative Artificial Intelligence",
  "ex:place": "Genoa",

  "ex:hasSpeaker": {
    "@id": "Speaker#1",
    "@type": "Person",
    "url": ["http://homepage.com", "http://office.com"]
  }
}
```

Navigation bar: Expanded, Compacted, Flattened, **Framed**, N-Quads, Canonized

```
{
  "@graph": [
    {
      "@id": "Event#0005",
      "@type": "http://www.exampleOntol.org#Event",
      "http://www.exampleOntol.org#hasSpeaker": {
        "@id": "Speaker#1",
        "@type": "http://schema.org/Person",
        "http://schema.org/url": [
          "http://homepage.com",
          "http://office.com"
        ]
      },
      "http://www.exampleOntol.org#name": "Generative Artificial Intelligence",
      "http://www.exampleOntol.org#place": "Genoa"
    },
    {
      "@id": "Speaker#1",
      "@type": "http://schema.org/Person",
      "http://schema.org/url": [
        "http://homepage.com",
        "http://office.com"
      ]
    }
  ]
}
```