A Proposal for an OpenMath JSON Encoding

Tom Wiesing Michael Kohlhase
Computer Science, FAU Erlangen-Nürnberg

July 10, 2019
OpenMath Workshop
Conference on Intelligent Computer Mathematics
Prague, Czech Republic

What is JSON?

- JSON = **J**ava**S**cript **O**bject **N**otation
  - lightweight data-interchange format
  - subset of JavaScript (used a lot on the web)
  - defined independently
- Primitive types
  - Strings (e.g. `"Hello␣world"`)
  - Numbers (e.g. `42` or `3.14159265`)
  - Booleans (`true` and `false`)
  - `null`
- Composite types
  - Arrays (e.g. `[1, "two", false]`)
  - Objects (e.g. `{"foo": "bar", "answer": 42}`)

Why an OpenMath encoding for JSON?

- an OpenMath JSON encoding would make it easy to use across many languages
  - JSON support exists in most modern programming languages
    - corresponding native types common
    - serialization to/from JSON without external library
- some existing approaches for an OpenMath JSON encoding
  - discussed / suggested on the OpenMath mailing list
  - we will look at two examples here

XML as JSON

- **Idea:** Generically encode XML as JSON
- use the JSONML standard for this
- e.g. $\mathrm{plus}(x, 5)$ corresponds to:

```
[
    "OMOBJ",
    {"xmlns":"http://www.openmath.org/OpenMath"},
    [
        "OMA",
        ["OMS", {"cd": "arith1", "name": "plus"}],
        ["OMV", {"name": "x"}],
        ["OMI", "5"]
    ]
]
```

XML as JSON (2)

- Advantages
  - based on well-known XML encoding
  - easy to understand based on it
- does not make use of JSON structures
  - all attributes are encoded as strings, even numbers
  - e.g. `1e-10` (a valid JSON literal) can not be used
- retains some of the XML akwardness
  - introduces unnecessary overhead
  - e.g. some pseudo-elements (such as OMATP) are needed

OpenMath-JS

- OpenMath-JS
  - an (incomplete) implementation of OpenMath in JavaScript
  - developed by Nathan Carter for use with Lurch Math on the web
  - written in literate coffee script, a derivative language of JavaScript
- e.g. $\mathrm{plus}(x, 5)$ corresponds to:

```
{
    "t": "a",
    "c": [
      {"t": "sy", "cd": "arith1", "n": "plus"},
      {"t": "v", "n": "x"},
      {"t": "i", "v": "5"}
    ]
}
```

OpenMath-JS (2)

- does make use of JSON native structures
  - much better than *JSON-ML*
  - small property names keep size of transmitted objects small
- comes with some problems
  - hard to read for humans
  - written for *JavaScript*, not JSON
  - no formal schema

Towards an OpenMath JSON Formalization

- we need to write a new OpenMath JSON encoding
  - combine advantages of the above two
  - should be close to the XML encoding
  - should make use of JSON concepts
- we want to formalize this JSON encoding
  - to verify JSON objects
  - not done by existing approaches
- comes with some positive side effects
  - formalization of JSON $\Rightarrow$ structure definition in most languages
  - trivial to use advanced serialization tools
    - e.g. *Protocol Buffers*, *ZeroMQ*
- we can use JSON Schema
  - a vocabulary allowing us to validate and annotate JSON documents
  - tools for verification exist

Towards an OpenMath JSON Formalization (2)

- JSON schema is often tedious to write and read
  - especially when it comes to recusrive data types
  - but implementation of it still exist
- **Idea:** Write schema in a **TypeScript Definition file**, compile into a JSON schema
  - TypeScript = JavaScript + Type Annotations
  - easily writeable and understandable
  - a compiler from TypeScript Definitions into JSON Schema exists
- We have done this, will present some examples

Towards an OpenMath JSON Formalization (3)

- Wrote a JSON Schema
  - was written as described above
  - we will give an overview how this looks below
- Wrote a translator from OpenMath XML to JSON (we have actually built two)
  1. web demo on (`https://omjson.openmath.org`)
  2. as part of MMT (i.e. Scala) in the form of a RESTful API
- Wrote an extension of the *OpenMath Standard* to make it an official encoding

$\Rightarrow$ Let's look at the concrete proposal for examples