Network Working Group
Internet-Draft
Intended status: Standards Track
Expires: December 7, 2012

R. Bhat
P. Saint-Andre
Cisco Systems, Inc.

June 5, 2012

A JavaScript Object Notation (JSON) Representation for vCard
draft-bhat-vcarddav-json-00

Abstract

This document defines a representation of vCard data in JavaScript Object Notation (JSON).

Status of this Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at http://datatracker.ietf.org/drafts/current/.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on December 7, 2012.

Copyright Notice

Table of Contents

1. Introduction

vCard [RFC6350] is a data format for representing and exchanging information about individuals and other entities. It is a text-based format (as opposed to a binary format). This document defines a representation for vCard data in JavaScript Object Notation (JSON), a lightweight, text-based, language-independent data interchange format derived from the ECMAScript Programming Language Standard (see [RFC4627]). As with the XML representation of vCard defined in [RFC6351], the data structure is exactly the same as for plain vCard, enabling a 1-to-1 mapping between the plain vCard format and the JSON representation (or the XML representation). The JSON formatting might be preferred in some contexts where JSON facilities are readily available and can be reused instead of writing a standalone vCard parser.

2. Discussion Venue

The preferred discussion venue for this document is the vcarddav@ietf.org mailing list, for which subscription information and archives can be found at <https://www.ietf.org/mailman/listinfo/vcarddav>.

3. Terminology

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "NOT RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in [RFC2119].

4. Example

To ease comparison with both plain vCard and the XML representation,

the following example is a JSON representation of the same vCard (for Simon Perreault) that is also shown in [RFC6350] and [RFC6351].

```
{
    "version":"4.0",
    "fn":  "Simon Perreault",
    "n": {
        "surname": "Simon",
        "given": "Perreault",
        "suffix": [ "ing. jr", "M.Sc."]
    },
    "bday": { "date": "--0203" },
```



```
    "anniversary": { "date-time": "20090808T1430-0500" },
    "gender": { "sex" : "M" },
    "lang": [ {
        "pref": 1,
        "language-tag": "fr",
    },
    {
        "pref": 2,
        "language-tag": "en",
    },
    ],
    "org": {
        "type": "work",
        "text": "Viagenie"
    },
    "adr": {
        "type": "work",
        "label": "Simon Perreault
2875 boul. Laurier, suite D2-630
Quebec, QC, Canada G1V 2M2",
        "street": "2875 boul. Laurier, suite D2-630",
        "locality": "Quebec",
        "region": "QC",
        "code": "G1V 2M2",
        "country": "CA"
    },
    "tel": [ {
        "type":  ["work", "voice"],
        "uri": "tel:+1-418-656-9254;ext=102"
```

```
        },
        {
            "type": ["work", "text", "voice", "cell", "video"],
            "uri": "tel:+1-418-262-6501"
        }
        ],
        "email": {
        "type": "work",
        "text": "simon.perreault@viagenie.ca"
        },
        "geo": {
            "type": "work",
            "uri": "geo:46.766336,-71.28955"
        },
        "key": {
            "type": "work",
            "uri": "http://www.viagenie.ca/simon.perreault/simon.asc"
        },
        "tz": "America/Montreal",
```



```
        "url": {
            "type": "home",
            "uri": "http://nomis80.org"
        }
    }
```

Figure 1: JSON representation of RFC 6350 example

5. Design Considerations

The general idea is to map vCard parameters, properties and value types to JSON property/value pairs. For example, the "FN" property is mapped to the fn property. Value contains a text string that corresponds to the vCard property's value. vCard parameters are also mapped to JSON objects which are contained in the value object. For example, the "TYPE" parameter applied to the "TEL" property would look like the following in JSON.

```
        "tel": {
        "type": [ "voice", "video" ],
        "uri": "tel:+1-555-555-555"
```

```
    }
```

Figure 2: Mapping of a vCard parameter

Parameters taking a list of values and properties with cardinality of more than one are converted to a JSON array object. Properties having structured values (e.g., the "N" property) are expressed by nested JSON object trees. Properties within that tree ("surname", "given", etc.) are mapped as simple name/value pairs. These pairs should follow the schema defined by the XML mapping of vCard(Appendix A in [RFC6351]) Line folding is a non-issue in JSON. Therefore, the mapping from vCard to JSON is done after the unfolding procedure is carried out. Conversely, the mapping from JSON to vCard is done before the folding procedure is carried out. The group construct (Section 3.2 in [RFC6350]) is represented with the JSON object of the same name. For example:


```
{
    "version" : "4.0"
    "contact": {
        "fn": "...",
        "email": "..."
    },
    "media" : {
        "photo": "..."
    },
    "categories": "..."
}
```

Figure 3: Mapping of lists and structured values

... is equivalent to:

```
BEGIN:VCARD
VERSION:4.0
contact.FN=...
contact.EMAIL=...
media.PHOTO=...
CATEGORIES=...
END:VCARD
```

Figure 4: Plain vCard equivalent

The VALUE parameter from the plain VCARD format is used as the property name in the JSON format. If there is no VALUE parameter specified, it is treated as equivalent to VALUE=text. For example:

```
{
    "email": {
    "type": "work",
    "text": "simon.perreault@viagenie.ca"
    },
    "geo": {
        "type": "work",
        "uri": "geo:46.766336,-71.28955"
    },
    "bday": { "date": "--0203" },
}
```

Figure 5: Mapping of VALUE parameter

... is equivalent to:


```
BEGIN:VCARD
VERSION:4.0
EMAIL;VALUE=text;type=work:simon.perreault@viagenie.ca
GEO;VALUE=uri;type=work:geo:46.766336,-71.28955
BDATE;VALUE=date:--0203
END:VCARD
```

Figure 6: Plain vCard equivalent

In the plain vCard format, the "VERSION" property was mandatory and played a role in extensibility. In XML, this property was dropped in favor of the XML namespace mechanism. In the JSON mapping, we keep the "version" property, which plays a similar role as in the plain vCard format.

Finally, there is no reason to include a top-level name of "vcard" or "vcards", since the data type can be determined from the media type of the data file.

6. Extensibility

The plain vCard format is extensible. New properties, parameters, data types and values (collectively known as vCard elements, not to be confused with XML elements) can be registered with IANA (see [RFC6350], Section 10.2). It is expected that these vCard extensions will also specify extensions to the JSON format described in this document. New JSON vCard property and parameter element names MUST be lower-case. This is necessary to ensure that round-tripping between JSON and plain-text vCard works correctly. Unregistered extensions (i.e., those starting with "X-" and "VND-...-") are expressed in JSON by using properties starting with "x-" and "vnd-...-". Refer to Section 7 for the implications when converting between plain-text vCard and JSON. For example:

```
{
    "x-my-prop": {
        "pref":  1,
    "text": "value goes here"
    }
}
```

Figure 7: An example of extensibility

A vCard JSON parser MUST ignore JSON parameters and properties for which it doesn't recognize the name.

In the XML representation of vCard [RFC6351], extensibility is

handled in part by using XML namespaces [XML-NAMES] for properties

and parameters that have no equivalent in plain-text vCard. For extensions that might appear in both the JSON representation and the XML representation, it is RECOMMENDED to represent the JSON parameter or property name in "Clark Notation" [CLARK] by preceding the name itself with the Uniform Resource Identifier [RFC3986] of the XML namespace, enclosed in curly brackets ('{' and '}'); thus the "expanded name" will be of the form "{URI}name". For extensions that will appear in the JSON representation but not the XML representation, a mere (non-expanded) name can be used, or the name can be an expanded name formed in another manner (e.g., using the "reverse domain name" convention such as "com.example.vcard.foo").

The JSON format does not validate the cardinality of properties. This is a limitation of the JSON format specification. Cardinalities of the plain vCard format [RFC6350] MUST still be respected.

7. Format Conversions

To follow

8. Schema definition

To follow

9. Security Considerations

All the security considerations applicable to plain vCard [RFC6350] are also applicable to the JSON representation of vCard.

As explained in [RFC4627], JSON is a subset of JavaScript, but it is a safe subset that excludes assignment and invocation. A JSON text can be safely passed into JavaScript's eval() function (which compiles and executes a string) if all the characters not enclosed in strings are in the set of characters that form JSON tokens.

10. IANA Considerations

To: ietf-types@iana.org

Subject: Registration of media type application/vcard+json

```
Type name: application
Subtype name: vcard+json
Required parameters: (none)
Optional parameters: (none)
Encoding considerations:  8bit if UTF-8; binary if UTF-16 or UTF-32;
   see RFC 4627.
Security considerations:  All of the security considerations
   specified in RFC 4627 and RFC 6350 apply.
Interoperability considerations: (none)
Specification:  XXXX
Applications that use this media type:  vCard processors.
Additional information:
   Magic number(s):  (none)
   File extension(s):  .jcard
   Macintosh file type code(s):  TEXT
Person and email address to contact for further information:  vCard
   discussion mailing list, <vcarddav@ietf.org>
Intended usage:  COMMON
Restrictions on usage:  (none)
Author:  Peter Saint-Andre, <psaintan@cisco.com>
Change controller:  Peter Saint-Andre, <psaintan@cisco.com>
```

11. Acknowledgements

Thanks to Joe Hildebrand for his feedback.

Some text in this document was borrowed from [RFC6351].

12. References

12.1. Normative References

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.

[RFC4627] Crockford, D., "The application/json Media Type for JavaScript Object Notation (JSON)", RFC 4627, July 2006.

[RFC6350] Perreault, S., "vCard Format Specification", RFC 6350, August 2011.

12.2. Informative References

[CLARK] Clark, J., "Clark Notation", February 1999,

[RFC3986] Berners-Lee, T., Fielding, R., and L. Masinter, "Uniform Resource Identifier (URI): Generic Syntax", STD 66, RFC 3986, January 2005.

[RFC6351] Perreault, S., "xCard: vCard XML Representation", RFC 6351, August 2011.

[XML-NAMES] Thompson, H., Hollander, D., Layman, A., Bray, T., and R. Tobin, "Namespaces in XML 1.0 (Third Edition)", World Wide Web Consortium Recommendation REC-xml-names-20091208, December 2009, <http://www.w3.org/TR/2009/REC-xml-names-20091208>.

Authors' Addresses

Raghurama Bhat
Cisco Systems, Inc.
900 McCarthy Blvd.
Milpitas, CA 95035
USA

Phone: +1-408-902-2123
Email: ragbhat@cisco.com

Peter Saint-Andre
Cisco Systems, Inc.
1899 Wynkoop Street, Suite 600
Denver, CO 80202
USA

Phone: +1-303-308-3282
Email: psaintan@cisco.com