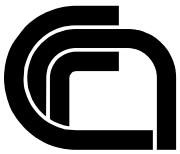

Consiglio Nazionale delle Ricerche

JSContact: A JSON representation of contact card information

R. Stepanek, M. Loffredo

IIT TR-05/2019

Technical Report

Ottobre 2019

Istituto di Informatica e Telematica

JSContact:
A JSON representation of contact card information

Stepanek R. – FastMail - rsto@fastmailteam.com
Loffredo M. – IIT CNR – mario.loffredo@iit.cnr.it

Summary

Abstract

This document defines a data model and JSON representation of contact card information that can be used for data storage and exchange in address book or directory applications. It aims to be an alternative to the vCard data format and to be unambiguous, extendable and simple to process. In contrast to the JSON-based jCard format, it is not a direct mapping from the vCard data model and expands semantics where appropriate.
The JSON contact representation described in this document has been implemented by the .it RDAP public test server as an alternative to jCard in order to provide contact information within the RDAP response.

1. Introduction

This document defines a data model for contact card data normally used in address book or directory applications and services. It aims to be an alternative to the vCard data format [RFC6350] and to provide a JSON-based standard representation of contact card data.

The key design considerations for this data model are as follows:

- Most of the initial set of attributes should be taken from the vCard data format [RFC6350] and extensions ([RFC6473], [RFC6474], [RFC6715], [RFC6869], [RFC8605]). The specification should add new attributes or value types, or not support existing ones, where appropriate. Conversion between the data formats need not fully preserve semantic meaning.

- The attributes of the cards data represented must be described as a simple key-value pair, reducing complexity of its representation.

- The data model should avoid all ambiguities and make it difficult to make mistakes during implementation.

- Extensions, such as new properties and components, MUST NOT lead to requiring an update to this document.

The representation of this data model is defined in the I-JSON format [RFC7493], which is a strict subset of the JavaScript Object Notation (JSON) Data Interchange Format [RFC8259]. Using JSON is mostly a pragmatic choice: its widespread use makes JSCard easier to adopt, and the availability of production-ready JSON implementations eliminates a whole category of parser-related interoperability issues.

1.1. Relation to the xCard and jCard formats

The xCard [RFC6351] and jCard [RFC7095] specifications define alternative representations for vCard data, in XML and JSON format respectively. Both explicitly aim to not change the underlying data model. Accordingly, they are regarded as equal to vCard in the context of this document.

2. JSCard

MIME type: "application/jscontact+json;type=jscard"

A JSCard object stores information about a person, organization or company.

2.1. Metadata properties

2.1.1 uid

Type: "String" (mandatory).

An identifier, used to associate the object as the same across different systems, addressbooks and views. [RFC4122] describes a range of established algorithms to generate universally unique identifiers (UUID), and the random or pseudo-random version is recommended. For compatibility with [RFC6350] UIDs, implementations MUST accept both URI and free-form text.

2.1.2 prodid

Type: "String" (optional).

The identifier for the product that created the JSCard object.

2.1.3 updated

Type: "String" (mandatory).

The date and time when the data in this JSCard object was last modified. The timestamp MUST be formatted as specified in [RFC3339].

2.1.4 kind

Type: "String" (optional).

The kind of the entity the Card represents. The value MUST be either one of the following values, registered in a future RFC, or a vendor-specific value:

- "individual": a single person
- "org": an organization
- "location": a named location
- "device": a device, such as appliances, computers, or network elements
- "application": a software application

2.2. Name and Organization properties

2.2.1. name

Type: "String" (mandatory).

The name of the entity represented by this JSCard. A Name object has the following properties:

- fullName: LocalizedString (mandatory). The full name (e.g. the personal name and surname of an individual, the name of an organization) of the entity represented by this card.
- prefix: String[] (optional). The honorific title(s), e.g. "Mr", "Ms", "Dr".
- personalName: String[] (optional). The personal name(s), also known as "first name", "give name".
- surname: String[] (optional). The surname(s) (also known as "last name", "family name").
- additionalName: String[] (optional). The additional name(s), also known as "middle name".
- suffix: String[] (optional). The honorific suffix(es), e.g. "B.A.", "Esq.".
- nickname: String[] (optional). The nickname(s) of the person represented by this card.

2.2.2. organization

Type: "LocalizedString[]" (optional).

The company or organization name and units associated with this card. The first entry in the list names the organization, and any following entries name organizational units.

2.2.3. jobTitle

Type: "LocalizedString[]" (optional).

The job title(s) or functional position(s) of the entity represented by this card.

2.2.4. role

Type: "LocalizedString[]" (optional).

The role(s), function(s) or part(s) played in a particular situation by the entity represented by this card. In contrast to a job title, the roles might differ for example in project contexts.

2.3. Contact and Resource properties

2.3.1. emails

Type: "Resource[]" (optional).

An array of Resource objects where the values are URLs in the [RFC2368] "mailto" scheme or free-text email addresses. Types are:

- "personal" The address is for emailing in a personal context.
- "work" The address is for emailing in a professional context.
- "other" The address is for some other purpose. A label property MAY be included to display next to the address to help the user identify its purpose.

2.3.2. phones

Type: "Resource[]" (optional).

An array of Resource objects where the values are URIs scheme or free-text phone numbers. Typical URI schemes are the [RFC3966] "tel" or [RFC3261] "sip" schemes, but any URI scheme is allowed. Contact method types are:

- "voice" The number is for calling by voice.
- "fax" The number is for sending faxes.
- "pager" The number is for a pager or beeper.
- "other" The number is for some other purpose. A label property MAY be included to display next to the number to help the user identify its purpose.

The following labels are pre-defined for phone contact methods:

- "private" The phone number should be used in a private context.
- "work" The phone number should be used in a professional context.

2.3.3. online

Type: "Resource[]" (optional).

An array of Resource objects where the values are URIs or usernames associated with the card for online services. Types are:

- "uri" The value is a URI, e.g. a website link.

- "username" The value is a username associated with the entity represented by this card (e.g. for social media, or an IM client). A label property SHOULD be included to identify what service this is for. For compatibility between clients, this label SHOULD be the canonical service name, including capitalization. e.g. "Twitter", "Facebook", "Skype", "GitHub", "XMPP".

- "other" The value is something else not covered by the above categories. A label property MAY be included to display next to the number to help the user identify its purpose.

2.3.4. preferredContactMethod

Type: "String" (optional).

Defines the preferred contact method. The value MUST be the property name of one of the Resource lists: "emails", "phones", "online", "other".

2.4. Address and Location properties

2.4.1. addresses

Type: "Address[]" (optional).

An array of Address objects, containing physical locations. An Address object has the following properties:

- type: String (mandatory). Specifies the context of the address information. The value MUST be either one of the following values, registered in a future RFC, or a vendor-specific value:
    - "home" An address of a residence.
    - "work" An address of a workplace.
    - "billing" An address to be used for billing.
    - "postal" An address to be used for delivering physical items.
    - "other" An address not covered by the above categories.
- label: String (optional). A label describing the value in more detail.
- fullAddress: LocalizedString (optional). The complete address, excluding type and label. This property is mainly useful to represent addresses of which the individual address components are unknown, or to provide localized representations.
- street: String (optional). The street address. This MAY be multiple lines; newlines MUST be preserved.
- extension: String (optional) The extended address, such as an apartment or suite number, or care-of address.
- postOfficeBox: String (optional) The post office box.
- locality: String (optional). The city, town, village, post town, or other locality within which the street address may be found.
- region: String (optional). The province, such as a state, county, or canton within which the locality may be found.
- postcode: String (optional). The postal code, post code, ZIP code or other short code associated with the address by the relevant country's postal system.
- country: String (optional). The country name.

- countryCode: String (optional). The ISO-3166-1 country code.

- coordinates: String (optional) A [RFC5870] "geo:" URI for the address.

- timeZone: String (optional) Identifies the time zone this address is located in. This SHOULD be a time zone name registered in the IANA Time Zone Database [1]. Unknown time zone identifiers MAY be ignored by implementations.

- isPreferred: Boolean (optional, default: "false"). Whether this Address is the preferred for its type. This SHOULD only be one per type.

2.5. Additional properties

2.5.1. anniversaries

Type: "Anniversary[]" (optional).

Memorable dates and events for the entity represented by this card. An Anniversary object has the following properties:

- type: String (mandatory). Specifies the type of the anniversary. This RFC predefines the following types, but implementations MAY use additional values:

    - "birth": a birth day anniversary
    - "death": a death day anniversary
    - "other": an anniversary not covered by any of the known types.

- date: String (mandatory). The date of this anniversary, in the form "YYYY-MM-DD" (any part may be all 0s for unknown) or a [RFC3339] timestamp.

- place: Address (optional). An address associated with this anniversary, e.g. the place of birth or death.

2.5.2. personalInfo

Type: " PersonalInformation[]" (optional).

A list of personal information about the entity represented by this card. A PersonalInformation object has the following properties:

- type: String (mandatory). Specifies the type for this personal information. Allowed values are:

    - "expertise": a field of expertise or credential
    - "hobby": a hobby
    - "interest": an interest

    - "other": an information not covered by the above categories

- value: String (mandatory). The actual information. This generally is free-text, but future specifications MAY restrict allowed values depending on the type of this PersonalInformation.

- level: String (optional) Indicates the level of expertise, or engagement in hobby or interest. Allowed values are: "high", "medium" and "low".

2.5.3. notes

Type: "LocalizedString[]" (optional).

Arbitrary notes about the entity represented by this card.

2.5.4. categories

Type: "String[]" (optional).

A list of free-text or URI categories that relate to the card.

2.6. Common JSCard types

2.6.1. LocalizedString

A LocalizedString object has the following properties:

- value: String (mandatory). The property value.

- language: String (optional). The [RFC5646] language tag of this value, if any.

- localizations: String[String] (optional). A map of any additional localized values of string, indexed by RFC5646 language tag.

2.6.2. Resource

A Resource object has the following properties:

- type: String (mandatory). Specifies the context of the contact method. This MUST be taken from the set of values allowed depending on whether this is part of the phones, emails or online property (see above).

- label: String (optional). A label describing the value in more detail, especially if the type property has value "other" (but MAY be included with any type).

- value: String (mandatory). The actual contact method, e.g. the email address or phone number.

- mediaType: String (mandatory). It is used with properties whose value is a URI. It provides the media type [RFC2046] of the resource identified by the URI.

- isPreferred: Boolean (optional, default: "false"). Whether this Resource is the preferred for its type. This SHOULD only be one per type.

3. JSCardGroup

MIME type: "application/jscontact+json;type=jscardgroup"

A JSCardGroup object represents a named set of JSCards.

3.1. Properties

3.1.1. uid

Type: "String" (mandatory).

A globally unique identifier. The same requirements as for the JSCard uid property apply.

3.1.2. name

Type: "String" (optional).

The user-visible name for the group, e.g. "Friends". This may be any UTF-8 string of at least 1 character in length and maximum 255 octets in size. The same name may be used by two different groups.

3.1.3. cardIds

Type: "String[]" (mandatory).

The ids of the cards in the group. Implementations MUST preserve the order of list entries.

4. JSCard in RDAP

There is a general feeling in the community of developers about jCard replacement in the RDAP response ([RFC7483]). The jCard representation is considered inefficient because it can't be serialized/deserialized straightforwardly by using the most common JSON libraries available on the web.

Since JSCard fixes those issues, it seems to be more suitable than jCard to represent contact information in the RDAP response.

The .it RDAP public test server (https://rdap.pubtest.nic.it) implements an optional capability to provide contact information according to the JSCard data model. Such capability is available for both lookup and search queries ([RFC7482]) and can requested by setting the parameter “jscard” to 1/true (e.g. https://rdap.pubtest.nic.it/domain/nic.it?jscard=1).

A comparison between the same contact data extracted from an RDAP response according to, respectively, jCard and JSCard is shown in the following:


```
…
"vcardArray": [
  "vcard",
  [
     ["version", { }, "text", "4.0"],
     ["fn", { }, "text", "ccTLD '.it' Registry - IIT/CNR"],
     ["adr", { "cc": "it" }, "text", [ "", "", "Via Giuseppe Moruzzi 1", "Pisa", "PI", "56124", "Italy"]],
     ["org", { }, "text","ccTLD '.it' Registry - IIT/CNR"],
     ["kind", { }, "text", "org"],
     ["tel", {"type": "voice"}, "uri", "tel:+39.0503139811"],
     ["tel", {"type": "fax"}, "uri", "tel:+39.050542420"],
     ["email", { }, "text", "hostmaster@nic.it"]
  ]
],
 …
```

```
…

"jscard": {
  "uid": "CNT-ITNIC-1036705",
  "kind": "org",
  "name": {
    "fullName": {"value": "ccTLD '.it' Registry - IIT/CNR"}
  },
  "organization": [
    {"value": "ccTLD '.it' Registry - IIT/CNR"}
  ],
  "emails": [
    {"type": "work", "value": "hostmaster@nic.it"}
  ],
  "phones": [
    {"type": "voice", "label": "work", "value": "+39.0503139811"},
    {"type": "fax", "label": "work", "value": "+39.050542420"}
  ],
  "addresses": [
    {"type": "work",
     "fullAddress": {"value": "Via Giuseppe Moruzzi 1 Pisa PI 56124 Italy IT"},
     "street": "Via Giuseppe Moruzzi 1",
     "locality": "Pisa",
     "region": "PI",
```

```
      "postcode": "56124",
      "country": "Italy",
      "countryCode": "it"
     }
   ],
   "updated": "2016-06-03T15:19:28+02:00"
},
…
```


5. References

4.1. Normative References

[RFC2046] Freed, N. and N. Borenstein, "Multipurpose Internet Mail Extensions (MIME) Part Two: Media Types", RFC 2046, DOI 10.17487/RFC2046, November 1996, <https://www.rfc-editor.org/info/rfc2046>.

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <https://www.rfc-editor.org/info/rfc2119>.

[RFC4122] Leach, P., Mealling, M., and R. Salz, "A Universally Unique IDentifier (UUID) URN Namespace", RFC 4122, DOI 10.17487/RFC4122, July 2005, <https://www.rfc-editor.org/info/rfc4122>.

[RFC5870] Mayrhofer, A. and C. Spanring, "A Uniform Resource Identifier for Geographic Locations ('geo' URI)", RFC 5870, DOI 10.17487/RFC5870, June 2010, <https://www.rfc-editor.org/info/rfc5870>.

[RFC6350] Perreault, S., "vCard Format Specification", RFC 6350, DOI 10.17487/RFC6350, August 2011, <https://www.rfc-editor.org/info/rfc6350>.

[RFC6351] Perreault, S., "xCard: vCard XML Representation", RFC 6351, DOI 10.17487/RFC6351, August 2011, <https://www.rfc-editor.org/info/rfc6351>.

[RFC7095] Kewisch, P., "jCard: The JSON Format for vCard", RFC 7095, DOI 10.17487/RFC7095, January 2014, <https://www.rfc-editor.org/info/rfc7095>.

[RFC7493] Bray, T., Ed., "The I-JSON Message Format", RFC 7493, DOI 10.17487/RFC7493, March 2015, <https://www.rfc-editor.org/info/rfc7493>.

[RFC8174] Leiba, B., "Ambiguity of Uppercase vs Lowercase in RFC 2119 Key Words", BCP 14, RFC 8174, DOI 10.17487/RFC8174, May 2017, <https://www.rfc-editor.org/info/rfc8174>.

[RFC8259] Bray, T., Ed., "The JavaScript Object Notation (JSON) Data Interchange Format", STD 90, RFC 8259, DOI 10.17487/RFC8259, December 2017, <https://www.rfc-editor.org/info/rfc8259>.

4.2. Informative References

[RFC2368] Hoffman, P., Masinter, L., and J. Zawinski, "The mailto URL scheme", RFC 2368, DOI 10.17487/RFC2368, July 1998, <https://www.rfc-editor.org/info/rfc2368>.

[RFC3261] Rosenberg, J., Schulzrinne, H., Camarillo, G., Johnston, A., Peterson, J., Sparks, R., Handley, M., and E. Schooler, "SIP: Session Initiation Protocol", RFC 3261, DOI 10.17487/RFC3261, June 2002, <https://www.rfc-editor.org/info/rfc3261>.

[RFC3339] Klyne, G. and C. Newman, "Date and Time on the Internet: Timestamps", RFC 3339, DOI 10.17487/RFC3339, July 2002, <https://www.rfc-editor.org/info/rfc3339>.

[RFC3966] Schulzrinne, H., "The tel URI for Telephone Numbers", RFC 3966, DOI 10.17487/RFC3966, December 2004, <https://www.rfc-editor.org/info/rfc3966>.

[RFC5646] Phillips, A., Ed. and M. Davis, Ed., "Tags for Identifying Languages", BCP 47, RFC 5646, DOI 10.17487/RFC5646, September 2009, <https://www.rfc-editor.org/info/rfc5646>.

[RFC6473] Saint-Andre, P., "vCard KIND:application", RFC 6473, DOI 10.17487/RFC6473, December 2011, <https://www.rfc-editor.org/info/rfc6473>.

[RFC6474] Li, K. and B. Leiba, "vCard Format Extensions: Place of Birth, Place and Date of Death", RFC 6474, DOI 10.17487/RFC6474, December 2011, <https://www.rfc-editor.org/info/rfc6474>.

[RFC6715] Cauchie, D., Leiba, B., and K. Li, "vCard Format Extensions: Representing vCard Extensions Defined by the Open Mobile Alliance (OMA) Converged Address Book (CAB) Group", RFC 6715, DOI 10.17487/RFC6715, August 2012, <https://www.rfc-editor.org/info/rfc6715>.

[RFC6869] Salgueiro, G., Clarke, J., and P. Saint-Andre, "vCard KIND:device", RFC 6869, DOI 10.17487/RFC6869, February 2013, <https://www.rfc-editor.org/info/rfc6869>.

[RFC7482] Newton, A. and S. Hollenbeck, "Registration Data Access Protocol (RDAP) Query Format", RFC 7482, DOI 10.17487/RFC7482, March 2015, <https://www.rfc-editor.org/info/rfc7482>.

[RFC7483] Newton, A. and S. Hollenbeck, "JSON Responses for the Registration Data Access Protocol (RDAP)", RFC 7483, DOI 10.17487/RFC7483, March 2015, <https://www.rfc-editor.org/info/rfc7483>.

[RFC8605] HOLLENBECK, S. AND R. CARNEY, "VCARD FORMAT EXTENSIONS: ICANN EXTENSIONS FOR THE REGISTRATION DATA ACCESS PROTOCOL (RDAP)", RFC 8605, DOI 10.17487/RFC8605, MAY 2019, <HTTPS://WWW.RFC-EDITOR.ORG/INFO/RFC8605>.

4.3. URIs

[1] HTTPS://WWW.IANA.ORG/TIME-ZONES