Workgroup: jmap
Internet-Draft:
draft-ietf-jmap-jscontact-vcard-09
Published: 16 January 2022
Intended Status: Standards Track
Expires: 20 July 2022
Authors: M. Loffredo
IIT-CNR/Registro.it
R. Stepanek
FastMail

JSContact: Converting from and to vCard

Abstract

This document defines how to convert contact information as defined in the JSContact [draft-ietf-jmap-jscontact] specification from and to vCard.

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 https://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 20 July 2022.

Copyright Notice

Table of Contents

1. Introduction

1.1. Motivation

The JSContact specification [draft-ietf-jmap-jscontact] has been defined to represent contact card information as a more efficient alternative to vCard [RFC6350] and its JSON-based version named jCard [RFC7095].

While new applications might adopt JSContact as their main format to exchange contact card data, they are likely to interoperate with services and clients that just support vCard/jCard. Similarly, existing contact data providers and consumers already using vCard/jCard might want to represent their data also according to the JSContact specification.

To facilitate this, this document defines how to convert contact information as defined in the JSContact [draft-ietf-jmap-jscontact] specification from and to vCard.

1.2. Scope and Caveats

JSContact and vCard have a lot of semantics in common, however some differences must be outlined:

* The JSContact data model defines some contact information that doesn't have a direct mapping with vCard properties. In particular, unlike vCard, JSContact distinguishes between a single contact card, named Card, and a group of contact cards, named CardGroup.

* The properties that can be present multiple times in a vCard are represented through different collections in JSContact; mainly as maps, sometimes as lists, in some cases condensed in a single value.

1.3. Conventions Used in This Document

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

In the following of this document, the vCard features, namely properties and parameters, are written in uppercase while the Card/CardGroup features are written in camel case wrapped in double quotes.

1.4. Extensions

While translating vCard to JSContact, any vCard property that doesn't have a direct counterpart in JSContact MUST be converted into a property whose name is prefixed by "ietf.org:<RFC defining the extension>:" (e.g. "ietf.org:rfc6350:").

Any custom extension MAY be added and its name MUST be prefixed with a specific domain name to avoid conflict, e.g. "example.com:customprop".

Likewise, while translating JSContact to vCard, a JSContact property that doesn't have a direct counterpart in vCard MUST be converted into a property whose name is prefixed with "X-" as specified in Section 6.10 of [RFC6350].

2. Translating vCard properties to JSContact

This section contains the translation rules from vCard to Card/CardGroup. The vCard properties are grouped according to the categories as defined in [RFC6350].

If a vCard represents a group of contacts, those vCard properties which don't have a counterpart in CardGroup are converted into related properties of the "CardGroup.card" object. In this case, the "uid" member of both the resulting CardGroup object and its "card" member MUST have the same value.

2.1. Common Parameters

The following mapping rules apply to parameters that are common to most of the vCard properties:

* The generic values of the TYPE parameter are mapped to the values of the "Context" type as defined in Section 1.5.1 of [draft-ietf-jmap-jscontact]. The "home" value corresponds to the "private" key. The mapping of those specific TYPE values used in the TEL and RELATED properties are defined in Section 2.6.1 and Section 2.8.5.

* The PREF parameter is mapped to the "pref" property.

* The MEDIATYPE parameter is mapped to the "mediaType" property. As described in Section 5.7 of [RFC6350], the media type of a resource can be identified by its URI. For example, "image/gif" can be derived from the ".gif" extension of a GIF image URI. JSContact producers MAY provide the media type information even when it is not specified in the vCard.

* The ALTID and LANGUAGE parameters are used in combination for associating the language-dependent alternatives with a given property. Such alternatives are represented by using the "localizations" map: the "localizations" key is the LANGUAGE value, the key of the related PatchObject map is the JSON pointer of the JSContact member matching the vCard property while the value is the JSContact member itself.

2.2. Unmapped JSContact Information

The rules to generate a map key of type Id as well as a value for "created", "language" and "preferredContactMethod" properties are out of the scope of this document.

2.3. General Properties

2.3.1. BEGIN and END

The BEGIN and END properties don't have a direct match with a JSContact feature.

2.3.2. SOURCE

A SOURCE property is represented as an entry of the "online" map (Figure 1). The entry value is a "Resource" object whose "type" member is set to "uri", the "label" member is set to "source" and the "resource" member is the SOURCE value.

The PREF and MEDIATYPE parameters are mapped according to the rules as defined in Section 2.1.

```
BEGIN:VCARD
VERSION:4.0
...
SOURCE:http://directory.example.com/addressbooks/jdoe/Jean%20Dupont.
...
END:VCARD
```

```
{
...
"online":{
   ...
   "a-source":{
     "type": "uri",
     "label": "source",
     "resource": "http://directory.example.com/addressbooks/jdoe/Jea
   },
   ...
},
...
}
```

Figure 1: SOURCE mapping example

2.3.3. KIND

The KIND property is mapped to the "kind" member (Figure 2). Allowed values are those described in Section 6.1.4 of [RFC6350] and extended with the values declared in [RFC6473] and [RFC6869]. The value "group" is reserved for a CardGroup instance.

```
BEGIN:VCARD
VERSION:4.0
...
KIND:individual
...
END:VCARD

{
...
"kind": "individual",
...
}
```

Figure 2: KIND mapping example

2.3.4. XML

The XML property doesn't have a direct match with a JSContact feature.

2.4. Identification Properties

2.4.1. FN

All the FN instances are represented through the "fullName" member (Figure 3). The presence of multiple instances is implicitly associated with the full name translations in various languages regardless of the presence of the ALTID parameter. Each translation is mapped according to the rules as defined in Section 2.1.

If the vCard represents a group of contacts, implementers MAY convert the FN property into either "CardGroup.card.fullName" or "CardGroup.name" or both properties.

2.4.2. N and NICKNAME

The N instances are converted into equivalent items of the "components" array of the "name" property (Figure 3): the N components are transformed into related "NameComponent" objects as presented in Table 1. Name components SHOULD be ordered such that their values joined by whitespace produce a valid full name of this entity.

Each NICKNAME instance is mapped to an item of "nickNames" array.

N component	"type" value
Honorific Prefixes	prefix
Given Names	personal
Family Names	surname

N component	"type" value
Additional Names	additional
Honorific Suffixes	suffix

Table 1: N components mapping

```
BEGIN:VCARD
VERSION:4.0
...
FN:Mr. John Q. Public\, Esq.
N:Public;John;Quinlan;Mr.;Esq.
NICKNAME:Johnny
...
END:VCARD
```

```
{
...
"fullName":{ "value": "Mr. John Q. Public, Esq." },
"name":{
    "components":[
      { "value":"Mr.", "type": "prefix" },
      { "value":"John", "type": "personal" },
      { "value":"Public", "type": "surname" },
      { "value":"Quinlan", "type": "additional" },
      { "value":"Esq.", "type": "suffix" }
    ]
},
"nickNames":[
  { "value": "Johnny" }
],
...
}
```

Figure 3: FN, N, NICKNAME mapping example

2.4.3. PHOTO

A PHOTO property is represented as an entry of the "photos" map (Figure 4). The entry value is a "File" object whose "href" member is the PHOTO value.

The PREF and MEDIATYPE parameters are mapped according to the rules as defined in Section 2.1.

```
BEGIN:VCARD
VERSION:4.0
...
PHOTO:http://www.example.com/pub/photos/jqpublic.gif
...
END:VCARD
```

```
{
...
"photos":{
   ...
   "a-photo":{
     "href": "http://www.example.com/pub/photos/jqpublic.gif"
   },
   ...
},
...
}
```

Figure 4: PHOTO mapping example

2.4.4. BDAY, BIRTHPLACE, DEATHDATE, DEATHPLACE, ANNIVERSARY

The BDAY and ANNIVERSARY properties and the extensions BIRTHPLACE, DEATHDATE, DEATHPLACE described in [RFC6350] are represented as "Anniversary" objects included in the "anniversaries" map (Figure 5):

- *BDAY and BIRTHPLACE are mapped to "date" and "place" where "type" is set to "birth";
- *DEATHDATE and DEATHPLACE are mapped to "date" and "place" where "type" is set to "death";
- *ANNIVERSARY is mapped to "date" where "type" is empty and "label" is set to a value describing in detail the kind of anniversary (e.g. "marriage date" for the wedding anniversary).

Both birth and death places are represented as instances of the "Address" object.

The BIRTHPLACE and DEATHPLACE properties that are represented as geo URIs are converted into "Address" instances including only the "coordinates" member. If the URI value is not a geo URI, the place is ignored.

The ALTID and LANGUAGE parameters of both BIRTHPLACE and DEATHPLACE properties are mapped according to the rules as defined in Section 2.1.

```
BEGIN:VCARD
VERSION:4.0
...
BDAY:19531015T231000Z
BIRTHPLACE:Mail Drop: TNE QB\n123 Main Street\nAny Town, CA 91921-12
DEATHDATE:19960415
DEATHPLACE:4445 Courtright Street\nNew England, ND 58647\nU.S.A.
ANNIVERSARY:19860201
...
END:VCARD
```

```
{
...
"anniversaries": {
  "ANNIVERSARY-1" : {
    "type": "birth",
    "date": "1953-10-15T23:10:00Z",
    "place":{
      "fullAddress":{
        "value": "Mail Drop: TNE QB\n123 Main Street\nAny Town, CA 9
      }
    }
  },
  "ANNIVERSARY-2" : {
    "type": "birth",
    "date": "1953-10-15T23:10:00Z",
    "place":{
      "fullAddress":{
        "value": "4445 Courtright Street\nNew England, ND 58647\nU.S
      }
    }
  },
  "ANNIVERSARY-3" : {
    "label": "marriage date",
    "date": "1986-02-01"
  }
},
...
}
```

Figure 5: BDAY, BIRTHPLACE, DEATHDATE, DEATHPLACE, ANNIVERSARY mapping example

2.4.5. GENDER

The GENDER property is a single structured value with two optional components: the biological sex and the gender information. The former is represented as an enumerated value, while the latter as a free-form text. As opposed to such a representation, the JSContact

specification includes the "SpeakToAs" object just to represent how to address, speak to or refer to the contact. In particular, some pre-defined values are allowed for the "grammaticalGender" member.

For the reasons stated above, the GENDER property doesn't have a direct match with the "SpeakToAs" object. However, on the assumption that the GENDER property doesn't store the actual biological sex of the contact, implementations MAY use the conversion rules shown in Table 2 and Table 3.

GENDER value	"SpeakToAs.grammaticalGender" value
M	male
F	female
N	neuter
O	animate
U	SpeakToAs = null

Table 2: GENDER to SpeakToAs conversion

"SpeakToAs.grammaticalGender" value	GENDER value
male	M
female	F
neuter	N
animate	O
inanimate	N;inanimate

Table 3: SpeakToAs to GENDER conversion

2.5. Delivery Addressing Properties

2.5.1. ADR

An ADR property is represented as an entry of the "addresses" map (Figure 6). The entry value is an "Address" object.

The ADR components are transformed into the "Address" members as presented in Table 4 and Table 5.

The "street address" and "extended address" ADR components MAY be converted into either a single StreetComponent item or a combination of StreetComponent items.

ADR component	Address member
locality	locality
region	region
postal code	postcode
country name	country

Table 4: ADR components vs. Address members mapping

ADR component	Single StreetComponent item	Combination of StreetComponent items
post office box	postOfficeBox	
extended address	extension	extension, building, floor, room, apartment
street address	name	name, number, direction

Table 5: ADR components vs. StreetComponent items mapping

The LABEL parameter is converted into the "fullAddress" member.

The GEO parameter is converted into the "coordinates" member.

The TZ parameter is converted into the "timeZone" member.

The CC parameter as defined in [RFC8605] is converted into the "countryCode" member.

The PREF and TYPE parameters are mapped according to the rules as defined in Section 2.1.

The ALTID and LANGUAGE parameters are mapped according to the rules as defined in Section 2.1. Each possible language-dependent alternative is represented as an entry of the PatchObject map where the key references the "fullAddress" member.

```
BEGIN:VCARD
VERSION:4.0
...
ADR;TYPE=work;CC=US:;;54321 Oak St;Reston;VA;20190;USA
ADR;TYPE=home;CC=US:;;12345 Elm St;Reston;VA;20190;USA
...
END:VCARD
```

```
{
...
"addresses":{
  "work-address" :{
    "contexts":{ "work": true },
    "fullAddress":{
      "value": "54321 Oak St\nReston\nVA\n20190\nUSA"
    },
    "street": [
       { "name": "Oak St" },
       { "number" : "54321" }
    ],
    "locality": "Reston",
    "region": "VA",
    "country": "USA",
    "postcode": "20190",
    "countryCode": "US"
  },
  "private-address":{
    "contexts":{ "private": true },
    "fullAddress":{
      "value": "12345 Elm St\nReston\nVA\n20190\nUSA"
    },
    "street": [
       { "name": "Elm St" },
       { "number" : "12345" }
    ],
    "locality": "Reston",
    "region": "VA",
    "country": "USA",
    "postcode": "20190",
    "countryCode": "US"
  }
},
...
}
```

Figure 6: ADR mapping example

2.6. Communications Properties

2.6.1. TEL

A TEL property is represented as an entry of the "phones" map (Figure 7). The entry value is a "Phone" object. The TEL-specific values of the TYPE parameter are mapped to the "features" map keys. The values that don't match a key are represented as comma-separated values of the "label" member. The "phone" member is set to the TEL value.

The PREF and TYPE parameters are mapped according to the rules as defined in Section 2.1.

```
BEGIN:VCARD
VERSION:4.0
...
TEL;VALUE=uri;PREF=1;TYPE="voice,home":tel:+1-555-555-5555;ext=5555
TEL;VALUE=uri;TYPE=home:tel:+33-01-23-45-67
...
END:VCARD
```

```
{
...
"phones":{
  "a-phone":{
    "contexts":{ "private": true },
    "features":{ "voice": true },
    "phone": "tel:+1-555-555-5555;ext=5555",
    "pref": 1
  },
  "another-phone":{
    "contexts":{ "private": true },
    "phone": "tel:+33-01-23-45-67"
  }
],
...
}
```

Figure 7: TEL mapping example

2.6.2. EMAIL

An EMAIL property is represented as an entry of the "emails" map (Figure 8). The entry value is an "EmailAddress" object. The "email" member is set to the EMAIL value.

The PREF and TYPE parameters are mapped according to the rules as defined in Section 2.1.

```
BEGIN:VCARD
VERSION:4.0
...
EMAIL;TYPE=work:jqpublic@xyz.example.com
EMAIL;PREF=1:jane_doe@example.com
...
END:VCARD
```

```
{
...
"emails":{
  "work-email":{
    "contexts":{ "work": true },
    "email": "jqpublic@xyz.example.com"
  },
  "private-email":{
    "contexts":{ "private", true },
    "email": "jane_doe@example.com",
    "pref": 1
  }
},
...
}
```

Figure 8: EMAIL mapping example

2.6.3. IMPP

An IMPP property is represented as an entry of the "online" map (Figure 9). The entry value is a "Resource" object whose "type" member is set to "username", the "label" member is set to "XMPP" and the "resource" member is the IMPP value.

The PREF and TYPE parameters are mapped according to the rules as defined in Section 2.1.

```
BEGIN:VCARD
VERSION:4.0
...
IMPP;PREF=1:xmpp:alice@example.com
...
END:VCARD
```

```
{
...
"online":{
  ...
  {
    "type": "username",
    "label": "XMPP",
    "value": "alice@example.com"
  },
  ...
},
...
}
```

Figure 9: IMPP mapping example

2.6.4. LANG

A LANG property is represented as an entry of the "preferredContactLanguages" map (Figure 10). The entry keys correspond to the language tags, the corresponding entry values are arrays of "ContactLanguage" objects.

The PREF and TYPE parameters are mapped according to the rules as defined in Section 2.1.

If both PREF and TYPE parameters are missing, the array of "ContactLanguage" objects MUST be empty.

```
BEGIN:VCARD
VERSION:4.0
...
LANG;TYPE=work;PREF=1:en
LANG;TYPE=work;PREF=2:fr
LANG;TYPE=home:fr
...
END:VCARD
```

```
{
...
"preferredContactLanguages":{
  "en":[
    {
      "context": "work",
      "pref": 1
    }
  ],
  "fr":[
    {
      "context": "work",
      "pref": 2
    },
    {
      "context": "private"
    }
  ]
},
...
}
```

Figure 10: LANG mapping example

2.7. Geographical Properties

The GEO and TZ properties are not directly mapped to topmost Card members because the same information is represented through equivalent "Address" members.

The ALTID parameter is used for associating both GEO and TZ properties with the related address information. When the ALTID parameter is missing, the matched members SHOULD be included in the first "Address" object.

2.7.1. Time Zone Representation

As specified in Section 6.5.1 of [RFC6350], the time zone information can be represented as a time zone name, as a UTC offset or as a URI.

* If the TZ value is defined in the IANA timezone database, it is directly matched by the "timeZone" member in JSContact.

* An UTC offset MUST be converted into the related "Etc/GMT" time zone (e.g. the value "-0500" converts to "Etc/GMT+5"). If the UTC offset value contains minutes information or is not an IANA timezone name, it requires special handling.

* Since there is no URI scheme defined for time zones [uri-schemes], any implementation that does use some a custom URI for a time zone is not interoperable anyway. In this case, if the URI corresponds to an IANA time zone [time-zones], this latter SHOULD be used. Otherwise, the URI value is dumped into a string.

2.8. Organizational Properties

2.8.1. TITLE and ROLE

Both TITLE and ROLE properties are represented as entries of the "titles" map (Figure 11). The entry value is a "Title" object whose "title" member includes information about the title or role. The rules to set the "organization" member are out of the scope of this document.

The ALTID and LANGUAGE parameters are mapped according to the rules as defined in Section 2.1.

```
BEGIN:VCARD
VERSION:4.0
...
TITLE:Research Scientist
ROLE:Project Leader
...
END:VCARD
```

```
{
...
"titles":{
  "a-title":{
    "title":{ "value" : "Project Leader" }
  },
  "another-title":{
    "title":{ "value" : "Research Scientist" }
  }
},
...
}
```

Figure 11: TITLE and ROLE mapping example

2.8.2. LOGO

A LOGO property is represented as an entry of the "online" map (Figure 12). The entry value is a "Resource" object whose "type" member is set to "uri", the "label" member is set to "logo" and the "resource" member is the LOGO value.

The PREF and TYPE parameters are mapped according to the rules as defined in Section 2.1.

```
BEGIN:VCARD
VERSION:4.0
...
LOGO:http://www.example.com/pub/logos/abccorp.jpg
...
END:VCARD
```

```
{
...
"online":{
  ...
  "a-logo":{
    "type": "uri",
    "label": "logo",
    "resource": "http://www.example.com/pub/logos/abccorp.jpg"
  },
  ...
},
...
}
```

Figure 12: LOGO mapping example

2.8.3. ORG

An ORG property is represented as an entry of the "organizations" map (Figure 13). The entry value is an "Organization" object whose "name" member contains the organizational name and the "units" member contains the organizational units.

The ALTID and LANGUAGE parameters are mapped according to the rules as defined in Section 2.1.

```
BEGIN:VCARD
VERSION:4.0
...
ORG:ABC\, Inc.;North American Division;Marketing
...
END:VCARD
```

```
{
...
"organizations":{
  "an-organization":{
    "name":{ "value": "ABC, Inc." },
    "units":[
      { "value": "North American Division" },
      { "value": "Marketing" }
    ]
  }
},
...
}
```

Figure 13: ORG mapping example

2.8.4. MEMBER

According to the JSContact specification, a group of contact cards is represented through a CardGroup (Figure 14). The uids of the contact cards composing the group are included in the "members" map.

In this case, the PREF parameter has not a JSContact counterpart; however, the implementers MAY insert the map entries by order of preference.

```
BEGIN:VCARD
VERSION:4.0
KIND:group
FN:The Doe family
MEMBER:urn:uuid:03a0e51f-d1aa-4385-8a53-e29025acd8af
MEMBER:urn:uuid:b8767877-b4a1-4c70-9acc-505d3819e519
END:VCARD
```

```
{
  "kind": "group",
  "fullName":{ "value": "The Doe family" },
  "uid": "urn:uuid:ab4310aa-fa43-11e9-8f0b-362b9e155667",
  "members":{
    "urn:uuid:03a0e51f-d1aa-4385-8a53-e29025acd8af": true,
    "urn:uuid:b8767877-b4a1-4c70-9acc-505d3819e519": true
  }
}
```

Figure 14: Group example

Only if the GROUP contains properties that don't have a mapping to CardGroup properties, then the CardGroup.card property MAY contain the optional Card object of this group.

```
{
  "name": "The Doe family",
  "uid": "urn:uuid:ab4310aa-fa43-11e9-8f0b-362b9e155667",
  "members":{
     "urn:uuid:03a0e51f-d1aa-4385-8a53-e29025acd8af": true,
     "urn:uuid:b8767877-b4a1-4c70-9acc-505d3819e519": true
  },
  "card": {
     "fullName":{ "value": "The Doe family" },
     "uid": "urn:uuid:ab4310aa-fa43-11e9-8f0b-362b9e155667",
     "photos":{
       "a-photo":{
         "href": "http://www.example.com/pub/photos/jqpublic.gif"
       }
     }
  }
}
```

Figure 15: card member of CardGroup object

2.8.5. RELATED

All the RELATED instances are converted into the "relatedTo" map (Figure 16): an entry for each entity the entity described by the

Card is associated with. The map keys are the "uid" values of the associated cards.

Each map value is a "Relation" object including only the "relation" member represented as a set of the RELATED-specific values of the TYPE parameter as defined in Section 6.6.6 of [RFC6350].

If the relation type is unspecified, the "relation" member MUST be empty.

```
BEGIN:VCARD
VERSION:4.0
...
RELATED;TYPE=friend:urn:uuid:f81d4fae-7dec-11d0-a765-00a0c91e6bf6
RELATED;TYPE=contact:http://example.com/directory/jdoe.vcf
RELATED;VALUE=text:Please contact my assistant Jane Doe for any inqu
...
END:VCARD
```

```
{
...
"relatedTo":{
  {
    "urn:uuid:f81d4fae-7dec-11d0-a765-00a0c91e6bf6":{
      "relation":{ "friend": true }
    }
  },
  {
    "http://example.com/directory/jdoe.vcf":{
      "relation":{ "contact": true }
    }
  },
  {
    "Please contact my assistant Jane Doe for any inquiries.":{
      "relation":{ }
    }
  }
},
...
}
```

Figure 16: RELATED mapping example

2.8.6. CONTACT-URI

A CONTACT-URI property as defined in [RFC8605] is represented as an entry of the "online" map (Figure 17). The entry value is a "Resource" object whose "type" member is set to "uri", the "label"

member is set to "contact-uri" and the "resource" member is the CONTACT-URI value.

The PREF and TYPE parameters are mapped according to the rules as defined in Section 2.1.

```
BEGIN:VCARD
VERSION:4.0
...
CONTACT-URI;PREF=1:mailto:contact@example.com
...
END:VCARD
```

```
{
...
"online":{
  ...
  "a-contact-uri":{
    "type": "uri",
    "label": "contact-uri",
    "resource": "mailto:contact@example.com",
    "pref": 1
  },
  ...
},
...
}
```

Figure 17: CONTACT-URI mapping example

2.9. Personal Information Properties

The LEVEL parameter as defined in [RFC6715] is directly mapped to the "level" property of the "PersonalInformation" type apart from when LEVEL is used in the EXPERTISE property; in this case, the values are converted as in the following:

* "beginner" is converted into "low";
* "average" is converted into "medium";
* "expert" is converted into "high".

2.9.1. EXPERTISE

An EXPERTISE property as defined in [RFC6715] is represented as a "PersonalInformation" object in the "personalInfo" map (Figure 18). The "type" member is set to "expertise".

The INDEX parameter is represented as the index of the expertise among the declared expertises.

```
BEGIN:VCARD
VERSION:4.0
...
EXPERTISE;LEVEL=beginner;INDEX=2:chinese literature
EXPERTISE;INDEX=1;LEVEL=expert:chemistry
...
END:VCARD
```

```
{
...
"personalInfo":{
  ...
  "PERSINFO-1" : {
    "type": "expertise",
    "value": "chemistry",
    "level": "high"
  },
  "PERSINFO-2" : {
    "type": "expertise",
    "value": "chinese literature",
     "level": "low"
  },
  ...
},
...
}
```

Figure 18: EXPERTISE mapping example

2.9.2. HOBBY

A HOBBY property as defined in [RFC6715] is represented as a "PersonalInformation" object in the "personalInfo" map (Figure 19). The "type" member is set to "hobby".

The INDEX parameter is represented as the index of the hobby among the declared hobbies.

```
BEGIN:VCARD
VERSION:4.0
...
HOBBY;INDEX=1;LEVEL=high:reading
HOBBY;INDEX=2;LEVEL=high:sewing
...
END:VCARD
```

```
{
...
"personalInfo":{
  ...
  "PERSINFO-1" : {
    "type": "hobby",
    "value": "reading",
    "level": "high"
  },
  "PERSINFO-2" : {
    "type": "hobby",
    "value": "sewing",
    "level": "high"
  },
  ...
},
...
}
```

Figure 19: HOBBY mapping example

2.9.3. INTEREST

An INTEREST property as defined in [RFC6715] is represented as a "PersonalInformation" object in the "personalInfo" map (Figure 20). The "type" member is set to "interest".

The INDEX parameter is represented as the index of the interest among the declared interests.

```
BEGIN:VCARD
VERSION:4.0
...
INTEREST;INDEX=1;LEVEL=medium:r&b music
INTEREST;INDEX=2;LEVEL=high:rock ’n’ roll music
...
END:VCARD
```

```
{
...
"personalInfo":{
  ...
  "PERSINFO-1" : {
    "type": "interest",
    "value": "r&b music",
    "level": "medium"
  },
  "PERSINFO-2" : {
    "type": "interest",
    "value": "rock ’n’ roll music",
    "level": "high"
  },
  ...
},
...
}
```

Figure 20: INTEREST mapping example

2.9.4. ORG-DIRECTORY

An ORG-DIRECTORY property is represented as an entry of the "online" map (Figure 21). The entry value is a "Resource" object whose "type" member is set to "uri", the "label" member is set to "org-directory" and the "resource" member is the ORG-DIRECTORY value.

The PREF and TYPE parameters are mapped according to the rules as defined in Section 2.1.

The INDEX parameter is represented as the index of the directory among the online resources with the "org-directory" label.

```
BEGIN:VCARD
VERSION:4.0
...
ORG-DIRECTORY;INDEX=1:http://directory.mycompany.example.com
ORG-DIRECTORY;PREF=1:ldap://ldap.tech.example/o=Example%20Tech,ou=En
...
END:VCARD
```

```
{
...
"online":{
  ...
  "an-org-directory":{
    "type": "uri",
    "label": "org-directory",
    "resource": "http://directory.mycompany.example.com"
  },
  "another-org-directory":{
    "type": "uri",
    "label": "org-directory",
    "resource": "ldap://ldap.tech.example/o=Example%20Tech,ou=Engine
    "pref": 1
  },
  ...
},
...
}
```

Figure 21: ORG-DIRECTORY mapping example

2.10. Explanatory Properties

2.10.1. CATEGORIES

A CATEGORIES property is converted into a set of entries of the "categories" map (Figure 22). The keys are the comma-separated text values of the CATEGORIES property.

In this case, the PREF parameter has not a JSContact counterpart; however, implementers MAY use a map preserving the order of insertion and the map entries can be inserted by order of preference.

```
BEGIN:VCARD
VERSION:4.0
...
CATEGORIES:INTERNET,IETF,INDUSTRY,INFORMATION TECHNOLOGY
...
END:VCARD
```

```
{
...
"categories":{
  "INTERNET": true,
  "IETF": true,
  "INDUSTRY": true,
  "INFORMATION TECHNOLOGY": true
},
...
}
```

Figure 22: CATEGORIES mapping example

2.10.2. NOTE

A NOTE property is mapped to the "notes" property (Figure 23). All the NOTE instances are condensed into a single note and separated by newline.

The ALTID and LANGUAGE parameters are mapped according to the rules as defined in Section 2.1.

```
BEGIN:VCARD
VERSION:4.0
...
NOTE:This fax number is operational 0800 to 1715 EST\, Mon-Fri.
...
END:VCARD
```

```
{
...
"notes": {
    "value": "This fax number is operational 0800 to 1715 EST, Mon-F
},
...
}
```

Figure 23: NOTE mapping example

2.10.3. PRODID

The PRODID property is converted into the "prodId" member (Figure 24).

```
BEGIN:VCARD
VERSION:4.0
...
PRODID:-//ONLINE DIRECTORY//NONSGML Version 1//EN
...
END:VCARD
```

```
{
...
"prodId": "-//ONLINE DIRECTORY//NONSGML Version 1//EN",
...
}
```

Figure 24: PRODID mapping example

2.10.4. REV

The REV property is transformed into the "updated" member (Figure 25).

```
BEGIN:VCARD
VERSION:4.0
...
REV:19951031T222710Z
...
END:VCARD
```

```
{
...
"updated": "1995-10-31T22:27:10Z",
...
}
```

Figure 25: REV mapping example

2.10.5. SOUND

A SOUND property is represented as an entry of the "online" map (Figure 26). The entry value is a "Resource" object whose "type" member is set to "uri", the "label" member is set to "sound" and the "resource" member is the SOUND value.

The PREF and TYPE parameters are mapped according to the rules as defined in Section 2.1.

```
BEGIN:VCARD
VERSION:4.0
...
SOUND:CID:JOHNQPUBLIC.part8.19960229T080000.xyzMail@example.com
...
END:VCARD
```

```
{
...
"online":{
  ...
  "a-sound":{
    "type": "uri",
    "label": "sound",
    "resource": "CID:JOHNQPUBLIC.part8.19960229T080000.xyzMail@examp
  },
  ...
},
...
}
```

Figure 26: SOUND mapping example

2.10.6. UID

The UID property corresponds to the "uid" property (Figure 27) in both Card and CardGroup.

```
BEGIN:VCARD
VERSION:4.0
...
UID:urn:uuid:f81d4fae-7dec-11d0-a765-00a0c91e6bf6
...
END:VCARD
```

```
{
...
"uid": "urn:uuid:f81d4fae-7dec-11d0-a765-00a0c91e6bf6",
...
}
```

Figure 27: UID mapping example

2.10.7. CLIENTPIDMAP and PID Parameter

The CLIENTPIDMAP property and the PDI parameter don't have a direct match with a Card feature.

2.10.8. URL

An URL property is represented as an entry of the "online" map (Figure 28). The entry value is a "Resource" object whose "type" member is set to "uri", the "label" member is set to "url" and the "resource" member is the URL value.

The PREF and TYPE parameters are mapped according to the rules as defined in Section 2.1.

```
BEGIN:VCARD
VERSION:4.0
...
URL:http://example.org/restaurant.french/~chezchic.html
...
END:VCARD
```

```
{
...
"online":{
  ...
  "an-url":{
    "type": "uri",
    "label": "url",
    "resource": "http://example.org/restaurant.french/~chezchic.html
  },
  ...
},
...
}
```

Figure 28: URL mapping example

2.10.9. VERSION

The VERSION property doesn't have a direct match with a JSContact feature.

2.11. Security Properties

2.11.1. KEY

A KEY property is represented as an entry of the "online" map (Figure 29). The entry value is a "Resource" object whose "type"

member is set to "uri", the "label" member is set to "key" and the "resource" member is the KEY value.

The PREF and TYPE parameters are mapped according to the rules as defined in Section 2.1.

```
BEGIN:VCARD
VERSION:4.0
...
KEY:http://www.example.com/keys/jdoe.cer
...
END:VCARD
```

```
{
...
"online":{
  ...
  "a-key":{
    "type": "uri",
    "label": "key",
    "resource": "http://www.example.com/keys/jdoe.cer"
  },
  ...
},
...
}
```

Figure 29: KEY mapping example

2.12. Calendar Properties

2.12.1. FBURL

An FBURL property is represented as an entry of the "online" map (Figure 30). The entry value is a "Resource" object whose "type" member is set to "uri", the "label" member is set to "fburl" and the "resource" member is the FBURL value.

The PREF and TYPE parameters are mapped according to the rules as defined in Section 2.1.

```
BEGIN:VCARD
VERSION:4.0
...
FBURL;PREF=1:http://www.example.com/busy/janedoe
FBURL;MEDIATYPE=text/calendar:ftp://example.com/busy/project-a.ifb
...
END:VCARD
```

```
{
...
"online":{
  ...
  "an-fburl":{
    "type": "uri",
    "label": "fburl",
    "resource": "http://www.example.com/busy/janedoe",
    "pref": 1
  },
  "another-fburl":{
    "type": "uri",
    "label": "fburl",
    "resource": "ftp://example.com/busy/project-a.ifb",
    "mediaType": "text/calendar"
  },
  ...
},
...
}
```

Figure 30: FBURL mapping example

2.12.2. CALADRURI

A CALADRURI property is represented as an entry of the "online" map (Figure 31). The entry value is a "Resource" object whose "type" member is set to "uri", the "label" member is set to "caladruri" and the "resource" member is the CALADRURI value.

The PREF and TYPE parameters are mapped according to the rules as defined in Section 2.1.

```
BEGIN:VCARD
VERSION:4.0
...
CALADRURI;PREF=1:mailto:janedoe@example.com
CALADRURI:http://example.com/calendar/jdoe
...
END:VCARD
```

```
{
...
"online":{
  ...
  "a-caladruri":{
    "type": "uri",
    "label": "caladruri",
    "resource": "mailto:janedoe@example.com",
    "pref": 1
  },
  "another-caladruri":{
    "type": "uri",
    "label": "caladruri",
    "resource": "http://example.com/calendar/jdoe"
  },
  ...
},
...
}
```

Figure 31: CALADRURI mapping example

2.12.3. CALURI

A CALURI property is represented as an entry of the "online" map (Figure 32). The entry value is a "Resource" object whose "type" member is set to "uri", the "label" member is set to "caluri" and the "resource" member is the CALURI value.

The PREF and TYPE parameters are mapped according to the rules as defined in Section 2.1.

```
BEGIN:VCARD
VERSION:4.0
...
CALURI;PREF=1:http://cal.example.com/calA
CALURI;MEDIATYPE=text/calendar:ftp://ftp.example.com/calA.ics
...
END:VCARD
```

```
{
...
"online":{
  ...
  "a-caluri":{
    "type": "uri",
    "label": "caluri",
    "resource": "http://cal.example.com/calA",
    "pref": 1
  },
  "another-caluri":{
    "type": "uri",
    "label": "caluri",
    "resource": "ftp://ftp.example.com/calA.ics",
    "mediaType": "text/calendar"
  },
  ...
},
...
}
```

Figure 32: CALURI mapping example

2.13. vCard Unmatched Properties

The unmatched vCard properties MAY be converted into JSContact properties whose name contains the prefix "ietf.org:rfc6350:" followed by property name in uppercase (i.e. ietf.org:rfc6350:CLIENTPIDMAP").

2.14. Card Required Properties

While converting a vCard into a Card/CardGroup, only the topmost "uid" member is mandatory. Implementers are REQUIRED to set it when it is missing.

3. Translating JSContact properties to vCard

In most of the cases, the rules about the translation from Card/CardGroup to vCard can be derived by reversing the rules presented in Section 2. The remaining cases are treated in the following of this section.

3.1. Id

Where a map key is of type Id, implementers are free to either ignore it or preserve it as a vCard information (e.g. a vCard parameter).

3.2. Localizations

Each PatchObject entry value of each "localizations" entry is converted into a instance of the vCard property matching the JSContact member referenced by the PatchObject entry key. The LANGUAGE parameter of such alternative MUST be set to the value of the given "localizations" entry. The LANGUAGE parameter of a vCard property presenting, at least, a language-dependent alternative MUST be set to the value of the JSContact "language" property if it is valued. Implementers MAY set the ALTID parameter to group language-based alternatives of the same value.

Note also that the components of some vCard values and their language-dependent alternatives are split into different JSContact values. For example, the "name" and "units" values for a given language must be grouped to make a single ORG value where components are separated by ";".

3.3. Date and Time Representations

The JSContact spec defines the "UTCDateTime" type to represent [RFC3339] "date-time" format with further restrictions. This means that the matched vCard format for a "UTCDateTime" value MUST be one of the formats shown in Section 4.3.5 of [RFC6350] (i.e. "19961022T140000Z").

In addition to such format, the "date" member of the "Anniversary" type allows to specify both dates without the time and partial dates. In such cases, the corresponding vCard format is that defined in Section 4.3.1.

3.4. Time Zone

The time zone name as represented by the "timeZone" property is mapped to the TZ parameter.

Implementers MAY map an "Etc/GMT" time zone either preserving the time zone name or converting it into a UTC offset.

3.5. JSContact Types Matching Multiple vCard Properties

3.5.1. Title

The "titles" property contains information about the job, the position or the role of the entity the card represents. In vCard, such information is split into the TITLE and the ROLE properties. This specification defines TITLE as the default target property when converting the "titles" property.

3.5.2. Resource

The "online" property includes resources that are usually represented through different vCard properties. The matched vCard property of a "Resource" object can be derived from the value of its "label" member.

Any resource included in the "online" map that doesn't match a vCard property MAY be converted into a vCard extended property.

3.6. CardGroup

A CardGroup object is converted into a vCard by merging its properties with the properties of "CardGroup.card" object. If the "CardGroup.card.fullName" property exists, it MUST be used to set the FN value.

3.7. Card Unmatched Properties

Both the "preferredContactMethod" and "created" members don't match any vCard property. Implementers MAY represent them as vCard extended properties.

3.8. vCard Required Properties

While converting a Card/CardGroup into a vCard, only the FN property is required. Since both the "Card.fullName" and "CardGroup.name" properties are optional, implementers are REQUIRED to generate an FN value when it is missing.

4. IANA Considerations

This document has no actions for IANA.

5. Implementation Status

NOTE: Please remove this section and the reference to RFC 7942 prior to publication as an RFC.

This section records the status of known implementations of the protocol as defined in this specification at the time of posting of this Internet-Draft, and is based on a proposal described in [RFC7942]. The description of implementations in this section is intended to assist the IETF in its decision processes in progressing drafts to RFCs. Please note that the listing of any individual implementation here does not imply endorsement by the IETF. Furthermore, no effort has been spent to verify the information presented here that was supplied by IETF contributors. This is not intended as, and must not be construed to be, a catalog of available implementations or their features. Readers are advised to note that other implementations may exist.

According to RFC 7942, "this will allow reviewers and working groups to assign due considerationto documents that have the benefit of running code, which may serve as evidence of valuable experimentation and feedback that have made the implemented protocols more mature. It is up to the individual working groups to use this information as they see fit".

5.1. CNR

- Responsible Organization: National Research Council (CNR) of Italy
- Location: https://github.com/consiglionazionaledellericerche/jscontact-tools
- Description: This implementation includes tools for JSContact creation, validation, serialization/deserialization, and conversion from vCard, xCard and jCard.
- Level of Maturity: This is an "alpha" test implementation.
- Coverage: This implementation includes all of the features described in this specification.
- Contact Information: Mario Loffredo, mario.loffredo@iit.cnr.it

6. Security Considerations

This document doesn't present any security consideration.

7. References

7.1. Normative References

[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>.

[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>.

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

[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>.

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

[RFC7942] Sheffer, Y. and A. Farrel, "Improving Awareness of Running Code: The Implementation Status Section", BCP 205, RFC 7942, DOI 10.17487/RFC7942, July 2016, <https://www.rfc-editor.org/info/rfc7942>.

[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>.

7.2. Informative References

[draft-ietf-jmap-jscontact] "JSContact: A JSON representation of contact data", <https://datatracker.ietf.org/doc/draft-ietf-jmap-jscontact/>.

[time-zones] "Time Zone Database", <https://www.iana.org/time-zones>.

[uri-schemes] "Uniform Resource Identifier (URI) Schemes", <https://www.iana.org/assignments/uri-schemes/uri-schemes.xhtml>.

Authors' Addresses

Mario Loffredo
IIT-CNR/Registro.it
Via Moruzzi,1
56124 Pisa
Italy

Email: mario.loffredo@iit.cnr.it
URI: http://www.iit.cnr.it

Robert Stepanek
FastMail
PO Box 234, Collins St West
Melbourne VIC 8007
Australia

Email: rsto@fastmailteam.com
URI: https://www.fastmail.com