Workgroup: calext
Internet-Draft:
draft-ietf-calext-jscontact-vcard-05
Published: 10 January 2023
Intended Status: Standards Track
Expires: 14 July 2023
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 defined in the 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 14 July 2023.



## Table of Contents

## 1. Introduction

### 1.1. Motivation

The JSContact data model and format [I-D.ietf-calext-jscontact] aims to be an alternative to the widely used vCard [RFC6350] standard and jCard [RFC7095].

While applications might prefer JSContact to exchange contact card data with other systems, they are likely to interoperate with services and clients that only support vCard or jCard. Similarly, existing contact data providers and consumers already using vCard or jCard might want to represent their contact data also in JSContact.

To achieve this, this document defines standard rules to convert contact data between JSContact and vCard (and consequently jCard).

### 1.2. Notational Conventions

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 BCP 14 [RFC2119] [RFC8174] when, and only when, they appear in all capitals, as shown here.

### 1.3. ABNF Notations

The ABNF definitions in this document use the notations of [RFC5234]. ABNF rules not defined in this document either are defined in [RFC5234] (such as the ABNF for CRLF, WSP, DQUOTE, VCHAR, ALPHA, and DIGIT) or [RFC6350].

## 2. Converting vCard to JSContact

This section contains the conversion rules from vCard to JSContact Card. Its follows the same structure as the vCard v4 RFC document [RFC6350]. Properties and parameters of vCard extension RFCs, including the vCard JSContact Extension RFC [I-D.ietf-calext-vcard-jscontact-extensions] are added to appropriate subsections.

## 2.1. General rules

### 2.1.1. The uid property

The UID property in vCard is optional, but the uid property in JSContact is mandatory. Implementations that convert a vCard without UID property **MUST** generate a unique identifier as a value for the uid property. They **SHOULD** generate the same identifier when repeatedly converting the same vCard, but they **MAY** generate different values. Consequently, a vCard without UID property **MAY** not convert to one exact instance of a JSContact card.

### 2.1.2. Choosing identifiers

Multi-valued properties in JSContact typically are represented as a JSON object where the object keys are of the Id type and the object values are the converted vCard property. In absence of the PROP-ID parameter (see Section 2.3.15), implementations are free to choose any identifier for such entries. Whatever identifier generation scheme implementations use, they **SHOULD** generate values of short character length. For example, this could be an incrementing number across all Ids or only unique within one JSON object.

## 2.2. vCard Value Data Types

### 2.2.1. BOOLEAN

This converts to the JSContact Boolean type.

### 2.2.2. DATE, TIME, DATE-TIME, DATE-AND-OR-TIME, and TIMESTAMP

The TIMESTAMP type generally converts to the UTCDateTime. It converts to the Timestamp type for anniversaries.

The DATE type converts to the PartialDate type, which only is relevant for anniversaries. This does not apply to DATE values that only define a month or a day.

The TIME, DATE-TIME, DATE-AND-OR-TIME types and DATE type values that only define a month or day do not convert to any JSContact type. vCard properties or parameters having such values **MAY** convert as defined in Section 2.16.

### 2.2.3. INTEGER

This converts to the JSContact Int and UnsignedInt types.

### 2.2.4. FLOAT

This converts to the JSContact Number type.

### 2.2.5. LANGUAGE-TAG

This converts to the JSContact String type.

#### 2.2.6. TEXT

This converts to the JSContact String type.

#### 2.2.7. URI

This converts to the JSContact String type.

#### 2.2.8. UTC-OFFSET

This either converts to a String containing an IANA TimeZone Database entry name (see Section 2.9.2), or it does not convert to any JSContact type. For the latter, vCard properties or parameters having such values **MAY** convert to JSContact as defined in Section 2.16.

### 2.3. vCard Parameters

#### 2.3.1. ALTID

This does not convert to an IANA-registered property in JSContact, but several conversion rules make use of this parameter to combine multiple vCard properties into a single JSContact object instance. For an example of this see Section 2.7.1. To preserve the verbatim value of the ALTID parameter, set the JSContact extension properties props or params defined in Section 2.16.

#### 2.3.2. AUTHOR

The AUTHOR parameter value of a vCard NOTE property converts to the uri property of the Author object for this note. To convert this parameter for other vCard properties, see Section 2.16.2.

Note: This parameter is defined in [I-D.ietf-calext-vcard-jscontact-extensions].

#### 2.3.3. AUTHOR-NAME

The AUTHOR-NAME parameter value of a vCard NOTE property converts to the name property of the Author object for this note. To convert this parameter for other vCard properties, see Section 2.16.2.

Note: This parameter is defined in [I-D.ietf-calext-vcard-jscontact-extensions].

#### 2.3.4. CALSCALE

This parameter set on a BDAY, DEATHDATE or ANNIVERSARY property converts to the calendarScale property of the PartialDate object type. To convert this parameter for other vCard properties, see Section 2.16.2.

#### 2.3.5. CREATED

The CREATED parameter value of a vCard NOTE property converts to the created property of the Note object. To convert this parameter for other vCard properties, see Section 2.16.2.

Note: This parameter is defined in [I-D.ietf-calext-vcard-jscontact-extensions].

#### 2.3.6. DERIVED

If this parameter is set to true on a vCard property, then implementations **MAY** choose to not convert that property. Note: This parameter is defined in [I-D.ietf-calext-vcard-jscontact-extensions].

#### 2.3.7. GEO

This parameter set on an ADR property converts to the JSContact coordinates property of the Address object that represents the vCard ADR. To convert this parameter for other vCard properties, see Section 2.16.2.

#### 2.3.8. GROUP

This parameter exclusively is for use in jCard (see Section 7.1 of [RFC7095]). It **MUST NOT** be set in a vCard. Preserving the exact group name when converting from vCard to JSContact and back to vCard is not necessary. Any group identifiers will do, as long as the resulting vCard groups its properties equally to the original vCard. Implementations that still wish to preserve the exact property group name of a vCard property **MAY** set the group parameter in the JSContact extension properties props or params defined in Section 2.16.

```
item1.TEL;VALUE=uri:tel:+1-555-555-5555
```

```
"phones": {
  "p1": {
    "@type": "Phone",
    "number": "tel:+1-555-555-5555",
    "vCardParams" : {
     "group" : "item1"
    }
  }
}
```

Figure 1: An example how to preserve the group name in params during conversion.

```
item2.X-FOO:bar
```

```
"vCardProps": [
  ["x-foo", {
    "group": "item2"
  }, "unknown", "bar"]
]
```

Figure 2: An example how to preserve the group name in props during conversion.

#### 2.3.9. INDEX

This parameter set on the EXPERTISE, HOBBY, INTEREST and ORG-DIRECTORY properties defined in [RFC6715] converts to the JSContact listAs property of the PersonalInfo and DirectoryResource objects. To convert this parameter for other vCard properties, see Section 2.16.2.

#### 2.3.10. LANGUAGE

This converts to an entry in the localizations property for the vCard property that this parameter is set on. The value of the LANGUAGE parameter defines the language tag key in the localizations property.

This specification does not define a single standard conversion rule for how to convert the property values. Instead, building the localizations value is implementation-specific.

Two options to populate the localizations property are:

* One Patch Per Property: For each vCard property with a LANGUAGE parameter, set the complete path in the PatchObject to the JSContact property that the vCard property converts to. The value of the patch is the converted property value. This is simple to process and adequate if the vCard only contains a few properties with LANGUAGE parameter.

* Bundle Patches by Parent: If a PatchObject contains multiple paths that have the same parent paths, then it might be possible to combine these patches into one patch that patches the parent property. This is possible if the property in the Card is patched in its entirety.

A non-localized Card **SHOULD NOT** provide less information than any of its localizations.

As a general rule, an implementation **SHOULD** choose those vCard properties to convert to the non-localized card, that either all have the same LANGUAGE parameter value or none at all. Figure 3 illustrates this.

```
FN;LANGUAGE=EN:John Doe
TITLE;LANGUAGE=EN:Boss
TITLE;LANGUAGE=fr:Patron

"locale": "en",
"fullName": "John Doe",
"titles": {
  "t1": {
    "@type": "Title",
    "title": "Boss"
  }
},
"localizations": {
  "fr": {
    "titles/t1/title": "Patron"
  }
}
```

Figure 3: LANGUAGE conversion example: one dominant language

As a special case, if one or more vCard properties of the same type do not have the LANGUAGE parameter set, then choose them to the non-localized Card. Convert any with LANGUAGE parameters to the localizations property.Figure 4 illustrates this.

```
FN:John Doe
TITLE:Boss
TITLE;LANGUAGE=fr:Patron

"titles": {
  "t1": {
    "@type": "Title",
    "title": "Boss"
  }
},
"localizations": {
  "fr": {
    "titles/t1/title": "Patron"
  }
}
```

Figure 4: LANGUAGE conversion example: property without language

As the least preferred option, Figure 5 illustrates how all vCard properties of the same type have the LANGUAGE parameter set, but none of their language tags match the locale of the main Card. In this case, implementations **MAY** choose to add the localized vCard properties only to the localizations object. Implementations **SHOULD** avoid this scenario as much as possible.

```
LOCALE:es
FN:Gabriel García Márquez
TITLE;LANGUAGE=en:Boss
TITLE;LANGUAGE=fr:Patron
```

```
"locale": "es",
"fullName": "Gabriel García Márquez",
"localizations": {
  "en": {
    "titles": {
      "t1": {
        "@type": "Title",
        "title": "Novelist"
      }
  },
  "fr": {
    "titles": {
      "t1": {
        "@type": "Title",
        "title": "Écrivain"
      }
  }
}
```

Figure 5: LANGUAGE conversion example: conflicting locale and language

#### 2.3.11. LEVEL

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

- "beginner" converts to "low";
- "average" converts to "medium";
- "expert" converts to "high".

To convert this parameter for other vCard properties, see Section 2.16.2.

#### 2.3.12. MEDIATYPE

This converts to the mediaType property of the Resource object type. To convert this parameter for other vCard properties, see Section 2.16.2.

#### 2.3.13. PID

This does not convert to an IANA-registered property in JSContact. To convert this parameter, see Section 2.16.2.

#### 2.3.14. PREF

This converts to the pref property.

### 2.3.15. PROP-ID

The PROP-ID parameter value of a vCard property converts to the Id of the JSContact property to which the vCard property converts to.

```
TEL;PROP-ID=PHONE-A;VALUE=uri;PREF=1;TYPE="voice,home"
  :tel:+1-555-555-5555;ext=5555
TEL;PROP-ID=PHONE-B;VALUE=uri;TYPE=home
  :tel:+33-01-23-45-67
```

```
"phones": {
  "PHONE-A": {
    "@type": "Phone",
    "contexts": { "private": true },
    "features": { "voice": true },
    "number": "tel:+1-555-555-5555;ext=5555",
    "pref": 1
  },
  "PHONE-B": {
    "@type": "Phone",
    "contexts": { "private": true },
    "number": "tel:+33-01-23-45-67"
  }
}
```

Figure 6: PROP-ID conversion example

Note: This parameter is defined in [I-D.ietf-calext-vcard-jscontact-extensions].

### 2.3.16. RANKS

The values of this parameter convert to the rank property of the NameComponent objects inferred from the N property value. To convert this parameter for other vCard properties, see Section 2.16.2. Note: This parameter is defined in [I-D.ietf-calext-vcard-jscontact-extensions].

### 2.3.17. SERVICE-TYPE

This converts to the service property of the OnlineService object type. To convert this parameter for other vCard properties, see Section 2.16.2. Note: This parameter is defined in [I-D.ietf-calext-vcard-jscontact-extensions].

### 2.3.18. SORT-AS

This converts to the sortAs properties defined for the Name, Organization and OrgUnit object types. To convert this parameter for other vCard properties, see Section 2.16.2.

#### 2.3.19. TYPE

This converts to the contexts property as well as property-specific type property values defined in later sections.

#### 2.3.20. TZ

This parameter set on an ADR property converts to the JSContact timeZone property of the Address object that represents the vCard ADR. Also see the conversion of the TZ property in Section 2.9.2. To convert this parameter for other vCard properties, see Section 2.16.2.

### 2.4. VALUE

This does not convert to an IANA-registered property in JSContact. To preserve properties with experimental values, see Section 2.16.2 and Section 2.16.1.

### 2.5. General Properties

#### 2.5.1. BEGIN and END

These do not convert to IANA-registered properties in JSContact.

#### 2.5.2. KIND

The KIND property converts to the kind property (Figure 7). Allowed values are those described in Section 6.1.4 of [RFC6350] and extended with the values declared in [RFC6473] and [RFC6869].

```
KIND:individual
```

```
"kind": "individual"
```

Figure 7: KIND conversion example

#### 2.5.3. SOURCE

A SOURCE property converts to an entry in the directories property (Figure 8). The entry value is a DirectoryResource object whose type property is set to entry and uri property is set to the SOURCE value.

The PREF and MEDIATYPE parameters convert according to the rules as defined in Section 2.3.

```
SOURCE:http://dir.example.com/addrbook/jdoe/Jean%20Dupont.vcf
```

```
"directories": {
   "ENTRY-1": {
     "@type": "DirectoryResource",
     "type": "entry",
     "uri": "http://dir.example.com/addrbook/jdoe/Jean%20Dupont.vcf"
   }
}
```

Figure 8: SOURCE conversion example

#### 2.5.4. XML

This does not convert to an IANA-registered property in JSContact.

### 2.6. Identification Properties

#### 2.6.1. BDAY, BIRTHPLACE, DEATHDATE, DEATHPLACE, ANNIVERSARY

The BDAY and ANNIVERSARY properties and the extensions BIRTHPLACE, DEATHDATE, DEATHPLACE described in [RFC6474] are represented as Anniversary objects included in the anniversaries property (Figure 9):

- *BDAY and BIRTHPLACE convert to date and place where type is set to "birth";
- *DEATHDATE and DEATHPLACE convert to date and place where type is set to "death";
- *ANNIVERSARY converts to date where type is "wedding".

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

The BIRTHPLACE and DEATHPLACE properties that are represented as geo URIs convert to Address instances including only the coordinates property. If the URI value is not a geo URI, the place is ignored.

The ALTID and LANGUAGE parameters of both BIRTHPLACE and DEATHPLACE properties convert according to the rules as defined in Section 2.3.

```
BDAY:19531015T231000Z
BIRTHPLACE:Mail Drop: TNE QB\n123 Main Street\nAny Town, CA 91921-1234\n
DEATHDATE:19960415
DEATHPLACE:4445 Courtright Street\nNew England, ND 58647\nU.S.A.
ANNIVERSARY:19860201
```

```
"anniversaries": {
  "ANNIVERSARY-1" : {
    "@type": "Anniversary",
    "type": "birth",
    "date": {
      "@type": "Timestamp",
      "utc": "1953-10-15T23:10:00Z"
    },
    "place": {
      "@type": "Address",
      "fullAddress": "Mail Drop: TNE QB\n123 Main Street\nAny Town, CA 9
    }
  },
  "ANNIVERSARY-2" : {
    "@type": "Anniversary",
    "type": "death",
    "date": {
      "@type": "PartialDate",
      "year": 1996,
      "month": 4,
      "year": 15
    },
    "place": {
      "@type": "Address",
      "fullAddress": "4445 Courtright Street\nNew England, ND 58647\nU.S
    }
  },
  "ANNIVERSARY-3" : {
    "@type": "Anniversary",
    "type": "wedding",
    "date": {
      "@type": "PartialDate",
      "year": 1986,
      "month": 2,
      "day": 1
    }
  }
}
```

Figure 9: BDAY, BIRTHPLACE, DEATHDATE, DEATHPLACE, ANNIVERSARY conversion example

### 2.6.2. GENDER

This does not map to an IANA-registered property in JSContact. To convert this property, see Section 2.16.1. Note the alternative JSContact speakToAs property which defines how to address and refer to an individual represented by the card, as do the newly defined vCard GRAMMATICAL-GENDER and PRONOUNS properties of [I-D.ietf-calext-vcard-jscontact-extensions].

### 2.6.3. GRAMMATICAL-GENDER and PRONOUNS

The GRAMMATICAL-GENDER property converts to the grammaticalGender property of the SpeakToAs object (Figure 10).

The PRONOUNS property converts to an entry in the pronouns property of the SpeakToAs object (Figure 10).

```
GRAMMATICAL-GENDER:NEUTER
PRONOUNS;PREF=2:they/them
PRONOUNS;PREF=1:xe/xir

"speakToAs": {
  "grammaticalGender": "neuter",
  "pronouns": {
    "PRONOUNS-1": {
      "@type": "Pronouns",
      "pronouns": "they/them",
      "pref": 2
    },
    "PRONOUNS-2": {
      "@type": "Pronouns",
      "pronouns": "xe/xir",
      "pref": 1
    }
  }
}
```

Figure 10: GRAMMATICAL-GENDER and PRONOUNS conversion example

### 2.6.4. FN

All the FN instances are represented through the fullName property (Figure 11). The presence of multiple instances is implicitly associated with the full name translation in various languages regardless of the presence of the ALTID parameter. Each translation converts according to the rules as defined in Section 2.3.

### 2.6.5. N and NICKNAME

The N instances convert to equivalent items of the components array of the name property (Figure 11): the N components convert 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 comma-separated item of the SORT-AS parameter value converts to an entry of the sortAs property where the key is the "type" value related to the sorted N component and the value is the corresponding item. Absence of a key in the sortAs indicates that its related part in the SORT-AS parameter value **MUST** either the empty string followed by COMMA, or no further SORT-AS parts are defined.

| N component | "type" value |
|---|---|
| Honorific Prefixes | prefix |
| Given Names | given |
| Family Names | surname |
| Additional Names | middle |
| Honorific Suffixes | suffix |

Table 1: N components conversion

A NICKNAME property converts to an entry in the nickNames property (Figure 11). The entry value is a NickName object. The name property is set to the NICKNAME value.

The PREF and TYPE parameters convert according to the rules as defined in Section 2.3.

```
"fullName": "Mr.  John Q.  Public, Esq.",
"name": {
    "@type": "Name",
    "components":[
      { "@type": "NameComponent", "type": "prefix", "value": "Mr."  },
      { "@type": "NameComponent", "type": "given", "value": "John" },
      { "@type": "NameComponent", "type": "surname", "value": "Public" }
      { "@type": "NameComponent", "type": "middle", "value": "Quinlan" }
      { "@type": "NameComponent", "type": "suffix", "value": "Esq." }
    ],
    "sortAs": {
        "surname": "Public",
        "given": "John"
    }
},
"nickNames": {
   "NICK-1": {
      "@type": "NickName",
      "name": "Johnny"
   }
}
```

```
"fullName": "Mr.  John Q.  Public, Esq.",
"name": {
    "@type": "Name",
    "components":[
      { "@type": "NameComponent", "type": "prefix", "value": "Mr."  },
      { "@type": "NameComponent", "type": "given", "value": "John" },
      { "@type": "NameComponent", "type": "surname", "value": "Public" }
      { "@type": "NameComponent", "type": "middle", "value": "Quinlan" }
      { "@type": "NameComponent", "type": "suffix", "value": "Esq." }
    ],
    "sortAs": {
        "surname": "Public",
        "given": "John"
    }
},
"nickNames": {
   "NICK-1": {
      "@type": "NickName",
      "name": "Johnny"
   }
}
```

Figure 11: FN, N, NICKNAME conversion example

#### 2.6.6. PHOTO

A PHOTO property converts to an entry in the media property (Figure 12). The entry value is a MediaResource object whose type property is set to photo and uri property is set to the PHOTO value.

The PREF and MEDIATYPE parameters convert according to the rules as defined in Section 2.3.

```
PHOTO:http://www.example.com/pub/photos/jqpublic.gif
```

```
"media": {
   "PHOTO-1": {
     "@type": "MediaResource",
     "type": "photo",
     "uri": "http://www.example.com/pub/photos/jqpublic.gif"
   }
}
```

Figure 12: PHOTO conversion example

### 2.7. Delivery Addressing Properties

#### 2.7.1. ADR

An ADR property converts to an entry in the addresses property (Figure 13). The entry value is an Address object.

The ADR components convert into the Address properties as presented in Table 2 and Table 3.

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

Table 2: ADR components vs. Address members conversion

| 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 3: ADR components vs. StreetComponent items conversion

Applications **SHOULD** separate the ADR "street address" component into street name, number and directions. If they can not determine the respective street components from the ADR "street address"", then they **SHOULD** convert these to a single name StreetComponent object.

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

The LABEL parameter converts to the fullAddress property.

The GEO parameter converts to the coordinates property.

The TZ parameter converts to the timeZone property.

The CC parameter as defined in [RFC8605] converts to the countryCode property.

The PREF and TYPE parameters convert according to the rules as defined in Section 2.3.

The ALTID and LANGUAGE parameters convert according to the rules as defined in Section 2.3. Each possible language-dependent alternative is represented as an entry of the PatchObject map where the key references the fullAddress property.

```
ADR;TYPE=work;CC=US:;;54321 Oak St;Reston;VA;20190;USA
ADR;TYPE=home;CC=US:;;12345 Elm St;Reston;VA;20190;USA
```

```
"addresses": {
  "ADDR-1" : {
    "@type": "Address",
    "contexts": { "work": true },
    "fullAddress": "54321 Oak St\nReston\nVA\n20190\nUSA",
    "street": [
       { "@type": "StreetComponent", "type": "name", "value": "Oak St" }
       { "@type": "StreetComponent", "type": "number", "value": "54321"
    ],
    "locality": "Reston",
    "region": "VA",
    "country": "USA",
    "postcode": "20190",
    "countryCode": "US"
  },
  "ADDR-2": {
    "@type": "Address",
    "contexts": { "private": true },
    "fullAddress":  "12345 Elm St\nReston\nVA\n20190\nUSA",
    "street": [
       { "@type": "StreetComponent", "type": "name", "value": "Elm St" }
       { "@type": "StreetComponent", "type": "number", "value": "12345"
    ],
    "locality": "Reston",
    "region": "VA",
    "country": "USA",
    "postcode": "20190",
    "countryCode": "US"
  }
}
```

Figure 13: ADR conversion example

## 2.8. Communications Properties

### 2.8.1. CONTACT-CHANNEL-PREF

A CONTACT-CHANNEL-PREF property converts to an entry in the preferredContactChannels property (Figure 14). The entry key set is defined in Table 4, the related entry values are arrays of ContactChannelPreference objects.

| CONTACT-CHANNEL-PREF value | "preferredContactChannels" key |
|---|---|
| ADR | addresses |
| EMAIL | emails |
| IMPP | onlineServices |
| TEL | phones |

Table 4: CONTACT-CHANNEL-PREF values conversion

The PREF and TYPE parameters convert according to the rules as defined in Section 2.3.

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

```
CONTACT-CHANNEL-PREF;PREF=1:EMAIL
CONTACT-CHANNEL-PREF;PREF=2:TEL

"preferredContactChannels": {
  "emails":[
    {
      "@type": "ContactChannelPreference",
      "pref": 1
    }
  ],
  "phones":[
    {
      "@type": "ContactChannelPreference",
      "pref": 2
    }
  ]
}
```

Figure 14: LANG conversion example

Note: This property is defined in [I-D.ietf-calext-vcard-jscontact-extensions].

### 2.8.2. EMAIL

An EMAIL property converts to an entry in the emails property (Figure 15). The entry value is an EmailAddress object. The address property is set to the EMAIL value.

The PREF and TYPE parameters convert according to the rules as defined in Section 2.3.

```
EMAIL;TYPE=work:jqpublic@xyz.example.com
EMAIL;PREF=1:jane_doe@example.com

"emails": {
  "EMAIL-1": {
    "@type": "EmailAddress",
    "contexts": { "work": true },
    "address": "jqpublic@xyz.example.com"
  },
  "EMAIL-2": {
    "@type": "EmailAddress",
    "address": "jane_doe@example.com",
    "pref": 1
  }
}
```

Figure 15: EMAIL conversion example

### 2.8.3. IMPP

An IMPP property converts to an entry in the onlineServices property (Figure 16). The type of the OnlineService is set to impp. The entry value is a OnlineService object and the user property is set to the IMPP value.

The PREF and TYPE parameters convert according to the rules as defined in Section 2.3.

```
IMPP;PREF=1:xmpp:alice@example.com
```

```
"onlineServices": {
  "OS-1": {
    "@type": "OnlineService",
    "user": "xmpp:alice@example.com",
    "type": "impp",
    "pref": 1
  },
}
```

Figure 16: IMPP conversion example

### 2.8.4. LANG

A LANG property converts to an entry in the preferredLanguages property (Figure 17). The entry keys correspond to the language tags, the related entry values are arrays of LanguagePreference objects.

The PREF and TYPE parameters convert according to the rules as defined in Section 2.3.

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

```
LANG;TYPE=work;PREF=1:en
LANG;TYPE=work;PREF=2:fr
LANG;TYPE=home:fr
```

```
"preferredLanguages": {
  "en":[
    {
      "@type": "LanguagePreference",
      "contexts": { "work": true },
      "pref": 1
    }
  ],
  "fr":[
    {
      "@type": "LanguagePreference",
      "contexts": { "work": true },
      "pref": 2
    },
    {
      "@type": "LanguagePreference",
      "contexts": { "private": true }
    }
  ]
}
```

Figure 17: LANG conversion example

#### 2.8.5. LOCALE

The LOCALE property converts to the locale property (Figure 18).

```
LOCALE:de-AT

"locale": "de-AT"
```

Figure 18: LOCALE conversion example

Note: This property is defined in [I-D.ietf-calext-vcard-jscontact-extensions].

#### 2.8.6. SOCIALPROFILE

A SOCIALPROFILE property converts to an entry in the onlineServices property (Figure 19). If the value type of the SOCIALPROFILE is TEXT, then the type of the OnlineService is set to username. Otherwise, the type is set to uri. The entry value is a OnlineService object and the user property is set to the property value.

The PREF and TYPE parameters convert according to the rules as defined in Section 2.3.

```
SOCIALPROFILE;SERVICE-TYPE=Twitter:https://twitter.com/ietf
```

```
"onlineServices": {
  ...
  "OS-1": {
    "@type": "OnlineService",
    "service": "Twitter",
    "user": "https://twitter.com/ietf",
    "type": "uri"
  }
}
```

Figure 19: IMPP conversion example

Note: This property is defined in [I-D.ietf-calext-vcard-jscontact-extensions].

#### 2.8.7. TEL

A TEL property converts to an entry in the phones property (Figure 20). The entry value is a Phone object. The TEL-specific values of the TYPE parameter convert to the features property keys. The number property is set to the TEL value.

The PREF and TYPE parameters convert according to the rules as defined in Section 2.3.

```
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

"phones": {
  "PHONE-1": {
    "@type": "Phone",
    "contexts": { "private": true },
    "features": { "voice": true },
    "number": "tel:+1-555-555-5555;ext=5555",
    "pref": 1
  },
  "PHONE-2": {
    "@type": "Phone",
    "contexts": { "private": true },
    "number": "tel:+33-01-23-45-67"
  }
}
```

Figure 20: TEL conversion example

### 2.9. Geographical Properties

#### 2.9.1. GEO

This converts to the coordinates property of the Address object. Also see Section 2.9.3 to determine which Address object instance to convert to.

### 2.9.2. TZ

A value of type TEXT converts to the timeZone property in the Address object.

A value of type UTC-OFFSET converts to the timeZone property in the Address object if the offset has zero minutes and the hour offset is in the range -12 <= 14:

- If the hour offset is zero, use the time zone name Etc/UTC.
- Otherwise construct the time zone name with ETC/GMT suffixed with the string representation of the reversed sign hour offset, including the sign but excluding leading zeros and minutes. For example, the UTC offset value -0500 converts to ETC/GMT+5.

For such property values, also see Section 2.9.3 to determine which Address object instance to convert to.

Any other value of type UTC-OFFSET or URI does not convert to an IANA-registered property in JSContact. To convert such property, see Section 2.16.1.

### 2.9.3. Combining geographical properties

In vCard the properties ADR, GEO and TZ occur independently of each other. In JSContact, they all convert to properties of an Address object. It is implementation-specific if these vCard properties convert to *separate* address instances in JSContact, or if some or all of them convert to the *same* address. That being said, implementations **SHOULD** convert the properties to the *same* address for the following cases:

- The ALTID parameter values of the properties match.
- The ALTID parameters are not set, but are set on any other ADR, GEO and TZ properties.

## 2.10. Organizational Properties

### 2.10.1. CONTACT-URI

A CONTACT-URI property as defined in [RFC8605] is represented as an entry of the links property (Figure 21). The entry value is a LinkResource object whose type property is set to contact and uri property is set to the CONTACT-URI value.

The PREF and TYPE parameters convert according to the rules as defined in Section 2.3.

```
CONTACT-URI;PREF=1:mailto:contact@example.com
```

```
"links": {
  "CONTACT-1": {
    "@type": "LinkResource",
    "type": "contact",
    "uri": "mailto:contact@example.com",
    "pref": 1
  }
}
```

Figure 21: CONTACT-URI conversion example

#### 2.10.2. LOGO

A LOGO property converts to an entry in the media property (Figure 22). The entry value is a MediaResource object whose type property is set to logo and uri property is set to the LOGO value.

The PREF and TYPE parameters convert according to the rules as defined in Section 2.3.

```
LOGO:http://www.example.com/pub/logos/abccorp.jpg
```

```
"media": {
  "LOGO-1": {
    "@type": "MediaResource",
    "type": "logo",
    "uri": "http://www.example.com/pub/logos/abccorp.jpg"
  }
}
```

Figure 22: LOGO conversion example

#### 2.10.3. MEMBER

The uids of the contact cards composing the group are included in the members property (Figure 23).

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

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

```
"kind": "group",
"fullName": "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 23: Group example

#### 2.10.4. ORG

An ORG property converts to an entry in the organizations property (Figure 24). The entry value is an Organization object whose name property contains the organizational name, and the units property is an array of OrgUnit objects each containing the organizational unit name in the name property.

Implementations **MAY** allow representation of organizational units without the organizational name. In this case, the first component of the ORG value **MUST** be an empty string (e.g. ORG:;DepartmentA).

The ALTID, LANGUAGE parameters convert according to the rules as defined in Section 2.3.

The first item of the comma-separated SORT-AS parameter value converts to the sortAs property of the Organization object. The subsequent items convert to the sortAs property of the corresponding OrgUnit object

The TYPE parameter converts according to the rules as defined in Section 2.3.

```
ORG;SORT-AS="ABC":ABC\, Inc.;North American Division;Marketing
```

```
"organizations": {
  "ORG-1": {
    "@type": "Organization",
    "name": "ABC, Inc.",
    "units":[
      {"@type": "OrgUnit", "name": "North American Division"},
      {"@type": "OrgUnit", "name": "Marketing"}
    ],
    "sortAs": "ABC"
  }
}
```

Figure 24: ORG conversion example

#### 2.10.5. RELATED

This converts to an entry in the relatedTo property (Figure 25). The property value converts to the key in the relatedTo property. The

TYPE parameters convert to the relation of the Relation object value. Any other parameters convert as defined in Section 2.16.2.

```
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 inquirie
```

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

Figure 25: RELATED conversion example

### 2.10.6. TITLE and ROLE

Both TITLE and ROLE properties are represented as entries of the titles property (Figure 26). The entry value is a Title object whose type property is set to title or role for TITLE and ROLE vCard properties, respectively. The name property is set to the vCard property value.

The value of the organization property can be derived if the TITLE or ROLE property is member of a vCard property group and exactly one other ORG property also is part of that group.

The ALTID and LANGUAGE parameters convert according to the rules as defined in Section 2.3.

```
TITLE:Research Scientist
group1.ROLE:Project Leader
group1.ORG:ABC, Inc.
```

```
"titles": {
  "TITLE-1": {
    "@type": "Title",
    "type": "title",
    "name": "Project Leader"
  },
  "TITLE-2": {
    "@type": "Title",
    "type": "role",
    "name": "Research Scientist",
    "organization": "ORG-1"
  }
},
"organizations": {
  "ORG-1": {
    "@type": "Organization",
    "name": "ABC, Inc."
  }
}
```

Figure 26: TITLE and ROLE conversion example

## 2.11. Personal Information Properties

### 2.11.1. EXPERTISE

An EXPERTISE property as defined in [RFC6715] is represented as a PersonalInfo object in the personalInfo property (Figure 27). The type property is set to "expertise".

The INDEX parameter converts according to the rules as defined in Section 2.3.

```
EXPERTISE;LEVEL=beginner;INDEX=2:chinese literature
EXPERTISE;INDEX=1;LEVEL=expert:chemistry

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

Figure 27: EXPERTISE conversion example

### 2.11.2. HOBBY

A HOBBY property as defined in [RFC6715] is represented as a PersonalInfo object in the personalInfo property (Figure 28). The type property is set to "hobby".

The INDEX parameter converts according to the rules as defined in Section 2.3.

```
HOBBY;INDEX=1;LEVEL=high:reading
HOBBY;INDEX=2;LEVEL=high:sewing
```

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

Figure 28: HOBBY conversion example

### 2.11.3. INTEREST

An INTEREST property as defined in [RFC6715] is represented as a PersonalInfo object in the personalInfo property (Figure 29). The type property is set to "interest".

The INDEX parameter converts according to the rules as defined in Section 2.3.

```
INTEREST;INDEX=1;LEVEL=medium:r&b music
INTEREST;INDEX=2;LEVEL=high:rock ’n’ roll music
```

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

Figure 29: INTEREST conversion example

#### 2.11.4. ORG-DIRECTORY

An ORG-DIRECTORY property as defined in [RFC6715] is represented as an entry of the directories property (Figure 30). The entry value is a DirectoryResource object whose type property is set to directory and uri property is set to the ORG-DIRECTORY value.

The INDEX, PREF and TYPE parameters convert according to the rules as defined in Section 2.3.

```
ORG-DIRECTORY;INDEX=1:http://directory.mycompany.example.com
ORG-DIRECTORY;PREF=1:ldap://ldap.tech.example/o=Example%20Tech,ou=Engine

"directories": {
  "DIRECTORY-1": {
    "@type": "DirectoryResource",
    "type": "directory",
    "uri": "http://directory.mycompany.example.com",
    "position": 1
  },
  "DIRECTORY-2": {
    "@type": "DirectoryResource",
    "type": "directory",
    "uri": "ldap://ldap.tech.example/o=Example%20Tech,ou=Engineering",
    "pref": 1
  }
}
```

Figure 30: ORG-DIRECTORY conversion example

## 2.12. Explanatory Properties

### 2.12.1. CATEGORIES

A CATEGORIES property converts to a set of entries of the keywords property (Figure 31). The keys are the comma-separated text values of the CATEGORIES property.

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

```
CATEGORIES:internet,IETF,Industry,Information Technology
```

```
"keywords": {
  "internet": true,
  "IETF": true,
  "Industry": true,
  "Information Technology": true
}
```

Figure 31: CATEGORIES conversion example

### 2.12.2. CLIENTPIDMAP and PID Parameter

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

### 2.12.3. CREATED

The CREATED property converts to the created property (Figure 32).

```
CREATED:19940930T143510Z
```

```
"created": "1994-09-30T14:35:10Z"
```

Figure 32: CREATED conversion example

Note: This property is defined in [I-D.ietf-calext-vcard-jscontact-extensions].

### 2.12.4. NOTE

A NOTE property converts to a Note object in the notes map (Figure 33)

The ALTID and LANGUAGE parameters convert according to the rules as defined in Section 2.3.

```
NOTE;CREATED=20221123T150132Z;AUTHOR-NAME="John":
  Office hours are from 0800 to 1715 EST\, Mon-Fri.
```

```
"notes": {
    "NOTE-1" : {
       "note": "Office hours are from 0800 to 1715 EST, Mon-Fri.",
       "created": "2022-11-23T15:01:32Z",
       "author": {
         "@type": "Author",
         "name": "John"
       }
    }
}
```

Figure 33: NOTE conversion example

#### 2.12.5. PRODID

The PRODID property converts to the prodId property (Figure 34).

```
PRODID:-//ONLINE DIRECTORY//NONSGML Version 1//EN

"prodId": "-//ONLINE DIRECTORY//NONSGML Version 1//EN"
```

Figure 34: PRODID conversion example

#### 2.12.6. REV

The REV property converts to the updated property (Figure 35).

```
REV:19951031T222710Z

"updated": "1995-10-31T22:27:10Z"
```

Figure 35: REV conversion example

#### 2.12.7. SOUND

A SOUND property converts to an entry in the media property (Figure 36). The entry value is a MediaResource object whose type property is set to sound and uri property is set to the SOUND value.

The PREF and TYPE parameters convert according to the rules as defined in Section 2.3.

```
SOUND:CID:JOHNQPUBLIC.part8.19960229T080000.xyzMail@example.com
```

```
"media": {
  ...
  "SOUND-1": {
    "@type": "MediaResource",
    "type": "sound",
    "uri": "CID:JOHNQPUBLIC.part8.19960229T080000.xyzMail@example.com"
  }
}
```

Figure 36: SOUND conversion example

#### 2.12.8. UID

The UID property corresponds to the uid property (Figure 37).

```
UID:urn:uuid:f81d4fae-7dec-11d0-a765-00a0c91e6bf6

"uid": "urn:uuid:f81d4fae-7dec-11d0-a765-00a0c91e6bf6"
```

Figure 37: UID conversion example

#### 2.12.9. URL

An URL property converts to an entry in the links property (Figure 38). The entry value is a LinkResource object whose uri property is set to the URL value.

The PREF and TYPE parameters convert according to the rules as defined in Section 2.3.

```
URL:http://example.org/restaurant.french/~chezchic.html

"links": {
  "LINK-1": {
    "@type": "LinkResource",
    "uri": "http://example.org/restaurant.french/~chezchic.html"
  }
}
```

Figure 38: URL conversion example

#### 2.12.10. VERSION

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

#### 2.12.11. X-ABLabel

This property is experimental but widely in use in existing vCard data. It converts to the label property of a JSContact object type. The X-ABLabel property is preceded by a vCard property group name,

and the label converts to the JSContact object which got converted from a vCard property having the same group.

The group name is not preserved, implementations are free to choose any unique group name when converting back to vCard. For an example how to preserve the group name see Section 2.3.8.

```
item1.TEL;VALUE=uri:tel:+1-555-555-5555
item1.X-ABLabel:foo
```

```
"phones": {
  "p1": {
    "@type": "Phone",
    "number": "tel:+1-555-555-5555",
    "label": "foo"
  }
}
```

Figure 39: X-ABLabel conversion example

## 2.13. Security Properties

### 2.13.1. KEY

A KEY property converts to an entry in the cryptoKeys property (Figure 40). The entry value is a CryptoResource object whose uri property is set to the KEY value.

The PREF and TYPE parameters convert according to the rules as defined in Section 2.3.

```
KEY:http://www.example.com/keys/jdoe.cer
```

```
"cryptoKeys": {
  "KEY-1": {
    "@type": "CryptoResource",
    "uri": "http://www.example.com/keys/jdoe.cer"
  }
}
```

Figure 40: KEY conversion example

## 2.14. Calendar Properties

### 2.14.1. CALADRURI

A CALADRURI property converts to an entry in the schedulingAddresses property (Figure 41). The entry value is a SchedulingAddress object whose uri property is set to the CALADRURI value.

The PREF parameter converts according to the rules as defined in Section 2.3.

```
CALADRURI;PREF=1:mailto:janedoe@example.com
CALADRURI:http://example.com/calendar/jdoe
```

```
"schedulingAddresses": {
  "SCHEDULING-1": {
    "@type": "SchedulingAddress",
    "uri": "mailto:janedoe@example.com",
    "pref": 1
  },
  "SCHEDULING-2": {
    "@type": "SchedulingAddress",
    "uri": "http://example.com/calendar/jdoe"
  }
}
```

Figure 41: CALADRURI conversion example

### 2.14.2. CALURI

A CALURI property converts to an entry in the calendars property (Figure 42). The entry value is a CalendarResource object whose type property is set to calendar and uri property is set to the CALURI value.

The PREF and TYPE parameters convert according to the rules as defined in Section 2.3.

```
CALURI;PREF=1:http://cal.example.com/calA
CALURI;MEDIATYPE=text/calendar:ftp://ftp.example.com/calA.ics
```

```
"calendars": {
  "CAL-1": {
    "@type": "CalendarResource",
    "type": "calendar",
    "uri": "http://cal.example.com/calA",
    "pref": 1
  },
  "CAL-2": {
    "@type": "CalendarResource",
    "type": "calendar",
    "uri": "ftp://ftp.example.com/calA.ics",
    "mediaType": "text/calendar"
  }
}
```

Figure 42: CALURI conversion example

### 2.14.3. FBURL

An FBURL property converts to an entry in the calendars property (Figure 43). The entry value is a CalendarResource object whose type

property is set to freeBusy and uri property is set to the FBURL value.

The PREF and TYPE parameters convert according to the rules as defined in Section 2.3.

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

```
"calendars": {
  "FBURL-1": {
    "@type": "CalendarResource",
    "type": "freeBusy",
    "uri": "http://www.example.com/busy/janedoe",
    "pref": 1
  },
  "FBURL-2": {
    "@type": "CalendarResource",
    "type": "freeBusy",
    "uri": "ftp://example.com/busy/project-a.ifb",
    "mediaType": "text/calendar"
  }
}
```

Figure 43: FBURL conversion example

### 2.15. Extended Properties and Parameters

These convert as specified in Section 2.16.

### 2.16. New JSContact properties

vCards may contain properties or parameters for which no IANA-registered JSContact property is defined. For example, a vCard may contain extended properties and parameters of which the semantics or purposes are unknown Section 6.10 of [RFC6350].

This section defines JSContact extension properties by which such vCard properties and parameters **MAY** be represented in JSContact. Implementations **MAY** choose to convert differently if they deem that more appropriate.

#### 2.16.1. Property vCardProps

**Name:** vCardProps

**Type:** JCardProp[], where JCardProp denotes a jCard-encoded vCard property as defined in Section 3.3 of [RFC7095].

**Definition:** This property is set on a JSContact object that represents a vCard. It contains properties that are set in the vCard represented by this JSContact object. Each entry in this list typically represents a vCard property for which no

conversion to an IANA-registered JSContact property is defined. It **MAY** contain vCard IANA-registered properties which also got converted to an IANA-registered property in the same JSContact object. In case of conflict, the values of the JSContact property **MUST** be used.

**Example:** This illustrates how to convert a vCard extension property:

```
item1.X-FOO;X-BAR=Hello:World!
```

```
"vCardProps": [
  ["x-foo", {
    "x-bar": "Hello",
    "group": "item1"
  }, "unknown", "World!"]
]
```

Figure 44: JSContact props example

### 2.16.2. Property vCardParams

**Name:** vCardParams

**Type:** String[String|String[]]

**Definition:** This property is set on a JSContact object that represents a vCard property. Its value **MUST** be a JSON object containing vCard property parameters, defined as array element 2 in Section 3.3 of [RFC7095]. Each entry represents a parameter of the vCard property that converts to the JSContact object. Typically, these are parameters for which no IANA-registered property is defined in the JSContact object. It **MAY** contain vCard IANA-registered parameters which also got converted to an IANA-registered property in the same JSContact object. In case of conflict, the values of the JSContact property **MUST** be used.

**Example:** This illustrates how to convert a vCard extension parameter:

```
EMAIL;X-FOO=Bar:jane_doe@example.com
```

```
"emails": {
  "email1": {
    "@type": "EmailAddress",
    "address": "jane_doe@example.com",
    "vCardParams": {
      "x-foo": "Bar"
    }
  }
}
```

Figure 45: JSContact vCardParams example

## 3. Converting JSContact to vCard

### 3.1. Conversion Rules

A Card converts to vCard by applying the reverse rules of converting vCard to JSContact. Table 5 lists the relevant document sections for each JSContact object type and property. The following additional rules apply:

* Multi-valued JSContact properties convert to separate instances of their equivalent vCard property, and for each the PROP-ID parameter **MUST** be set to the Id of the converted value (see Section 2.3.15).

* The fullName property in JSContact is optional, but it is mandatory in vCard. If the fullName is not set but the name property is, then implementations **MAY** derive the value of the FN property from it. In this case, they **MUST** set the DERIVED parameter on the FN property. Otherwise, they **MUST** set the FN property with an empty value.

* Vendor-extension and unknown properties convert to vCard as outlined in section Section 3.2.

| JSContact Type | Property Name | Relevant Section(s) |
|---|---|---|
| Address | @type | not applicable |
| Address | contexts | Section 2.3.19 |
| Address | coordinates | Section 2.3.7, Section 2.9.1 |
| Address | country | Section 2.7.1 |
| Address | countryCode | Section 2.7.1 |
| Address | fullAddress | Section 2.7.1 |
| Address | locality | Section 2.7.1 |
| Address | postcode | Section 2.7.1 |
| Address | pref | Section 2.3.14 |
| Address | region | Section 2.7.1 |
| Address | street | Section 2.7.1 |
| Address | timeZone | Section 2.3.20, Section 2.9.2 |
| Anniversary | @type | not applicable |
| Anniversary | date | Section 2.6.1 |
| Anniversary | place | Section 2.6.1 |
| Anniversary | type | Section 2.6.1 |
| Author | @type | not applicable |
| Author | name | Section 2.3.3 |
| Author | uri | Section 2.3.2 |
| CalendarResource | @type | not applicable |
| CalendarResource | contexts | Section 2.3.19 |
| CalendarResource | label | Section 2.12.11 |
| CalendarResource | mediaType | Section 2.3.12 |
| CalendarResource | pref | Section 2.3.14 |
| CalendarResource | type | Section 2.14.1, Section 2.14.3 |
| CalendarResource | uri | Section 2.14.1, Section 2.14.3 |

| JSContact Type | Property Name | Relevant Section(s) |
|---|---|---|
| Card | @type | not applicable |
| Card | @version | not applicable |
| Card | addresses | Section 2.7.1 |
| Card | anniversaries | Section 2.6.1 |
| Card | calendars | Section 2.14.1, Section 2.14.3 |
| Card | created | Section 2.12.3 |
| Card | directories | Section 2.5.3, Section 2.11.4 |
| Card | emails | Section 2.8.2 |
| Card | fullName | Section 2.6.4 |
| Card | keywords | Section 2.12.1 |
| Card | kind | Section 2.5.2 |
| Card | links | Section 2.10.1, Section 2.12.9 |
| Card | locale | Section 2.8.5 |
| Card | localizations | Section 2.3.10 |
| Card | media | Section 2.6.6, Section 2.10.2, Section 2.12.7 |
| Card | members | Section 2.10.3 |
| Card | name | Section 2.6.5 |
| Card | nickNames | Section 2.6.5 |
| Card | notes | Section 2.12.4 |
| Card | onlineServices | Section 2.8.3 |
| Card | organizations | Section 2.10.4 |
| Card | personalInfo | Section 2.11.1, Section 2.11.2, Section 2.11.3 |
| Card | phones | Section 2.8.7 |
| Card | preferredContactChannels | Section 2.8.1 |
| Card | preferredLanguages | Section 2.8.4 |
| Card | prodId | Section 2.12.5 |
| Card | relatedTo | Section 2.10.5 |
| Card | schedulingAddresses | Section 2.14.1 |
| Card | speakToAs | Section 2.6.3 |
| Card | titles | Section 2.10.6 |
| Card | uid | Section 2.12.8 |
| Card | updated | Section 2.12.6 |
| ContactChannelPreference | @type | not applicable |
| ContactChannelPreference | contexts | Section 2.3.19 |
| ContactChannelPreference | pref | Section 2.3.14 |
| CryptoResource | @type | not applicable |
| CryptoResource | contexts | Section 2.3.19 |
| CryptoResource | label | Section 2.12.11 |
| CryptoResource | mediaType | Section 2.3.12 |
| CryptoResource | pref | Section 2.3.14 |
| CryptoResource | type | not applicable |
| CryptoResource | uri | Section 2.13.1 |
| DirectoryResource | @type | not applicable |
| DirectoryResource | contexts | Section 2.3.19 |
| DirectoryResource | label | Section 2.12.11 |
| DirectoryResource | listAs | Section 2.3.9 |
| DirectoryResource | mediaType | Section 2.3.12 |

| JSContact Type | Property Name | Relevant Section(s) |
|---|---|---|
| DirectoryResource | pref | Section 2.3.14 |
| DirectoryResource | type | Section 2.5.3, Section 2.11.4 |
| DirectoryResource | uri | Section 2.5.3, Section 2.11.4 |
| EmailAddress | @type | not applicable |
| EmailAddress | address | Section 2.8.2 |
| EmailAddress | contexts | Section 2.3.19 |
| EmailAddress | label | Section 2.12.11 |
| EmailAddress | pref | Section 2.3.14 |
| LanguagePreference | @type | not applicable |
| LanguagePreference | contexts | Section 2.3.19 |
| LanguagePreference | pref | Section 2.3.14 |
| LinkResource | @type | not applicable |
| LinkResource | contexts | Section 2.3.19 |
| LinkResource | label | Section 2.12.11 |
| LinkResource | mediaType | Section 2.3.12 |
| LinkResource | pref | Section 2.3.14 |
| LinkResource | type | Section 2.10.1, Section 2.12.9 |
| LinkResource | uri | Section 2.10.1, Section 2.12.9> |
| MediaResource | @type | not applicable |
| MediaResource | contexts | Section 2.3.19 |
| MediaResource | label | Section 2.12.11 |
| MediaResource | mediaType | Section 2.3.12 |
| MediaResource | pref | Section 2.3.14 |
| MediaResource | type | Section 2.6.6, Section 2.10.2, Section 2.12.7 |
| MediaResource | uri | Section 2.6.6, Section 2.10.2, Section 2.12.7 |
| Name | @type | not applicable |
| Name | components | Section 2.6.5 |
| Name | sortAs | Section 2.3.18 |
| NameComponent | @type | not applicable |
| NameComponent | rank | Section 2.3.16 |
| NameComponent | type | Section 2.6.5 |
| NameComponent | value | Section 2.6.5 |
| NickName | @type | not applicable |
| NickName | contexts | Section 2.3.19 |
| NickName | name | Section 2.6.5 |
| NickName | pref | Section 2.3.14 |
| Note | @type | not applicable |
| Note | author | Section 2.3.2, Section 2.3.3 |
| Note | created | Section 2.3.5 |
| Note | note | Section 2.12.4 |
| OnlineService | @type | not applicable |
| OnlineService | contexts | Section 2.3.19 |
| OnlineService | label | Section 2.12.11 |
| OnlineService | pref | Section 2.3.14 |
| OnlineService | service | Section 2.3.17 |
| OnlineService | type | Section 2.8.3, Section 2.8.6 |
| OnlineService | user | Section 2.8.3, Section 2.8.6 |

| JSContact Type | Property Name | Relevant Section(s) |
|---|---|---|
| OrgUnit | @type | not applicable |
| OrgUnit | name | Section 2.10.4 |
| OrgUnit | sortAs | Section 2.3.18 |
| Organization | @type | not applicable |
| Organization | contexts | Section 2.3.19 |
| Organization | name | Section 2.10.4 |
| Organization | sortAs | Section 2.3.18 |
| Organization | units | Section 2.10.4 |
| PartialDate | @type | not applicable |
| PartialDate | calendarScale | Section 2.3.4 |
| PartialDate | day | Section 2.2.2 |
| PartialDate | month | Section 2.2.2 |
| PartialDate | year | Section 2.2.2 |
| PatchObject | @type | not applicable |
| PersonalInfo | @type | not applicable |
| PersonalInfo | listAs | Section 2.3.9 |
| PersonalInfo | level | Section 2.3.11 |
| PersonalInfo | type | Section 2.11.1, Section 2.11.2, Section 2.11.3 |
| PersonalInfo | value | Section 2.11.1, Section 2.11.2, Section 2.11.3 |
| Phone | @type | not applicable |
| Phone | contexts | Section 2.3.19 |
| Phone | features | Section 2.8.7 |
| Phone | label | Section 2.12.11 |
| Phone | number | Section 2.8.7 |
| Phone | pref | Section 2.3.14 |
| Pronouns | @type | not applicable |
| Pronouns | contexts | Section 2.3.19 |
| Pronouns | pref | Section 2.3.14 |
| Pronouns | pronouns | Section 2.6.3 |
| Relation | @type | not applicable |
| Relation | relation | Section 2.10.5 |
| Resource | @type | not applicable |
| SchedulingAddress | @type | not applicable |
| SchedulingAddress | contexts | Section 2.3.19 |
| SchedulingAddress | label | Section 2.12.11 |
| SchedulingAddress | pref | Section 2.3.14 |
| SchedulingAddress | uri | Section 2.14.1 |
| SpeakToAs | @type | not applicable |
| SpeakToAs | grammaticalGender | Section 2.6.3 |
| SpeakToAs | pronouns | Section 2.6.3 |
| StreetComponent | @type | not applicable |
| StreetComponent | type | Section 2.7.1 |
| StreetComponent | value | Section 2.7.1 |
| Timestamp | @type | not applicable |
| Timestamp | utc | Section 2.2.2 |
| Title | @type | not applicable |
| Title | name | Section 2.10.6 |

| JSContact Type | Property Name | Relevant Section(s) |
|---|---|---|
| Title | organization | Section 2.10.6 |
| Title | type | Section 2.10.6 |

Table 5: Conversion rules by JSContact property

### 3.2. New vCard Properties and Parameters

JSContact object types may contain properties for which no IANA-registered vCard property is defined. For example, a JSContact object may contain vendor-specific properties of which the semantics or purpose are unknown.

This section defines new vCard properties and parameters by which such JSContact properties **MAY** be represented in JSContact. Implementations **MAY** choose to convert differently if they deem that more appropriate.

#### 3.2.1. Property JSCONTACT-PROP

**Property name:** JSCONTACT-PROP

**Purpose:** This property represents a JSContact property in vCard.

**Value type:** TEXT, also see Format Definition for value restrictions.

**Conformance:** The property can be specified multiple times in a vCard.

**Property parameters:** The JSPTR parameter **MUST** be set exactly once for this property. The VALUE parameter **MAY** be set once, in which case its value **MUST** be URI. Other IANA-registered and experimental property parameters can be specified on this property.

**Description:** This property converts a JSContact property to vCard. The vCard property value is the JSON-encoded value of the JSContact property, represented as a TEXT value. The format of the JSON value **SHOULD** be compact, e.g. without insignificant whitespace.

The value of the JSPTR parameter defines the path to that JSContact value within the Card. The last segment of the JSON pointer either defines the property name or array position of the JSContact value. The root of the JSON pointer always is the Card object that this vCard converts to, irrespective if the JSON pointer starts with the SOLIDUS (U+002F) character. If any but the last segment of the JSON pointer points to a non-existent JSContact property or array entry in the Card, then the

JSCONTACT-PROP property **MUST** be ignored and **SHOULD** be discarded from the vCard.

**Format definition:** This property is defined by the following notation:

```
jscontact-prop = "JSCONTACT-PROP" jscontact-prop-param ":" TEXT CRLF

jscontact-prop-param  = *(
            ; The following are MANDATORY and MUST NOT
            ; occur more than once
            ( ";" jspath-param ) /    ; see next section
            ( ";" "VALUE" "=" "TEXT")
            ;
            ; The following is OPTIONAL,
            ; and MAY occur more than once.
            ;
            (";" other-param)
            ;
            )
```

**Example(s):** This illustrates how to convert a property at the top-level in a Card object that is unknown to the implementation.

```
"someUnknownProperty": true

JSCONTACT-PROP;VALUE=TEXT;JSPTR="someUnknownProperty":true
```

Figure 46: Unknown property example

This illustrates how to convert a vendor-extension property at the top-level of a Card object. Note the required use of quoted string for the JSPTR value which allows the path to include the COLON (U+003A) character.

```
"example.com:foo": {
  "bar": 1234
}

JSCONTACT-PROP;VALUE=TEXT;JSPTR="example.com:foo":{"bar":1234}
```

Figure 47: Vendor-extension conversion example

This illustrates how to convert a vendor-extension property at a nested level in a Card object using a path relative to the Card object. Although not recommended, the property name includes the SOLIDUS (U+002F) character which requires escaping in the JSON pointer.

```
"phones": {
  "phone1": {
   "@type": "Phone",
   "number": "tel:+33-01-23-45-67"
    "example.com:foo/bar": "tux hux"
 }
}

TEL:tel:+33-01-23-45-67
JSCONTACT-PROP;VALUE=TEXT;JSPTR="phones/phone1/example.com:foo~1bar":"tu
```

Figure 48: Nested property example with path relative to Card

### 3.2.2. Parameter JSPTR

**Parameter name:** JSPTR

**Purpose:** This parameter contains a JSON pointer [RFC6901] that relates its vCard property to some JSON data. Its exact semantics depend on the definition of the property where this parameter is set on.

**Description:** This parameter has a single value that **MUST** be a valid JSON pointer as defined in [RFC6901]. Currently, its semantics only are defined for the JSCONTACT-PROP property (see Section 3.2.1), but it may also be used for other use cases in the future.

**Format definition:**

```
jsptr-param = "JSPTR" "=" DQUOTE *QSAFE-CHAR DQUOTE
              ; also see param-value in RFC 6350, section 3.3
```

**Example(s):** This illustrates a simple example. For further examples see Section 3.2.1.

```
JSCONTACT-PROP;VALUE=TEXT;JSPTR="example.com:foo":"bar"
```

## 4. Security Considerations

This specification defines how to convert between the JSContact and vCard formats. The security considerations for parsing and formatting such data apply and are outlined in Section 5 of [I-D.ietf-calext-jscontact] and Section 9 of [RFC6350].

## 5. IANA Considerations

### 5.1. New vCard Properties

IANA is requested to add the following entries to the "vCard Properties" registry, defined in Section 10.3.1. of [RFC6350].

| Namespace | Property | Reference |
|---|---|---|
| | JSCONTACT-PROP | This document, Section 3.2.1 |

Table 6: New VCARD Properties

### 5.2. New vCard Parameters

IANA is requested to add the following entries to the "vCard Parameters" registry, defined in Section 10.3.2. of [RFC6350].

| Namespace | Parameter | Reference |
|---|---|---|
| | JSPTR | This document, Section 3.2.2 |

Table 7: New VCARD Parameters

### 5.3. New JSContact Properties

IANA is requested to add the following entries to the "JSContact Properties" registry.

| Property Name | Property Type | Property Context | Reference or Description | Intended Usage | Since Version | Until Version | Change Contro |
|---|---|---|---|---|---|---|---|
| vCardParams | String[String] | Any JSContact object type that contains the @type property. | Section 2.16.2 | common | 1.0 | | IETF |
| vCardProps | JCardProp[] | Card | Section 2.16.1 | common | 1.0 | | IETF |

Table 8: Initial Contents of the "JSContact Properties" Registry

### 5.4. New JSContact Types

IANA is requested to add the following entries to the "JSContact Types" registry.

| Type Name | Reference or Description | Intended Usage | Since Version | Until Version | Change Controller |
|---|---|---|---|---|---|
| JCardProp | Section 2.16.1 | common | 1.0 | | IETF |

Table 9: Additional Contents of the "JSContact Types" Registry

## 6. 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 consideration to 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".

### 6.1. 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

## 7. References

### 7.1. Normative References


**[RFC2119]** Bradner, S. and RFC Publisher, "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>.

**[RFC5234]** Crocker, D., Ed. and P. Overell, "Augmented BNF for Syntax Specifications: ABNF", STD 68, RFC 5234, DOI

10.17487/RFC5234, January 2008, <https://www.rfc-editor.org/info/rfc5234>.

**[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>.

**[RFC6901]** Bryan, P., Ed., Zyp, K., and M. Nottingham, Ed., "JavaScript Object Notation (JSON) Pointer", RFC 6901, DOI 10.17487/RFC6901, April 2013, <https://www.rfc-editor.org/info/rfc6901>.

**[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>.

### 7.2. Informative References

**[I-D.ietf-calext-jscontact]** Stepanek, R. and M. Loffredo, "JSContact: A JSON representation of contact data", Work in Progress, Internet-Draft, draft-ietf-calext-jscontact-07, 10 January 2023, <https://datatracker.ietf.org/doc/html/draft-ietf-calext-jscontact-07>.

**[I-D.ietf-calext-vcard-jscontact-extensions]** Stepanek, R. and M. Loffredo, "vCard Format Extension for JSContact", Work in Progress, Internet-Draft, draft-ietf-calext-vcard-jscontact-extensions-03, 9 December 2022, <https://

datatracker.ietf.org/doc/html/draft-ietf-calext-vcard-jscontact-extensions-03>.

**[RFC8174]** Leiba, B. and RFC Publisher, "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>.

**[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>.

## 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