iCarol

iCarol API Reference

The iCarol API provides controlled, programmatic access to data residing in iCarol systems. You will find it uses industry-standard ways to expose data to you, the API consumer. We are continuously expanding the features and data that are available through the API as well as the actions you can perform with it. It is our goal that third party applications can creatively access and present this data in useful and interesting ways to the benefit of our clients who use iCarol, but only with their expressed and ongoing permission for that data to be made available.

This document provides technical detail on how to use the API. Please contact us regularly for updates as the API is evolving and expanding over time. Revision history can be found on the next page.

Legal notice

Access to the iCarol API is made available by “keys” published by your iCarol Account Manager, as described later in this document. Should the iCarol client who pays for their iCarol system decide to cease permission to your access of their data, the API Key would be revoked and data previously accessible by that API Key would cease immediately. Neither the iCarol team nor our client is obliged to provide notification of such an event. You may not use the API or its documentation to reverse engineer or to build a competitive product or service.

Revision History

Revision Number	Description	Revision Date
001	Version 1 Released	04/19/2013
002	ResourceVerification/Initiate and Email/Templates methods are added.	05/01/2013
003	Resource/Update is added. The Accept-Language header is honored now in some methods.	05/15/2013
004	Added Resource/Meta. Added *customFieldValueIDs* parameter to Resource/Search. Added *customField* and *coverage* to *resourceDetail* object.	05/30/2013
005	Added examples API calls for Resource/Update.	06/07/2013
006	Added Notes on Forward Compatibility, and additional properties to *resourceDetail*.	06/17/2013
007	Added the ability when searching for resources to query and sort Resources according to their last-modified date.	06/19/2013
008	Added summary of language code formatting to the Language Preference section. Added *translations* property to *resourceDetail* object.	07/15/2013
009	Added Request Detail Logging.	09/03/2013
010	Added information about mapping fields between the iCarol API and iCarol CSV/Excel Resource Exports.	10/13/2013
011	Added new Messaging/Status method.	11/01/2013
012	Added *descriptionText* to the *resourceDetail* object.	12/12/2013
013	Added *searchHints* to the *resourceDetail* object.	03/05/2014

Developer Documentation

Available developer-targeted information, updates to this document and more can be found at https://api.icarol.com/Help.

References

HTTP 1.1 Specification (IETF RFC 2616)
http://tools.ietf.org/html/rfc2616

OAuth 2.0 Specification (IETF RFC 6749)
http://tools.ietf.org/html/rfc6749

OAuth 2.0 Bearer Tokens (IETF RFC 6750)
http://tools.ietf.org/html/rfc6750

JSON (IETF RFC 4627)
http://www.ietf.org/rfc/rfc4627

A Note on Forward Compatibility

As a warning to our consumers, to prevent your application from breaking when we add new features, please be forward thinking. Familiarize yourself with the referenced W3C Specifications, and when possible, use well-known libraries to assist you in making requests and parsing responses. Here are some points to pay specific attention to:

- For now, the iCarol API will return JSON only. In the future, we will be adding additional formats. Use of the HTTP Accept header will ensure that your application continues receiving content in the `application/json` format.
- iCarol will be incrementally adding new services, as well as enhancing existing services. This means that additional properties may be added to existing/current responses, and that the ordering of parameters (especially in json) may change at any time. When parsing API responses, you should ensure that additional and/or missing properties will not break your application. HTTP Headers can, and will change as we add support for additional Request/Response metadata. These too, should not break your application when modified. We will try to maintain compatibility as much as possible within these guidelines, and if necessary, will increment the base URL when introducing potential breaking changes like renaming or relocating data structures in the API response scheme.
- While we make every effort to keep the iCarol API as responsive as possible, we reserve the right to throttle API clients that may be affecting the performance of other users. The reliability of mobile data connections also dictate the unfortunate necessity of reminding you that occasional timeouts or slow responses can happen and should be gracefully handled to ensure the best possible quality of your application or service and experience for your users. It is recommended to pay attention to the HTTP response codes and the possibility of retrying your requests after a sufficient delay.
- The iCarol API and this document may contain mistakes or oversights. Please bring any discrepancy to our attention, and we will address it as quickly as possible.

Authorization

OAuth 2.0 Bearer Token Authorization

The iCarol Web API supports the use of Bearer Tokens supported by OAuth 2.0 and defined in IETF RFC 6750. Please see the RFC for a complete description of the OAuth process, but we will summarize it here.

Bearer Token Authorization is sometimes called One-Legged OAuth because the client skips the Authorization Request and Grant steps of the authentication flow and passes the access token directly to the resource server. This token can be any piece of encoded data.

Access Token

The Access Token is passed directly in the HTTP Authorization Request Header using the "Bearer" scheme:

```
GET https://api.icarol.com/v1/Database
Authorization: Bearer 28D3091E-7D69-4DE2-A164-AFD179B66388
```

For compatibility, the bearer token may also be presented in the request query string:

```
GET https://api.icarol.com/v1/Database?access_token=28D3091E…
```

Or using form encoding in the request body: (only when the request uses the `application/x-www-form-urlencoded` content type)

```
POST https://api.icarol.com/v1/Database HTTP/1.1
Host: api.icarol.com:445
Content-Type: application/x-www-form-urlencoded

access_token=28D3091E-7D69-4DE2-A164-AFD179B66388
```

Unauthenticated Requests

If the client attempts to request a resource without presenting a valid token, they will receive an HTTP 401 Unauthorized response with a corresponding WWW-Authenticate header:

```
HTTP/1.1 401 Unauthorized
WWW-Authenticate: Bearer realm="icarol.com";
```

Error Responses

If the server does not accept your bearer token, you will receive an error message in the WWW-Authenticate header. Example cases of an invalid token include: malformed tokens, expired tokens, voluntarily disabled tokens, and revoked tokens. The response body MAY contain additional information.

```
HTTP/1.1 401 Unauthorized
WWW-Authenticate: Bearer realm="icarol.com";error="invalid_token"
Date: Fri, 19 Apr 2013 17:09:36 GMT
Content-Length: 139

{
  "error": "invalid_token",
  "error_description": "The authentication token is malformed.",
  "scope": "icarol.com"
}
```

Database Service

Operations

Get Database Info

Will return information about the databases you have access to.

Request Format

```
GET https://api.icarol.com/v1/Database
```

Response Format

```
HTTP/1.1 200 OK
Content-Type: application/json; charset=utf-8
Date: Wed, 05 Dec 2012 18:41:36 GMT
Content-Length: 19878

[{databaseInfo}]
```

Return Value: An array of *databaseInfo* objects.

Object Definitions

databaseInfo

	Description
{	
"id":*identifier*,	Integer ID of the database.
"owner":"*string*",	Name of the database owner.
"contactDetails": [{*contactMethod*}]	Array of *contactMethod* objects.
"services":["*string*"],	An array of services available for this database. Possible values are: • Email • Resource • ResourceVerification
"coverage":[{*postalAddress*}]	Array of *postalAddress* objects to indicate physical area(s) intended to be covered by this database.
}	

Resource Service

Operations

Search

The search method takes a number of parameters and will return up to 100 results at a time. Use the Take and Skip parameters to do paged results.

Request Format

```
GET https://api.icarol.com/v1/Resource/Search
```

```
Content-Length: [length]
Content-Type: application/json

{searchParameters}
```

> Note that the HTTP specification does not allow for GET requests to contain a body, but some proxy or firewalls may not. You may alternatively use a POST request when calling /v1/Resource/Search.

searchParameters Object

	Req'd?	Description
{		
"term":"*string*",	Yes	Search phrase. (hint: you can use * to return all resources)
"db":[*int*],	No	Limit search scope to specified database IDs. Default: all databases you have access to
"namesOnly":*bool*,	No	If true, match resource names only. Default: false
"resourceType": "*resourceType*",	No	Limit results to resources of a specific type or types. Comma-separated list of the following values: (Default: Any) • Any • Agency • Site • Program
"status":"*string*",	No	Filter results by current resource status. (Any, Active, Inactive) Default: Active
"referenceLocations": [{*physicalLocation*}],	No	List of physical locations to use as reference locations. (array of *physicalLocation* objects, see *contactMethods*) The resource's distance will be calculated from the first element in this array, if defined.
"locationFilter":"*string*",	No	Specifies how the referenceLocations should be used, if defined. Default: None • None: Results will not be filtered by geographic location or coverage area. • Within: Results will contain only resources located within the areas specified. (hint: specify larger regions or areas instead of specific locations) • Serving: Results will contain only resources servicing the areas specified. (hint: specify smaller areas or specific locations) (Hint: See the iCarol website for a description of how these geographic filters work.)
"taxonomyCodes": ["*string*"],	No	Array of 2-1-1 Taxonomy Codes to search within. Taxonomy codes can be as
"taxonomyOperator":	No	Boolean operator to use between *taxonomyCodes*.

"*operator*",		(AND, OR) Default: AND
"taxonomyIncludeChildren": *bool*,	No	Default: true • True: Include results from child terms. • False: Include results only from the specified terms.
"categoryIDs":[*int*],	No	Array of category IDs to search within.
"categoryLinkedIDs": [*int*],	No	
"categoryOperator": "*operator*",	No	Boolean operator to use between *categoryIDs*. (AND, OR) Default: AND
"sort":"*string*",	No	Specifies how the results should be sorted. Default: Alphabetical • Alphabetical: Sort results by resource name. • Proximity: Sort results by proximity to the first *referenceLocation*. • Modifed: Sort results according to when they were previously modified.
"modifiedBefore": "*dateString*",	No	Will return only resources modified on or before the date/time specified. (ISO 8601)
"modifiedSince": "*dateString*",	No	Will return only resources modified on or after the date/time specified. (ISO 8601)
"customFieldValueIDs": [*int*],	No	Array of *customFieldValue* IDs to match. (see the Resource/Meta method)
"take":*int*,	No	The number of results to return. (0-100) Default: 10
"skip":*int*	No	Skip the number of records before returning results. Default: 0
}		

> Note that when the Resource/Search method returns *resourceDetail* objects, the object will not be fully populated. If you need more details about a resource, please make an additional call to Resource/Get by ID for each resource ID you need full details for.

Response Format

```
HTTP/1.1 200 OK
Content-Type: application/json; charset=utf-8
Date: Wed, 05 Dec 2012 18:41:36 GMT
Content-Length: 19878

{searchResult}
```

Return Value: JSON-formatted array of *searchResult* objects.

searchResult Object

	Description
{	
"totalResultCount": *int*,	The total number of results found. Hint: use the Skip and Take parameters to get multiple pages of results.
"referenceLocation": {*physicalLocation*},	If a *referenceLocation* was specified in the search parameters, the location used will be output here.

"results":[{	Array of *searchResult* objects.
"distance": *int*,	If a *referenceLocation* was specified, the resource's distance. (depending on the country of the database being searched: Miles for U.S. databases, Kilometers for all others)
"resource":{*resourceDetail*}	An instance of *resourceDetail*, see below.
}],	
}	

Example Usage

Using jQuery.AJAX to search for resources

> Note that iCarol does not currently support JSONP, so this will not work from your public website. This is here in JavaScript for your reference only.

```
var authToken="dsfjkfjh43j43njke4ernkjerknj"; //put your token here
var searchParams = {
    "term":"food pantry",
    "referenceLocations":[
      {"latitude":3.145, "longitude":86.323},
      {"postalCode":"94553", "stateProvince":"CA", "country":"USA"}
    ],
    "locationFilter":"within",
    "sort":"Proximity",
    "take":10
  };

$.ajax({
  url: "https://api.icarol.com/v1/Resource/Search",
  type: "POST",
  dataType: "json", // expected format for response
  contentType: "application/json", // send searchParams as JSON
  data: searchParams,
  headers: {"Authorization": "Bearer " + authToken},

  complete: function() {
    //called when complete
  },

  success: function() {
    //called when successful
  },

  error: function() {
    //called when there is an error
  },
});
```

Get by ID

Get the *resourceDetails* for a set of resources by specifying the resource IDs. Note that you can pass multiple resource IDs at one time to get back multiple resource details.

Request Format

```
GET https://api.icarol.com/v1/Resource/id
GET https://api.icarol.com/v1/Resource/?id=id1&id=id2&id=id3…
```

Request Parameters

	Req'd?	Description
id=*int*	Yes	Integer resource ID or IDs. A maximum of 10 IDs can be requested at one time.

Response Format

```
HTTP/1.1 200 OK
Content-Type: application/json; charset=utf-8
Date: Wed, 05 Dec 2012 18:41:36 GMT
Content-Length: length

[{resourceDetail}]
```

Return Value: An array of JSON-formatted *resourceDetail* objects.

Taxonomy

Find and retrieve descriptions of the standard 2-1-1 taxonomy code. Requesting information on a single code will return information on that code and all of its child codes.

Request Format

```
GET https://api.icarol.com/v1/Resource/Taxonomy?id=code&id=code&id=...
GET https://api.icarol.com/v1/Resource/Taxonomy?id=code&db=databaseid
                            &resourceCountMin=int&search=text
```

Request Parameters

	Req'd?	Description
id=*taxonomyCode*	No	A single *taxonomyCode* string. (Linked terms must be broken apart first.) If left empty, a list of root terms will be returned. A maximum of 10 IDs can be requested at one time.
db=*databaseID*	No	*DatabaseID* to apply customizations from. Some databases have customized the taxonomy terms for their own usage. If null, the standard taxonomy names and descriptions will be used.
resourceCountMin=*int*	No	If specified, child terms will only be returned if they have at least *int* resources assigned to those terms (or term descendants). If the current context has multiple databases, *db* must also be specified.
resourceDepthMax=int	No	For *resourceCountMin*: if unspecified, default is *any*.
search=*text*	No	If specified, will return a list of taxonomy terms with either a name or a description that matches the given *text*.
searchDepthMax=*int*	No	For the *search* parameters: if unspecified, default is *any*.

Response Format

```
HTTP/1.1 200 OK
```

```
Content-Type: application/json; charset=utf-8
Date: Wed, 05 Dec 2012 18:41:36 GMT
Content-Length: length

[{taxonomyCodeInfo}]
```

Return Value: A JSON-formatted array of *taxonomyCodeInfo* objects.

Categories

Find and retrieve information about resource categories. Requesting information on a single category will return that category and all of its child categories.

Request Format

```
GET https://api.icarol.com/v1/Resource/Categories?id=cid&id=cid&id=...
GET https://api.icarol.com/v1/Resource/Categories?id=cid&db=databaseID
```

Request Parameters

	Req'd?	Description
id=*categoryID*	No	Integer category ID. If null, will return all root categories. A maximum of 10 IDs can be requested at one time.
db=*databaseID*	No	DatabaseID. Required if the current context has access to multiple databases.
resourceCountMin=*int*	No	If specified, child categories will only be returned if they have at least *int* resources assigned to those categories (or category descendants).
search=*text*	No	If specified, will return a list of categories with either a name or a description that matches the given *text*.
depthMax=*int*	No	For the search and resourceCountMin parameters: if unspecified, default is *any*.

Response Format

```
HTTP/1.1 200 OK
Content-Type: application/json; charset=utf-8
Date: Wed, 05 Dec 2012 18:41:36 GMT
Content-Length: length

[{categoryInfo}]
```

Return Value: A JSON-formatted array of *categoryInfo* objects.

Meta

Retrieve the definition of the custom resource fields.

> Currently, this method will only return information about custom fields. Standard fields will not be included in the response.

Request Format

```
GET https://api.icarol.com/v1/Resource/Meta?db={id}&scope=Custom
```

Request Parameters:

	Req'd?	Description
scope=*string*	Yes	Must be "Custom".
db=*databaseID*	Yes	

Response Format

```
HTTP/1.1 200 OK
Content-Type: application/json; charset=utf-8
Date: Wed, 05 Dec 2012 18:41:36 GMT
Content-Length: length

{resourceDefinition}
```

Return Value: A *resourceDefinition* object.

resourceDefinition

	Description
{	
"customFields":[{	Array of *customFieldDef* objects.
"id":*int*,	
"isReadOnly":*boolean*,	
"isRequired":*boolean*,	
"isSearchable":*boolean*,	
"label":"*string*",	The display name of this field.
"type":"*string*",	The type of this field. Possible Values: • Text • MultilineText • Picklist • MultiPicklist
"validForRecordTypes","*string*",	Limit results to resources of a specific type or types. Comma-separated list of the following values: • Any • Agency • Site • Program
"possibleValues":[{	For fields of type *Picklist* or *MultiPicklist*: Array of *customFieldValueDef* objects.
"id":*int*,	The ID of this value. **This value may be used in the *searchParameters.customFieldValueIDs* property.**
"label":"*string*",	The display name of this value.
"isDefault":*boolean*,	
"isUsableInSearchParameters": *boolean*	Indicates that this value may be used in the searchParameters.customFieldValueIDs property. (see Resource/Search)
}],	
"minSelectionCount":*int*,	
"maxSelectionCount":*int*	
}]	
}	

Update

Request Format

```
1  PATCH https://api.icarol.com/v1/Resource/{id}
2  Content-Type: application/json; charset=utf-8
3  Content-Language: en-US
4  Date: Wed, 05 Dec 2012 18:41:36 GMT
5  Content-Length: length
6
7  {resourceDetailDelta}
```

This method will accept a partial *resourceDetail* object and update only the fields sent. When updating fields belonging to child objects, you must pass the entire child object. Note that for *contactDetail* objects, you must include the type and purpose properties and that they must match the appropriate value.

> Note that this method is not fully implemented yet. It will only work for Mailing Address and Physical Location.

Specifying Content Language

(see line 3 in the Request Format example) If you wish to update any text value translations other than the resource's default language, you must specify a `Content-Language` header. (Note that it is different from the `Accept-Language` header in that you can only specify a single value rather than a list of accepted values.)

> Not yet Implemented: All updates will currently be applied to the resource's default language.

Response Format

```
HTTP/1.1 200 OK
Date: Wed, 05 Dec 2012 18:41:36 GMT
```

The response body will be empty.

Example Usage

Updating a Resource's Status

```
PATCH https://api.icarol.com/v1/Resource/554565
Authorization: Bearer authToken
Content-Type: application/json; charset=utf-8
Content-Language: en-US
Date: Wed, 05 Dec 2012 18:41:36 GMT
Content-Length: length

{
    "id":554565,
    "databaseID":2222,
    "status":"Inactive"
}
```

Updating a Resource's PostalAddress

```
PATCH https://api.icarol.com/v1/Resource/554565
Authorization: Bearer authToken
```

```
Content-Type: application/json; charset=utf-8
Content-Language: en-US
Date: Wed, 05 Dec 2012 18:41:36 GMT
Content-Length: length

{
   "id":554565,
   "databaseID":2222,
   "contactDetails":[{
      "contact":{
         "type":"PhysicalLocation",
         "purpose":"Physical",
         "line1":"P.O. Box 5001",
         "city":"Albuquerque",
         "stateProvince":"NM",
         "zipPostalCode":"87153"
         "latitude":35.110833,
         "longitude":-106.61
      }
   }]
}
```

Updating a Custom Field

```
PATCH https://api.icarol.com/v1/Resource/554565
Authorization: Bearer authToken
Content-Type: application/json; charset=utf-8
Content-Language: en-US
Date: Wed, 05 Dec 2012 18:41:36 GMT
Content-Length: length

{
   "id":554565,
   "databaseID":2222,
   "customFields":[
      {
         "id":555,
         "valueText":"New Value"
      },
      {
         "id":556,
         "selectedValues":{
            5344:"Value A",
            5345:"Value B"
         }
      }
   ]
}
```

Object Definitions

resourceDetail

> Note that when the Resource/Search method returns *resourceDetail* objects, the object will not be fully populated. If you need more details about a

> resource, please make an additional call to Resource/Get by ID for each resource ID you need full details for.

	Description
{	
"id":"*identifier*",	
"databaseID":*int*,	Database ID
"names":[{	Array of *resourceName* objects.
"value":"*string*",	Name of the resource
"purpose":"*string*",	The purpose of this name. Typically one of these values: • Primary • Alternate • Official
"language":"*languageCode*"	If this name is specific to a particular language, a language code following the IETF RFC 4646 standard.
}],	
"type":"*resourceType*",	Type of the resource. • Agency • Program • Site • ProgramAtSite (Hint: see the iCarol website for a description of how these types interact)
"status":"*string*",	Current status of this resource. Common values: • Active • Inactive
"taxonomy":["*taxonomyCode*"],	Array of *taxonomyCode* strings which may be linked by joining multiple terms together in one string separated by the delimiter " * ".
"categories":[*categoryID*],	Array of *categoryID* integers.
"description":"*string*",	Primary description of the resource. Note: **This field can contain HTML.**
"descriptionText":"*string*",	Primary description of the resource. Note: **This field contains the same text as *description*, but any HTML has been stripped but may still contain line breaks.**
"searchHints":["*string*"],	A list of keywords used during resource searching.
"temporaryMessage":"*string*",	If defined, contains a message that should be displayed along with the resource description.
"verificationExpired":"*dateTime*",	If defined, the date on which this resources' previous validation has expired. This resource record may contain obsolete or incorrect information. (ISO 8601)
"legalStatus":"*string*",	If defined, the legal status of this resource.
"related":[{	Array of *resourceRelationship* objects.
"id":"*identifier*",	Resource ID
"name":"*string*",	
"type":"*resourceType*"	
}],	

"contactDetails":[{	Array of *contactDetail* objects.
"isConfidential":*boolean*,	**If true, this *contactMethod* should not be exposed to public users.**
"contact":{*contactMethod*},	A single *contactMethod* object. See Below.
}],	
"languagesOffered":"*string*"	Describes the language services offered by this resource. (**Plain Text**)
"isFeatured":*boolean*,	If true, this resource record should be displayed at the top of search results.
"isConfidential":*boolean*,	**If true, this resource should not be exposed to public users.**
"coverage":[{*postalAddress*}],	Array of *postalAddress* objects denoting the areas serviced by this resource. See *contactMethods*.
"coverageNote":"*string*",	
"specialNotes":["string"],	
"modified":"*dateTime*",	Date/Time when this resource was last modified. (UTC, ISO 8601)
"hours":{*resourceHours*},	A single *resourceHours* object.
"notes":{ "*type*":"*string*", … },	If defined, special notes fields for internal staff of various *types*. • Editors • Internal
"accessibility":{ "*type*":"*string*" },	If defined, accessibility details of various *types*: • Bus • Disabled • Public
"fees":"*string*",	
"requiredDocumentation":"*string*",	
"applicationProcess":"*string*",	
"customFields":[{	An array of *customFieldValue* objects.
"id":*int*,	The ID of this field definition
"label":"*string*",	The display name of the field.
"valueText":"*string*",	For text fields.

"selectedValues":{
"*int*":"*string*",
…
}
	For checkboxes, radio buttons and multi-selects, a dictionary of key-value pairs. **Note that this is NOT an array, but may have multiple key/value pairs.**

}],	
"fein":"*string*",	Federal EIN number
"eligibility":"string",	
"capacity":"string",	
"sourceOfFunds":"string",	
"volunteering":{	If defined, a *volunteerInfo* object.
"opportunitiesAvailable": *boolean*,	
"requirements":"*string*",	
"duties":"*string*"	
},	

`"translations":["string"]`	A list of languages for which complete or partial translations of this record exist. (IETF RFC 4646 standard language codes)
`}`	

categoryInfo

	Description
`{`	
`"id":"identifier",`	
`"databaseID": int,`	
`"name":"string",`	
`"description":"string",`	
`"taxonomyCode":"string",`	If defined, this category's equivalent 2-1-1 Taxonomy Code.
`"visible":boolean,`	When false, this category should not be displayed. (Default: true)
`}`	

taxonomyCodeInfo

Valid 2-1-1 Taxonomy Codes must match this regular expression:

```
^[A-Z]{1,2}(\-\d{4}?(?:\.\d{4}(?:\-\d{3}(?:\.\d{1,4})?)?)?)?$
```

	Description
`{`	
`"code":"string",`	A 2-1-1 Taxonomy Code
`"databaseID":int,`	If defined, indicates that this record was customized by the specified database owner.
`"name":"string",`	Display Name of this taxonomy code.
`"path":"string",`	A description of this code's full hierarchy. Terms names are comma-separated.
`"linkedCodes":["string"]`	Array of taxonomy code strings. (see *code* parameter)
`"definition":"string",`	Description of the taxonomy code.
`"synonyms":["string"]`	A list of alternate terms that can be used to describe this Taxonomy Code.
`}`	

resourceHours

	Description
`{`	
`"notes":"string",`	
`"days":[{`	Array of no more than 7 *resourceHoursDetail* objects.
`"dayOfWeek":"string",`	A 3-letter abbreviation for the day of the week.
`"opens":"string",`	
`"closes":"string",`	
`"notes":"string"`	
`}]`	
`}`	

Export Field Mapping

Most users use iCarol to export Excel or CSV views of their resources. The organization of data is, necessarily, different due to the columnar nature of CSV exports. The iCarol API can communicate using richer data structures and as such, has no such limitation. Not all data is yet represented in the iCarol API; more will be added over time. Here is a brief summary of how to match CSV export data to the API structure.

> The mapping below uses XPath notation, even when the response is given in JSON format.
>
> Where the resourceDetail column is empty, the property is not yet present in the iCarol API.

CSV Field	*resourceDetail* Object Property
ResourceAgencyNum	/id
PublicName	/names[@purpose="Primary"]/value
AlternateName	/names[@purpose="Alternate"]/value
OfficialName	/names[@purpose="Official"]/value
TaxonomyLevelName	/type
ParentAgency	/related[@type="Agency"]/name
ParentAgencyNum	/related[@type="Agency"]/id
RecordOwner	/databaseID
UniqueIDPriorSystem	
MailingAttentionName	/contactDetails/contact[@type="postalAddress" && @purpose="Mailing"]/attention
MailingAddress1	/contactDetails/contact[@type="postalAddress" && @purpose="Mailing"]/address1
MailingAddress2	/contactDetails/contact[@type="postalAddress" && @purpose="Mailing"]/address2
MailingCity	/contactDetails/contact[@type="postalAddress" && @purpose="Mailing"]/city
MailingStateProvince	/contactDetails/contact[@type="postalAddress" && @purpose="Mailing"]/stateProvince
MailingPostalCode	/contactDetails/contact[@type="postalAddress" && @purpose="Mailing"]/zipPostalCode
MailingCountry	/contactDetails/contact[@type="postalAddress" && @purpose="Mailing"]/country
MailingAddressIsPrivate	/contactDetails/isPrivate[contact/@type="postalAddress" && contact/@purpose="Mailing"]
PhysicalAddress1	/contactDetails/contact[@type="postalAddress" && @purpose="Physical"]/address1
PhysicalAddress2	/contactDetails/contact[@type="postalAddress" && @purpose="Physical"]/address2
PhysicalCity	/contactDetails/contact[@type="postalAddress" && @purpose="Physical"]/city
PhysicalStateProvince	/contactDetails/contact[@type="postalAddress" && @purpose="Physical"]/stateProvince

PhysicalPostalCode	/contactDetails/contact[@type="postalAddress" && @purpose="Physical"]/zipPostalCode
PhysicalCountry	/contactDetails/contact[@type="postalAddress" && @purpose="Physical"]/country
PhysicalAddressIsPrivate	/contactDetails/isPrivate[contact/@type="postalAddress" && contact/@purpose="Physical"]
OtherAddress1	/contactDetails/contact[@type="postalAddress" && @purpose=null]/address1
OtherAddress2	/contactDetails/contact[@type="postalAddress" && @purpose=null]/address2
OtherCity	/contactDetails/contact[@type="postalAddress" && @purpose=null]/city
OtherStateProvince	/contactDetails/contact[@type="postalAddress" && @purpose=null]/stateProvince
OtherPostalCode	/contactDetails/contact[@type="postalAddress" && @purpose=null]/ zipPostalCode
OtherCountry	/contactDetails/contact[@type="postalAddress" && @purpose=null]/ country
OtherAddressIsPrivate	/contactDetails/isPrivate[contact/@type="postalAddress" && contact/@purpose=null]
Latitude	/contactDetails/contact[@type="physicalLocation" && @purpose="Physical"]/latitude
Longitude	/contactDetails/contact[@type="physicalLocation" && @purpose="Physical"]/longitude
HoursOfOperation	/hours/note
Phone1Number	(/contactDetails/contact[@type="phoneNumber"]/number)[1]
Phone1Name	(/contactDetails/contact[@type="phoneNumber"]/label)[1]
Phone2Number	(/contactDetails/contact[@type="phoneNumber"]/number)[2]
Phone2Name	(/contactDetails/contact[@type="phoneNumber"]/label)[2]
Phone3Number	(/contactDetails/contact[@type="phoneNumber"]/number)[3]
Phone3Name	(/contactDetails/contact[@type="phoneNumber"]/label)[3]
Phone4Number	(/contactDetails/contact[@type="phoneNumber"]/number)[4]
Phone4Name	(/contactDetails/contact[@type="phoneNumber"]/label)[4]
Phone5Number	(/contactDetails/contact[@type="phoneNumber"]/number)[5]
Phone5Description	(/contactDetails/contact[@type="phoneNumber"]/label)[5]
PhoneFax	/contactDetails/contact[@type="phoneNumber" && @purpose="Fax"]/number
PhoneTTY	/contactDetails/contact[@type="phoneNumber" && @purpose="TTY"]/number
PhoneTollFree	/contactDetails/contact[@type="phoneNumber" && @purpose="TollFree"]/number
PhoneNumberHotline	
PhoneNumberBusinessLine	
PhoneNumberOutOfArea	
PhoneNumberAfterHours	/contactDetails/contact[@type="phoneNumber" && @purpose="AfterHours"]/number
EmailAddressMain	/contactDetails/contact[@type="email" && @label="Main"]/emailAddress
WebsiteAddress	/contactDetails/contact[@type="website"]/url

AgencyStatus	/status
AgencyClassification	
AgencyDescription	/description
SearchHints	
CoverageArea	/coverageNote
CoverageAreaText	/coverage
Eligibility	/eligibility
SeniorWorkerName	/contactDetails/contact[@type="person" && @label="Senior Worker"]/name
SeniorWorkerTitle	/contactDetails/contact[@type="person" && @label="Senior Worker"]/title
SeniorWorkerEmailAddress	/contactDetails/contact[@type="person" && @label="Senior Worker"]/contactMethods/[@type="email"]/emailAddress
SeniorWorkerPhoneNumber	/contactDetails/contact[@type="person" && @label="Main"]/contactMethods/[@type="phoneNumber"]/number
SeniorWorkerIsPrivate	/contactDetails[contact@type="person" && contact@label="Senior Worker"]/isPrivate
MainContactName	/contactDetails/contact[@type="person" && @label="Main"]/name
MainContactTitle	/contactDetails/contact[@type="person" && @label="Main"]/title
MainContactEmailAddress	/contactDetails/contact[@type="person" && @label="Main"]/contactMethods/[@type="email"]/emailAddress
MainContactPhoneNumber	/contactDetails/contact[@type="person" && @label="Main"]/contactMethods/[@type="phoneNumber"]/number
MainContactType	
MainContactIsPrivate	/contactDetails[contact@type="person" && contact@label="Main"]/isPrivate
LicenseAccreditation	
IRSStatus	
FEIN	/fein
YearIncorporated	
AnnualBudgetTotal	
LegalStatus	/legalStatus
SourceOfFunds	/sourceOfFunds
ExcludeFromWebsite	/isConfidential
ExcludeFromDirectory	
DissabilitiesAccess	/accessibility/disabled
PhysicalLocationDescription	
BusServiceAccess	/accessibility/bus
PublicAccessTransportation	/accessibility/public
FeeStructureSource	/fees
ApplicationProcess	/applicationProcess
ResourceInfo	
DocumentsRequired	/requiredDocumentation
LanguagesOffered	/languagesOffered
ServiceNotAlwaysAvailability	
CapacityType	Combined into /capacity

ServiceCapacity	
TemporaryMessage	/temporaryMessage
TemporaryMessageAppears	
TemporaryMessageExpires	
EnteredOn	
UpdatedOn	/modified
MadeInactiveOn	
InternalNotes	/notes/internal
InternalNotesForEditorsAndViewers	/notes/editors
HighlightedResource	
LastVerifiedOn	Computed and, if the verification frequency has lapsed, the verification date will be in /verificationExpired
LastVerifiedByName	
LastVerifiedByTitle	
LastVerifiedByPhoneNumber	
LastVerifiedByEmailAddress	
LastVerificationApprovedBy	
AvailableForDirectory	
AvailableForReferral	
AvailableForResearch	
PreferredProvider	/isFeatured
ConnectsToSiteNum	/related[@type="Site"]/id
ConnectsToProgramNum	/related[@type="Program"]/id
LanguageOfRecord	HTTP Response Header: Content-Language
Categories	/categories/id
TaxonomyTerm	
TaxonomyTerms	Use the /Resource/Taxonomy service.
TaxonomyTermsNotDeactivated	
TaxonomyCodes	/taxonomy
Coverage	
Hours	/hours/notes
Custom_...	/customFields

ResourceVerification Service

The ResourceVerification service is used to verify and keep resources up-to-date by involving resource operators in the verification process.

Operations

VerificationJob Create

This method will accept a list of resource IDs for which to initiate the resource verification process.

Request Format

```
POST https://api.icarol.com/v1/ResourceVerification/RequestNewJob
Content-Length: [length]
Content-Type: application/json

{verificationJobRequest}
```

Verification Job Creation Body Parameters

	Req'd?	Description
{		
"resourceIDs":[*int*],	Yes	Array of resource IDs for which to start verification. **Resources must all be from the same database.**
"emailTemplateID":*int*,	Yes	The ID of the *emailTemplate* to use. (See Email/Templates)
"databaseID":*int*,	No	The databaseID that all of the above resources belong to. Required if the current user has access to multiple datbases.
"fromName":"*string*"	No	The name of the sender that should appear in the outgoing verification emails.
}		

Response Format

```
HTTP/1.1 201 CREATED
ETag: "{id: 1234}"

{verificationJob}
```

Object Definitions

verificationJob

	Description
{	
"id":"*identifier*",	
"emailTemplateID": *int*,	
"requestorDatabaseID":*int*,	
"requestedBy":*int*,	
"requestedOn":*dateTime*,	
"status":"*string*",	
"resourceIDs":[*int*],	
"fromEmailAddress":"*string*",	
"fromName":"*string*"	
}	

verificationRequest

	Description
{	
}	

verificationResponse

	Description
{	
}	

Email Service

Operations

Templates

Request Format

```
GET https://api.icarol.com/v1/Email/Templates/
GET https://api.icarol.com/v1/Email/Templates/id
GET https://api.icarol.com/v1/Email/Templates/?id=id&id=id&id=...
```

Request Parameters

	Req'd?	Description
id=*int*	No	Integer template ID or IDs. A maximum of 10 IDs can be requested at one time.

Response Format

```
HTTP/1.1 200 OK
Content-Type: application/json; charset=utf-8
Date: Wed, 05 Dec 2012 18:41:36 GMT
Content-Length: 19878

[{emailTemplate}]
```

Return Value: An array of *emailTemplate* objects.

emailTemplate Object

	Description
{	
"id": *int*,	
"databaseID":*int*,	
"name":"*string*",	
"fromName": "*string*",	
"fromEmailAddress":"*string*",	
"cc":"*string*",	
"bcc":"*string*",	
"subject":"*string*",	
"bodyHtml":"*string*",	
"category":"*string*"	If specified, the display category containing this template.
}	

Messaging Service

Operations

Status

Request Format

```
GET https://api.icarol.com/v1/Messaging/
GET https://api.icarol.com/v1/Messaging/?db=id
```

Request Parameters

	Req'd?	Description
db=*int*	No*	If the ApiKey used has access to multiple databases, you must specify the database ID of the messaging instance to work with.

Response Format

```
HTTP/1.1 200 OK
Content-Type: application/json; charset=utf-8
Date: Wed, 05 Dec 2012 18:41:36 GMT
Content-Length: 19878

{messagingStatus}
```

Return Value: A single *messagingStatus* object.

messagingStatus Object

	Description
{	
"databaseID":*int*,	
"portals":[{	If this messaging instance has multiple portals, this will contain an array of *portalStatus* objects.
"portalID":*int*,	The ID of the Portal.
"name":"*string*",	
"registrationUrl":"*url*",	
"conversations":[{	A list of currently-active conversations.
"started":*dateTime*,	The date/time that this conversation was initiated. (ISO 8601)
"ended":*dateTime*,	The date/time that this conversation was either ended manually or automatically timed out. (ISO 8601)
"lastActive":*dateTime*,	The date/time of the last message sent or received as part of this conversation. (ISO 8601)
"messageCount":*int*,	The number of messages sent as part of this conversation.
"workerID":*int*,	If this chat has been assigned to a worker, the ID of the worker currently servicing this chat.
"channel":"*string*"	A string indicating the conversation channel. Current values: • "icm": Web-based chat. • "sms": Chat via mobile SMS.

`}],`	
`"availableWorkers":[{`	An array of *workerStatus* objects listing the currently available workers.
`"id":int,`	The Worker's ID.
`"activeConversationCount": int`	The number of currently active conversations the worker is participating in.
`}]`	
`}]`	
`}`	

Common to All Services

All iCarol API services will share these common definitions. The iCarol API is written on top of the HTTP protocol specification and inherits a lot of power from HTTP. Some of these are detailed below.

Status Codes

All API methods use the common REST convention of returning API call success and error status using standard HTTP/1.1 protocol response status codes. Some common codes are:

> For more detailed listing of all possible status codes, please see RFC 2616 section 10.

2xx Success Status Messages

200 OK

The request has succeeded.

201 Created

The request has been fulfilled and resulted in a new resource being created.

202 Accepted

The request has been accepted for processing, but the processing has not been completed. There is no need to re-send your request.

204 No Content

Your request has been fulfilled, but there is no content to send to you. There may be no results or the requested resource may be empty.

3xx Redirection Messages

301 Moved Permanently

Please update your source with the new URI and reissue your request to the URI specified in the `Location` header.

304 Not Modified

See Writing Cache-Aware Applications.

307 Temporary Redirect

The request should be re-sent to the new URI specified in the `Location` header, but the source need not be updated.

4xx Client Error Messages

400 Bad Request

There is a problem with the request or the request parameters.

401 Unauthorized

You are either not authenticated or not authorized to perform the requested operation.

403 Forbidden

You are not authorized to perform the requested operation on the specified resource.

404 Not Found

The requested resource does not exist.

5xx Server Error Messages

500 Internal Server Error

This is a general catchall for all possible server-side errors encountered while processing your request. Please see the response body for more detailed error information, if available.

501 Not Implemented

The request was accepted but could not be completed at this time. The service that you requested does not exist.

503 Service Unavailable

Temporary conditions prevent us from processing your request at this time. You may re-issue your request at a later time.

Language Preference

You can specify your preferred language using the standard `Accept-Language` HTTP request header (see the W3C HTTP Specification for more info):

```
GET https://api.icarol.com/ HTTP/1.1
Host: api.icarol.com:445
Accept-Language: en-US,en;q=0.8,*
```

You can specify multiple languages and q-numbers. For fields that iCarol has translations for, iCarol will return data with fields in the language specified. If no translation for a field exists, iCarol will attempt to fill the field with translations in the order specified. If no translations are available for any of the specified languages, the fields will be empty. The `*` value should be used at the end of the `Accept-Languages` header to indicate that the client will accept any language (typically the Resource or Database's principal language) and ensure that all possible fields have values regardless of the language of the value.

Language values should be in the format `languagecode2-scripttag-country/regioncode2` according to IETF RFC 4646.

languagecode2	A two-letter code derived from ISO 639-1.

scripttag	An optional 4-letter ISO 15924 code describing the written script.
country/regioncode2	An optional (usually 2-letter) ISO 3166-1 code describing the country or region.

Response Compression

Compression is achieved by using the Accept-Encoding HTTP request header:

```
GET https://api.icarol.com/ HTTP/1.1
Host: api.icarol.com:445
Accept-Encoding: gzip,deflate,sdch
```

The HTTP response stream will only be compressed if the server supports one or more requested encoding scheme.

Character Encoding

Character encoding is especially important for non-English language text. We recommend that clients use UTF-8 encoding, when supported, but since not all clients will support UTF character encoding, you may specify alternative encodings if necessary.

```
GET https://api.icarol.com/ HTTP/1.1
User-Agent: Fiddler
Host: api.icarol.com:445
Accept-Charset: utf-8,ISO-8859-1;q=0.7,*;q=0.3
```

Response Formatting

The iCarol API supports returning data in XML or JSON formats. The layout of the objects will be the same, or similar. XML is sometimes more preferable in some environments. Note that the XML returned will not have an XSD or be strongly-typed. Use the Accept HTTP request header to specify your preference for XML or JSON.

```
GET https://api.icarol.com/ HTTP/1.1
Host: api.icarol.com:445
Accept: application/json,text/json,text/xml;q=0.9
```

Currently, only `application/json` or `text/json` are supported. Future support for text/xml is being considered.

> To ensure that your application is future-compatible and that your application continues receiving your preferred response format, you should always include an HTTP Accept header value.

Writing Cache-Aware applications

If your client has built-in Caching, or does any client-side data caching, you can make use of standard HTTP cache control headers.

> Not yet implemented. Clients may include or expect these headers but not expect any difference yet in the response.

Cache-Aware Request

```
GET https://api.icarol.com/ HTTP/1.1
Host: api.icarol.com:445
If-Modified-Since: Fri, 25 Jan 2013 20:10:50 GMT
```

Cache-Aware Response for Unmodified resource

```
HTTP/1.1 304 Not Modified
Date: Fri, 25 Jan 2013 21:26:12 GMT
…
Cache-Control: max-age=604800
Last-Modified: Fri, 25 Jan 2013 20:10:50 GMT
```

Metrics and Tracking

iCarol will log and track API requests for each API key along with the preferred language and client agent used. To allow these metrics to be as useful as possible for your organization, please populate the HTTP Request Agent header:

```
GET https://api.icarol.com/v1/Database HTTP/1.1
User-Agent: Elmdale Helpline Resource Search App (iOS; iPhone; Retina)
Host: api.icarol.com:445
```

Error Responses

Error Responses will contain, when available, an *errorInfo* object. You should first check the response status code. See RFC 2616 for a description of the standard HTTP status codes.

	Description
{	
"error":"*string*",	A single, machine readable error code.
"error_title":"*string*",	A human readable error title.
"error_description":"*string*",	A brief description of the error.
"error_uri":"*uri*",	If available, a URL where detailed information about the error can be located.
"state":"*string*",	
"exception": {*exceptionInfo*},	
"argument_errors": [{*errors*}]	
}	

Logging Request Details

All requests to the iCarol API may be logged for internal statistics, but API Users may, additionally, save full Request/Response transcripts for a limited time.

> Note that the entire HTTP Request and HTTP Response will be saved along with all HTTP Headers, client information, request parameters, and full response details. This information almost certainly contains user-identifying information. These details will be available only to members of your organization and, with your permission, to iCarol technical support staff.

Enabling Request Detail Logging

For all requests made using a specific API Key

You may log into iCarol and use your API Management screen to enable Request Detail Logging. This will save full transcripts of all API requests made which contain your API Key. This setting will be on for a limited time before turning itself off again.

For only a single request

When issuing an API request, you may specify an optional `X-iCarol-LogRequestDetails` HTTP header.

```
GET /v1/version HTTP/1.1
Authorization: Bearer your_api_key
Host: api.icarol.com
X-iCarol-LogRequestDetails: true
```

Valid values for the `X-iCarol-LogRequestDetails` header are any Boolean values: "True", "False", "1", or "0".

Reviewing previously logged requests

From within your iCarol instance, go to your API Management screen to review any saved API Request/Response transcripts.

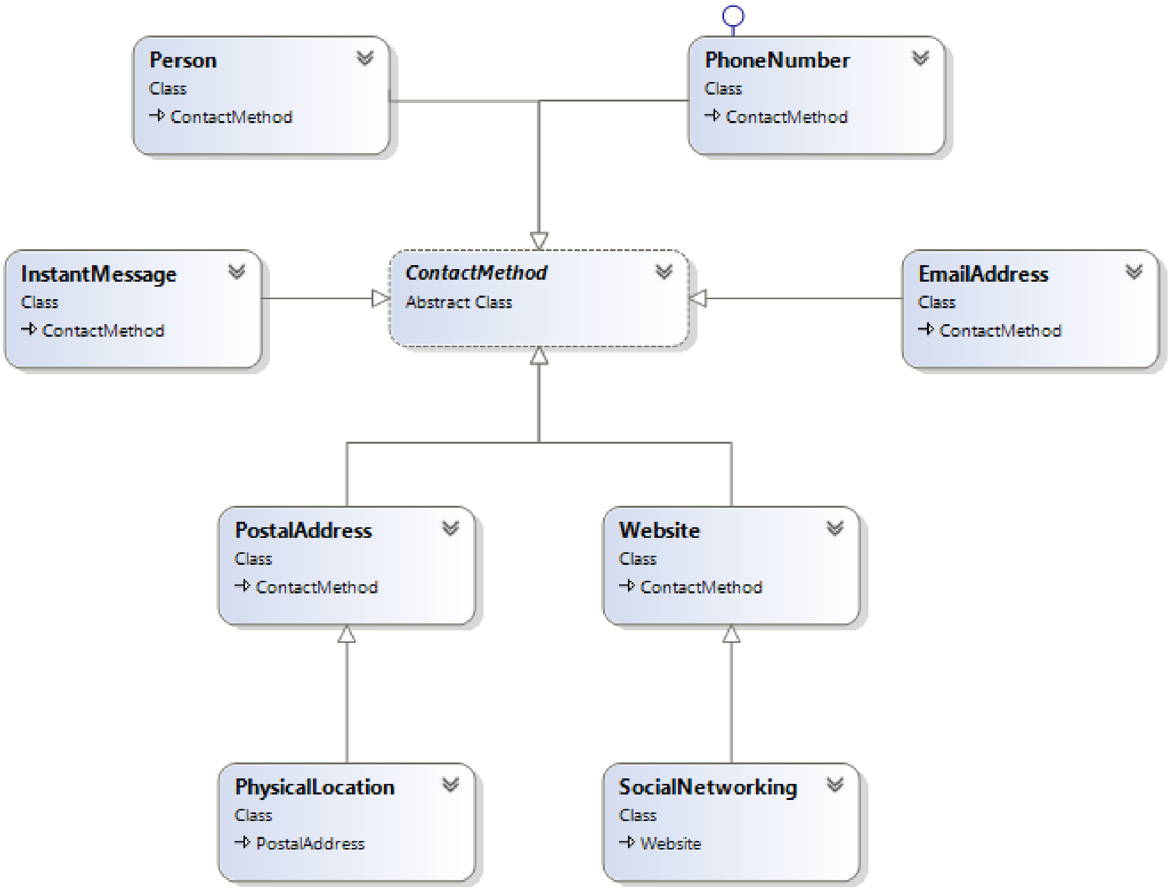
Request Detail Log Expiration

Requests logged using the Request Details feature will automatically expire after 48 hours and be purged from the iCarol system. You may override this setting by passing a Date/Time in the `X-iCarol-LogRequestDetailsExpires` HTTP header:

```
GET /v1/version HTTP/1.1
Authorization: Bearer your_api_key
Host: api.icarol.com
X-iCarol-LogRequestDetails: true
X-iCarol-LogRequestDetailsExpires: Thu, 10 Apr 2008 13:30:00 GMT
```

The date in the `X-iCarol-LogRequestDetailsExpires` header must conform to IETF RFC 1123 (Also known as HTTP date format) and must be no more than 7 days into the future.

Contact Method Objects



contactMethod

All *contactMethod* objects share these properties:

	Description
{	
"label":"*string*",	
"description":"*string*",	
"notes":"*string*"	
"type":"*contactType*",	The type of the *contactMethod* object. See below for a list of possible types.
}	

person

	Description
{	
"type":"person",	
Inherits from contactMethod	
"name":{	*personName* object
"firstName":"*string*",	
"surName":"*string*",	
"displayName":"string",	
"title":"*string*",	Personal title. Not to be confused with job title.
"prefix":"*string*",	
"suffix":"*string*",	
"nickname":"*string*"	
},	
"titles":["*string*"],	Array of string titles held by the *person*.
"companyName":"*string*",	
"contactMethods":[*contactMethod*]	Array of direct contact methods.
}	

postalAddress

	Description
{	
"type":"postalAddress",	
Inherits from contactMethod	
"purpose":"*postalPurposeString*",	Typical values: • Mailing • Physical • Home • Office • ReferenceLocation • CoverageArea
"careOf":"*string*",	
"attention":"*string*",	
"recipient":"*string*",	
"company":"*string*",	
"department":"string"	
"line1":"*string*",	Address Line 1
"line2":"*string*",	Address Line 2
"county":"*string*",	*formal*, County or Parish name
"district":"*string*",	*informal*, District (within a city)
"city":"*string*",	City name
"region":"*string*",	*informal*, Region (larger than city)
"stateProvince":"*string*",	State or Province abbreviation
"zipPostalCode":"*string*",	Zip or Postal code
"country":"*string*"	Country name or 2-letter code (ISO 3166-1 alpha-2)
}	

physicalLocation

Inherits all properties from *postalAddress* and *contactMethod*, above.

	Description
{	
"type":"physicalLocation",	
Inherits from contactMethod	
Inherits from postalAddress	
"latitude":*double*,	(-90 to 90, where 0 is the equator)
"longitude":*double*,	(-180 to 180, where 0 is the Prime Meridian)
"precision":"*string*",	General indication of the precision of the lat/long. • GeneralArea • AddressGeocode • Precise • Unknown
"source":"*string*"	Optional, the source of the Lat/Long measurement.
}	

website

Inherits all properties from *contactMethod*, above.

	Description
{	
"type":"website",	
Inherits from contactMethod	
"url":"*string*"	
}	

socialNetworking

Inherits all properties from *website* and *contactMethod*, above.

	Description
{	
"type":"socialNetworking",	
Inherits from contactMethod	
Inherits from website	
"network":"string",	Name of the Social Network
"name":"*string*"	Name of the Profile
}	

email

Inherits all properties from *contactMethod*, above.

	Description
{	
"type":"email",	
Inherits from contactMethod	
"emailAddress":"*string*"	
}	

phoneNumber

Inherits all properties from *contactMethod*, above.

	Description
{	
"type":"phoneNumber",	
Inherits from contactMethod	
"purpose":"*phonePurpose*",	Typical values: • Main • Direct • Fax • TTY • TollFree • AfterHours • Mobile • Home • Pager
"number":"string",	
"extension":"string",	
"isMobile":*boolean*,	
"isTTY":*boolean*,	
"isFax":*boolean*,	
"isTollFree":*boolean*	
}	