REST API Specification

interact.io

Mr. Michael Zirngibl

V2.0

Created by: interact.io Scrum Team

interact.io Cloud Solutions GmbH, Pappelallee 78-79, 10437 Berlin, Germany, Telefon: +49 (30) 2089 828-0,
Geschäftsführer: Michael Zirngibl, HRB 146195 B / Berlin-Charlottenburg

Current version: V2.0 (← Revisionsdaten bitte hier ändern)

Following changes have been made in order to create the current version:

Change History Overview

Date	Change	Version	Status
12.12.2013	Added CRUD pictures; Remo Zellmer	-	draft
13.12.2013	Added POST /admin/users/:id?cmd=wipeAccount; Remo Zellmer	-	draft
17.12.2013	Updated /interactions calls, Remo Zellmer	-	draft
06.01.2014	Added GET /triggerDevices?deviceId=:id; Remo Zellmer	-	draft
07.01.2014	Fixed confusing naming for authToken <> triggerToken; deviceToken <> deviceId; Remo Zellmer	-	draft
09.01.2014	Added CRUD custom fields; Remo Zellmer	-	draft
28.07.2014	Added JSONP parameters; Johannes Selig	-	draft

Table of contents

1 About this Document

This document describes the REST API of the interact.io platform. For basic understanding we distinguish 3 types of components:

1. The Trigger Device that triggers the actual event,
2. The Server and
3. The Device that is the end user application.

In the Overview chapter you will find a general overview of all provided API calls. Each of these API calls is documented in a chapter below. The documentation of each API call is based on a table format and is structured in the following way:

<table>
<tr><td><b>Title</b></td><td colspan="3"><The name of the API call></td></tr>
<tr><td><b>URL</b></td><td colspan="3"><The URL structure – path only, no root url.></td></tr>
<tr><td><b>Header attributes</b></td><td colspan="3"><Header attributes with names and values></td></tr>
<tr><td><b>Method</b></td><td colspan="3"><The request type – DELETE | GET | POST | PUT></td></tr>
<tr><td rowspan="2"><b>URL parameters</b></td><td colspan="3"><URL parameters if exist, specified in accordance with name mentioned in URL section></td></tr>
<tr><td><Parameter name = Parameter data type></td><td><Required marker></td><td><Example></td></tr>
<tr><td><b>Data</b></td><td colspan="3"><Payload for POST and PUT requests with data types, optional attributes and nodes are in lighter color></td></tr>
<tr><td><b>Success response</b></td><td colspan="3"><HTTP return code and response data with data types, optional attributes and nodes are in lighter color. The <i>entity</i> attribute contains the whole object.></td></tr>
<tr><td><b>Error response</b></td><td colspan="3"><Possible HTTP return codes and response data with data types, optional attributes and nodes are in lighter color></td></tr>
<tr><td><b>Code example</b></td><td colspan="3"><AJAX example to make life easier and more predictable ;-)></td></tr>
<tr><td><b>Notes</b></td><td colspan="3"><Section where all uncertainties and commentary go.></td></tr>
</table>

Table 1 API call documentation template

In case one of the sections does not apply to a particular API call, the table cell value is *none*.

2 API Specification

Note: The Server's root URL is https://api.interact.io/v2.

2.1 Prerequisites

The character set used for encoding response data and decoding request data is UTF-8.

2.2 General API Pattern

In general the following simple pattern is used in the API:

- Nouns are used as resource names.
- Verbs (http methods) are used to operate on entities.

CRUD (**C**reate-**R**ead-**U**pdate-**D**elete) operations are represented by the http methods PUT, GET, POST and DELETE. Entities are defined by integer IDs.

Resource	**PUT** create	**GET** read	**POST** update	**DELETE** delete
/v2/foos	Create new foo	List foos	Bulk update foos	Delete foos
/v2/foos/:id	ERROR	Show foo	If exists update foo, If not exists ERROR	Delete foo if exists
/v2/foos/:id/bars	Create new bars associated to foo	Show bars associated to foo	Associate all bars to foo and replace former associations	Delete associations from foo to bars

Table 2 General API pattern

In addition there are some extended functions available. These functions are represented by a query parameter named *cmd*.

The API is able to respond to GET requests in "JSON with padding" format (JSONP). This function is represented by a query parameter named *callback*. See *http://www.json-p.org/* for further informations.

2.3 Overview

API call	**PUT**	**GET**	**POST**	**DELETE**
/v2/activities		X		
/v2/admin/companies	X	X		X

/v2/admin/companies/:id		X		
/v2/admin/companies/:id/customFields		X		
/v2/admin/companies/:id/groups		X		
/v2/admin/customFields	X			X
/v2/admin/customFields/:id		X	X	
/v2/admin/groups	X			X
/v2/admin/groups/:id		X	X	X
/v2/admin/users	X	X		X
/v2/admin/users/:id		X	X	X
/v2/admin/users/:id/groups			X	
/v2/admin/users/:id/roles			X	
/v2/basicInfos	X			
/v2/basicInfos/:id			X	
/v2/cleanup		X		
/v2/contacts	X	X	X	X
/v2/contacts/:id		X	X	X
/v2/contacts/:id/contacts		X		
/v2/contacts/:id/deals		X		
/v2/contacts/:id/interactions		X		
/v2/contacts/:id/notes		X		
/v2/contacts/:id/pictures		X		
/v2/contacts/:id/profiles		X		
/v2/contacts/:id/searchTerms		X		
/v2/contacts/:id/tags		X		
/v2/contacts/:id/tasks		X		
/v2/deals	X	X		X
/v2/deals/:id		X	X	X

/v2/devices	X	X		
/v2/devices/:id			X	X
/v2/import			X	
/v2/interactions	X	X		
/v2/interactions/:id		X	X	
/v2/login			X	
/v2/logout			X	
/v2/myAccount			X	
/v2/notes	X	X		X
/v2/notes/:id			X	X
/v2/pictures	X			X
/v2/pictures/:id		X	X	X
/v2/sync		X		
/v2/tags		X		
/v2/tasks	X	X		X
/v2/tasks/:id		X	X	X
/v2/taskLists	X	X		
/v2/taskLists/:id		X	X	X
/v2/triggerDevices	X	X		

Table 3 API call overview

2.4 Push Notifications (Server → Device)

2.4.1 Push an Interaction Notification

<table>
<tr><td><b>Title</b></td><td>Notify Device about interaction</td></tr>
<tr><td><b>Data</b></td><td><pre>{
  "type" : <enum(call-incoming,call-outgoing,sms-incoming,sms-outgoing)>,
  "data" : <alphanumeric>,
  "match" :
    {
      "contactId" : <integer>,
      "contactName" : <alphanumeric>,
      "contactType" : <alphanumeric>,
      "basicInfoId" : <integer>
    },
  "tentatives" : [
    {
      "contactId" : <integer>,
      "contactName" : <alphanumeric>,
      "contactType" : <alphanumeric>,
      "basicInfoId" : <integer>
    }
  ]
  "interactionId" : <integer>
}</pre></td></tr>
<tr><td><b>Code example</b></td><td><pre>var ws = new WebSocket(
  "ws://"+window.location.host+"/interact-
server/socket?deviceId=s243df35fd6g43&authToken=bTIgh4uUt5rj6T");
ws.onmessage = function(event) {
  console.log(event.data);
};</pre></td></tr>
<tr><td><b>Notes</b></td><td>This is actually not an API call but a push notification from the Server to the Device. To receive push notifications the Device must have established a websocket connection to the Server with the <i>deviceId</i> and a valid <i>authToken</i> as parameters.

The attribute <i>match</i> is set only when <i>data</i> is associated to a known contact in the database.

If <i>match</i> is not set, <i>tentatives</i> may be set. A tentative match may represent a contact. In this case all attributes of the tentative match will be set. If the tentative match represents a basicinfo not assigned to a contact, only the <i>tentatives[i].basicInfoId</i> attribute will be set and the <i>tentatives[i].contactName</i> attribute will be set to “Unknown”.</td></tr>
</table>

Table 4 Push interaction notification

2.4.2 Push an Interaction Notification (via APN)

Title	Notify Apple Device about interaction

Data	`{`
	`  "aps" :`
	`    {`
	`      "alert" : <alphanumeric>`
	`    },`
	`  "data" : <alphanumeric>,`
	`  "type" : <enum(call-incoming,call-outgoing,sms-incoming,sms-outgoing)>`
	`}`

Notes	This is actually not an API call but a push notification from the Server to an Apple Device. To receive push notifications the Device must have enabled Apple Push Notifications. The alert message has the following form: *"<Incomning|Outgoing> <call|chat message|email|sms> <from|to> <contactname (phonenumber)|phonenumber>"*

Table 5 Push interaction notification (via APN)

2.4.3 Push a Synchronize Hint Message

Title	Notify Device that cloud synchronization should be performed

Data	`{`
	`  "message" : "cloud synchronization recommended",`
	`  "lastSyncTime" : <integer>`
	`}`

Code example	`var ws = new WebSocket(`
	`  "ws://"+window.location.host+"/interact-`
	`server/socket?deviceId=s243df35fd6g43&authToken=bTIgh4uUt5rj6T");`
	`ws.onmessage = function(event) {`
	`  console.log(event.data);`
	`};`

Notes	This is actually not an API call but a push notification from the Server to the Device. To receive push notifications the Device must have established a websocket connection to the Server with the *deviceId* and a valid *authToken* as parameters. When a Device establishes a websocket connection the Server checks for changes in the cloud DB occurred since the last time the Device was synchronized. If there were no changes, no push notification is sent at all. Otherwise the server sends the last determined sync time of the Device in the *lastSyncTime* attribute. This time is represented by the number of milliseconds since the standard base time known as "the epoch", namely January 1, 1970, 00:00:00 GMT. Then the device is supposed to {Synchronize Offline Data}.

Table 6 Push synchronization hint

2.5 API Calls

2.5.1 Authentication

Authentication is required for all other API calls.

2.5.1.1 Login

<table>
<tr><td><b>Title</b></td><td>Login and authenticate for further API calls</td></tr>
<tr><td><b>URL</b></td><td>/v2/login</td></tr>
<tr><td><b>Header attributes</b></td><td>none</td></tr>
<tr><td><b>Method</b></td><td>POST</td></tr>
<tr><td><b>URL parameters</b></td><td>none</td></tr>
<tr><td><b>Data</b></td><td><pre>
{
  "username" : <alphanumeric>,
  "password" : <alphanumeric>,
  "scope" : <enum(csr,device app, trigger app, web portal)>
}
</pre></td></tr>
<tr><td rowspan="2"><b>Success response</b></td><td>Code: 200 OK</td></tr>
<tr><td><pre>
{
  "user" :
    {
      "id" : <integer>,
      "active" : true,
      "email" : <alphanumeric>,
      "companyId" : <integer>,
      "groups" :
        [{
          "id" : <integer>,
          "name" : <alphanumeric>
        }],
      "roles" : [<alphanumeric>]
    },
  "token" :
    {
      "authToken" : <alphanumeric>,
      "expires" : <integer>
    }
}
</pre></td></tr>
<tr><td rowspan="2"><b>Error response</b></td><td>Code: 400 BAD REQUEST
Code: 401 UNAUTHORIZED</td></tr>
<tr><td><pre>
{"code" : <integer>, "error" : <integer>, "devmsg" : <alphanumeric>}
</pre></td></tr>
<tr><td><b>Code example</b></td><td><pre>
$.ajax({
  type : "POST",
  url : "/v2/login",
  dataType : "json",
  contentType : "application/json",
  data : JSON.stringify({
    "username" : "test@interact.io",
</pre></td></tr>
</table>

	`    "password" : "iwonttellyou",`
	`    "scope" : “device app”`
	`  }),`
	`  success : function(data) {`
	`    console.log(data);`
	`  },`
	`  error : function(data) {`
	`    console.log(data);`
	`  }`
	`});`

Notes	The *authToken* returned can be used for further API calls and must be passed in the header attribute of the request. The *expires* field indicates the time when this token will be invalidated. This time is represented by the number of milliseconds since the standard base time known as "the epoch", namely January 1, 1970, 00:00:00 GMT. Note that the token will be renewed with every API call.

Table 7 Login

2.5.1.2 Logout

Title	Logout and invalidate the authentication token
URL	/v2/logout
Header attributes	none
Method	POST
URL parameters	none

Data	`{`
	`  "authToken" : <alphanumeric>`
	`}`

Success response	Code: 200 OK
	`{"code" : 200, "devmsg" : "success"}`
Error response	Code: 400 BAD REQUEST
	`{"code" : <integer>, "error" : <integer>, "devmsg" : <alphanumeric>}`

Code example	`$.ajax({`
	`  type : "POST",`
	`  url : "/v2/logout",`
	`  dataType : "json",`
	`  contentType : "application/json",`
	`  data : JSON.stringify({`
	`    "authToken" : "gfUH43trfdkjg34"`
	`  }),`
	`  success : function(data) {`
	`    console.log(data);`
	`  },`
	`  error : function(data) {`
	`    console.log(data);`
	`  }`
	`});`

Notes	none

Table 8 Logout

2.5.2 Trigger Device → Server

2.5.2.1 Trigger an Event

Title	Trigger an interaction event		
URL	/v2/interactions		
Header attributes	**Attribute**	**Required?**	**Example**
	`triggerToken`	yes	`triggerToken=fSDAF8439fdsRdsfG5dfTf`
Method	PUT		
URL parameters	none		

Data	`{`
	`  "data" : <alphanumeric>,`
	`  "externalSystem" :`
	`    {`
	`      "name" : <alphanumeric>,`
	`      "threadId" : <alphanumeric>,`
	`      "entityId" : <alphanumeric>`
	`    },`
	`  "receiverId" : <alphanumeric>,`
	`  "time" : <integer>,`
	`  "type" : <enum(call-incoming,call-outgoing,email-incoming,email-outgoing,im-incoming,im-outgoing,sms-incoming,sms-outgoing)>`
	`}`		

Success response	Code: 201 CREATED		
	`{"code" : 201, "entity" : <interaction>, "devmsg" : "success"}`		
Error response	Code: 401 UNAUTHORIZED Code: 400 BAD REQUEST		
	`{"code" : <integer>, "error" : <integer>, "devmsg" : <alphanumeric>}`		

Code example	`$.ajax({`
	`  type : "PUT",`
	`  url : "/v2/interactions",`
	`  headers : {triggerToken : "fSDAF8439fdsRdsfG5dfTf"},`
	`  dataType : "json",`
	`  contentType : "application/json",`
	`  data : JSON.stringify({`
	`    "type" : "call-incoming”,`
	`    "data" : "555-12345",`
	`    "receiverId" : "john.doe@interact.io",`
	`  }),`
	`  success : function(data) {`
	`    console.log(data);`
	`  },`
	`  error : function(data) {`
	`    console.log(data);`
	`  }`
	`});`		

Notes	The given *triggerToken* must be a valid token retrieved by “ Register a Trigger Device”, otherwise the error response will be sent. If the owner of the Trigger Device has registered another Device which is connected to the Server, this second Device will receive a notification about the trigger event. (see “Push an Interaction Notification”)		

	If the *externalSystem* entity is given at least one of *externalSystem.threadId* or *externalSystem.entityId* must be given. While the threadId is meant to reference the interaction parent (e.g. the im-conversation id) the entityId is meant to reference the concrete interaction message (e.g. im-message id) inside that parent. The *time* attribute defines when the event occurred. It is represented by the number of milliseconds since the standard base time known as "the epoch", namely January 1, 1970, 00:00:00 GMT.

Table 9 Trigger event

2.5.2.2 Register a Trigger Device

Title	Register a new trigger device for interactions.		
URL	/v2/triggerDevices		
Header attributes	**Attribute**	**Required?**	**Example**
	`authToken`	Yes	`authToken=gfUH43trfdkjg34`
Method	PUT		
URL Parameters	none		

Data	`{`
	`  "deviceId" : <alphanumeric>`
	`}`		

Success response	Code: 201 CREATED		
	`{"code" : 201, "entity" : <trigger-device>, "devmsg" : "success"}`		
Error response	Code: 400 BAD REQUEST Code: 401 UNAUTHORIZED Code: 403 FORBIDDEN		
	`{"code" : <integer>, "error" : <integer>, "devmsg" : <alphanumeric>}`		

Code example	`$.ajax({`
	`  type : "PUT",`
	`  url : "/v2/triggerDevices",`
	`  headers : {authToken : "gfUH43trfdkjg34"},`
	`  dataType : "json",`
	`  contentType : "application/json",`
	`  data : JSON.stringify({`
	`    "deviceId" : "2155341-13582-5-38"`
	`  }),`
	`  success : function(data) {`
	`    console.log(data);`
	`  },`
	`  error : function(data) {`
	`    console.log(data);`
	`  }`
	`});`		

Notes	The *deviceId* should be the IMEI of the Trigger Device. The most important attribute of the returned trigger-device entity is the *triggerToken* attribute. This token represents a permanent token which allows the Trigger Device to sending notifications to the Server. See “Trigger an Event”.		

Table 10 Register a trigger device

2.5.2.3 Read a Trigger Token

<table>
<tr><td><b>Title</b></td><td colspan="3">Read the trigger token for a registered trigger Device.</td></tr>
<tr><td><b>URL</b></td><td colspan="3">/v2/triggerDevices?deviceId=:id</td></tr>
<tr><td rowspan="2"><b>Header attributes</b></td><td><b>Attribute</b></td><td><b>Required?</b></td><td><b>Example</b></td></tr>
<tr><td><code>authToken</code></td><td>Yes</td><td><code>authToken=gfUH43trfdkjg34</code></td></tr>
<tr><td><b>Method</b></td><td colspan="3">GET</td></tr>
<tr><td rowspan="2"><b>URL Parameters</b></td><td><b>Attribute</b></td><td><b>Required?</b></td><td><b>Example</b></td></tr>
<tr><td><code>id : <alphanumeric></code></td><td>Yes</td><td><code>deviceId=353092051071645</code></td></tr>
<tr><td><b>Data</b></td><td colspan="3">None</td></tr>
<tr><td rowspan="2"><b>Success response</b></td><td colspan="3">Code: 200 OK</td></tr>
<tr><td colspan="3"><code>{"triggerToken" : <alphanumeric>}</code></td></tr>
<tr><td rowspan="2"><b>Error response</b></td><td colspan="3">Code: 400 BAD REQUEST
Code: 401 UNAUTHORIZED
Code: 403 FORBIDDEN</td></tr>
<tr><td colspan="3"><code>{"code" : <integer>, "error" : <integer>, "devmsg" : <alphanumeric>}</code></td></tr>
<tr><td><b>Code example</b></td><td colspan="3"><pre>
$.ajax({
  type : "GET",
  url : "/v2/triggerDevices?deviceId=353092051071645",
  headers : {authToken : "gfUH43trfdkjg34"},
  dataType : "json",
  contentType : "application/json",
  success : function(data) {
    console.log(data);
  },
  error : function(data) {
    console.log(data);
  }
});
</pre></td></tr>
<tr><td><b>Notes</b></td><td colspan="3">The <i>deviceId</i> must be the id given when the trigger device was registerted, e.g. the IMEI of the mobile phone.</td></tr>
</table>

Table 11 Read the Trigger Token of a Trigger Device

2.5.3 Device (Web Portal, CSR resp.) → Server

2.5.3.1 Basic CRUD functions

Basic CRUD functions are represented by the http methods PUT, GET, POST and DELETE.

2.5.3.1.1 Read a List of Activities

<table>
<tr><td><b>Title</b></td><td colspan="3">Retrieve a list of activities</td></tr>
<tr><td><b>URL</b></td><td colspan="3">/v2/activities?offset=:offset&limit=:limit&callback=:callback</td></tr>
<tr><td rowspan="2"><b>Header attributes</b></td><td><b>Attribute</b></td><td><b>Required?</b></td><td><b>Example</b></td></tr>
<tr><td>authToken</td><td>yes</td><td>authToken=gfUH43trfdkjg34</td></tr>
<tr><td><b>Method</b></td><td colspan="3">GET</td></tr>
<tr><td rowspan="2"><b>URL parameters</b></td><td><b>Parameter</b></td><td><b>Required?</b></td><td><b>Example</b></td></tr>
<tr><td>offset : <integer>
limit : <integer>
callback : <string&gt;</td><td>no
no
no</td><td>?offset=5
&limit=25
&callback=jsonpcallback</td></tr>
<tr><td><b>Data</b></td><td colspan="3">none</td></tr>
<tr><td rowspan="2"><b>Success response</b></td><td colspan="3">Code: 200 OK</td></tr>
<tr><td colspan="3"><pre>
{
  "totalLength" : <integer>,
  "length" : <integer>,
  "startIndex" : <integer>,
  "data": [
    {
      "action" : enum(insert,update,delete),
      "entity" :
        {
          "oldState" : <object>,
          "newState" : <object>
        },
      "entityId" : <integer>,
      "entityType" : enum(note,interaction),
      "relatedTo" :
        {
          "id" : <integer>,
          "displayName" : <alphanumeric>
        },
      "time" : <integer>
    }
  ]
}
</pre></td></tr>
<tr><td rowspan="2"><b>Error response</b></td><td colspan="3">Code: 401 UNAUTHORIZED</td></tr>
<tr><td colspan="3"><pre>{"code" : 401, "error" : <integer>, "devmsg" : <alphanumeric&gt;}</pre></td></tr>
<tr><td><b>Code example</b></td><td colspan="3"><pre>
$.ajax({
  type : "GET",
  url : "/v2/activities",
  headers : {authToken : "gfUH43trfdkjg34"},
  dataType : "json",
  success : function(data) {
</pre></td></tr>
</table>

<table>
<tr><td></td><td><pre>
    console.log(data);
  },
  error : function(data) {
    console.log(data);
  }
});
</pre></td></tr>
<tr><td><b>Notes</b></td><td>The <i>time</i> attribute holds the date when the activity was saved in the db. This date is represented by the number of milliseconds since the standard base time known as "the epoch", namely January 1, 1970, 00:00:00 GMT.
According to the value of the <i>entityType</i> attribute the attributes <i>entity.oldState</i> and <i>entity.newState</i> will differ. E.g. if <i>entityType</i> is ‘note’ the state attributes will hold note objects.
If <i>action</i> is ‘delete’ the <i>entity.newState</i> attribute will not be set. On the other hand if action is ‘insert’ the <i>entity.oldState</i> attribute will not be set.
The <i>relatedTo</i> attribute represents the user who performed the action.
<i>offset</i> and <i>limit</i> parameters are used for pagination. If at least one of them is provided, the response will contain the start index and the according length.</td></tr>
</table>

Table 12 Activities list request

2.5.3.1.2 Create Person Contact

Title	Create given person contact		
URL	/v2/contacts		
Header attributes	**Attribute**	**Required?**	**Example**
	authToken	yes	authToken=gfUH43trfdkjg34
Method	PUT		
URL parameters	none		

Data

```
{
  "contactType" : "person",
  "title" : <alphanumeric>,
  "lastName" : <alphanumeric>,
  "firstName" : <alphanumeric>,
  "middleName" : <alphanumeric>,
  "gender" : <enum(f,m)>,
  "profession" : <alphanumeric>,
  "hot" : <boolean>,
  "basicInfos" :
    [{
      "id" : <integer>,
      "type" : <enum(email,fax,mobile,phone,website)>,
      "subtype" : <enum(home,work,other)>,
      "identifier“ : <alphanumeric>,
      "value" : <alphanumeric>,
      "excluded" : <boolean>
    }],
  "addresses" :
    [{
      "id" : <integer>,
      "addressLine1" : <alphanumeric>,
      "addressLine2" : <alphanumeric>,
      "addressLine3" : <alphanumeric>,
      "addressType" : <enum(home,work,other)>,
      "postCode" : <alphanumeric>,
      "city" : <alphanumeric>,
      "country" : <alphanumeric>
    }],
  "relatedTo" : [<integer>],
  "tags" : [<alphanumeric>],
  "searchTerms" : [<alphanumeric>],
  "profiles" :
    {
    "facebook" :
      {
        "connected" : <boolean>,
        "username" : <alphanumeric>,
        "url" : <alphanumeric>
      },
    "aim" :
      {
        "connected" : <boolean>,
        "username" : <alphanumeric>,
        "url" : <alphanumeric>
      },
    "gplus" :
      {
        "connected" : <boolean>,
        "username" : <alphanumeric>,
        "url" : <alphanumeric>
      },
    "gtalk" :
      {
        "connected" : <boolean>,
        "username" : <alphanumeric>,
        "url" : <alphanumeric>
      },
    "icq" :
        {
          "connected" : <boolean>,
          "username" : <alphanumeric>,
          "url" : <alphanumeric>
        },
      "jabber" :
        {
          "connected" : <boolean>,
          "username" : <alphanumeric>,
          "url" : <alphanumeric>
        },
      "linkedIn" :
        {
          "connected" : <boolean>,
          "username" : <alphanumeric>,
          "url" : <alphanumeric>
        },
      "skype" :
        {
          "connected" : <boolean>,
          "username" : <alphanumeric>,
          "url" : <alphanumeric>
        },
      "twitter" :
        {
          "connected" : <boolean>,
          "username" : <alphanumeric>,
          "url" : <alphanumeric>
        },
      "xing" :
        {
          "connected" : <boolean>,
          "username" : <alphanumeric>,
          "url" : <alphanumeric>
        }
    }
}
```

Success response

Code: 201 CREATED

```
{"code" : 201, "entity" : <contact>, "devmsg" : "success"}
```

Error response

Code: 400 BAD REQUEST

Code: 401 UNAUTHORIZED

Code: 403 FORBIDDEN

```
{"code" : <integer>, "error" : <integer>, "devmsg" : <alphanumeric>}
```

Code example

```
$.ajax({
  type : "PUT",
  url : "/v2/contacts",
  headers : {authToken : "gfUH43trfdkjg34"},
  dataType : "json",
  contentType : "application/json",
  data : JSON.stringify({
    "firstName" : "John",
    "contactType" : "person",
    "basicInfos" : [{
        "id" : 498,
        "type" : "phone",
        "value" : "02332667998"
    },{
        "type" : "fax",
        "value" : "023326670"
    },{
        "type" : "website",
        "identifier" : "instagram",
        "value" : "http://instagram.com/instagram"
    }],
  }),
  success : function(data) {
    console.log(data);
  },
  error : function(data) {
    console.log(data);
```

	`}`
	`});`

Notes	At least the *lastName* or *firstName* attribute has to be set.

Table 13 Person creation

2.5.3.1.3 Create Organization Contact

<table>
<tr><td><b>Title</b></td><td colspan="3">Create given organization contact</td></tr>
<tr><td><b>URL</b></td><td colspan="3">/v2/contacts</td></tr>
<tr><td rowspan="2"><b>Header attributes</b></td><td><b>Attribute</b></td><td><b>Required?</b></td><td><b>Example</b></td></tr>
<tr><td>authToken</td><td>yes</td><td>authToken=gfUH43trfdkjg34</td></tr>
<tr><td><b>Method</b></td><td colspan="3">PUT</td></tr>
<tr><td><b>URL parameters</b></td><td colspan="3">none</td></tr>
<tr><td><b>Data</b></td><td colspan="3"><pre>
{
  "contactType" : "organization",
  "organizationName" : <alphanumeric>,
  "business" : <alphanumeric>,
  "hot" : <boolean>,
  "basicInfos" :
    [{
      "id" : <integer>,
      "type" : <enum(email,fax,mobile,phone,website)>,
      "subtype" : <enum(home,work,other)>,
      "identifier“ : <alphanumeric>,
      "value" : <alphanumeric>,
      "excluded" : <boolean>
    }],
  "addresses" :
    [{
      "id" : <integer>,
      "addressLine1" : <alphanumeric>,
      "addressLine2" : <alphanumeric>,
      "addressLine3" : <alphanumeric>,
      "addressType" : <enum(home,work,other)>,
      "postCode" : <alphanumeric>,
      "city" : <alphanumeric>,
      "country" : <alphanumeric>
    }],
  "relatedTo" : [<integer>],
  "tags" : [<alphanumeric>],
  "searchTerms" : [<alphanumeric>],
  "profiles" :
    {
    "facebook" :
      {
        "connected" : <boolean>,
        "username" : <alphanumeric>,
        "url" : <alphanumeric>
      },
    "aim" :
      {
        "connected" : <boolean>,
        "username" : <alphanumeric>,
        "url" : <alphanumeric>
      },
    "gplus" :
      {
        "connected" : <boolean>,
        "username" : <alphanumeric>,
        "url" : <alphanumeric>
      },
    "gtalk" :
      {
        "connected" : <boolean>,
        "username" : <alphanumeric>,
        "url" : <alphanumeric>
      },
    "icq" :
      {
        "connected" : <boolean>,
        "username" : <alphanumeric>,
        "url" : <alphanumeric>
      },
    "jabber" :
      {
</pre></td></tr>
</table>

```
        "connected" : <boolean>,
        "username" : <alphanumeric>,
        "url" : <alphanumeric>
      },
    "linkedIn" :
      {
        "connected" : <boolean>,
        "username" : <alphanumeric>,
        "url" : <alphanumeric>
      },
    "skype" :
      {
        "connected" : <boolean>,
        "username" : <alphanumeric>,
        "url" : <alphanumeric>
      },
    "twitter" :
      {
        "connected" : <boolean>,
        "username" : <alphanumeric>,
        "url" : <alphanumeric>
      },
    "xing" :
      {
        "connected" : <boolean>,
        "username" : <alphanumeric>,
        "url" : <alphanumeric>
      }
  }
}
```

Success response	Code: 201 CREATED `{"code" : 201, "entity" : <contact>, "devmsg" : "success"}`
Error response	Code: 400 BAD REQUEST Code: 401 UNAUTHORIZED Code: 403 FORBIDDEN `{"code" : <integer>, "error" : <integer>, "devmsg" : <alphanumeric>}`

Code example

```
$.ajax({
  type : "PUT",
  url : "/v2/contacts",
  headers : { authToken : "gfUH43trfdkjg34" },
  dataType : "json",
  contentType : "application/json",
  data : JSON.stringify({
    "organizationName" : "xaimed",
    "contactType" : "organization",
    "basicInfos" : [{
        "id" : 498,
        "type" : "phone",
        "value" : "02332667998"
    },{
        "type" : "fax",
        "value" : "023326670"
    }],
  }),
  success : function(data) {
    console.log(data);
  },
  error : function(data) {
    console.log(data);
  }
});
```

Notes	At least *organizationName* attribute has to be set.

Table 14 Organization creation

2.5.3.1.4 Read a List of Contacts

<table>
<tr><td><b>Title</b></td><td colspan="3">Retrieve a list of contacts</td></tr>
<tr><td><b>URL</b></td><td colspan="3">/v2/contacts?projection=:proj&offset=:offset&limit=:limit&callback=:callback</td></tr>
<tr><td rowspan="2"><b>Header attributes</b></td><td><b>Attribute</b></td><td><b>Required?</b></td><td><b>Example</b></td></tr>
<tr><td>authToken</td><td>yes</td><td>authToken=gfUH43trfdkjg34</td></tr>
<tr><td><b>Method</b></td><td colspan="3">GET</td></tr>
<tr><td rowspan="5"><b>URL parameters</b></td><td><b>Parameter</b></td><td><b>Required?</b></td><td><b>Example</b></td></tr>
<tr><td>proj : <enum(thin,base,flat)></td><td>no</td><td>?projection=thin</td></tr>
<tr><td>offset : <integer></td><td>no</td><td>&offset=5</td></tr>
<tr><td>limit : <integer></td><td>no</td><td>&limit=25</td></tr>
<tr><td>callback : <string></td><td>no</td><td>&callback=jsonpcallback</td></tr>
<tr><td><b>Data</b></td><td colspan="3">none</td></tr>
<tr><td rowspan="2"><b>Success response</b></td><td colspan="3">Code: 200 OK</td></tr>
<tr><td colspan="3">

```
{
  "data" : [<contact>],
  "length" : <integer>,
  "startIndex" : <integer>,
  "totalLength" : <integer>
}
```

</td></tr>
<tr><td rowspan="2"><b>Error response</b></td><td colspan="3">Code: 400 BAD REQUEST
Code: 401 UNAUTHORIZED
Code: 403 FORBIDDEN</td></tr>
<tr><td colspan="3">{"code" : <integer>, "error" : <integer>, "devmsg" : <alphanumeric>}</td></tr>
<tr><td><b>Code example</b></td><td colspan="3">

```
$.ajax({
  type : "GET",
  url : "/v2/contacts?offset=5&limit=10",
  headers : { authToken : "gfUH43trfdkjg34" },
  dataType : "json",
  success : function(data) {
    console.log(data);
  },
  error : function(data) {
    console.log(data);
  }
});
```

</td></tr>
<tr><td><b>Notes</b></td><td colspan="3">

offset and *limit* parameters are used for pagination. If both are specified, the response will contain according *length* and *startIndex* attributes as well as an additional *totalLength* attribute to indicate whether there are more contacts available.

If *projection* is not specified the default is *base*.

If projection is thin, the *extra* attribute of the contact objects contains the organization name if *contactType* is person and the person contact is member of EXACTLY one organization. It contains the number of organizations if *contactType* is person and the person contact is member of MORE THAN one organizations. Vice

</td></tr>
</table>

	versa, it contains the person name if *contactType* is organization and the organization contact has EXACTLY one person contact assigned to it – or contains the number of person contacts if *contactType* is organization and the organization has MORE THAN one person contacts assigned to it. The attribute is not defined or empty if neither of the conditions above matches. If projection is base, the contact objects are returned as described in Read a Person Contact, Read an Organization Contact resp. Otherwise, if projection is *flat* the return data is very similar to projection *base* but more flatten.

Table 15 Contact list request

2.5.3.1.5 Read a List of Contacts Associated with a Contact

<table>
<tr><td><b>Title</b></td><td colspan="3">Retrieve a list of contacts related to a particular contact</td></tr>
<tr><td><b>URL</b></td><td colspan="3">/v2/contacts/:id/contacts?projection=:proj&callback=:callback</td></tr>
<tr><td rowspan="2"><b>Header attributes</b></td><td><b>Attribute</b></td><td><b>Required?</b></td><td><b>Example</b></td></tr>
<tr><td>authToken</td><td>yes</td><td>authToken=gfUH43trfdkjg34</td></tr>
<tr><td><b>Method</b></td><td colspan="3">GET</td></tr>
<tr><td rowspan="2"><b>URL parameters</b></td><td><b>Parameter</b></td><td><b>Required?</b></td><td><b>Example</b></td></tr>
<tr><td>id : <integer>
proj : <enum(thin,base,flat)>
callback : <string></td><td>yes
no
no</td><td>/23/
?projection=thin
&callback=jsonpcallback</td></tr>
<tr><td><b>Data</b></td><td colspan="3">none</td></tr>
<tr><td rowspan="2"><b>Success response</b></td><td colspan="3">Code: 200 OK</td></tr>
<tr><td colspan="3">{ "data" : [<contact>] }</td></tr>
<tr><td rowspan="2"><b>Error response</b></td><td colspan="3">Code: 400 BAD REQUEST
Code: 401 UNAUTHORIZED
Code: 403 FORBIDDEN</td></tr>
<tr><td colspan="3">{"code" : <integer>, "error" : <integer>, "devmsg" : <alphanumeric>}</td></tr>
<tr><td><b>Code example</b></td><td colspan="3"><pre>
$.ajax({
  type : "GET",
  url : "/v2/contacts/23/contacts?projection=thin",
  headers : {authToken : "gfUH43trfdkjg34"},
  dataType : "json",
  success : function(data) {
    console.log(data);
  },
  error : function(data) {
    console.log(data);
  }
});
</pre></td></tr>
<tr><td><b>Notes</b></td><td colspan="3">If <i>projection</i> is not specified the default is <i>base</i>.
If projection is thin, the <i>extra</i> attribute of the contact objects contains the organization name if <i>contactType</i> is person and the person contact is member of EXACTLY one organization. It contains the number of organizations if <i>contactType</i> is person and the person contact is member of MORE THAN one organizations. Vice versa, it contains the person name if <i>contactType</i> is organization and the organization contact has EXACTLY one person contact assigned to it – or contains the number of person contacts if <i>contactType</i> is organization and the organization</td></tr>
</table>

	has MORE THAN one person contacts assigned to it. The attribute is not defined or empty if neither of the conditions above matches. If projection is base, the contact objects are returned as described in Read a Person Contact, Read an Organization Contact resp. Otherwise, if projection is *flat* the return data is very similar to projection *base* but more flatten.

Table 16 Contacts list request - associated with another contact

2.5.3.1.6 Read a Person Contact

Title	Retrieve detail information of a person contact		
URL	/v2/contacts/:id?projection=:proj&callback=:callback		
Header attributes	**Attribute**	**Required?**	**Example**
	`authToken`	yes	`authToken=gfUH43trfdkjg34`
Method	GET		
URL parameters	**Parameter**	**Required?**	**Example**
	`id : <integer>`	yes	`/47`
	`proj : <enum(thin,base,flat)>`	no	`?projection=base`
	`callback : <string>`	no	`&callback=jsonpcallback`
Data	none		
Success response	Code: 200 OK		

Content for projection *base*:

```
{
  "id" : <integer>,
  "version" : <integer>,
  "contactType" : "person",
  "title" : <alphanumeric>,
  "lastName" : <alphanumeric>,
  "firstName" : <alphanumeric>,
  "middleName" : <alphanumeric>,
  "profession" : <alphanumeric>,
  "gender" : <enum(m,f)>,
  "hot" : <boolean>,
  "basicInfos" :
    [{
      "id" : <integer>,
      "type" : <enum(email,fax,mobile,phone,website)>,
      "subtype" : <enum(home,work,other)>,
      "identifier“ : <alphanumeric>,
      "value" : <alphanumeric>,
      "excluded" : <boolean>
    }],
  "addresses" :
    [{
      "id" : <integer>,
      "addressLine1" : <alphanumeric>,
      "addressLine2" : <alphanumeric>,
      "addressLine3" : <alphanumeric>,
      "addressType" : <enum(home,work,other)>,
      "postCode" : <alphanumeric>,
      "city" : <alphanumeric>,
      "country" : <alphanumeric>
    }],
  "deals" : [<integer>],
  "notes" : [<integer>],
  "tags" : [<alphanumeric>],
  "tasks" : [<integer>],
  "interactions" : [<integer>],
  "relatedTo" : [<integer>],
  "pictures" :
    [{
      "type" : <alphanumeric>,
      "url" : <alphanumeric>
    }],
  "searchTerms" : [<alphanumeric>],
  "profiles" :
    {
    "facebook" :
      {
        "connected" : <boolean>,
        "username" : <alphanumeric>,
        "url" : <alphanumeric>
      },
    "aim" :
      {
        "connected" : <boolean>,
        "username" : <alphanumeric>,
        "url" : <alphanumeric>
      },
    "gplus" :
      {
        "connected" : <boolean>,
        "username" : <alphanumeric>,
        "url" : <alphanumeric>
      },
    "gtalk" :
      {
        "connected" : <boolean>,
        "username" : <alphanumeric>,
        "url" : <alphanumeric>
      },
    "icq" :
      {
        "connected" : <boolean>,
        "username" : <alphanumeric>,
        "url" : <alphanumeric>
      },
    "jabber" :
      {
        "connected" : <boolean>,
        "username" : <alphanumeric>,
        "url" : <alphanumeric>
      },
    "linkedIn" :
      {
        "connected" : <boolean>,
        "username" : <alphanumeric>,
        "url" : <alphanumeric>
      },
    "skype" :
      {
        "connected" : <boolean>,
        "username" : <alphanumeric>,
        "url" : <alphanumeric>
      },
    "twitter" :
      {
        "connected" : <boolean>,
        "username" : <alphanumeric>,
        "url" : <alphanumeric>
      },
    "xing" :
      {
        "connected" : <boolean>,
        "username" : <alphanumeric>,
        "url" : <alphanumeric>
      }
  }
}
```

Error response

Code: 400 BAD REQUEST

Code: 401 UNAUTHORIZED

Code: 403 FORBIDDEN

```
{"code" : <integer>, "error" : <integer>, "devmsg" : <alphanumeric>}
```

Code example

```
$.ajax({
  type : "GET",
  url : "/v2/contacts/47",
  headers : {authToken : "gfUH43trfdkjg34"},
  dataType : "json",
  success : function(data) {
    console.log(data);
  },
```

	`  error : function(data) {`
	`    console.log(data);`
	`  }`
	`});`

Notes	The attributes *deals*, *notes*, *tasks*, *interactions* and *relatedTo* contain comma-separated lists of ids. At least the *lastName* or *firstName* attribute will be set. The structure for attribute *picture* depends on the value of the *projection* http query parameter: • base: `"pictures" : [{"type" : <alphanumeric>,"url" : <alphanumeric>}]` • flat: `"pictures" : [<alphanumeric>]` • thin: `"picture" : <alphanumeric>` When projection *thin* is used, the *picture* attribute holds the default picture only.

Table 17 Preson contact request

2.5.3.1.7 Read an Organization Contact

Title	Retrieve detail information of an organization contact		
URL	/v2/contacts/:id?projection=:proj&callback=:callback		
Header attributes	**Attribute**	**Required?**	**Example**
	`authToken`	yes	`authToken=gfUH43trfdkjg34`
Method	GET		
URL parameters	**Parameter**	**Required?**	**Example**

	`id : <integer>`
	`proj : <enum(thin,base,flat)>`
	`callback : <string>`
		yes
		no
		no
			`/22`
			`?projection=base`
			`&callback=jsonpcallback`

Data	none		
Success response	Code: 200 OK		
	Content for projection *base*:		

```
{
  "id" : <integer>,
  "version" : <integer>,
  "contactType" : "organization",
  "organizationName" : <alphanumeric>,
  "business" : <alphanumeric>,
  "hot" : <boolean>,
  "basicInfos" :
    [{
      "id" : <integer>,
      "type" : <enum(email,fax,mobile,phone,website)>,
      "subtype" : <enum(home,work,other)>,
      "identifier“ : <alphanumeric>,
      "value" : <alphanumeric>,
      "excluded" : <boolean>
    }],
  "addresses" :
    [{
      "id" : <integer>,
      "addressLine1" : <alphanumeric>,
      "addressLine2" : <alphanumeric>,
      "addressLine3" : <alphanumeric>,
      "addressType" : <enum(home,work,other)>,
      "postCode" : <alphanumeric>,
      "city" : <alphanumeric>,
      "country" : <alphanumeric>
    }],
  "deals" : [<integer>],
  "notes" : [<integer>],
  "tags" : [<alphanumeric>],
  "tasks" : [<integer>],
  "interactions" : [<integer>],
  "relatedTo" : [<integer>],
  "pictures" :
    [{
      "type" : <alphanumeric>,
      "url" : <alphanumeric>
    }],
  "searchTerms" : [<alphanumeric>],
  "profiles" :
    {
    "facebook" :
      {
        "connected" : <boolean>,
        "username" : <alphanumeric>,
        "url" : <alphanumeric>
      },
    "aim" :
      {
        "connected" : <boolean>,
        "username" : <alphanumeric>,
        "url" : <alphanumeric>
      },
    "gplus" :
      {
        "connected" : <boolean>,
        "username" : <alphanumeric>,
        "url" : <alphanumeric>
      },
    "gtalk" :
      {
        "connected" : <boolean>,
        "username" : <alphanumeric>,
        "url" : <alphanumeric>
      },
    "icq" :
      {
        "connected" : <boolean>,
        "username" : <alphanumeric>,
        "url" : <alphanumeric>
      },
    "jabber" :
      {
        "connected" : <boolean>,
        "username" : <alphanumeric>,
        "url" : <alphanumeric>
      },
    "linkedIn" :
      {
        "connected" : <boolean>,
        "username" : <alphanumeric>,
        "url" : <alphanumeric>
      },
    "skype" :
      {
        "connected" : <boolean>,
        "username" : <alphanumeric>,
        "url" : <alphanumeric>
      },
    "twitter" :
      {
        "connected" : <boolean>,
        "username" : <alphanumeric>,
        "url" : <alphanumeric>
      },
    "xing" :
      {
        "connected" : <boolean>,
        "username" : <alphanumeric>,
        "url" : <alphanumeric>
      }
  }
}
```

Error response

Code: 400 BAD REQUEST

Code: 401 UNAUTHORIZED

Code: 403 FORBIDDEN

```
{"code" : <integer>, "error" : <integer>, "devmsg" : <alphanumeric>}
```

Code example

```
$.ajax({
  type : "GET",
  url : "/v2/contacts/22",
  headers : {authToken : "gfUH43trfdkjg34"},
  dataType : "json",
  success : function(data) {
    console.log(data);
  },
  error : function(data) {
    console.log(data);
  }
});
```

Notes	The attributes *deals*, *notes*, *tasks*, *interactions* and *relatedTo* contain comma-separated lists of ids. The structure for attribute *picture* depends on the value of the *projection* http query parameter: • base: `"pictures" : [{"type" : <alphanumeric>,"url" : <alphanumeric>}]` • flat: `"pictures" : [<alphanumeric>]` • thin: `"picture" : <alphanumeric>` When projection *thin* is used, the *picture* attribute holds the first picture only.

Table 18 Organization request

2.5.3.1.8 Update a Person Contact

Title	Update a particular person contact		
URL	/v2/contacts/:id		
Header attributes	**Attribute**	**Required?**	**Example**
	authToken	Yes	authToken=gfUH43trfdkjg34
Method	POST		
URL parameters	**Parameter**	**Required?**	**Example**
	id : <integer>	yes	/285
Data			

```
{
  "version" : <integer>,
  "title" : <alphanumeric>,
  "lastName" : <alphanumeric>,
  "firstName" : <alphanumeric>,
  "gender" : <enum(f,m)>,
  "middleName" : <alphanumeric>,
  "profession" : <alphanumeric>,
  "hot" : <boolean>,
  "basicInfos" :
    [{
      "id" : <integer>,
      "type" : <enum(email,fax,mobile,phone,website)>,
      "subtype" : <enum(home,work,other)>,
      "identifier“ : <alphanumeric>,
      "value" : <alphanumeric>,
      "excluded" : <boolean>
    }],
  "addresses" :
    [{
      "id" : <integer>,
      "addressLine1" : <alphanumeric>,
      "addressLine2" : <alphanumeric>,
      "addressLine3" : <alphanumeric>,
      "addressType" : <enum(home,work,other)>,
      "postCode" : <alphanumeric>,
      "city" : <alphanumeric>,
      "country" : <alphanumeric>
    }],
  "relatedTo" : [<integer>],
  "tags" : [<alphanumeric >],
  "searchTerms" : [<alphanumeric >],
  "profiles" :
    {
      "facebook" :
        {
          "connected" : <boolean>,
          "username" : <alphanumeric>,
          "url" : <alphanumeric>
        },
      "aim" :
        {
          "connected" : <boolean>,
          "username" : <alphanumeric>,
          "url" : <alphanumeric>
        },
      "gplus" :
        {
          "connected" : <boolean>,
          "username" : <alphanumeric>,
          "url" : <alphanumeric>
        },
      "gtalk" :
        {
          "connected" : <boolean>,
          "username" : <alphanumeric>,
          "url" : <alphanumeric>
        },
```

<table>
<tr>
<td></td>
<td>

```
      "icq" :
        {
          "connected" : <boolean>,
          "username" : <alphanumeric>,
          "url" : <alphanumeric>
        },
      "jabber" :
        {
          "connected" : <boolean>,
          "username" : <alphanumeric>,
          "url" : <alphanumeric>
        },
      "linkedIn" :
        {
          "connected" : <boolean>,
          "username" : <alphanumeric>,
          "url" : <alphanumeric>
        },
      "skype" :
        {
          "connected" : <boolean>,
          "username" : <alphanumeric>,
          "url" : <alphanumeric>
        },
      "twitter" :
        {
          "connected" : <boolean>,
          "username" : <alphanumeric>,
          "url" : <alphanumeric>
        },
      "xing" :
        {
          "connected" : <boolean>,
          "username" : <alphanumeric>,
          "url" : <alphanumeric>
        }
    }
}
```

</td>
</tr>
<tr>
<td rowspan="2"><b>Success response</b></td>
<td>Code: 200 OK</td>
</tr>
<tr>
<td>

```
{"code" : 200, "entity" : <contact>, "devmsg" : "success"}
```

</td>
</tr>
<tr>
<td rowspan="2"><b>Error response</b></td>
<td>Code: 400 BAD REQUEST
Code: 401 UNAUTHORIZED
Code: 403 FORBIDDEN
Code: 409 CONFLICT</td>
</tr>
<tr>
<td>

```
{"code" : <integer>, "error" : <integer>, "devmsg" : <alphanumeric>}
```

</td>
</tr>
<tr>
<td><b>Code example</b></td>
<td>

```
$.ajax({
  type : "POST",
  url : "/v2/contacts/285",
  headers : {authToken : "gfUH43trfdkjg34"},
  dataType : "json",
  contentType : "application/json",
  data : JSON.stringify({
    "version" : 12,
    "firstName" : "Remo",
    "basicInfos" : [{
      "id" : 46,
      "type" : "mobile",
      "value" : "+4917134455667"
    },{
      "type" : "phone",
      "value" : "033056434239"
    },{
      "type" : "website"
      "value" : "http://instagram.com/instagram",
      "identifier" : "instagram"
    }]
  }),
  success : function(data) {
    console.log(data);
```

</td>
</tr>
</table>

	`  },`
	`  error : function(data) {`
	`    console.log(data);`
	`  }`
	`});`

Notes	Attribute values of *basicInfo.type* and *basicInfo.value* must not be empty (and thus are not allowed to be deleted). Updating *firstName* and *lastName* to an empty string is also not allowed. In the case of a version conflict the error response is a 409 Conflict with the devmsg “version conflict”.

Table 19 Person update

2.5.3.1.9 Update an Organization Contact

<table>
<tr><td><b>Title</b></td><td colspan="3">Update a particular organization contact</td></tr>
<tr><td><b>URL</b></td><td colspan="3">/v2/contacts/:id</td></tr>
<tr><td rowspan="2"><b>Header attributes</b></td><td><b>Attribute</b></td><td><b>Required?</b></td><td><b>Example</b></td></tr>
<tr><td>authToken</td><td>yes</td><td>authToken=gfUH43trfdkjg34</td></tr>
<tr><td><b>Method</b></td><td colspan="3">POST</td></tr>
<tr><td rowspan="2"><b>URL parameters</b></td><td><b>Parameter</b></td><td><b>Required?</b></td><td><b>Example</b></td></tr>
<tr><td>id : &lt;integer&gt;</td><td>yes</td><td>/1243</td></tr>
<tr><td><b>Data</b></td><td colspan="3"><pre>
{
  "version" : <integer>,
  "name" : <alphanumeric>,
  "business" : <alphanumeric>,
  "hot" : <boolean>,
  "basicInfos" :
    [{
      "id" : <integer>,
      "type" : <enum(email,fax,mobile,phone,website)>,
      "subtype" : <enum(home,work,other)>,
      "identifier“ : <alphanumeric>,
      "value" : <alphanumeric>,
      "excluded" : <boolean>
    }],
  "addresses" :
    [{
      "id" : <integer>,
      "addressLine1" : <alphanumeric>,
      "addressLine2" : <alphanumeric>,
      "addressLine3" : <alphanumeric>,
      "addressType" : <enum(home,work,other)>,
      "postCode" : <alphanumeric>,
      "city" : <alphanumeric>,
      "country" : <alphanumeric>
    }],
  "relatedTo" : [<integer>],
  "tags" : [<alphanumeric>],
  "searchTerms" : [<alphanumeric>],
  "profiles" :
    {
      "facebook" :
        {
          "connected" : <boolean>,
          "username" : <alphanumeric>,
          "url" : <alphanumeric>
        },
      "aim" :
        {
          "connected" : <boolean>,
          "username" : <alphanumeric>,
          "url" : <alphanumeric>
        },
      "gplus" :
        {
          "connected" : <boolean>,
          "username" : <alphanumeric>,
          "url" : <alphanumeric>
        },
      "gtalk" :
        {
          "connected" : <boolean>,
          "username" : <alphanumeric>,
          "url" : <alphanumeric>
        },
      "icq" :
        {
          "connected" : <boolean>,
</pre></td></tr>
</table>

```
          "username" : <alphanumeric>,
          "url" : <alphanumeric>
        },
      "jabber" :
        {
          "connected" : <boolean>,
          "username" : <alphanumeric>,
          "url" : <alphanumeric>
        },
      "linkedIn" :
        {
          "connected" : <boolean>,
          "username" : <alphanumeric>,
          "url" : <alphanumeric>
        },
      "skype" :
        {
          "connected" : <boolean>,
          "username" : <alphanumeric>,
          "url" : <alphanumeric>
        },
      "twitter" :
        {
          "connected" : <boolean>,
          "username" : <alphanumeric>,
          "url" : <alphanumeric>
        },
      "xing" :
        {
          "connected" : <boolean>,
          "username" : <alphanumeric>,
          "url" : <alphanumeric>
        }
    }
}
```

Success response

Code: 200 OK

```
{"code" : 200, "entity" : <contact>, "devmsg" : "success"}
```

Error response

Code: 400 BAD REQUEST

Code: 401 UNAUTHORIZED

Code: 403 FORBIDDEN

Code: 409 CONFLICT

```
{"code" : <integer>, "error" : <integer>, "devmsg" : <alphanumeric>}
```

Code example

```
$.ajax({
  type : "POST",
  url : /v2/contacts/1243 ",
  headers : {authToken : "gfUH43trfdkjg34"},
  dataType : "json",
  contentType : "application/json",
  data : JSON.stringify({
    "version" : 21,
    "name" : "TU Berlin",
    "basicInfos" : [{
      "id" : 189,
      "type" : " phone",
      "value" : "0303140"
    }]
  }),
  success : function(data) {
    console.log(data);
  },
  error : function(data) {
    console.log(data);
  }
});
```

Notes

Attribute values of *basicInfo.type* and *basicInfo.value* must not be empty (and thus are not allowed to be deleted).

	Updating *organizationName* to an empty string is also not allowed. In the case that of a version conflict the error response is a 409 Conflict with the devmsg “version conflict”.

Table 20 Organization update

2.5.3.1.10 Delete Contacts

Title	Delete contacts with the given ids		
URL	/v2/contacts?ids=:ids		
Header attributes	**Attribute**	**Required?**	**Example**
	authToken	yes	authToken=gfUH43trfdkjg34
Method	DELETE		
URL parameters	**Parameter**	**Required?**	**Example**
	ids : [<integer>]	Yes	ids=1,3,45,6
Data	none		
Success response	Code: 200 OK		
	{"code" : 204, "devmsg" : "success"}		
Error response	Code: 400 BAD REQUEST Code: 401 UNAUTHORIZED Code: 403 FORBIDDEN		
	{"code" : <integer>, "error" : <integer>, "devmsg" : <alphanumeric>}		

Code example	$.ajax({
	  type : "DELETE",
	  url : "/v2/contacts?ids=1,3,45,6",
	  headers : {authToken : "gfUH43trfdkjg34"},
	  dataType : "json",
	  success : function(data) {
	    console.log(data);
	  },
	  error : function(data) {
	    console.log(data);
	  }
	});		

Notes	none		

Table 21 Contact(s) deletion

2.5.3.1.11 Delete a Contact

Title	Delete a particular person or organization contact		
URL	/v2/contacts/:id		
Header attributes	**Attribute**	**Required?**	**Example**
	`authToken`	yes	`authToken=gfUH43trfdkjg34`
Method	`DELETE`		
URL parameters	**Parameter**	**Required?**	**Example**
	`id : <integer>`	yes	`/12`
Data	none		
Success response	Code: 200 OK		
	`{"code" : 204, "devmsg" : "success"}`		
Error response	Code: 400 BAD REQUEST Code: 401 UNAUTHORIZED Code: 403 FORBIDDEN		
	`{"code" : <integer>, "error" : <integer>, "devmsg" : <alphanumeric>}`		

Code example	`$.ajax({`
	`  type : "DELETE",`
	`  url : "/v2/contacts/12",`
	`  headers : {authToken : "gfUH43trfdkjg34"},`
	`  dataType : "json",`
	`  success : function(data) {`
	`    console.log(data);`
	`  },`
	`  error : function(data) {`
	`    console.log(data);`
	`  }`
	`});`		

Notes	none		

Table 22 Contact deletion

2.5.3.1.12 Upload a contact picture

Title	Provides uploading pictures for contacts		
URL	/v2/pictures		
Header attributes	**Attribute**	**Required?**	**Example**
	authToken	yes	authToken=gfUH43trfdkjg34
Method	PUT		
URL parameters	none		

Data	{
	"contactId" : <integer>,
	"base64" : <base64string>,
	"isDefault" : <boolean>
	}		

Success response	Code: 200 OK		
	{"code" : 201, "devmsg" : "success", "entity" : <picture>}		
Error response	Code: 400 BAD REQUEST Code: 401 UNAUTHORIZED Code: 403 FORBIDDEN		
	{"code" : <integer>, "error" : <integer>, "devmsg" : <alphanumeric>}		

Code example	$.ajax({
	type : "PUT",
	url : "/v2/pictures",
	headers : {authToken : "gfUH43trfdkjg34"},
	dataType : "json",
	data : JSON.stringify({
	"contactId" : 41,
	"base64" : "DDgF4g833gSq3FKl9d8lEWshf+rh1//9k="
	}),
	success : function(data) {
	console.log(data);
	},
	error : function(data) {
	console.log(data);
	}
	});		

Notes	none		

Table 23 Picture upload

2.5.3.1.13 Read a Picture

Title	Retrieve data of a particular picture		
URL	/v2/pictures/:id?callback=:callback		
Header attributes	**Attribute**	**Required?**	**Example**
	authToken	yes	authToken=gfUH43trfdkjg34
Method	GET		
URL parameters	**Parameter**	**Required?**	**Example**
	id : <integer> callback : <string>	Yes No	/751 ?callback=jsonpcallback
Data	none		
Success response	Code: 200 OK		

	{
	"id" : <integer>,
	"isDefault" : <boolean>,
	"type" : <enum(facebook,google,gravatar,interact,twitter)>,
	"url" : <alphanumeric>
	}		

Error response	Code: 400 BAD REQUEST Code: 401 UNAUTHORIZED Code: 403 FORBIDDEN		
	{"code" : <integer>, "error" : <integer>, "devmsg" : <alphanumeric>}		

Code example	$.ajax({
	type : "GET",
	url : "/v2/pictures/751",
	headers : {authToken : "gfUH43trfdkjg34"},
	dataType : "json",
	success : function(data) {
	console.log(data);
	},
	error : function(data) {
	console.log(data);
	}
	});		

Notes	none		

Table 24 Picture request

2.5.3.1.14 Read a List of Pictures Associated with a Contact

Title	Retrieve a list of pictures associated with a particular contact		
URL	/v2/contacts/:id/pictures?callback=:callback		
Header attributes	**Attribute**	**Required?**	**Example**
	`authToken`	yes	`authToken=gfUH43trfdkjg34`
Method	GET		
URL parameters	**Parameter**	**Required?**	**Example**

	`id : <integer>`
	`callback : <string>`
		Yes
		No
			`/23/`
			`?callback=myjsonpcallback`

Data	none		
Success response	Code: 200 OK		

	`[{`
	`"id" : <integer>,`
	`"isDefault" : <boolean>,`
	`"type" : <alphanumeric>,`
	`"url" : <alphanumeric>`
	`}]`		

Error response	Code: 400 BAD REQUEST Code: 401 UNAUTHORIZED Code: 403 FORBIDDEN		
	`{"code" : <integer>, "error" : <integer>, "devmsg" : <alphanumeric>}`		

Code example	`$.ajax({`
	`type : "GET",`
	`url : "/v2/contacts/23/pictures",`
	`headers : {authToken : "gfUH43trfdkjg34"},`
	`dataType : "json",`
	`success : function(data) {`
	`console.log(data);`
	`},`
	`error : function(data) {`
	`console.log(data);`
	`}`
	`});`		

Notes	none		

Table 25 Picture list request – associated with a contact

2.5.3.1.15 Update a contact picture

<table>
<tr><td><b>Title</b></td><td colspan="3">Allows updating pictures for contacts</td></tr>
<tr><td><b>URL</b></td><td colspan="3">/v2/pictures/:id</td></tr>
<tr><td rowspan="2"><b>Header attributes</b></td><td><b>Attribute</b></td><td><b>Required?</b></td><td><b>Example</b></td></tr>
<tr><td>authToken</td><td>yes</td><td>authToken=gfUH43trfdkjg34</td></tr>
<tr><td><b>Method</b></td><td colspan="3">POST</td></tr>
<tr><td rowspan="2"><b>URL parameters</b></td><td><b>Parameter</b></td><td><b>Required?</b></td><td><b>Example</b></td></tr>
<tr><td>id : <integer></td><td>yes</td><td>/85</td></tr>
<tr><td><b>Data</b></td><td colspan="3"><pre>{
  "base64" : <base64string>,
  "isDefault" : <boolean>
}</pre></td></tr>
<tr><td rowspan="2"><b>Success response</b></td><td colspan="3">Code: 200 OK</td></tr>
<tr><td colspan="3">{"code" : 200, "devmsg" : "success", "entity" : <picture>}</td></tr>
<tr><td rowspan="2"><b>Error response</b></td><td colspan="3">Code: 400 BAD REQUEST
Code: 401 UNAUTHORIZED
Code: 403 FORBIDDEN</td></tr>
<tr><td colspan="3">{"code" : <integer>, "error" : <integer>, "devmsg" : <alphanumeric>}</td></tr>
<tr><td><b>Code example</b></td><td colspan="3"><pre>$.ajax({
  type : "POST",
  url : "/v2/pictures/85",
  headers : {authToken : "gfUH43trfdkjg34"},
  dataType : "json",
  data : JSON.stringify({
    "base64" : "DDgF4g833gSq3FKl9d8lEWshf+rh1//9k="
  }),
  success : function(data) {
    console.log(data);
  },
  error : function(data) {
    console.log(data);
  }
});</pre></td></tr>
<tr><td><b>Notes</b></td><td colspan="3">If the picture specified by the given id is not an uploaded picture but a picture from an external system (e.g. facebook, twitter, ...) the <i>base64</i> attribute must not be given.</td></tr>
</table>

Table 26 Picture update

2.5.3.1.16 Delete Pictures

Title	Delete pictures with the given ids		
URL	/v2/pictures?ids=:ids		
Header attributes	**Attribute**	**Required?**	**Example**
	`authToken`	yes	`authToken=gfUH43trfdkjg34`
Method	DELETE		
URL parameters	**Parameter**	**Required?**	**Example**
	`ids : [<integer>]`	Yes	`ids=12,435,98`
Data	none		
Success response	Code: 200 OK		
	`{"code" : 204, "devmsg" : "success"}`		
Error response	Code: 400 BAD REQUEST Code: 401 UNAUTHORIZED Code: 403 FORBIDDEN		
	`{"code" : <integer>, "error" : <integer>, "devmsg" : <alphanumeric>}`		

Code example	`$.ajax({`
	`  type : "DELETE",`
	`  url : "/v2/pictures?ids=12,435,98",`
	`  headers : {authToken : "gfUH43trfdkjg34"},`
	`  dataType : "json",`
	`  success : function(data) {`
	`    console.log(data);`
	`  },`
	`  error : function(data) {`
	`    console.log(data);`
	`  }`
	`});`		

Notes	If any picture specified by the ids parameter is not an uploaded picture but a picture from an external system (e.g. facebook, twitter, ...) it can not be deleted.		

Table 27 Pictures deletion

2.5.3.1.17 Delete Picture

<table>
<tr><td><b>Title</b></td><td colspan="3">Delete a particular picture</td></tr>
<tr><td><b>URL</b></td><td colspan="3">/v2/pictures/:id</td></tr>
<tr><td rowspan="2"><b>Header attributes</b></td><td><b>Attribute</b></td><td><b>Required?</b></td><td><b>Example</b></td></tr>
<tr><td>authToken</td><td>yes</td><td>authToken=gfUH43trfdkjg34</td></tr>
<tr><td><b>Method</b></td><td colspan="3">DELETE</td></tr>
<tr><td rowspan="2"><b>URL parameters</b></td><td><b>Parameter</b></td><td><b>Required?</b></td><td><b>Example</b></td></tr>
<tr><td>id : <integer></td><td>Yes</td><td>/98</td></tr>
<tr><td><b>Data</b></td><td colspan="3">none</td></tr>
<tr><td rowspan="2"><b>Success response</b></td><td colspan="3">Code: 200 OK</td></tr>
<tr><td colspan="3">{"code" : 204, "devmsg" : "success"}</td></tr>
<tr><td rowspan="2"><b>Error response</b></td><td colspan="3">Code: 400 BAD REQUEST
Code: 401 UNAUTHORIZED
Code: 403 FORBIDDEN</td></tr>
<tr><td colspan="3">{"code" : <integer>, "error" : <integer>, "devmsg" : <alphanumeric>}</td></tr>
<tr><td><b>Code example</b></td><td colspan="3"><pre>
$.ajax({
  type : "DELETE",
  url : "/v2/pictures/98",
  headers : {authToken : "gfUH43trfdkjg34"},
  dataType : "json",
  success : function(data) {
    console.log(data);
  },
  error : function(data) {
    console.log(data);
  }
});
</pre></td></tr>
<tr><td><b>Notes</b></td><td colspan="3">If the picture specified by the id parameter is not an uploaded picture but a picture from an external system (e.g. facebook, twitter, ...) it can not be deleted.</td></tr>
</table>

Table 28 Picture deletion

2.5.3.1.18 Create Note Associated with a Contact

<table>
<tr><td><b>Title</b></td><td colspan="3">Create given note</td></tr>
<tr><td><b>URL</b></td><td colspan="3">/v2/notes</td></tr>
<tr><td rowspan="2"><b>Header attributes</b></td><td><b>Attribute</b></td><td><b>Required?</b></td><td><b>Example</b></td></tr>
<tr><td>authToken</td><td>yes</td><td>authToken=gfUH43trfdkjg34</td></tr>
<tr><td><b>Method</b></td><td colspan="3">PUT</td></tr>
<tr><td rowspan="2"><b>URL parameters</b></td><td><b>Parameter</b></td><td><b>Required?</b></td><td><b>Example</b></td></tr>
<tr><td>id : <integer></td><td>yes</td><td>/312/</td></tr>
<tr><td><b>Data</b></td><td colspan="3"><pre>{
  "message" : <alphanumeric>,
  "relatedTo" :
    {
      "id" : <integer>,
      "type" : "contact"
    }
}</pre></td></tr>
<tr><td rowspan="2"><b>Success response</b></td><td colspan="3">Code: 201 CREATED</td></tr>
<tr><td colspan="3"><pre>{"code" : 201, "entity" : <note>, "devmsg" : "success"}</pre></td></tr>
<tr><td rowspan="2"><b>Error response</b></td><td colspan="3">Code: 400 BAD REQUEST
Code: 401 UNAUTHORIZED
Code: 403 FORBIDDEN</td></tr>
<tr><td colspan="3"><pre>{"code" : <integer>, "error" : <integer>, "devmsg" : <alphanumeric>}</pre></td></tr>
<tr><td><b>Code example</b></td><td colspan="3"><pre>$.ajax({
  type : "PUT",
  url : "/v2/notes",
  headers : {authToken : "gfUH43trfdkjg34"},
  dataType : "json",
  contentType : "application/json",
  data : JSON.stringify({
    "message" : "deadline postponed",
    "relatedTo" : {
      "id" : 327,
      "type" : "contact"
    }
  }),
  success : function(data) {
    console.log(data);
  },
  error : function(data) {
    console.log(data);
  }
});</pre></td></tr>
<tr><td><b>Notes</b></td><td colspan="3">none</td></tr>
</table>

Table 29 Notes creation

2.5.3.1.19 Read a List of Notes

Title	Retrieve a list of all notes created by the system user		
URL	/v2/notes?callback=:callback		
Header attributes	**Attribute**	**Required?**	**Example**
	authToken	yes	authToken=gfUH43trfdkjg34
Method	GET		
URL parameters	**Parameter**	**Required?**	**Example**
	callback : <string>	no	?callback=jsonpcallback
Data	none		
Success response	Code: 200 OK		

	[{
	  "id" : <integer>,
	  "timestamp" : <integer>,
	  "message" : <alphanumeric>,
	  "relatedTo" :
	    {
	      "id" : <integer>,
	      "type" : <enum(contact,deal)>,
	      "displayName" : <alphanumeric>
	    }
	}]		

Error response	Code: 401 UNAUTHORIZED Code: 403 FORBIDDEN		
	{"code" : <integer>, "error" : <integer>, "devmsg" : <alphanumeric>}		

Code example	$.ajax({
	  type : "GET",
	  url : "/v2/notes",
	  headers : {authToken : "gfUH43trfdkjg34"},
	  dataType : "json",
	  success : function(data) {
	    console.log(data);
	  },
	  error : function(data) {
	    console.log(data);
	  }
	});		

Notes	The *timestamp* attribute holds the date when the note was last updated. This date is represented by the number of milliseconds since the standard base time known as "the epoch", namely January 1, 1970, 00:00:00 GMT.		

Table 30 Note list request

2.5.3.1.20 Read a List of Notes Associated with a Contact

Title	Retrieve a list of notes associated with a particular contact		
URL	/v2/contacts/:id/notes?callback=:callback		
Header attributes	**Attribute**	**Required?**	**Example**
	`authToken`	yes	`authToken=gfUH43trfdkjg34`
Method	GET		
URL parameters	**Parameter**	**Required?**	**Example**

	`id : <integer>`
	`callback : <string>`
		Yes
		no
			`/23/`
			`?callback=jsonpcallback`

Data	none		
Success response	Code: 200 OK		

	`[{`
	`  "id" : <integer>,`
	`  "timestamp" : <integer>,`
	`  "message" : <alphanumeric>`
	`}]`		

Error response	Code: 400 BAD REQUEST Code: 401 UNAUTHORIZED Code: 403 FORBIDDEN		
	`{"code" : <integer>, "error" : <integer>, "devmsg" : <alphanumeric>}`		

Code example	`$.ajax({`
	`  type : "GET",`
	`  url : "/v2/contacts/23/notes",`
	`  headers : {authToken : "gfUH43trfdkjg34"},`
	`  dataType : "json",`
	`  success : function(data) {`
	`    console.log(data);`
	`  },`
	`  error : function(data) {`
	`    console.log(data);`
	`  }`
	`});`		

Notes	The *timestamp* attribute holds the date when the note was last updated. This date is represented by the number of milliseconds since the standard base time known as "the epoch", namely January 1, 1970, 00:00:00 GMT.		

Table 31 Note list request – associated with a contact

2.5.3.1.21 Update a Note

Title	Update a particular note		
URL	/v2/notes/:id		
Header attributes	**Attribute**	**Required?**	**Example**
	authToken	yes	authToken=gfUH43trfdkjg34
Method	POST		
URL parameters	**Parameter**	**Required?**	**Example**
	id : <integer>	yes	/184

Data	{
	  "message" : <alphanumeric>
	}		

Success response	Code: 200 OK		
	{"code" : 200, "entity" : <note>, "devmsg" : "success"}		
Error response	Code: 400 BAD REQUEST Code: 401 UNAUTHORIZED Code: 403 FORBIDDEN		
	{"code" : <integer>, "error" : <integer>, "devmsg" : <alphanumeric>}		

Code example	$.ajax({
	  type : "POST",
	  url : "/v2/notes/184",
	  headers : {authToken : "gfUH43trfdkjg34"},
	  dataType : "json",
	  contentType : "application/json",
	  data : JSON.stringify({
	    "message" : "Don’t forget to ask for address data."
	  }),
	  success : function(data) {
	    console.log(data);
	  },
	  error : function(data) {
	    console.log(data);
	  }
	});		

Notes	none		

Table 32 Note update

2.5.3.1.22 Delete Notes

<table>
<tr><td><b>Title</b></td><td colspan="3">Delete notes with the given ids</td></tr>
<tr><td><b>URL</b></td><td colspan="3">/v2/notes?ids=:ids</td></tr>
<tr><td rowspan="2"><b>Header attributes</b></td><td><b>Attribute</b></td><td><b>Required?</b></td><td><b>Example</b></td></tr>
<tr><td>authToken</td><td>yes</td><td>authToken=gfUH43trfdkjg34</td></tr>
<tr><td><b>Method</b></td><td colspan="3">DELETE</td></tr>
<tr><td rowspan="2"><b>URL parameters</b></td><td><b>Parameter</b></td><td><b>Required?</b></td><td><b>Example</b></td></tr>
<tr><td>ids : [<integer>]</td><td>Yes</td><td>ids=12,435,98</td></tr>
<tr><td><b>Data</b></td><td colspan="3">none</td></tr>
<tr><td rowspan="2"><b>Success response</b></td><td colspan="3">Code: 200 OK</td></tr>
<tr><td colspan="3">{"code" : 204, "devmsg" : "success"}</td></tr>
<tr><td rowspan="2"><b>Error response</b></td><td colspan="3">Code: 400 BAD REQUEST
Code: 401 UNAUTHORIZED
Code: 403 FORBIDDEN</td></tr>
<tr><td colspan="3">{"code" : <integer>, "error" : <integer>, "devmsg" : <alphanumeric>}</td></tr>
<tr><td><b>Code example</b></td><td colspan="3"><pre>
$.ajax({
  type : "DELETE",
  url : "/v2/notes?ids=1,3,45,6",
  headers : {authToken : "gfUH43trfdkjg34"},
  dataType : "json",
  success : function(data) {
    console.log(data);
  },
  error : function(data) {
    console.log(data);
  }
});
</pre></td></tr>
<tr><td><b>Notes</b></td><td colspan="3">none</td></tr>
</table>

Table 33 Notes deletion

2.5.3.1.23 Delete a Note

<table>
<tr><td><b>Title</b></td><td colspan="3">Delete a particular note.</td></tr>
<tr><td><b>URL</b></td><td colspan="3">/v2/notes/:id</td></tr>
<tr><td rowspan="2"><b>Header attributes</b></td><td><b>Attribute</b></td><td><b>Required?</b></td><td><b>Example</b></td></tr>
<tr><td>authToken</td><td>yes</td><td>authToken=gfUH43trfdkjg34</td></tr>
<tr><td><b>Method</b></td><td colspan="3">DELETE</td></tr>
<tr><td rowspan="2"><b>URL parameters</b></td><td><b>Parameter</b></td><td><b>Required?</b></td><td><b>Example</b></td></tr>
<tr><td>id : &lt;integer&gt;</td><td>yes</td><td>/12</td></tr>
<tr><td><b>Data</b></td><td colspan="3">none</td></tr>
<tr><td rowspan="2"><b>Success response</b></td><td colspan="3">Code: 200 OK</td></tr>
<tr><td colspan="3">{"code" : 204, "devmsg" : "success"}</td></tr>
<tr><td rowspan="2"><b>Error response</b></td><td colspan="3">Code: 400 BAD REQUEST
Code: 401 UNAUTHORIZED
Code: 403 FORBIDDEN</td></tr>
<tr><td colspan="3">{"code" : <integer>, "error" : <integer>, "devmsg" : <alphanumeric&gt;}</td></tr>
<tr><td><b>Code example</b></td><td colspan="3"><pre>$.ajax({
  type : "DELETE",
  url : "/v2/notes/12",
  headers : {authToken : "gfUH43trfdkjg34"},
  dataType : "json",
  success : function(data) {
    console.log(data);
  },
  error : function(data) {
    console.log(data);
  }
});</pre></td></tr>
<tr><td><b>Notes</b></td><td colspan="3">none</td></tr>
</table>

Table 34 Note deletion

2.5.3.1.24 Read a List of Profiles Associated with a Contact

Title	Retrieve a list of social profiles and IMs associated with a particular contact		
URL	/v2/contacts/:id/profiles?callback=:callback		
Header attributes	**Attribute**	**Required?**	**Example**
	authToken	yes	authToken=gfUH43trfdkjg34
Method	GET		
URL parameters	**Parameter**	**Required?**	**Example**
	id : <integer> callback : <string>	Yes No	/471/ ?callback=jsonpcallback
Data	None		
Success response	Code: 200 OK		

```
{
  "facebook" :
    {
      "connected" : <boolean>,
      "username" : <alphanumeric>,
      "url" : <alphanumeric>
    },
  "aim" :
    {
      "connected" : <boolean>,
      "username" : <alphanumeric>,
      "url" : <alphanumeric>
    },
  "gplus" :
    {
      "connected" : <boolean>,
      "username" : <alphanumeric>,
      "url" : <alphanumeric>
    },
  "gtalk" :
    {
      "connected" : <boolean>,
      "username" : <alphanumeric>,
      "url" : <alphanumeric>
    },
  "icq" :
    {
      "connected" : <boolean>,
      "username" : <alphanumeric>,
      "url" : <alphanumeric>
    },
  "jabber" :
    {
      "connected" : <boolean>,
      "username" : <alphanumeric>,
      "url" : <alphanumeric>
    },
  "linkedIn" :
    {
      "connected" : <boolean>,
      "username" : <alphanumeric>,
      "url" : <alphanumeric>
    },
  "skype" :
    {
      "connected" : <boolean>,
      "username" : <alphanumeric>,
      "url" : <alphanumeric>
    },
```

	"twitter" :
	  {
	    "connected" : <boolean>,
	    "username" : <alphanumeric>,
	    "url" : <alphanumeric>
	  },
	"xing" :
	  {
	    "connected" : <boolean>,
	    "username" : <alphanumeric>,
	    "url" : <alphanumeric>
	  }
	}

Error response	Code: 400 BAD REQUEST Code: 401 UNAUTHORIZED Code: 403 FORBIDDEN
	{"code" : <integer>, "error" : <integer>, "devmsg" : <alphanumeric>}

Code example	$.ajax({
	  type : "GET",
	  url : "/v2/contacts/471/profiles",
	  headers : {authToken : "gfUH43trfdkjg34"},
	  dataType : "json",
	  success : function(data) {
	    console.log(data);
	  },
	  error : function(data) {
	    console.log(data);
	  }
	});

Notes	none

Table 35 Profiles list request – associated with a contact

2.5.3.1.25 Read a List of Search Terms Associated with a Contact

Title	Retrieve a list of search terms associated with a particular contact		
URL	/v2/contacts/:id/searchTerms?callback=:callback		
Header attributes	**Attribute**	**Required?**	**Example**
	`authToken`	yes	`authToken=gfUH43trfdkjg34`
Method	`GET`		
URL parameters	**Parameter**	**Required?**	**Example**

	`id : <integer>`
	`callback : <string>`
		Yes
		No
			`/65/`
			`?callback=jsonpcallback`

Data	none		
Success response	Code: 200 OK		
	`[<alphanumeric>]`		
Error response	Code: 400 BAD REQUEST Code: 401 UNAUTHORIZED Code: 403 FORBIDDEN		
	`{"code" : <integer>, "error" : <integer>, "devmsg" : <alphanumeric>}`		

Code example	`$.ajax({`
	`  type : "GET",`
	`  url : "/v2/contacts/65/searchTerms",`
	`  headers : {authToken : "gfUH43trfdkjg34"},`
	`  dataType : "json",`
	`  success : function(data) {`
	`    console.log(data);`
	`  },`
	`  error : function(data) {`
	`    console.log(data);`
	`  }`
	`});`		

Notes	none		

Table 36 Search term list request – associated with a contact

2.5.3.1.26 Read a List of Tags

Title	Retrieve a list of all tags used by the system user		
URL	/v2/tags?callback=:callback		
Header attributes	**Attribute**	**Required?**	**Example**
	authToken	yes	authToken=gfUH43trfdkjg34
Method	GET		
URL parameters	**Parameter**	**Required?**	**Example**
	callback : <string>	no	?callback=jsonpcallback
Data	none		
Success response	Code: 200 OK		
Success response	[<alphanumeric>]		
Error response	Code: 401 UNAUTHORIZED Code: 403 FORBIDDEN		
Error response	{"code" : <integer>, "error" : <integer>, "devmsg" : <alphanumeric>}		

Code example	`$.ajax({`
	`  type : "GET",`
	`  url : "/v2/tags",`
	`  headers : {authToken : "gfUH43trfdkjg34"},`
	`  dataType : "json",`
	`  success : function(data) {`
	`    console.log(data);`
	`  },`
	`  error : function(data) {`
	`    console.log(data);`
	`  }`
	`});`		

Notes	none		

Table 37 Tag list request

2.5.3.1.27 Read a List of Tags Associated with a Contact

Title	Retrieve a list of tags associated with a particular contact		
URL	/v2/contacts/:id/tags?callback=:callback		
Header attributes	**Attribute**	**Required?**	**Example**
	authToken	yes	authToken=gfUH43trfdkjg34
Method	GET		
URL parameters	**Parameter**	**Required?**	**Example**
	id : <integer> callback : <string>	Yes No	/231/ ?callback=jsonpcallback
Data	none		
Success response	Code: 200 OK		
	[<alphanumeric>]		
Error response	Code: 400 BAD REQUEST Code: 401 UNAUTHORIZED Code: 403 FORBIDDEN		
	{"code" : <integer>, "error" : <integer>, "devmsg" : <alphanumeric>}		

Code example	$.ajax({
	  type : "GET",
	  url : "/v2/contacts/231/tags",
	  headers : {authToken : "gfUH43trfdkjg34"},
	  dataType : "json",
	  success : function(data) {
	    console.log(data);
	  },
	  error : function(data) {
	    console.log(data);
	  }
	});		

Notes	none		

Table 38 Tag list request – associated with a contact

2.5.3.1.28 Read a List of Interactions

Title	Returns all interactions associated with the requesting user		
URL	/v2/interactions?callback=:callback		
Header attributes	**Attribute**	**Required?**	**Example**
	authToken	yes	authToken=gfUH43trfdkjg34
Method	GET		
URL parameters	**Parameter**	**Required?**	**Example**
	callback : <string>	no	?callback=jsonpcallback
Data	none		
Success response	Code: 200 OK		

Success response

```
[{
  "id" : <integer>,
  "organization" :
    {
      "id" : <integer>,
      "displayName" : <alphanumeric>
    },
  "person" :
    {
      "id" : <integer>,
      "displayName" : <alphanumeric>
    }
  "basicInfoId" : <integer>,
  "type" : <enum(call-incoming,call-outgoing,email-incoming,email-outgoing,im-
incoming,im-outgoing,sms-incoming,sms-outgoing)>,
  "data" : <alphanumeric>,
  "duration" : <integer>,
  "externalSystems" :
    [{
      "systemId" : <alphanumeric>,
      "threadId" : <alphanumeric>,
      "entityId" : <alphanumeric>
    }]
  "time" : <integer>
}]
```

Error response

Code: 401 UNAUTHORIZED

Code: 403 FORBIDDEN

Error response

```
{"code" : <integer>, "error" : <integer>, "devmsg" : <alphanumeric>}
```

Code example

```
$.ajax({
  type : "GET",
  url : "/v2/interactions",
  headers : {authToken : "gfUH43trfdkjg34"},
  dataType : "json",
  success : function(data) {
    console.log(data);
  },
  error : function(data) {
    console.log(data);
  }
});
```

Notes	The *time* attribute holds the date when the interaction was created. This date is represented by the number of milliseconds since the standard base time known as "the epoch", namely January 1, 1970, 00:00:00 GMT. The interaction list is sorted by time, descending.

Table 39 Interaction list request

2.5.3.1.29 Read a List of Interactions Associated with a Contact

<table>
<tr><td><b>Title</b></td><td colspan="3">Retrieve a list of interactions associated with a particular contact</td></tr>
<tr><td><b>URL</b></td><td colspan="3">/v2/contacts/:id/interactions?callback=:callback</td></tr>
<tr><td rowspan="2"><b>Header attributes</b></td><td><b>Attribute</b></td><td><b>Required?</b></td><td><b>Example</b></td></tr>
<tr><td>authToken</td><td>yes</td><td>authToken=gfUH43trfdkjg34</td></tr>
<tr><td><b>Method</b></td><td colspan="3">GET</td></tr>
<tr><td rowspan="2"><b>URL parameters</b></td><td><b>Parameter</b></td><td><b>Required?</b></td><td><b>Example</b></td></tr>
<tr><td>id : &lt;integer>
callback : <string&gt;</td><td>Yes
no</td><td>/23/
?callback=jsonpcallback</td></tr>
<tr><td><b>Data</b></td><td colspan="3">none</td></tr>
<tr><td rowspan="2"><b>Success response</b></td><td colspan="3">Code: 200 OK</td></tr>
<tr><td colspan="3"><pre>
[{
  "id" : <integer>,
  "organization":
    {
      "id" : <integer>,
      "displayName" : <alphanumeric>
    },
  "person" :
    {
      "id" : <integer>,
      "displayName" : <alphanumeric>
    }
  "basicInfoId" : <integer>,
  "type" : <enum(call-incoming,call-outgoing,email-incoming,email-outgoing,im-
incoming,im-outgoing,sms-incoming,sms-outgoing)>,
  "data" : <alphanumeric>,
  "duration" : <integer>,
  "externalSystems" :
    [{
      "systemId" : <alphanumeric>,
      "threadId" : <alphanumeric>,
      "entityId" : <alphanumeric>
    }]
  "time" : <integer>
 }]
</pre></td></tr>
<tr><td rowspan="2"><b>Error response</b></td><td colspan="3">Code: 400 BAD REQUEST
Code: 401 UNAUTHORIZED
Code: 403 FORBIDDEN</td></tr>
<tr><td colspan="3"><pre>{"code" : <integer>, "error" : <integer>, "devmsg" : <alphanumeric>}</pre></td></tr>
<tr><td><b>Code example</b></td><td colspan="3"><pre>
$.ajax({
  type : "GET",
  url : "/v2/contacts/23/interactions",
  headers : {authToken : "gfUH43trfdkjg34"},
  dataType : "json",
  success : function(data) {
    console.log(data);
  },
  error : function(data) {
    console.log(data);
  }
});
</pre></td></tr>
</table>

Notes	The *time* attribute holds the date when the interaction was created. This date is represented by the number of milliseconds since the standard base time known as "the epoch", namely January 1, 1970, 00:00:00 GMT. The interaction list is sorted by time, descending.

Table 40 Interaction list request - associated with a contact

2.5.3.1.30 Read an Interaction

<table>
<tr><td><b>Title</b></td><td colspan="3">Retrieve data of a particular interaction</td></tr>
<tr><td><b>URL</b></td><td colspan="3">/v2/interactions/:id?callback=:callback</td></tr>
<tr><td rowspan="2"><b>Header attributes</b></td><td><b>Attribute</b></td><td><b>Required?</b></td><td><b>Example</b></td></tr>
<tr><td>authToken</td><td>yes</td><td>authToken=gfUH43trfdkjg34</td></tr>
<tr><td><b>Method</b></td><td colspan="3">GET</td></tr>
<tr><td rowspan="2"><b>URL parameters</b></td><td><b>Parameter</b></td><td><b>Required?</b></td><td><b>Example</b></td></tr>
<tr><td>id : &lt;integer>
callback : <string&gt;</td><td>Yes
No</td><td>/79
?callback=jsonpcallback</td></tr>
<tr><td><b>Data</b></td><td colspan="3">none</td></tr>
<tr><td><b>Success response</b></td><td colspan="3">Code: 200 OK
Content:
<pre>
{
  "id" : <integer>,
  "organization":
    {
      "id" : <integer>,
      "displayName" : <alphanumeric>
    },
  "person" :
    {
      "id" : <integer>,
      "displayName" : <alphanumeric>
    }
  "basicInfoId" : <integer>,
  "type" : <enum(call-incoming,call-outgoing,email-incoming,email-outgoing,im-
incoming,im-outgoing,sms-incoming,sms-outgoing)>,
  "data" : <alphanumeric>,
  "duration" : <integer>,
  "externalSystems" :
    [{
      "systemId" : <alphanumeric>,
      "threadId" : <alphanumeric>,
      "entityId" : <alphanumeric>
    }]
  "time" : <integer>
}
</pre></td></tr>
<tr><td><b>Error response</b></td><td colspan="3">Code: 400 BAD REQUEST
Code: 401 UNAUTHORIZED
Code: 403 FORBIDDEN
Code: 404 NOT FOUND
Content:
{"code" : <integer>, "error" : <integer>, "devmsg" : <alphanumeric&gt;}</td></tr>
<tr><td><b>Code example</b></td><td colspan="3">
<pre>
$.ajax({
  type : "GET",
  url : "/v2/interactions/79",
  headers : {authToken : "gfUH43trfdkjg34"},
  dataType : "json",
  success : function(data) {
    console.log(data);
</pre></td></tr>
</table>

	`  },`
	`  error : function(data) {`
	`    console.log(data);`
	`  }`
	`});`

Notes	The *time* attribute holds the date when the interaction was created. This date is represented by the number of milliseconds since the standard base time known as "the epoch", namely January 1, 1970, 00:00:00 GMT. The interaction list is sorted by time, descending.

Table 41 Interaction request

2.5.3.1.31 Update an Interaction

Title	Update a particular interaction		
URL	/v2/interactions/:id		
Header attributes	**Attribute**	**Required?**	**Example**
	authToken	yes	authToken=gfUH43trfdkjg34
Method	POST		
URL parameters	**Parameter**	**Required?**	**Example**
	id : <integer>	yes	/1387

Data	{
	  "basicInfoId" : <integer>,
	  "customTags" : <alphanumeric>,
	  "duration" : <integer>,
	  "externalSystems" :
	    [{
	      "name" : <alphanumeric>,
	      "threadId" : <alphanumeric>,
	      "entityId" : <alphanumeric>
	    }],
	  "organization":
	    {
	      "id" : <integer>
	    },
	  "person" :
	    {
	      "id" : <integer>
	    },
	  "status" : <alphanumeric>
	}		

Success response	Code: 200 OK		
	{"code" : 200, "entity" : <interaction>, "devmsg" : "success"}		
Error response	Code: 400 BAD REQUEST		
	{"code" : <integer>, "error" : <integer>, "devmsg" : <alphanumeric>}		

Code example	$.ajax({
	  type : "POST",
	  url : "/v2/interactions/1387",
	  headers : {authToken : "gfUH43trfdkjg34"},
	  dataType : "json",
	  contentType : "application/json",
	  data : JSON.stringify({
	    "organizationId" : 12,
	    "personId" : 34
	  }),
	  success : function(data) {
	    console.log(data);
	  },
	  error : function(data) {
	    console.log(data);
	  }
	});		

Notes	The purpose of this API call is linking a contact to a formerly unknown interaction list entry or setting a concrete contact person for an interaction with an organization. It is also possible du update an interaction (of type *call*) with the duration when the call		

	ended, set custom tags and assign an ID for that interaction inside an external system. If the *externalSystem* entity is given at least one of *externalSystem.threadId* or *externalSystem.entityId* must be given. While the threadId is meant to reference the interaction parent (e.g. the im-conversation id) the entityId is meant to reference the concrete interaction message (e.g. im-message id) inside that parent. The *status* field is meant to represent the technical state of the interaction, e.g. “missed”, “voicemail”, “answered”, …

Table 42 Interaction update

2.5.3.1.32 Create Basicinfo

Title	Create given basicinfo		
URL	/v2/basicInfos		
Header attributes	**Attribute**	**Required?**	**Example**
	`authToken`	yes	`authToken=gfUH43trfdkjg34`
Method	PUT		
URL parameters	none		

Data	`{`
	`  "excluded" : <boolean>,`
	`  "subtype" : <enum(home,work,other)>,`
	`  "type" : <enum(email,fax,mobile,phone,website)>,`
	`  "value" : <alphanumeric>`
	`}`		

Success response	Code: 201 CREATED Content: `{"code" : 201, "entity" : <basicinfo>, "devmsg" : "success"}`		
Error response	Code: 400 BAD REQUEST Code: 401 UNAUTHORIZED Code: 403 FORBIDDEN Content: `{"code" : <integer>, "error" : <integer>, "devmsg" : <alphanumeric>}`		

Code example	`$.ajax({`
	`  type : "PUT",`
	`  url : "/v2/basicInfos",`
	`  headers : {authToken : "gfUH43trfdkjg34"},`
	`  dataType : "json",`
	`  contentType : "application/json",`
	`  data : JSON.stringify({`
	`    "type" : "phone",`
	`    "value" : "+491774528462"`
	`    "excluded" : true`
	`  }),`
	`  success : function(data) {`
	`    console.log(data);`
	`  },`
	`  error : function(data) {`
	`    console.log (data);`
	`  }`
	`});`		

Notes	The general purpose of this API call is excluding unknown phone numbers from Interact.		

Table 43 Basicinfos creation

2.5.3.1.33 Update a Basicinfo

Title	Update a particular basicinfo		
URL	/v2/basicInfos/:id		
Header attributes	**Attribute**	**Required?**	**Example**
	`authToken`	yes	`authToken=gfUH43trfdkjg34`
Method	POST		
URL parameters	**Parameter**	**Required?**	**Example**
	`id : <integer>`	yes	`/246`

Data	`{`
	`  "type" : <enum(email,fax,mobile,phone,website)>,`
	`  "subtype" : <enum(home,work,other)>,`
	`  "value" : <alphanumeric>,`
	`  "excluded" : <boolean>`
	`}`		

Success response	Code: 200 OK Content: `{"code" : 200, "entity" : <basicinfo>, "devmsg" : "success"}`		
Error response	Code: 400 BAD REQUEST Code: 401 UNAUTHORIZED Code: 403 FORBIDDEN Content: `{"code" : <integer>, "error" : <integer>, "devmsg" : <alphanumeric>}`		

Code example	`$.ajax({`
	`  type : "POST",`
	`  url : "/v2/basicInfos/246",`
	`  headers : {authToken : "gfUH43trfdkjg34"},`
	`  dataType : "json",`
	`  contentType : "application/json",`
	`  data : JSON.stringify({`
	`    "value" : "+493029352225",`
	`    "subtype" : "work",`
	`    "excluded" : true`
	`  }),`
	`  success : function(data) {`
	`    console.log(data);`
	`  },`
	`  error : function(data) {`
	`    console.log(data);`
	`  }`
	`});`		

Notes	The general purpose of this API call is excluding known phone numbers from Interact.		

Table 44 Basicinfo update

2.5.3.1.34 Create Task

<table>
<tr><td><b>Title</b></td><td colspan="3">Create a task</td></tr>
<tr><td><b>URL</b></td><td colspan="3">/v2/tasks</td></tr>
<tr><td rowspan="2"><b>Header attributes</b></td><td><b>Attribute</b></td><td><b>Required?</b></td><td><b>Example</b></td></tr>
<tr><td>authToken</td><td>yes</td><td>authToken=gfUH43trfdkjg34</td></tr>
<tr><td><b>Method</b></td><td colspan="3">PUT</td></tr>
<tr><td><b>URL parameters</b></td><td colspan="3">none</td></tr>
<tr><td><b>Data</b></td><td colspan="3"><pre>
{
  "assignedTo" :
    {
      "id" : <integer>
    },
  "concreteDueDate" : <integer>,
  "content" : <alphanumeric>,
  "created" : <integer>,
  "duration" : <integer>,
  "fuzzyDueDate" : <enum(tomorrow,next_week,in_a_month,someday)>,
  "hot" : <boolean>,
  "listId" : <integer>,
  "phoneNumberIds" : [<integer>],
  "relatedTo" :
    {
      "id" : <integer>,
      "type" : <enum(contact,deal)>
    },
  "status" : <enum(completed,open)>,
  "taskType" : <enum(task,scheduled_call>,
  "title" : <alphanumeric>
}
</pre></td></tr>
<tr><td><b>Success response</b></td><td colspan="3">Code: 201 CREATED
Content:
{"code" : 201, "entity" : <task>, "devmsg" : "success"}</td></tr>
<tr><td><b>Error response</b></td><td colspan="3">Code: 400 BAD REQUEST
Code: 401 UNAUTHORIZED
Code: 403 FORBIDDEN
Content:
{"code" : <integer>, "error" : <integer>, "devmsg" : <alphanumeric>}</td></tr>
<tr><td><b>Code example</b></td><td colspan="3"><pre>
$.ajax({
  type : "PUT",
  url : "/v2/tasks",
  headers : {authToken : "gfUH43trfdkjg34"},
  dataType : "json",
  contentType : "application/json",
</pre></td></tr>
</table>

```
  data : JSON.stringify({
    "content" : "Meeting with Remo at Joeys’",
    "created" : 1360190185200,
    "concreteDueDate" : 1360190188000,
    "listId" : 95
    "relatedTo" : {
      "id" : 327,
      "type" : "contact"
    },
    "status" : "open",
    "taskType" : "task",
    "title" : "Lunch with Remo"
  }]),
  success : function(data) {
    console.log(data);
  },
  error : function(data) {
    console.log(data);
  }
});
```

Notes	The attributes *duration* and *phoneNumberIds* are only relevant when *taskType* is set to “scheduled_call”. *created* and *concreteDueDate* are represented by the number of milliseconds since the standard base time known as "the epoch", namely January 1, 1970, 00:00:00 GMT. *fuzzyDueDate* rules are the following: “tomorrow” => *concreteDueDate* = current date + 1d “next_week” => *concreteDueDate* = current date + 7d “in_a_month” => *concreteDueDate* = current date + 30d “someday” => *concreteDueDate* = current date + 99d

Table 45 Tasks creation

2.5.3.1.35 Read a List of Tasks

<table>
<tr><td><b>Title</b></td><td colspan="3">Returns all the tasks associated to the requesting user.</td></tr>
<tr><td><b>URL</b></td><td colspan="3">/v2/tasks?callback=:callback</td></tr>
<tr><td rowspan="2"><b>Header attributes</b></td><td><b>Attribute</b></td><td><b>Required?</b></td><td><b>Example</b></td></tr>
<tr><td>authToken</td><td>yes</td><td>authToken=gfUH43trfdkjg34</td></tr>
<tr><td><b>Method</b></td><td colspan="3">GET</td></tr>
<tr><td rowspan="2"><b>URL parameters</b></td><td><b>Parameter</b></td><td><b>Required?</b></td><td><b>Example</b></td></tr>
<tr><td>callback : <string&gt;</td><td>no</td><td>?callback=jsonpcallback</td></tr>
<tr><td><b>Data</b></td><td colspan="3">none</td></tr>
<tr><td><b>Success response</b></td><td colspan="3">Code: 200 OK
Content:<pre>
[{
  "id" : <integer>,
  "assignedTo" :
    {
      "id" : <integer>,
      "displayName" : <alphanumeric>
    },
  "concreteDueDate" : <integer>,
  "content" : <alphanumeric>,
  "created" : <integer>,
  "duration" : <integer>,
  "fuzzyDueDate" : <enum(tomorrow,next_week,in_a_month,someday)>,
  "hot" : <boolean>,
  "listId" : <integer>,
  "phoneNumbers" :
    [{
      "id" : <integer>,
      "type" : <enum(mobile,phone)>,
      "subtype" : <enum(home,work,other)>,
      "value" : <alphanumeric>,
      "excluded" : <boolean>
    }],
  "relatedTo" :
    {
      "id" : <integer>,
      "type" : <enum(contact,deal)>,
      "displayName" : <alphanumeric>
    },
  "status" : <enum(completed,open)>,
  "taskType" : <enum(task,scheduled_call>,
  "title" : <alphanumeric>
}]
</pre></td></tr>
<tr><td><b>Error response</b></td><td colspan="3">Code: 401 UNAUTHORIZED
Content:
{"code" : <integer>, "error" : <integer>, "devmsg" : <alphanumeric&gt;}</td></tr>
</table>

Code example	`$.ajax({`
	`  type : "GET",`
	`  url : "/v2/tasks",`
	`  headers : {authToken : "gfUH43trfdkjg34"},`
	`  dataType : "json",`
	`  success : function(data) {`
	`    console.log(data);`
	`  },`
	`  error : function(data) {`
	`    console.log(data);`
	`  }`
	`});`

Notes	*created* and *concreteDueDate* are represented by the number of milliseconds since the standard base time known as "the epoch", namely January 1, 1970, 00:00:00 GMT. The list is sorted by *concreteDueDate*, descending. The attributes *duration* and *phoneNumbers* may only be set when *taskType* is “scheduled_call”.

Table 46 Task list request

2.5.3.1.36 Read a List of Tasks Associated with a Contact

<table>
<tr><td><b>Title</b></td><td colspan="3">Retrieve a list of tasks associated with a particular contact</td></tr>
<tr><td><b>URL</b></td><td colspan="3">/v2/contacts/:id/tasks?callback=:callback</td></tr>
<tr><td rowspan="2"><b>Header attributes</b></td><td><b>Attribute</b></td><td><b>Required?</b></td><td><b>Example</b></td></tr>
<tr><td>authToken</td><td>yes</td><td>authToken=gfUH43trfdkjg34</td></tr>
<tr><td><b>Method</b></td><td colspan="3">GET</td></tr>
<tr><td rowspan="2"><b>URL parameters</b></td><td><b>Parameter</b></td><td><b>Required?</b></td><td><b>Example</b></td></tr>
<tr><td>id : &lt;integer>
callback : <string&gt;</td><td>Yes
No</td><td>/517/
?callback=jsonpcallback</td></tr>
<tr><td><b>Data</b></td><td colspan="3">none</td></tr>
<tr><td><b>Success response</b></td><td colspan="3">Code: 200 OK
Content:
<pre>
[{
  "id" : <integer>,
  "assignedTo" :
    {
      "id" : <integer>,
      "displayName" : <alphanumeric>
    },
  "concreteDueDate" : <integer>,
  "content" : <alphanumeric>,
  "created" : <integer>,
  "duration" : <integer>,
  "fuzzyDueDate" : <enum(tomorrow,next_week,in_a_month,someday)>,
  "hot" : <boolean>,
  "listId" : <integer>,
  "phoneNumbers" :
    [{
      "id" : <integer>,
      "type" : <enum(mobile,phone)>,
      "subtype" : <enum(home,work,other)>,
      "value" : <alphanumeric>,
      "excluded" : <boolean>
    }],
  "relatedTo" :
    {
      "id" : <integer>,
      "type" : <enum(contact,deal)>,
      "displayName" : <alphanumeric>
    },
  "status" : <enum(completed,open)>,
  "taskType" : <enum(task,scheduled_call>,
  "title" : <alphanumeric>
}]
</pre></td></tr>
<tr><td><b>Error response</b></td><td colspan="3">Code: 401 UNAUTHORIZED
Code: 403 FORBIDDEN</td></tr>
</table>

<table>
<tr><td></td><td>Code: 404 NOT FOUND
Content:
<code>{"code" : <integer>, "error" : <integer>, "devmsg" : <alphanumeric&gt;}</code></td></tr>
<tr><td><b>Code example</b></td><td><pre><code>$.ajax({
  type : "GET",
  url : "/v2/contacts/517/tasks",
  headers : {authToken : "gfUH43trfdkjg34"},
  dataType : "json",
  success : function(data) {
    console.log(data);
  },
  error : function(data) {
    console.log(data);
  }
});</code></pre></td></tr>
<tr><td><b>Notes</b></td><td>The <i>created</i> attribute holds the date when the task was created. This date is represented by the number of milliseconds since the standard base time known as "the epoch", namely January 1, 1970, 00:00:00 GMT. Same behavior is used for <i>concreteDueDate</i>.
The list is sorted by <i>concreteDueDate</i>, descending.
The attributes <i>duration</i> and <i>phoneNumbers</i> may only be set when <i>taskType</i> is “scheduled_call”.</td></tr>
</table>

Table 47 Tasks list request - associated with a contact

2.5.3.1.37 Read a Task

Title	Retrieve data of a particular task		
URL	/v2/tasks/:id?callback=:callback		
Header attributes	**Attribute**	**Required?**	**Example**
	authToken	yes	authToken=gfUH43trfdkjg34
Method	GET		
URL parameters	**Parameter**	**Required?**	**Example**
	id : <integer> callback : <string>	Yes No	/47 ?callback=jsonpcallback
Data	none		
Success response	Code: 200 OK Content:		

```
{
  "id" : <integer>,
  "assignedTo" :
    {
      "id" : <integer>,
      "displayName" : <alphanumeric>
    },
  "concreteDueDate" : <integer>,
  "content" : <alphanumeric>,
  "created" : <integer>,
  "duration" : <integer>,
  "fuzzyDueDate" : <enum(tomorrow,next_week,in_a_month,someday)>,
  "hot" : <boolean>,
  "listId" : <integer>,
  "phoneNumbers" :
    [{
      "id" : <integer>,
      "type" : <enum(mobile,phone)>,
      "subtype" : <enum(home,work,other)>,
      "value" : <alphanumeric>,
      "excluded" : <boolean>
    }],
  "relatedTo" :
    {
      "id" : <integer>,
      "type" : <enum(contact,deal)>,
      "displayName" : <alphanumeric>
    },
  "status" : <enum(completed,open)>,
  "taskType" : <enum(task,scheduled_call>,
  "title" : <alphanumeric>
}
```

Error response	Code: 401 UNAUTHORIZED Code: 403 FORBIDDEN		

	Code: 404 NOT FOUND Content: `{"code" : <integer>, "error" : <integer>, "devmsg" : <alphanumeric>}`

Code example	`$.ajax({`
	`  type : "GET",`
	`  url : "/v2/tasks/47",`
	`  headers : {authToken : "gfUH43trfdkjg34"},`
	`  dataType : "json",`
	`  success : function(data) {`
	`    console.log(data);`
	`  },`
	`  error : function(data) {`
	`    console.log(data);`
	`  }`
	`});`

Notes	*created* and *concreteDueDate* are represented by the number of milliseconds since the standard base time known as "the epoch", namely January 1, 1970, 00:00:00 GMT. The attributes *duration* and *phoneNumbers* may only be set when *taskType* is “scheduled_call”.

Table 48 Task request

2.5.3.1.38 Update a Task

Title	Update a particular task		
URL	/v2/tasks/:id		
Header attributes	**Attribute**	**Required?**	**Example**
	authToken	yes	authToken=gfUH43trfdkjg34
Method	POST		
URL parameters	**Parameter**	**Required?**	**Example**
	id : <integer>	yes	/734

Data	{
	  "assignedTo" :
	    {
	      "id" : <integer>,
	    },
	  "concreteDueDate" : <integer>,
	  "content" : <alphanumeric>,
	  "created" : <integer>,
	  "duration" : <integer>,
	  "fuzzyDueDate" : <enum(tomorrow,next_week,in_a_month,someday)>,
	  "hot" : <boolean>,
	  "listId" : <integer>,
	  "phoneNumberIds" : [<integer>],
	  "relatedTo" :
	    {
	      "id" : <integer>,
	      "type" : <enum(contact,deal)>,
	    },
	  "status" : <enum(completed,open)>,
	  "title" : <alphanumeric>
	}		

Success response	Code: 200 OK Content: {"code" : 200, "entity" : <task>, "devmsg" : "success"}		
Error response	Code: 400 BAD REQUEST Code: 401 UNAUTHORIZED Code: 403 FORBIDDEN Content: {"code" : <integer>, "error" : <integer>, "devmsg" : <alphanumeric>}		

Code example	$.ajax({
	  type : "POST",
	  url : "/v2/tasks/734",
	  headers : {authToken : "gfUH43trfdkjg34"},
	  dataType : "json",
	  contentType : "application/json",		

```
    data : JSON.stringify({
      "content" : "Call Nancy",
      "dueDate" : 1360190157000,
      "relatedTo" : {
          "id" : 462,
          "type" : "contact"
      },
      "status" : "completed",
      "title" : "Birthday"
    }),
    success : function(data) {
      console.log(data);
    },
    error : function(data) {
      console.log(data);
    }
  });
```

Notes	*dueDate* is represented by the number of milliseconds since the standard base time known as "the epoch", namely January 1, 1970, 00:00:00 GMT. The attributes *duration* and *phoneNumberIds* are only relevant when the task is a “scheduled call”.

Table 49 Task update

2.5.3.1.39 Delete Tasks

Title	Delete tasks with the given ids		
URL	/v2/tasks?ids=:ids		
Header attributes	**Attribute**	**Required?**	**Example**
	authToken	yes	authToken=gfUH43trfdkjg34
Method	DELETE		
URL parameters	**Parameter**	**Required?**	**Example**
	ids : [<integer>]	Yes	ids=1056,973,1294
Data	none		
Success response	Code: 200 OK Content: {"code" : 204, "devmsg" : "success"}		
Error response	Code: 400 BAD REQUEST Code: 401 UNAUTHORIZED Code: 403 FORBIDDEN Content: {"code" : <integer>, "error" : <integer>, "devmsg" : <alphanumeric>}		

<table>
<tr><td><b>Code example</b></td><td>

```
$.ajax({
  type : "DELETE",
  url : "/v2/tasks?ids=1056,973,1294",
  headers : {authToken : "gfUH43trfdkjg34"},
  dataType : "json",
  success : function(data) {
    console.log(data);
  },
  error : function(data) {
    console.log(data);
  }
});
```

</td></tr>
<tr><td><b>Notes</b></td><td>none</td></tr>
</table>

Table 50 Tasks deletion

2.5.3.1.40 Delete a Task

Title	Delete a particular task		
URL	/v2/tasks/:id		
Header attributes	**Attribute**	**Required?**	**Example**
	`authToken`	yes	`authToken=gfUH43trfdkjg34`
Method	`DELETE`		
URL parameters	**Parameter**	**Required?**	**Example**
	`id : <integer>`	yes	`/481`
Data	none		
Success response	Code: 200 OK Content: `{"code" : 204, "devmsg" : "success"}`		
Error response	Code: 401 UNAUTHORIZED Code: 403 FORBIDDEN Content: `{"code" : <integer>, "error" : <integer>, "devmsg" : <alphanumeric>}`		

Code example	`$.ajax({`
	`  type : "DELETE",`
	`  url : "/v2/tasks/481",`
	`  headers : {authToken : "gfUH43trfdkjg34"},`
	`  dataType : "json",`
	`  success : function(data) {`
	`    console.log(data);`
	`  },`
	`  error : function(data) {`
	`    console.log(data);`
	`  }`
	`});`		

Notes	none		

Table 51 Task deletion

2.5.3.1.41 Create a Tasklist

Title	Create a tasklist		
URL	/v2/taskLists		
Header attributes	**Attribute**	**Required?**	**Example**
	`authToken`	yes	`authToken=gfUH43trfdkjg34`
Method	PUT		
URL parameters	none		

Data	`{`
	`  "title" : <alphanumeric>`
	`}`		

Success response	Code: 201 CREATED		
	`{"code" : 201, "entity" : <taskList>, "devmsg" : "success"}`		
Error response	Code: 400 BAD REQUEST Code: 401 UNAUTHORIZED Code: 403 FORBIDDEN		
	`{"code" : <integer>, "error" : <integer>, "devmsg" : <alphanumeric>}`		

Code example	`$.ajax({`
	`  type : "PUT",`
	`  url : "/v2/taskLists",`
	`  headers : {authToken : "gfUH43trfdkjg34"},`
	`  dataType : "json",`
	`  contentType : "application/json",`
	`  data : JSON.stringify({`
	`    "title" : "EMEA"`
	`  }]),`
	`  success : function(data) {`
	`    console.log(data);`
	`  },`
	`  error : function(data) {`
	`    console.log(data);`
	`  }`
	`});`		

Notes	none		

Table 52 Tasklist creation

2.5.3.1.42 Read a List of Tasklists

Title	Returns all the tasks lists associated to the requesting user.		
URL	/v2/taskLists?callback=:callback		
Header attributes	**Attribute**	**Required?**	**Example**
	authToken	yes	authToken=gfUH43trfdkjg34
Method	GET		
URL parameters	**Parameter**	**Required?**	**Example**
	callback : <string>	no	?callback=jsonpcallback
Data	none		
Success response	Code: 200 OK		

Success response

Error response

```
[{
  "id" : <integer>,
  "title" : <alphanumeric>,
  "tasks" :
    [{
      "id" : <integer>,
      "assignedTo" :
        {
          "id" : <integer>,
          "displayName" : <alphanumeric>
        },
      "concreteDueDate" : <integer>,
      "content" : <alphanumeric>,
      "created" : <integer>,
      "duration" : <integer>,
      "fuzzyDueDate" : <enum(tomorrow,next_week,in_a_month,someday)>,
      "hot" : <boolean>,
      "listId" : <integer>,
      "phoneNumbers" :
        [{
          "id" : <integer>,
          "type" : <enum(mobile,phone)>,
          "subtype" : <enum(home,work,other)>,
          "value" : <alphanumeric>,
          "excluded" : <boolean>
        }],
      "relatedTo" :
        {
          "id" : <integer>,
          "type" : <enum(contact,deal)>,
          "displayName" : <alphanumeric>
        },
      "status" : <enum(completed,open)>,
      "taskType" : <enum(task,scheduled_call>,
      "title" : <alphanumeric>
    }]
}]
```

Code: 401 UNAUTHORIZED

Code: 403 FORBIDDEN

Error response	{"code" : <integer>, "error" : <integer>, "devmsg" : <alphanumeric>}
Code example	$.ajax({ type : "GET",

	`  url : "/v2/taskLists",`
	`  headers : {authToken : "gfUH43trfdkjg34"},`
	`  dataType : "json",`
	`  success : function(data) {`
	`    console.log(data);`
	`  },`
	`  error : function(data) {`
	`    console.log(data);`
	`  }`
	`});`

Notes	none

Table 53 Tasklist list request

2.5.3.1.43 Read a Tasklist

Title	Returns the requested task list that is associated to the user.		
URL	/v2/taskLists/:id?callback=:callback		
Header attributes	**Attribute**	**Required?**	**Example**
	authToken	yes	authToken=gfUH43trfdkjg34
Method	GET		
URL parameters	**Parameter**	**Required?**	**Example**
	id : <integer> callback : <string>	Yes No	/47 ?callback=jsonpcallback
Data	none		
Success response	Code: 200 OK Content:		

```
{
  "id" : <integer>,
  "title" : <alphanumeric>,
  "tasks" :
    [{
      "id" : <integer>,
      "assignedTo" :
        {
          "id" : <integer>,
          "displayName" : <alphanumeric>
        },
      "concreteDueDate" : <integer>,
      "content" : <alphanumeric>,
      "created" : <integer>,
      "duration" : <integer>,
      "fuzzyDueDate" : <enum(tomorrow,next_week,in_a_month,someday)>,
      "hot" : <boolean>,
      "listId" : <integer>,
      "phoneNumbers" :
        [{
          "id" : <integer>,
          "type" : <enum(mobile,phone)>,
          "subtype" : <enum(home,work,other)>,
          "value" : <alphanumeric>,
          "excluded" : <boolean>
        }],
      "relatedTo" :
        {
          "id" : <integer>,
          "type" : <enum(contact,deal)>,
          "displayName" : <alphanumeric>
        },
      "status" : <enum(completed,open)>,
      "taskType" : <enum(task,scheduled_call>,
      "title" : <alphanumeric>
    }]
}
```

Error response	Code: 401 UNAUTHORIZED Code: 403 FORBIDDEN Code: 404 NOT FOUND Content: `{"code" : <integer>, "error" : <integer>, "devmsg" : <alphanumeric>}`

Code example	`$.ajax({`
	`  type : "GET",`
	`  url : "/v2/taskLists/47",`
	`  headers : {authToken : "gfUH43trfdkjg34"},`
	`  dataType : "json",`
	`  success : function(data) {`
	`    console.log(data);`
	`  },`
	`  error : function(data) {`
	`    console.log(data);`
	`  }`
	`});`

Notes	*none*

Table 54 Tasklist request

2.5.3.1.44 Update a Tasklist

<table>
<tr><td><b>Title</b></td><td colspan="3">Update a particular tasklist</td></tr>
<tr><td><b>URL</b></td><td colspan="3">/v2/tasksLists/:id</td></tr>
<tr><td rowspan="2"><b>Header attributes</b></td><td><b>Attribute</b></td><td><b>Required?</b></td><td><b>Example</b></td></tr>
<tr><td>authToken</td><td>yes</td><td>authToken=gfUH43trfdkjg34</td></tr>
<tr><td><b>Method</b></td><td colspan="3">POST</td></tr>
<tr><td rowspan="2"><b>URL parameters</b></td><td><b>Parameter</b></td><td><b>Required?</b></td><td><b>Example</b></td></tr>
<tr><td>id : <integer></td><td>yes</td><td>/734</td></tr>
<tr><td><b>Data</b></td><td colspan="3"><pre>
{
  "title" : <alphanumeric>
}
</pre></td></tr>
<tr><td><b>Success response</b></td><td colspan="3">Code: 200 OK
Content:
{"code" : 200, "entity" : <taskList>, "devmsg" : "success"}</td></tr>
<tr><td><b>Error response</b></td><td colspan="3">Code: 400 BAD REQUEST
Code: 401 UNAUTHORIZED
Code: 403 FORBIDDEN
Content:
{"code" : <integer>, "error" : <integer>, "devmsg" : <alphanumeric>}</td></tr>
<tr><td><b>Code example</b></td><td colspan="3"><pre>
$.ajax({
  type : "POST",
  url : "/v2/taskLists/734",
  headers : {authToken : "gfUH43trfdkjg34"},
  dataType : "json",
  contentType : "application/json",
  data : JSON.stringify({
    "title" : "IBM EMEA"
  }),
  success : function(data) {
    console.log(data);
  },
  error : function(data) {
    console.log(data);
  }
});
</pre></td></tr>
<tr><td><b>Notes</b></td><td colspan="3"><i>none</i></td></tr>
</table>

Table 55 Tasklist update

2.5.3.1.45 Delete a Tasklist

<table>
<tr><td><b>Title</b></td><td colspan="3">Deletes the task list defined by the id path parameter.</td></tr>
<tr><td><b>URL</b></td><td colspan="3">/v2/taskLists/:id</td></tr>
<tr><td rowspan="2"><b>Header attributes</b></td><td><b>Attribute</b></td><td><b>Required?</b></td><td><b>Example</b></td></tr>
<tr><td>authToken</td><td>yes</td><td>authToken=gfUH43trfdkjg34</td></tr>
<tr><td><b>Method</b></td><td colspan="3">DELETE</td></tr>
<tr><td rowspan="2"><b>URL parameters</b></td><td><b>Parameter</b></td><td><b>Required?</b></td><td><b>Example</b></td></tr>
<tr><td>id : &lt;integer&gt;</td><td>yes</td><td>/481</td></tr>
<tr><td><b>Data</b></td><td colspan="3">none</td></tr>
<tr><td><b>Success response</b></td><td colspan="3">Code: 200 OK
Content:
{"code" : 204, "devmsg" : "success"}</td></tr>
<tr><td><b>Error response</b></td><td colspan="3">Code: 401 UNAUTHORIZED
Code: 403 FORBIDDEN
Content:
{"code" : <integer>, "error" : <integer>, "devmsg" : <alphanumeric&gt;}</td></tr>
<tr><td><b>Code example</b></td><td colspan="3"><pre>
$.ajax({
  type : "DELETE",
  url : "/v2/taskLists/481",
  headers : {authToken : "gfUH43trfdkjg34"},
  dataType : "json",
  success : function(data) {
    console.log(data);
  },
  error : function(data) {
    console.log(data);
  }
});
</pre></td></tr>
<tr><td><b>Notes</b></td><td colspan="3">none</td></tr>
</table>

Table 56 Tasklist deletion

2.5.3.1.46 Create Deal

Title	Create a deal		
URL	/v2/deals		
Header attributes	**Attribute**	**Required?**	**Example**
	authToken	yes	authToken=gfUH43trfdkjg34
Method	PUT		
URL parameters	none		

Data	{
	  "name" : <alphanumeric>,
	  "value" : <integer>,
	  "currency" : <alphanumeric>,
	  "hot" : <boolean>,
	  "relatedTo" :
	    {
	      "id" : <integer>
	    },
	  "source" : <enum(website,referral,word_of_mouth,cold_call)>,
	  "stage" :
	<enum(incoming,qualified,quote,closing,won,lost,unqualified)>
	}		

Success response	Code: 201 CREATED Content: {"code" : 201, "entity" : <deal>, "devmsg" : "success"}		
Error response	Code: 400 BAD REQUEST Code: 401 UNAUTHORIZED Code: 403 FORBIDDEN Content: {"code" : <integer>, "error" : <integer>, "devmsg" : <alphanumeric>}		

Code example	$.ajax({
	  type : "PUT",
	  url : "/v2/deals",
	  headers : {authToken : "gfUH43trfdkjg34"},
	  dataType : "json",
	  contentType : "application/json",
	  data : JSON.stringify({
	    "name" : "Apple Deal",
	    "value" : 100000,
	    "currency" : "USD",
	    "hot" : yes,
	    "relatedTo" : { "id" : 317 },
	    "source" : "word_of_mouth",
	    "stage" : "incoming"
	  })		

	`  success : function(data) {`
	`    console.log(data);`
	`  },`
	`  error : function(data) {`
	`    console.log(data);`
	`  }`
	`});`

Notes	none

Table 57 Deals creation

2.5.3.1.47 Read a List of Deals

Title	Returns all the deals associated to the requesting user		
URL	/v2/deals?callback=:callback		
Header attributes	**Attribute**	**Required?**	**Example**
	authToken	yes	authToken=gfUH43trfdkjg34
Method	GET		
URL parameters	**Parameter**	**Required?**	**Example**
	callback : <string>	no	?callback=jsonpcallback
Data	none		

Success response	Code: 200 OK
	Content:
	[{
	  "id" : <integer>,
	  "name" : <alphanumeric>,
	  "value" : <integer>,
	  "currency" : <alphanumeric>,
	  "hot" : <boolean>,
	  "relatedTo" :
	    {
	      "id" : <integer>,
	      "name" : <alphanumeric>
	    },
	  "source" : <enum(website, referral,word_of_mouth,cold_call)>,
	  "stage" :
	<enum(incoming,qualified,quote,closing,won,lost,unqualified)>,)>
	}]		

Error response	Code: 401 UNAUTHORIZED Content: {"code" : <integer>, "error" : <integer>, "devmsg" : <alphanumeric>}		

Code example	$.ajax({
	  type : "GET",
	  url : "/v2/deals",
	  headers : {authToken : "gfUH43trfdkjg34"},
	  dataType : "json",
	  success : function(data) {
	    console.log(data);
	  },
	  error : function(data) {
	    console.log(data);
	  }
	});		

Notes	none		

Table 58 Deal list request

2.5.3.1.48 Read a List of Deals Associated with a Contact

Title	Retrieve a list of deals associated with a particular contact		
URL	/v2/contacts/:id/deals?callback=:callback		
Header attributes	**Attribute**	**Required?**	**Example**
	authToken	yes	authToken=gfUH43trfdkjg34
Method	GET		
URL parameters	**Parameter**	**Required?**	**Example**
	id : <integer> callback : <string>	Yes No	/517/ ?callback=jsonpcallback
Data	none		

Success response	Code: 200 OK
	Content:
	[{
	"id" : <integer>,
	"name" : <alphanumeric>,
	"value" : <integer>,
	"currency" : <alphanumeric>,
	"hot" : <boolean>,
	"relatedTo" :
	{
	"id" : <integer>,
	"name" : <alphanumeric>
	},
	"source" : <enum(website,referral,word_of_mouth,cold_call)>,
	"stage" :
	<enum(incoming,qualified,quote,closing,won,lost,unqualified)>
	}]		

Error response	Code: 401 UNAUTHORIZED Code: 403 FORBIDDEN Code: 404 NOT FOUND Content: {"code" : <integer>, "error" : <integer>, "devmsg" : <alphanumeric>}		

Code example	$.ajax({
	type : "GET",
	url : "/v2/contacts/517/deals",
	headers : {authToken : "gfUH43trfdkjg34"},
	dataType : "json",
	success : function(data) {		

	`    console.log(data);`
	`  },`
	`  error : function(data) {`
	`    console.log(data);`
	`  }`
	`});`

Notes	none

Table 59 Deal list request - associated with a contact

2.5.3.1.49 Read a Deal

<table>
<tr><td><b>Title</b></td><td colspan="3">Retrieve data of a particular deal</td></tr>
<tr><td><b>URL</b></td><td colspan="3">/v2/deal/:id?callback=:callback</td></tr>
<tr><td rowspan="2"><b>Header attributes</b></td><td><b>Attribute</b></td><td><b>Required?</b></td><td><b>Example</b></td></tr>
<tr><td>authToken</td><td>yes</td><td>authToken=gfUH43trfdkjg34</td></tr>
<tr><td><b>Method</b></td><td colspan="3">GET</td></tr>
<tr><td rowspan="2"><b>URL parameters</b></td><td><b>Parameter</b></td><td><b>Required?</b></td><td><b>Example</b></td></tr>
<tr><td>id : &lt;integer>
callback : <string&gt;</td><td>Yes
No</td><td>/587
?callback=jsonpcallback</td></tr>
<tr><td><b>Data</b></td><td colspan="3">none</td></tr>
<tr><td rowspan="2"><b>Success response</b></td><td colspan="3">Code: 200 OK</td></tr>
<tr><td colspan="3"><pre>
{
  "id" : <integer>,
  "name" : <alphanumeric>,
  "value" : <integer>,
  "currency" : <alphanumeric>,
  "hot" : <boolean>,
  "relatedTo" :
    {
      "id" : <integer>,
      "name" : <alphanumeric>
    },
  "source" : <enum(website,referral,word_of_mouth,cold_call)>,
  "stage" : <enum(incoming,qualified,quote,closing,won,lost,unqualified)>
}
</pre></td></tr>
<tr><td rowspan="2"><b>Error response</b></td><td colspan="3">Code: 400 BAD REQUEST
Code: 401 UNAUTHORIZED
Code: 403 FORBIDDEN</td></tr>
<tr><td colspan="3">{"code" : <integer>, "error" : <integer>, "devmsg" : <alphanumeric&gt;}</td></tr>
<tr><td><b>Code example</b></td><td colspan="3"><pre>
$.ajax({
  type : "GET",
  url : "/v2/deals/587",
  headers : {authToken : "gfUH43trfdkjg34"},
  dataType : "json",
  success : function(data) {
    console.log(data);
  },
  error : function(data) {
    console.log(data);
  }
});
</pre></td></tr>
<tr><td><b>Notes</b></td><td colspan="3">none</td></tr>
</table>

Table 60 Deal request

2.5.3.1.50 Update a Deal

Title	Update a particular deal		
URL	/v2/deals/:id		
Header attributes	**Attribute**	**Required?**	**Example**
	authToken	yes	authToken=gfUH43trfdkjg34
Method	POST		
URL parameters	**Parameter**	**Required?**	**Example**
	id : <integer>	yes	/3812

Data	{
	"name" : <alphanumeric>,
	"value" : <integer>,
	"currency" : <alphanumeric>,
	"hot" : <boolean>,
	"relatedTo" :
	{
	"id" : <integer>
	},
	"source" : <enum(website,referral,word_of_mouth,cold_call)>,
	"stage" :
	<enum(incoming,qualified,quote,closing,won,lost,unqualified)>,)>
	}		

Success response	Code: 200 OK Content: {"code" : 200, "entity" : <deal>, "devmsg" : "success"}		
Error response	Code: 400 BAD REQUEST Code: 401 UNAUTHORIZED Code: 403 FORBIDDEN Content: {"code" : <integer>, "error" : <integer>, "devmsg" : <alphanumeric>}		

Code example	$.ajax({
	type : "POST",
	url : "/v2/deals/3812",
	headers : {authToken : "gfUH43trfdkjg34"},
	dataType : "json",
	contentType : "application/json",
	data : JSON.stringify({
	"name" : "xailabs interact.io",
	"value" : 1555000,
	"currency" : "EUR",
	"hot" : yes,
	"relatedTo" : { "id" : 38 },
	"source" : "word_of_mouth",
	"stage" : "qualified"		

	`  })`
	`  success : function(data) {`
	`    console.log(data);`
	`  },`
	`  error : function(data) {`
	`    console.log(data);`
	`  }`
	`});`

Notes	none

Table 61 Deal update

2.5.3.1.51 Delete Deals

Title	Delete deals with the given ids		
URL	/v2/deals?ids=:ids		
Header attributes	**Attribute**	**Required?**	**Example**
	`authToken`	yes	`authToken=gfUH43trfdkjg34`
Method	`DELETE`		
URL parameters	**Parameter**	**Required?**	**Example**
	`ids : [<integer>]`	Yes	`ids=56,2731,894`
Data	none		
Success response	Code: 200 OK Content: `{"code" : 204, "devmsg" : "success"}`		
Error response	Code: 400 BAD REQUEST Code: 401 UNAUTHORIZED Code: 403 FORBIDDEN Content: `{"code" : <integer>, "error" : <integer>, "devmsg" : <alphanumeric>}`		

Code example	`$.ajax({`
	`  type : "DELETE",`
	`  url : "/v2/deals?ids=56,2731,894",`
	`  headers : {authToken : "gfUH43trfdkjg34"},`
	`  dataType : "json",`
	`  success : function(data) {`
	`    console.log(data);`
	`  },`
	`  error : function(data) {`
	`    console.log(data);`
	`  }`
	`});`		

Notes	none		

Table 62 Deals deletion

2.5.3.1.52 Delete a Deal

Title	Delete a particular deal		
URL	/v2/deals/:id		
Header attributes	**Attribute**	**Required?**	**Example**
	authToken	yes	authToken=gfUH43trfdkjg34
Method	DELETE		
URL parameters	**Parameter**	**Required?**	**Example**
	id : <integer>	yes	/122
Data	none		
Success response	Code: 200 OK Content: {"code" : 204, "devmsg" : "success"}		
Error response	Code: 400 BAD REQUEST Code: 401 UNAUTHORIZED Code: 403 FORBIDDEN Content: {"code" : <integer>, "error" : <integer>, "devmsg" : <alphanumeric>}		

Code example	$.ajax({
	  type : "DELETE",
	  url : "/v2/deals/122",
	  headers : {authToken : "gfUH43trfdkjg34"},
	  dataType : "json",
	  success : function(data) {
	    console.log(data);
	  },
	  error : function(data) {
	    console.log(data);
	  }
	});		

Notes	none		

Table 63 Deal deletion

2.5.3.1.53 Create User

<table>
<tr><td><b>Title</b></td><td colspan="3">Create given user</td></tr>
<tr><td><b>URL</b></td><td colspan="3">/v2/admin/users</td></tr>
<tr><td rowspan="2"><b>Header attributes</b></td><td><b>Attribute</b></td><td><b>Required?</b></td><td><b>Example</b></td></tr>
<tr><td>authToken</td><td>yes</td><td>authToken=gfUH43trfdkjg34</td></tr>
<tr><td><b>Method</b></td><td colspan="3">PUT</td></tr>
<tr><td><b>URL parameters</b></td><td colspan="3">none</td></tr>
<tr><td><b>Data</b></td><td colspan="3"><pre>
{
  "email" : <alphanumeric>,
  "password" : <alphanumeric>,
  "companyId" : <integer>
}
</pre></td></tr>
<tr><td><b>Success response</b></td><td colspan="3">Code: 201 CREATED
Content:
{"code" : 201, "entity" : <user>, "devmsg" : "success"}</td></tr>
<tr><td><b>Error response</b></td><td colspan="3">Code: 400 BAD REQUEST
Code: 401 UNAUTHORIZED
Code: 409 CONFLICT
Content:
{"code" : <integer>, "error" : <integer>, "devmsg" : <alphanumeric>}</td></tr>
<tr><td><b>Code example</b></td><td colspan="3"><pre>
$.ajax({
  type : "PUT",
  url : "/v2/admin/users",
  headers : {authToken : "gfUH43trfdkjg34"},
  dataType : "json",
  contentType : "application/json",
   data : JSON.stringify({
     "email" : "John@gmail.com",
     "password" : "john04"
  }),
  success : function(data) {
    console.log(data);
  },
  error : function(data) {
    console.log(data);
  }
});
</pre></td></tr>
<tr><td><b>Notes</b></td><td colspan="3">The <i>companyId</i> can be given on creation. If not it can be given on update. However the <i>companyId</i> can NOT be changed after it was set in the first time.</td></tr>
</table>

Table 64 User creation

2.5.3.1.54 Read a List of Users

<table>
<tr><td><b>Title</b></td><td>Retrieve a list of users.</td></tr>
<tr><td><b>URL</b></td><td>/v2/admin/users?callback=:callback</td></tr>
<tr><td><b>Header attributes</b></td><td><table><tr><th>Attribute</th><th>Required?</th><th>Example</th></tr><tr><td>authToken</td><td>yes</td><td>authToken=gfUH43trfdkjg34</td></tr></table></td></tr>
<tr><td><b>Method</b></td><td>GET</td></tr>
<tr><td><b>URL parameters</b></td><td><table><tr><th>Parameter</th><th>Required?</th><th>Example</th></tr><tr><td>callback : <string&gt;</td><td>no</td><td>?callback=jsonpcallback</td></tr></table></td></tr>
<tr><td><b>Data</b></td><td>none</td></tr>
<tr><td><b>Success response</b></td><td>Code: 200 OK
Content:
<pre>
[{
  "active" : <boolean>,
  "companyId" : <integer>,
  "dateCreated" : <integer>,
  "email" : <alphanumeric>,
  "groups" :
    [{
      "id" : <integer>,
      "name" : <alphanumeric>
    }],
  "id" : <integer>,
  "interactionsCount" : <integer>,
  "roles" : [<alphanumeric>]
}]
</pre></td></tr>
<tr><td><b>Error response</b></td><td>Code: 401 UNAUTHORIZED
Content:
{"code" : <integer>, "error" : <integer>, "devmsg" : <alphanumeric&gt;}</td></tr>
<tr><td><b>Code example</b></td><td><pre>
$.ajax({
  type : "GET",
  url : "/v2/admin/users",
  headers : {authToken : "gfUH43trfdkjg34"},
  dataType : "json",
  success : function(data) {
    console.log(data);
  },
  error : function(data) {
    console.log(data);
  }
});
</pre></td></tr>
</table>

Notes	none

Table 65 User list request

2.5.3.1.55 Read a User

Title	Retrieve data about a particular user.		
URL	/v2/admin/users/:id?callback=:callback		
Header attributes	**Attribute**	**Required?**	**Example**
	authToken	yes	authToken=gfUH43trfdkjg34
Method	GET		
URL parameters	**Parameter**	**Required?**	**Example**
	id : <integer> callback : <string>	Yes No	/587 ?callback=jsonpcallback
Data	none		

Success response	Code: 200 OK
	Content:
	{
	"active" : <boolean>,
	"companyId" : <integer>,
	"dateCreated" : <integer>,
	"email" : <alphanumeric>,
	"groups" :
	[{
	"id" : <integer>,
	"name" : <alphanumeric>
	}],
	"id" : <integer>,
	"interactionsCount" : <integer>,
	"roles" : [<alphanumeric>]
	}		

Error response	Code: 401 UNAUTHORIZED Code: 404 NOT FOUND Content: {"code" : <integer>, "error" : <integer>, "devmsg" : <alphanumeric>}		

Code example	$.ajax({
	type : "GET",
	url : "/v2/admin/users/587",
	headers : {authToken : "gfUH43trfdkjg34"},
	dataType : "json",
	success : function(data) {
	console.log(data);
	},		

	error : function(data) {
	console.log(data);
	}
	});

Notes	none

Table 66 Single user request

2.5.3.1.56 Delete Users

Title	Delete users with the given ids		
URL	/v2/admin/users?ids=:ids		
Header attributes	**Attribute**	**Required?**	**Example**
	`authToken`	Yes	`authToken=gfUH43trfdkjg34`
Method	DELETE		
URL parameters	**Parameter**	**Required?**	**Example**
	`ids : [<integer>]`	Yes	`ids=56,2731,894`
Data	`none`		
Success response	Code: 200 OK		
	`{"code" : 204, "devmsg" : "success"}`		
Error response	Code: 401 UNAUTHORIZED Code: 403 FORBIDDEN		
	`{"code" : <integer>, "error" : <integer>, "devmsg" : <alphanumeric>}`		

Code example	`$.ajax({`
	`type : "DELETE",`
	`url : "/v2/admin/users?ids=56,2731,894",`
	`headers : {authToken : "gfUH43trfdkjg34"},`
	`dataType : "json",`
	`success : function(data) {`
	`console.log(data);`
	`},`
	`error : function(data) {`
	`console.log(data);`
	`}`
	`});`		

Notes	none		

Table 67 User deletion

2.5.3.1.57 Delete a User

Title	Delete the user with the given id		
URL	/v2/admin/users/:id		
Header attributes	**Attribute**	**Required?**	**Example**
	`authToken`	yes	`authToken=gfUH43trfdkjg34`
Method	`DELETE`		
URL parameters	**Parameter**	**Required?**	**Example**
	`id : <integer>`	Yes	`/734`
Data	none		
Success response	Code: 200 OK Content: `{"code" : 204, "devmsg" : "success"}`		
Error response	Code: 401 UNAUTHORIZED Code: 403 FORBIDDEN Content: `{"code" : <integer>, "error" : <integer>, "devmsg" : <alphanumeric>}`		

Code example	`$.ajax({`
	`  type : "DELETE",`
	`  url : "/v2/admin/users/734",`
	`  headers : {authToken : "gfUH43trfdkjg34"},`
	`  dataType : "json",`
	`  success : function(data) {`
	`    console.log(data);`
	`  },`
	`  error : function(data) {`
	`    console.log(data);`
	`  }`
	`});`		

Notes	none		

Table 68 User deletion

2.5.3.1.58 Create a Company

Title	Create a customer company.		
URL	/v2/admin/companies		
Header attributes	**Attribute**	**Required?**	**Example**
	authToken	yes	authToken=gfUH43trfdkjg34
Method	PUT		
URL parameters	none		

Data	{
	  "name" : <alphanumeric>
	}		

Success response	Code: 201 CREATED Content: {"code" : 201, "entity" : <company>, "devmsg" : "success"}		
Error response	Code: 400 BAD REQUEST Code: 401 UNAUTHORIZED Code: 403 FORBIDDEN Content: {"code" : <integer>, "error" : <integer>, "devmsg" : <alphanumeric>}		

Code example	$.ajax({
	  type : "PUT",
	  url : "/v2/companies",
	  headers : {authToken : "gfUH43trfdkjg34"},
	  dataType : "json",
	  contentType : "application/json",
	  data : JSON.stringify({
	    "name" : "xailabs GmbH"
	  })
	  success : function(data) {
	    console.log(data);
	  },
	  error : function(data) {
	    console.log(data);
	  }
	});		

Notes	none		

Table 69 Company creation

2.5.3.1.59 Read a List Companies

<table>
<tr><td><b>Title</b></td><td>Retrieve a list of customer companies</td></tr>
<tr><td><b>URL</b></td><td>/v2/admin/companies?callback=:callback</td></tr>
<tr><td><b>Header attributes</b></td><td><table><tr><th>Attribute</th><th>Required?</th><th>Example</th></tr><tr><td>authToken</td><td>yes</td><td>authToken=gfUH43trfdkjg34</td></tr></table></td></tr>
<tr><td><b>Method</b></td><td>GET</td></tr>
<tr><td><b>URL parameters</b></td><td><table><tr><th>Parameter</th><th>Required?</th><th>Example</th></tr><tr><td>callback : <string&gt;</td><td>no</td><td>?callback=jsonpcallback</td></tr></table></td></tr>
<tr><td><b>Data</b></td><td>none</td></tr>
<tr><td><b>Success response</b></td><td>Code: 200 OK
Content:
<pre>
[{
  "id" : <integer>,
  "name" : <alphanumeric>,
  "companyId" : <integer>,
  "users" :
    [{
      "active" : <boolean>,
      "companyId" : <integer>,
      "dateCreated" : <integer>,
      "email" : <alphanumeric>,
      "groups" :
        [{
          "id" : <integer>,
          "name" : <alphanumeric>
        }],
      "id" : <integer>,
      "interactionsCount" : <integer>,
      "roles" : [<alphanumeric>]
    }]
}]
</pre></td></tr>
<tr><td><b>Error response</b></td><td>Code: 401 UNAUTHORIZED
Code: 403 FORBIDDEN
Code: 404 NOT FOUND
Content:
<pre>
{"code" : <integer>, "error" : <integer>, "devmsg" : <alphanumeric>}
</pre></td></tr>
<tr><td><b>Code example</b></td><td><pre>
$.ajax({
  type : "GET",
  url : "/v2/contacts/517/deals",
  headers : {authToken : "gfUH43trfdkjg34"},
  dataType : "json",
  success : function(data) {
    console.log(data);
  },
</pre></td></tr>
</table>

	`  error : function(data) {`
	`    console.log(data);`
	`  }`
	`});`

Notes	none

Table 70 Company list request

2.5.3.1.60 Read a Company

Title	Retrieve data about a particular company.		
URL	/v2/admin/companies/:id?callback=:callback		
Header attributes	**Attribute**	**Required?**	**Example**
	authToken	yes	authToken=gfUH43trfdkjg34
Method	GET		
URL parameters	**Parameter**	**Required?**	**Example**
	id : <integer> callback : <string>	Yes No	/552 ?callback=jsonpcallback
Data	none		

Success response	Code: 200 OK
	Content:
	{
	"id" : <integer>,
	"name" : <alphanumeric>,
	"companyId" : <integer>,
	"users" :
	[{
	"active" : <boolean>,
	"companyId" : <integer>,
	"dateCreated" : <integer>,
	"email" : <alphanumeric>,
	"groups" :
	[{
	"id" : <integer>,
	"name" : <alphanumeric>
	}],
	"id" : <integer>,
	"interactionsCount" : <integer>,
	"roles" : [<alphanumeric>]
	}]
	}		

Error response	Code: 401 UNAUTHORIZED Code: 404 NOT FOUND Content: {"code" : <integer>, "error" : <integer>, "devmsg" : <alphanumeric>}		

Code example	$.ajax({
	type : "GET",
	url : "/v2/admin/companies/552",
	headers : {authToken : "gfUH43trfdkjg34"},
	dataType : "json",
	success : function(data) {
	console.log(data);
	},		

	`  error : function(data) {`
	`    console.log(data);`
	`  }`
	`});`

Notes	none

Table 71 Single company request

2.5.3.1.61 Delete Companies

Title	Delete companies with the given ids		
URL	/v2/admin/companies?ids=:ids		
Header attributes	**Attribute**	**Required?**	**Example**
	`authToken`	yes	`authToken=gfUH43trfdkjg34`
Method	`DELETE`		
URL parameters	**Parameter**	**Required?**	**Example**
	`ids : [<integer>]`	Yes	`ids=456,84,654`
Data	none		
Success response	Code: 200 OK Content: `{"code" : 204, "devmsg" : "success"}`		
Error response	Code: 400 BAD REQUEST Code: 401 UNAUTHORIZED Code: 403 FORBIDDEN Content: `{"code" : <integer>, "error" : <integer>, "devmsg" : <alphanumeric>}`		

Code example

```
$.ajax({
  type : "DELETE",
  url : "/v2/admin/companies?ids=456,84,654",
  headers : {authToken : "gfUH43trfdkjg34"},
  dataType : "json",
  success : function(data) {
    console.log(data);
  },
  error : function(data) {
    console.log(data);
  }
});
```

Notes	none

Table 72 Companies deletion

2.5.3.1.62 Create a Group

Title	Create a group for sharing contacts with users.		
URL	/v2/admin/groups		
Header attributes	**Attribute**	**Required?**	**Example**
	`authToken`	yes	`authToken=gfUH43trfdkjg34`
Method	PUT		
URL parameters	none		

Data	`{`
	`"name" : <alphanumeric>,`
	`"companyId" : <integer>`
	`}`		

Success response	Code: 201 CREATED Content: `{"code" : 201, "entity" : <group>, "devmsg" : "success"}`		
Error response	Code: 400 BAD REQUEST Code: 401 UNAUTHORIZED Code: 403 FORBIDDEN Content: `{"code" : <integer>, "error" : <integer>, "devmsg" : <alphanumeric>}`		

Code example	`$.ajax({`
	`type : "PUT",`
	`url : "/v2/admin/groups",`
	`headers : {authToken : "gfUH43trfdkjg34"},`
	`dataType : "json",`
	`contentType : "application/json",`
	`data : JSON.stringify({`
	`"name" : "Sales dept.",`
	`"companyId" : 12`
	`})`
	`success : function(data) {`
	`console.log(data);`
	`},`
	`error : function(data) {`
	`console.log(data);`
	`}`
	`});`		

Notes	none		

Table 73 Group creation

2.5.3.1.63 Read a List of Groups Associated with a Company

<table>
<tr><td><b>Title</b></td><td colspan="3">Retrieve a list of groups associated with a particular company</td></tr>
<tr><td><b>URL</b></td><td colspan="3">/v2/admin/companies/:id/groups?callback=:callback</td></tr>
<tr><td rowspan="2"><b>Header attributes</b></td><td><b>Attribute</b></td><td><b>Required?</b></td><td><b>Example</b></td></tr>
<tr><td>authToken</td><td>yes</td><td>authToken=gfUH43trfdkjg34</td></tr>
<tr><td><b>Method</b></td><td colspan="3">GET</td></tr>
<tr><td rowspan="2"><b>URL parameters</b></td><td><b>Parameter</b></td><td><b>Required?</b></td><td><b>Example</b></td></tr>
<tr><td>id : &lt;integer>
callback : <string&gt;</td><td>Yes
No</td><td>/758/
?callback=jsonpcallback</td></tr>
<tr><td><b>Data</b></td><td colspan="3">none</td></tr>
<tr><td><b>Success response</b></td><td colspan="3">Code: 200 OK
Content:
<pre>
[{
  "id" : <integer>,
  "name" : <alphanumeric>,
  "companyId" : <integer>,
  "users" :
    [{
      "active" : <boolean>,
      "companyId" : <integer>,
      "dateCreated" : <integer>,
      "email" : <alphanumeric>,
      "groups" :
        [{
          "id" : <integer>,
          "name" : <alphanumeric>
        }],
      "id" : <integer>,
      "interactionsCount" : <integer>,
      "roles" : [<alphanumeric>]
    }]
}]
</pre></td></tr>
<tr><td><b>Error response</b></td><td colspan="3">Code: 401 UNAUTHORIZED
Code: 403 FORBIDDEN
Code: 404 NOT FOUND
Content:
{"code" : <integer>, "error" : <integer>, "devmsg" : <alphanumeric&gt;}</td></tr>
<tr><td><b>Code example</b></td><td colspan="3"><pre>
$.ajax({
  type : "GET",
  url : "/v2/admin/companies/758/groups",
  headers : {authToken : "gfUH43trfdkjg34"},
  dataType : "json",
  success : function(data) {
</pre></td></tr>
</table>

	`    console.log(data);`
	`  },`
	`  error : function(data) {`
	`    console.log(data);`
	`  }`
	`});`

Notes	none

Table 74 Group list request - associated with a company

2.5.3.1.64 Read a Group

Title	Retrieve data about a particular group.		
URL	/v2/admin/groups/:id?callback=:callback		
Header attributes	**Attribute**	**Required?**	**Example**
	authToken	yes	authToken=gfUH43trfdkjg34
Method	GET		
URL parameters	**Parameter**	**Required?**	**Example**
	id : <integer> callback : <string>	Yes No	/875 ?callback=jsonpcallback
Data	none		

Success response	Code: 200 OK
	Content:
	{
	  "id" : <integer>,
	  "name" : <alphanumeric>,
	  "companyId" : <integer>,
	  "users" :
	    [{
	      "active" : <boolean>,
	      "companyId" : <integer>,
	      "dateCreated" : <integer>,
	      "email" : <alphanumeric>,
	      "groups" :
	        [{
	          "id" : <integer>,
	          "name" : <alphanumeric>
	        }],
	      "id" : <integer>,
	      "interactionsCount" : <integer>,
	      "roles" : [<alphanumeric>]
	    }]
	}		

Error response	Code: 401 UNAUTHORIZED Code: 404 NOT FOUND Content: {"code" : <integer>, "error" : <integer>, "devmsg" : <alphanumeric>}		

Code example	$.ajax({
	  type : "GET",
	  url : "/v2/admin/groups/875",
	  headers : {authToken : "gfUH43trfdkjg34"},
	  dataType : "json",
	  success : function(data) {
	    console.log(data);
	  },		

	`  error : function(data) {`
	`    console.log(data);`
	`  }`
	`});`

Notes	none

Table 75 Single group request

2.5.3.1.65 Update a Group

<table>
<tr><td><b>Title</b></td><td colspan="3">Update a particular group</td></tr>
<tr><td><b>URL</b></td><td colspan="3">/v2/admin/groups/:id</td></tr>
<tr><td rowspan="2"><b>Header attributes</b></td><td><b>Attribute</b></td><td><b>Required?</b></td><td><b>Example</b></td></tr>
<tr><td>authToken</td><td>yes</td><td>authToken=gfUH43trfdkjg34</td></tr>
<tr><td><b>Method</b></td><td colspan="3">POST</td></tr>
<tr><td rowspan="2"><b>URL parameters</b></td><td><b>Parameter</b></td><td><b>Required?</b></td><td><b>Example</b></td></tr>
<tr><td>id : &lt;integer&gt;</td><td>yes</td><td>/46</td></tr>
<tr><td><b>Data</b></td><td colspan="3"><pre>{
  "name" : <alphanumeric>
}</pre></td></tr>
<tr><td rowspan="2"><b>Success response</b></td><td colspan="3">Code: 200 OK</td></tr>
<tr><td colspan="3">{"code" : 200, "entity" : <group>, "devmsg" : "success"}</td></tr>
<tr><td rowspan="2"><b>Error response</b></td><td colspan="3">Code: 400 BAD REQUEST
Code: 401 UNAUTHORIZED
Code: 403 FORBIDDEN</td></tr>
<tr><td colspan="3">{"code" : <integer>, "error" : <integer>, "devmsg" : <alphanumeric&gt;}</td></tr>
<tr><td><b>Code example</b></td><td colspan="3"><pre>$.ajax({
  type : "POST",
  url : "/v2/admin/group/46",
  headers : {authToken : "gfUH43trfdkjg34"},
  dataType : "json",
  contentType : "application/json",
  data : JSON.stringify({
    "name" : "sales department"
  })
  success : function(data) {
    console.log(data);
  },
  error : function(data) {
    console.log(data);
  }
});</pre></td></tr>
<tr><td><b>Notes</b></td><td colspan="3">none</td></tr>
</table>

Table 76 Group update

2.5.3.1.66 Delete Groups

<table>
<tr><td><b>Title</b></td><td colspan="3">Delete groups with the given ids</td></tr>
<tr><td><b>URL</b></td><td colspan="3">/v2/admin/groups?ids=:ids</td></tr>
<tr><td rowspan="2"><b>Header attributes</b></td><td><b>Attribute</b></td><td><b>Required?</b></td><td><b>Example</b></td></tr>
<tr><td>authToken</td><td>yes</td><td>authToken=gfUH43trfdkjg34</td></tr>
<tr><td><b>Method</b></td><td colspan="3">DELETE</td></tr>
<tr><td rowspan="2"><b>URL parameters</b></td><td><b>Parameter</b></td><td><b>Required?</b></td><td><b>Example</b></td></tr>
<tr><td>ids : [<integer>]</td><td>Yes</td><td>ids=56,2731,894</td></tr>
<tr><td><b>Data</b></td><td colspan="3">none</td></tr>
<tr><td><b>Success response</b></td><td colspan="3">Code: 200 OK
Content:
{"code" : 204, "devmsg" : "success"}</td></tr>
<tr><td><b>Error response</b></td><td colspan="3">Code: 400 BAD REQUEST
Code: 401 UNAUTHORIZED
Code: 403 FORBIDDEN
Content:
{"code" : <integer>, "error" : <integer>, "devmsg" : <alphanumeric>}</td></tr>
<tr><td><b>Code example</b></td><td colspan="3"><pre>
$.ajax({
  type : "DELETE",
  url : "/v2/admin/groups?ids=56,2731,894",
  headers : {authToken : "gfUH43trfdkjg34"},
  dataType : "json",
  success : function(data) {
    console.log(data);
  },
  error : function(data) {
    console.log(data);
  }
});
</pre></td></tr>
<tr><td><b>Notes</b></td><td colspan="3">none</td></tr>
</table>

Table 77 Groups deletion

2.5.3.1.67 Delete a Group

Title	Delete a particular group		
URL	/v2/admin/groups/:id		
Header attributes	**Attribute**	**Required?**	**Example**
	`authToken`	yes	`authToken=gfUH43trfdkjg34`
Method	`DELETE`		
URL parameters	**Parameter**	**Required?**	**Example**
	`id : <integer>`	yes	`/289`
Data	none		
Success response	Code: 200 OK Content: `{"code" : 204, "devmsg" : "success"}`		
Error response	Code: 400 BAD REQUEST Code: 401 UNAUTHORIZED Code: 403 FORBIDDEN Content: `{"code" : <integer>, "error" : <integer>, "devmsg" : <alphanumeric>}`		
Code example	see below		
Notes	none		

Code example:

```
$.ajax({
  type : "DELETE",
  url : "/v2/admin/groups/289",
  headers : {authToken : "gfUH43trfdkjg34"},
  dataType : "json",
  success : function(data) {
    console.log(data);
  },
  error : function(data) {
    console.log(data);
  }
});
```

Table 78 Group deletion

2.5.3.1.68 Register a Device

Title	Registers a new device for push notifications.		
URL	/v2/devices		
Header attributes	**Attribute**	**Required?**	**Example**
	`authToken`	Yes	`authToken=gfUH43trfdkjg34`
Method	PUT		
URL Parameters	none		

Data	`{`
	`  "active" : <boolean>,`
	`  "decoded" : <boolean>,`
	`  "deviceId" : <alphanumeric>,`
	`  "name" : <alphanumeric>`
	`}`		

Success response	Code: 201 CREATED		
	`{"code" : 201, "entity" : <device>, "devmsg" : "success"}`		
Error response	Code: 400 BAD REQUEST Code: 401 UNAUTHORIZED Code: 403 FORBIDDEN		
	`{"code" : <integer>, "error" : <integer>, "devmsg" : <alphanumeric>}`		

Code example	`$.ajax({`
	`  type : "PUT",`
	`  url : "/v2/devices",`
	`  headers : {authToken : "gfUH43trfdkjg34"},`
	`  dataType : "json",`
	`  contentType : "application/json",`
	`  data : JSON.stringify({`
	`    "deviceId" : "6g79bcd2b124bcd835a0769cas3f9b12bbd2b8a779f7fde29b7c11897dcff4q8",`
	`    "name" : "Ipad 2",`
	`    "decoded" : true,`
	`    "active" : true`
	`  }),`
	`  success : function(data) {`
	`    console.log(data);`
	`  },`
	`  error : function(data) {`
	`    console.log(data);`
	`  }`
	`});`		

Notes	The *deviceId* can be passed as base64-encoded string of the raw-byte-data or as decoded hex-string (with 64 characters representing the 8 octets of the deviceId). If second option is used the *decoded* attribute must be set to *TRUE*.		

Table 79 Register a device

2.5.3.1.69 Read a List of Devices

<table>
<tr><td><b>Title</b></td><td colspan="3">Read a list of devices.</td></tr>
<tr><td><b>URL</b></td><td colspan="3">/v2/devices?callback=:callback</td></tr>
<tr><td rowspan="2"><b>Header attributes</b></td><td><b>Attribute</b></td><td><b>Required?</b></td><td><b>Example</b></td></tr>
<tr><td><code>authToken</code></td><td>Yes</td><td><code>authToken=gfUH43trfdkjg34</code></td></tr>
<tr><td><b>Method</b></td><td colspan="3"><code>GET</code></td></tr>
<tr><td rowspan="2"><b>URL parameters</b></td><td><b>Parameter</b></td><td><b>Required?</b></td><td><b>Example</b></td></tr>
<tr><td><code>callback : <string></code></td><td>no</td><td><code>?callback=jsonpcallback</code></td></tr>
<tr><td><b>Data</b></td><td colspan="3"><code>none</code></td></tr>
<tr><td rowspan="2"><b>Success response</b></td><td colspan="3">Code: 200 OK</td></tr>
<tr><td colspan="3"><pre>
[{
   "id" : <integer>,
   "active" : <boolean>,
   "deviceId" : <alphanumeric>,
   "name" : <alphanumeric>
}]
</pre></td></tr>
<tr><td rowspan="2"><b>Error response</b></td><td colspan="3">Code: 401 UNAUTHORIZED</td></tr>
<tr><td colspan="3"><code>{"code" : <integer>, "error" : <integer>, "devmsg" : <alphanumeric>}</code></td></tr>
<tr><td><b>Code example</b></td><td colspan="3"><pre>
$.ajax({
  type : "GET",
  url : "/v2/devices",
  headers : {authToken : "gfUH43trfdkjg34"},
  dataType : "json",
  success : function(data) {
    console.log(data);
  },
  error : function(data) {
    console.log(data);
  }
});
</pre></td></tr>
<tr><td><b>Notes</b></td><td colspan="3">none</td></tr>
</table>

Table 80 Read a list of Devices

2.5.3.1.70 Update a Device

Title	Update a registered device		
URL	/v2/devices/:id		
Header attributes	**Attribute**	**Required?**	**Example**
	`authToken`	Yes	`authToken=gfUH43trfdkjg34`
Method	POST		
URL parameters	**Parameter**	**Required?**	**Example**
	`id : <integer>`	yes	`/31`

Data	{
	  "name" : <alphanumeric>,
	  "active" : <boolean>
	}		

Success response	Code: 200 OK Content: {"code" : 200, "entity" : <device>, "devmsg" : "success"}		
Error response	Code: 400 BAD REQUEST Code: 401 UNAUTHORIZED Code: 403 FORBIDDEN Content: {"code" : <integer>, "error" : <integer>, "devmsg" : <alphanumeric>}		

Code example	$.ajax({
	  type : "POST",
	  url : "/v2/devices/31",
	  headers : {authToken : "gfUH43trfdkjg34"},
	  dataType : "json",
	  contentType : "application/json",
	  data : JSON.stringify({
	    "name" : "Ipad 2",
	    "active" : false
	  }),
	  success : function(data) {
	    console.log(data);
	  },
	  error : function(data) {
	    console.log(data);
	  }
	});		

Notes	none		

Table 81 Update a device

2.5.3.1.71 Delete a Device

<table>
<tr><td><b>Title</b></td><td colspan="3">Delete a device</td></tr>
<tr><td><b>URL</b></td><td colspan="3">/v2/devices/:id</td></tr>
<tr><td rowspan="2"><b>Header attributes</b></td><td><b>Attribute</b></td><td><b>Required?</b></td><td><b>Example</b></td></tr>
<tr><td><code>authToken</code></td><td>Yes</td><td><code>authToken=gfUH43trfdkjg34</code></td></tr>
<tr><td><b>Method</b></td><td colspan="3">DELETE</td></tr>
<tr><td rowspan="2"><b>URL parameters</b></td><td><b>Parameter</b></td><td><b>Required</b></td><td><b>Example</b></td></tr>
<tr><td><code>id : &lt;integer&gt;</code></td><td>Yes</td><td><code>/3</code></td></tr>
<tr><td><b>Data</b></td><td colspan="3">none</td></tr>
<tr><td rowspan="2"><b>Success response</b></td><td colspan="3">Code: 200 OK</td></tr>
<tr><td colspan="3"><code>{"code" : 204, "devmsg" : "success"}</code></td></tr>
<tr><td rowspan="2"><b>Error response</b></td><td colspan="3">Code: 400 BAD REQUEST
Code: 401 UNAUTHORIZED
Code: 403 FORBIDDEN</td></tr>
<tr><td colspan="3"><code>{"code" : <integer>, "error" : <integer>, "devmsg" : <alphanumeric&gt;}</code></td></tr>
<tr><td><b>Code example</b></td><td colspan="3"><pre>$.ajax({
  type : "DELETE",
  url : "/v2/devices/3",
  headers : {authToken : "gfUH43trfdkjg34"},
  dataType : "json",
  success : function(data) {
    console.log(data);
  },
  error : function(data) {
    console.log(data);
  }
});</pre></td></tr>
<tr><td><b>Notes</b></td><td colspan="3">none</td></tr>
</table>

Table 82 Delete a device

2.5.3.1.72 Create a Custom Field

Title	Create a custom field for an entity type.		
URL	/v2/admin/customFields		
Header attributes	**Attribute**	**Required?**	**Example**
	`authToken`	yes	`authToken=gfUH43trfdkjg34`
Method	PUT		
URL parameters	none		

Data	{
	  "companyId" : <integer>,
	  "dataType" : <enum(varchar)>,
	  "entityType" : <enuam(contact)>,
	  "name" : <alphanumeric>
	}		

Success response	Code: 201 CREATED		
	{"code" : 201, "entity" : <customField>, "devmsg" : "success"}		
Error response	Code: 400 BAD REQUEST Code: 401 UNAUTHORIZED Code: 403 FORBIDDEN		
	{"code" : <integer>, "error" : <integer>, "devmsg" : <alphanumeric>}		

Code example	$.ajax({
	  type : "PUT",
	  url : "/v2/customFields",
	  headers : {authToken : "gfUH43trfdkjg34"},
	  dataType : "json",
	  contentType : "application/json",
	  data : JSON.stringify({
	    "name" : "family status",
	    "dataType" : "varchar",
	    "entityType" : "contact"
	  })
	  success : function(data) {
	    console.log(data);
	  },
	  error : function(data) {
	    console.log(data);
	  }
	});		

Notes	For now only text fields with *datatype* varchar can be created for entities of type contact. If no *companyId* is given, the creating user must be a member of a company himself and his companyId will be used.		

Table 83 Custom field creation

2.5.3.1.73 Read a List of Custom Fields Associated with a Company

<table>
<tr><td><b>Title</b></td><td>Retrieve a list of custom fields associated with a company.</td></tr>
<tr><td><b>URL</b></td><td>/v2/admin/company/:id/customFields?entityType=:type&callback=:callback</td></tr>
<tr><td><b>Header attributes</b></td><td><table><tr><th>Attribute</th><th>Required?</th><th>Example</th></tr><tr><td>authToken</td><td>yes</td><td>authToken=gfUH43trfdkjg34</td></tr></table></td></tr>
<tr><td><b>Method</b></td><td>GET</td></tr>
<tr><td><b>URL parameters</b></td><td><table><tr><th>Parameter</th><th>Required?</th><th>Example</th></tr><tr><td>id : &lt;integer&gt;</td><td>yes</td><td>/23</td></tr><tr><td>type : <alphanumeric&gt;</td><td>no</td><td>?entityType=contact</td></tr><tr><td>callback : <string&gt;</td><td>no</td><td>&callback=jsonpcallback</td></tr></table></td></tr>
<tr><td><b>Data</b></td><td>none</td></tr>
<tr><td><b>Success response</b></td><td>Code: 200 OK<pre>
[{
  "id" : <integer>,
  "dataType" : <enuam(varchar)>,
  "entityType" : <enuam(contact)>,
  "name" : <alphanumeric>
}]
</pre></td></tr>
<tr><td><b>Error response</b></td><td>Code: 400 BAD REQUEST
Code: 401 UNAUTHORIZED
Code: 403 FORBIDDEN<pre>
{"code" : <integer>, "error" : <integer>, "devmsg" : <alphanumeric>}
</pre></td></tr>
<tr><td><b>Code example</b></td><td><pre>
$.ajax({
  type : "GET",
  url : "/v2/admin/customFields?offset=0&limit=25",
  headers : { authToken : "gfUH43trfdkjg34" },
  dataType : "json",
  success : function(data) {
    console.log(data);
  },
  error : function(data) {
    console.log(data);
  }
});
</pre></td></tr>
<tr><td><b>Notes</b></td><td><i>offset</i> and <i>limit</i> parameters are used for pagination. If both are specified, the response will contain according <i>length</i> and <i>startIndex</i> attributes as well as an additional <i>totalLength</i> attribute to indicate whether there are more custom fields available.
The entityType parameter can be used to filter custom fields for a particular type of entity. For now, the only supported entity type is “contact” – so setting a different type would lead to an empty result list.</td></tr>
</table>

Table 84 Custom field list request – associated with a company

2.5.3.1.74 Read a Custom Field

Title	Retrieve data of a particular custom field		
URL	/v2/admin/customFields/:id?callback=:callback		
Header attributes	**Attribute**	**Required?**	**Example**
	`authToken`	yes	`authToken=gfUH43trfdkjg34`
Method	GET		
URL parameters	**Parameter**	**Required?**	**Example**

	`id : <integer>`
	`callback : <string>`
		Yes
		No
			`/261`
			`?callback=jsonpcallback`

Data	none		
Success response	Code: 200 OK		
	`{ "id" : <integer>, "dataType" : <enuam(varchar)>, "entityType" : <enuam(contact)>, "name" : <alphanumeric> }`		
Error response	Code: 400 BAD REQUEST Code: 401 UNAUTHORIZED Code: 403 FORBIDDEN		
	`{"code" : <integer>, "error" : <integer>, "devmsg" : <alphanumeric>}`		
Code example	see below		
Notes	none		

Success response body:

```
{
  "id" : <integer>,
  "dataType" : <enuam(varchar)>,
  "entityType" : <enuam(contact)>,
  "name" : <alphanumeric>
}
```

Code example:

```
$.ajax({
  type : "GET",
  url : "/v2/admin/customFields/261",
  headers : {authToken : "gfUH43trfdkjg34"},
  dataType : "json",
  success : function(data) {
    console.log(data);
  },
  error : function(data) {
    console.log(data);
  }
});
```

Table 85 Custom field request

2.5.3.1.75 Update a Custom Field

Title	Update a particular custom field.		
URL	/v2/admin/customFields/:id		
Header attributes	**Attribute**	**Required?**	**Example**
	`authToken`	yes	`authToken=gfUH43trfdkjg34`
Method	POST		
URL parameters	**Parameter**	**Required?**	**Example**
	`id : <integer>`	yes	/463

Data	`{`
	`  "name" : <alphanumeric>`
	`}`		

Success response	Code: 200 OK		
	`{"code" : 200, "entity" : <customField>, "devmsg" : "success"}`		
Error response	Code: 400 BAD REQUEST Code: 401 UNAUTHORIZED Code: 403 FORBIDDEN		
	`{"code" : <integer>, "error" : <integer>, "devmsg" : <alphanumeric>}`		

Code example	`$.ajax({`
	`  type : "POST",`
	`  url : "/v2/admin/customFields/463",`
	`  headers : {authToken : "gfUH43trfdkjg34"},`
	`  dataType : "json",`
	`  contentType : "application/json",`
	`  data : JSON.stringify({`
	`    "name" : "hair color"`
	`  })`
	`  success : function(data) {`
	`    console.log(data);`
	`  },`
	`  error : function(data) {`
	`    console.log(data);`
	`  }`
	`});`		

Notes	none		

Table 86 Custom field update

2.5.3.1.76 Delete Custom Fields

Title	Delete custom fields with the given ids		
URL	/v2/admin/customFields?ids=:ids		
Header attributes	**Attribute**	**Required?**	**Example**
	authToken	Yes	authToken=gfUH43trfdkjg34
Method	DELETE		
URL parameters	**Parameter**	**Required?**	**Example**
	ids : [<integer>]	Yes	ids=371,455
Data	none		
Success response	Code: 200 OK		
	{"code" : 204, "devmsg" : "success"}		
Error response	Code: 401 UNAUTHORIZED Code: 403 FORBIDDEN		
	{"code" : <integer>, "error" : <integer>, "devmsg" : <alphanumeric>}		

Code example	$.ajax({
	  type : "DELETE",
	  url : "/v2/admin/customFields?ids=371,455",
	  headers : {authToken : "gfUH43trfdkjg34"},
	  dataType : "json",
	  success : function(data) {
	    console.log(data);
	  },
	  error : function(data) {
	    console.log(data);
	  }
	});		

Notes	none		

Table 87 Custom fields deletion

2.5.3.2 Extended functions

Extended functions are represented by a *cmd* query parameter.

2.5.3.2.1 Synchronize Offline Data

<table>
<tr><td><b>Title</b></td><td colspan="3">Synchronize local DB with data from cloud DB.</td></tr>
<tr><td><b>URL</b></td><td colspan="3">/v2/sync?deviceId=:token&since=:time</td></tr>
<tr><td rowspan="2"><b>Header attributes</b></td><td><b>Attribute</b></td><td><b>Required?</b></td><td><b>Example</b></td></tr>
<tr><td>authToken</td><td>Yes</td><td>authToken=gfUH43trfdkjg34</td></tr>
<tr><td><b>Method</b></td><td colspan="3">GET</td></tr>
<tr><td rowspan="2"><b>URL parameters</b></td><td><b>Parameter</b></td><td><b>Required?</b></td><td><b>Example</b></td></tr>
<tr><td>token : <alphanumeric>
time : <integer&gt;</td><td>yes
no</td><td>?token=12345
&time=1366819829170</td></tr>
<tr><td><b>Data</b></td><td colspan="3">none</td></tr>
<tr><td><b>Success response</b></td><td colspan="3">Code: 200 OK
Content:<pre>
{
  "contacts" : [<contact>],
  "deals" : [<deal>],
  "interactions" : [<interaction>],
  "notes" : [<note>],
  "taskLists" : [<tasklist>],
  "tasks" : [<task>],
  "deleted" :
    [{
      "id" : <integer>,
      "type" : <enum(contact,deal,interaction,note,taskList,task)>
    }]
}
</pre></td></tr>
<tr><td><b>Error response</b></td><td colspan="3">Code: 400 BAD REQUEST
Code: 401 UNAUTHORIZED
Code: 403 FORBIDDEN
Content:
{"code" : <integer>, "error" : <integer>, "devmsg" : <alphanumeric&gt;}</td></tr>
<tr><td><b>Code example</b></td><td colspan="3"><pre>
$.ajax({
  type : "GET",
  url : "/v2/sync?deviceId=12345",
  headers : {authToken : "gfUH43trfdkjg34"},
  dataType : "json",
  success : function(data) {
    console.log(data);
  },
</pre></td></tr>
</table>

<table>
<tr><td></td><td><pre>
  error : function(data) {
    console.log(data);
  }
});
</pre></td></tr>
<tr><td><b>Notes</b></td><td>The given <i>deviceId</i> must be a token formerly given to “
Register a Device”.
The <i>time</i> parameter defines the start point from which to get changed data. It is represented by the number of milliseconds since the standard base time known as "the epoch", namely January 1, 1970, 00:00:00 GMT. If the time parameter is omitted or it is set to a time after the last sync-time of the specified device, that last sync-time is used instead.</td></tr>
</table>

Table 88 Synchronize offline data

2.5.3.2.2 Wipe-out a User Account

Title	Wipe-out data of a particular user account		
URL	/v2/admin/users/:id?cmd=wipeAccount		
Header attributes	**Attribute**	**Required?**	**Example**
	authToken	Yes	authToken=gfUH43trfdkjg34
Method	POST		
URL parameters	**Parameter**	**Required?**	**Example**
	id : <integer>	yes	/23
Data	{} or null		
Success response	Code: 200 OK		
	{"code" : 200, "devmsg" : "success"}		
Error response	Code: 400 BAD REQUEST Code: 401 UNAUTHORIZED Code: 403 FORBIDDEN		
	{"code" : <integer>, "error" : <integer>, "devmsg" : <alphanumeric>}		

Code example	$.ajax({
	  type : "POST",
	  url :"/v2/admin/users/23?cmd=wipeAccount",
	  headers : {authToken : "gfUH43trfdkjg34"},
	  dataType : "json",
	  contentType : "application/json",
	  data : JSON.stringify({}),
	  success : function(data) {
	    console.log(data);
	  },
	  error : function(data) {
	    console.log(data);
	  }
	});		

Notes	none		

Table 89 User account wipe-out

2.5.3.2.3 Read a List of Contact Groups From an External System

Title	Read contact groups defined in an external system		
URL	/v2/import?cmd=getContactGroups		
Header attributes	**Attribute**	**Required?**	**Example**
	`authToken`	Yes	`authToken=gfUH43trfdkjg34`
Method	POST		
URL parameters	none		

Data	`{`
	`  "externalSystem" : "google",`
	`  "username" : <alphanumeric>,`
	`  "password" : <alphanumeric>`
	`}`		

Success response	Code: 200 OK		

	`[{`
	`  "externalId" : <alphanumeric>,`
	`  "name" : <alphanumeric>`
	`}]`		

Error response	Code: 401 UNAUTHORIZED Code: 403 FORBIDDEN Code: 503 SERVICE UNAVAILABLE		
	`{"code" : <integer>, "error" : <integer>, "devmsg" : <alphanumeric>}`		

Code example	`$.ajax({`
	`  type : "POST",`
	`  url :"/v2/import?cmd=getContactGroups",`
	`  headers : {authToken : "gfUH43trfdkjg34"},`
	`  dataType : "json",`
	`  contentType : "application/json",`
	`  data : JSON.stringify({`
	`    "externalSystem" : "google",`
	`    "username" : "john.doe@googlemail.com"`
	`    "password" : "pa55w0rd"`
	`  }),`
	`  success : function(data) {`
	`    console.log(data);`
	`  },`
	`  error : function(data) {`
	`    console.log(data);`
	`  }`
	`});`		

Notes	Ids of returned data can be passed as query parameters for " Import Person Contacts". For now the only supported external system is google (gmail).		

Table 90 External contact groups request

2.5.3.2.4 Import Person Contacts

Title	Import contacts from an external system		
URL	/v2/import?cmd=importContacts&nameRequired=:reqN&phoneRequired=:reqP&emailRequired=:reqE&updated-min=:time&groupIds=:ids		
Header attributes	**Attribute**	**Required?**	**Example**
	authToken	Yes	authToken=gfUH43trfdkjg34
Method	POST		
URL parameters	**Parameter**	**Required?**	**Example**
	reqN : <boolean>	no	&nameRequired=1
	reqP : <boolean>	no	&phoneRequired=0
	reqE : <boolean>	no	&emailRequired=0
	time : <datetime>	no	&updated-min=2012-12-31T00:00:00
	ids : [<alphanumeric>]	no	&groupIds=dfdg4f,6get54,none

Data

```
{
  "externalSystem" : "google",
  "username" : <alphanumeric>,
  "password" : <alphanumeric>
}
```

Success response

Code: 200 OK

```
{
  "externalSystem" : "google",
  "externalSystemAccount" : <alphanumeric>,
  "importTime" : <integer>,
  "failed" : [<integer>],
  "created" : [<integer>],
  "updated" : [<integer>],
  "deleted" : [<integer>]
}
```

Error response

Code: 401 UNAUTHORIZED

Code: 403 FORBIDDEN

Code: 503 SERVICE UNAVAILABLE

```
{"code" : <integer>, "error" : <integer>, "devmsg" : <alphanumeric>}
```

Code example

```
$.ajax({
  type : "POST",
  url : "/v2/import?cmd=importContacts&nameRequired=1&groupIds=none",
  headers : {authToken : "gfUH43trfdkjg34"},
  dataType : "json",
  contentType : "application/json",
  data : JSON.stringify({
    "externalSystem" : "google",
    "username" : "john.doe@googlemail.com"
    "password" : "pa55w0rd"
  }),
  success : function(data) {
    console.log(data);
  },
  error : function(data) {
    console.log(data);
```

	`  }`
	`});`

Notes	Data returned by the failed array is not stored as contact data in the database. User is supposed to resolve these data and call create/update/merge later. As group ids you can pass values earlier retrieved by Synchronize Offline Data. In addition you can pass none as groupId to import contacts without any group membership in the external system. If the updated-min parameter is set, only contacts created or updated after the specified time (in UTC datetime format yyyy-MM-dd'T'HH:mm:ss.SSS'Z') will be downloaded. Note that if updated-min is not set, the default value is the last download time. In this case only contacts created or changed after the last download will be downloaded. For now the only supported external system is google (gmail).

Table 91 Person contacts import

2.5.3.2.5 Find Duplicates of Basicinfos

Title	Find basicinfos, phone numbers especially, which are assigned to multiple contacts.		
URL	/v2/cleanup?cmd=getDuplicates		
Header attributes	**Attribute**	**Required?**	**Example**
	`authToken`	Yes	`authToken=gfUH43trfdkjg34`
Method	`GET`		
URL parameters	none		
Data	none		
Success response	Code: 200 OK		
	`[<alphanumeric>]`		
Error response	Code: 401 UNAUTHORIZED Code: 403 FORBIDDEN		
	`{"code" : <integer>, "error" : <integer>, "devmsg" : <alphanumeric>}`		

Code example	`$.ajax({`
	`  type : "GET",`
	`  url : "/v2/cleanup?cmd=getDuplicates",`
	`  headers : {authToken : "gfUH43trfdkjg34"},`
	`  dataType : "json",`
	`  success : function(data) {`
	`    console.log(data);`
	`  },`
	`  error : function(data) {`
	`    console.log(data);`
	`  }`
	`});`		

Notes	Data returned contains all values which are assigned multiple times. The values may be assigned to actual contacts but also to failed imports. The values retrieved can passed to " Get Conflicted Contacts".		

Table 92 Find basicinfo duplicates

2.5.3.2.6 Get Conflicted Contacts

Title	Return contacts having the same basicinfo value assigned.		
URL	/v2/cleanup?cmd=getConflicts&value=:value		
Header attributes	**Attribute**	**Required?**	**Example**
	`authToken`	Yes	`authToken=gfUH43trfdkjg34`
Method	`GET`		
URL parameters	**Parameter**	**Required?**	**Example**
	`value : <alphanumeric>`	yes	`&value=+12025552846`
Data	none		
Success response	Code: 200 OK		

	`{`
	`  "data" : [<object>]`
	`}`		

Error response	Code: 401 UNAUTHORIZED Code: 403 FORBIDDEN		
	`{"code" : <integer>, "error" : <integer>, "devmsg" : <alphanumeric>}`		

Code example	`$.ajax({`
	`  type : "GET",`
	`  url : "/v2/cleanup?cmd=getConflicts&value=12025552846",`
	`  headers : {authToken : "gfUH43trfdkjg34"},`
	`  dataType : "json",`
	`  success : function(data) {`
	`    console.log(data);`
	`  },`
	`  error : function(data) {`
	`    console.log(data);`
	`  }`
	`});`		

Notes	Data returned is an array of contact objects using *projection=base* (as described in “ Read a Person Contact”, ”Read an Organization Contact” resp.) The user is supposed to merge the contact data manually. Then the client should pass the result to “ Merge Contact Data”.		

Table 93 Find conflicted contacts

2.5.3.2.7 Merge Contact Data

<table>
<tr><td><b>Title</b></td><td colspan="3">Merge contact data from different contacts into one contact.</td></tr>
<tr><td><b>URL</b></td><td colspan="3">/v2/contacts?cmd=merge</td></tr>
<tr><td rowspan="2"><b>Header attributes</b></td><td><b>Attribute</b></td><td><b>Required?</b></td><td><b>Example</b></td></tr>
<tr><td>authToken</td><td>Yes</td><td>authToken=gfUH43trfdkjg34</td></tr>
<tr><td><b>Method</b></td><td colspan="3">POST</td></tr>
<tr><td><b>URL parameters</b></td><td colspan="3">none</td></tr>
<tr><td><b>Data</b></td><td colspan="3"><pre>
{
  "contactIds" : [<integer>],
  "importContactIds" : [<integer>],
  "contact" : <contact>
}
</pre></td></tr>
<tr><td rowspan="2"><b>Success response</b></td><td colspan="3">Code: 200 OK</td></tr>
<tr><td colspan="3">{"code" : 200, "entity" : <contact>, "devmsg" : "success"}</td></tr>
<tr><td rowspan="2"><b>Error response</b></td><td colspan="3">Code: 400 BAD REQUEST
Code: 401 UNAUTHORIZED
Code: 403 FORBIDDEN</td></tr>
<tr><td colspan="3">{"code" : <integer>, "error" : <integer>, "devmsg" : <alphanumeric>}</td></tr>
<tr><td><b>Code example</b></td><td colspan="3"><pre>
$.ajax({
  type : "POST",
  url : "/v2/contacts?cmd=merge",
  headers : {authToken : "gfUH43trfdkjg34"},
  dataType : "json",
  contentType : "application/json",
  data : JSON.stringify({
    "contactIds" : [12],
    "importContactIds" : [11,23,32],
    "contact" : {
      "id" : 52,
      "version" : 4,
      "firstName" : "Remo",
      "basicInfos" : [
        { "id" : 46 },{ "id" : "23" }, { "id" : "142" }
      ]
    }
  }),
  success : function(data) {
    console.log(data);
  },
  error : function(data) {
    console.log(data);
  }
});
</pre></td></tr>
<tr><td><b>Notes</b></td><td colspan="3">All related entities (deals, interactions, notes, tasks) of the former separated contacts will be attached to the new merged contact.
Contacts represented by the <i>contactIds</i> attribute will be deleted, imported contacts (which are non-active contacts) represented by the <i>importContactsIds</i> will be deleted as well. Those two attributes are supposed to reflect the contact data which has been merged into the new/updated contact represented by the <i>contact</i> attribute.</td></tr>
</table>

Table 94 Merge contacts

2.5.3.2.8 Share Contact With Group(s)

<table>
<tr><td><b>Title</b></td><td colspan="3">Share contact data with one or multiple groups</td></tr>
<tr><td><b>URL</b></td><td colspan="3">/v2/contacts/:id?cmd=share&groups=:groups</td></tr>
<tr><td rowspan="2"><b>Header attributes</b></td><td><b>Attribute</b></td><td><b>Required?</b></td><td><b>Example</b></td></tr>
<tr><td>authToken</td><td>Yes</td><td>authToken=gfUH43trfdkjg34</td></tr>
<tr><td><b>Method</b></td><td colspan="3">POST</td></tr>
<tr><td rowspan="2"><b>URL parameters</b></td><td><b>Parameter</b></td><td><b>Required?</b></td><td><b>Example</b></td></tr>
<tr><td>id : <integer>
groups : [<integer>]</td><td>yes
yes</td><td>/31
&groups=12,32</td></tr>
<tr><td><b>Data</b></td><td colspan="3">{} or null</td></tr>
<tr><td rowspan="2"><b>Success response</b></td><td colspan="3">Code: 200 OK</td></tr>
<tr><td colspan="3">{"code" : 200, "entity" : <contact>, "devmsg" : "success"}</td></tr>
<tr><td rowspan="2"><b>Error response</b></td><td colspan="3">Code: 400 BAD REQUEST
Code: 401 UNAUTHORIZED
Code: 403 FORBIDDEN</td></tr>
<tr><td colspan="3">{"code" : <integer>, "error" : <integer>, "devmsg" : <alphanumeric>}</td></tr>
<tr><td><b>Code example</b></td><td colspan="3"><pre>
$.ajax({
  type : "POST",
  url : "/v2/contacts/31?cmd=share&groups=12,32",
  headers : {authToken : "gfUH43trfdkjg34"},
  dataType : "json",
  data : JSON.stringify({}),
  success : function(data) {
    console.log(data);
  },
  error : function(data) {
    console.log(data);
  }
});
</pre></td></tr>
<tr><td><b>Notes</b></td><td colspan="3">Sharing means that each user – even the sharing user himself – receives a shallow-copy of the shared contact. That's why the <b>returned contact will have a different id</b>.
To share with all members of the company they have to be put into a group to share the contact with.</td></tr>
</table>

Table 95 Share a contact

2.5.3.2.9 Assign User Roles

Title	Assign roles to a user.		
URL	/v2/admin/users/:id/roles?cmd=:cmd		
Header attributes	**Attribute**	**Required?**	**Example**
	`authToken`	Yes	`authToken=gfUH43trfdkjg34`
Method	POST		
URL parameters	**Parameter**	**Required?**	**Example**

	`id : <integer>`
	`cmd : <enum(add,remove,set)>`
		yes
		yes
			`/31`
			`?cmd=set`

Data	`[<alphanumeric>]`		
Success response	Code: 200 OK		

	`{`
	`  "active" :  <boolean>,`
	`  "companyId" : <integer>,`
	`  "dateCreated" : <integer>,`
	`  "email" : <alphanumeric>,`
	`  "groups" :`
	`    [{`
	`      "id" : <integer>,`
	`      "name" : <alphanumeric>`
	`    }],`
	`  "id" : <integer>,`
	`  "interactionsCount" : <integer>,`
	`  "roles" : [<alphanumeric>]`
	`}`		

Error response	Code: 400 BAD REQUEST Code: 401 UNAUTHORIZED Code: 403 FORBIDDEN		
	`{"code" : <integer>, "error" : <integer>, "devmsg" : <alphanumeric>}`		

Code example	`$.ajax({`
	`  type : "POST",`
	`  url : "/v2/admin/users/31/roles?cmd=set",`
	`  headers : {authToken : "gfUH43trfdkjg34"},`
	`  dataType : "json",`
	`  contentType : "application/json",`
	`  data : JSON.stringify([`
	`    "interact_admin", "customer_admin", "user"`
	`  ]),`
	`  success : function(data) {`
	`    console.log(data);`
	`  },`
	`  error : function(data) {`
	`    console.log(data);`
	`  }`
	`});`		

Notes	The payload data of the request is an array of role names.		

Table 96 Assign roles to user

2.5.3.2.10 Assign User Groups

<table>
<tr><td><b>Title</b></td><td colspan="3">Assign groups to a user.</td></tr>
<tr><td><b>URL</b></td><td colspan="3">/v2/admin/users/:id/groups?cmd=:cmd</td></tr>
<tr><td rowspan="2"><b>Header attributes</b></td><td><b>Attribute</b></td><td><b>Required?</b></td><td><b>Example</b></td></tr>
<tr><td>authToken</td><td>Yes</td><td>authToken=gfUH43trfdkjg34</td></tr>
<tr><td><b>Method</b></td><td colspan="3">POST</td></tr>
<tr><td rowspan="2"><b>URL parameters</b></td><td><b>Parameter</b></td><td><b>Required?</b></td><td><b>Example</b></td></tr>
<tr><td>id : &lt;integer>
cmd : <enum(add,remove,set)&gt;</td><td>yes
yes</td><td>/31
?cmd=set</td></tr>
<tr><td><b>Data</b></td><td colspan="3">[&lt;integer&gt;]</td></tr>
<tr><td rowspan="2"><b>Success response</b></td><td colspan="3">Code: 200 OK</td></tr>
<tr><td colspan="3"><pre>
{
  "active" : <boolean>,
  "companyId" : <integer>,
  "dateCreated" : <integer>,
  "email" : <alphanumeric>,
  "groups" :
    [{
      "id" : <integer>,
      "name" : <alphanumeric>
    }],
  "id" : <integer>,
  "interactionsCount" : <integer>,
  "roles" : [<alphanumeric>]
}
</pre></td></tr>
<tr><td rowspan="2"><b>Error response</b></td><td colspan="3">Code: 400 BAD REQUEST
Code: 401 UNAUTHORIZED
Code: 403 FORBIDDEN</td></tr>
<tr><td colspan="3">{"code" : <integer>, "error" : <integer>, "devmsg" : <alphanumeric&gt;}</td></tr>
<tr><td><b>Code example</b></td><td colspan="3"><pre>
$.ajax({
  type : "POST",
  url : "/v2/admin/users/31/groups?cmd=add",
  headers : {authToken : "gfUH43trfdkjg34"},
  dataType : "json",
  contentType : "application/json",
  data : JSON.stringify([
    12,453,28,199
  ]),
  success : function(data) {
    console.log(data);
  },
  error : function(data) {
    console.log(data);
  }
});
</pre></td></tr>
<tr><td><b>Notes</b></td><td colspan="3">The payload data of the request is an array of group ids.</td></tr>
</table>

Table 97 Assign groups to user

2.5.3.2.11 Reset User Password

Title	Reset the password of a user.
URL	/v2/myAccount?cmd=resetPwd
Header Attributes	none
Method	POST
URL parameters	none

Data	`{`
	`  "email" : <alphanumeric>`
	`}`

Success response	Code: 200 OK
	`{ "code" : 200, "devmsg" : "success" }`
Error response	Code: 400 BAD REQUEST
	`{ "code" : <integer>, "error" : <integer>, "devmsg" : <alphanumeric> }`

Code example	`$.ajax({`
	`  type : "POST",`
	`  url : "/v2/myAccount?cmd=resetPwd",`
	`  dataType : "json",`
	`  contentType : "application/json",`
	`  data : JSON.stringify({`
	`    "email" : "test@interact.io"`
	`  }),`
	`  success : function(data) {`
	`    console.log(data);`
	`  },`
	`  error : function(data) {`
	`    console.log(data);`
	`  }`
	`});`

Notes	none

Table 98 Reset the password of a user

2.5.3.2.12 Change a User password

Title	Change the password of a suser.		
URL	/v2/myAccount?cmd=changePwd		
Header attributes	**Attribute**	**Required?**	**Example**
	authToken	Yes	authToken=gfUH43trfdkjg34
Method	POST		
URL parameters	none		

Data	{
	  "oldPassword" : <alphanumeric>,
	  "newPassword" : <alphanumeric>
	}		

Success response	Code: 200 OK		
	{"code" : 200, "devmsg" : "success"}		
Error response	Code: 400 BAD REQUEST		
	{"code" : <integer>, "error" : <integer>, "devmsg" : <alphanumeric>}		

Code example	$.ajax({
	  type : "POST",
	  url : "/v2/myAccount?cmd=changePwd",
	  dataType : "json",
	  contentType : "application/json",
	  data : JSON.stringify({
	    "oldPassword" : "bad",
	    "newPassword" : "$aFeP4Ssw0rD!",
	   }),
	  success : function(data) {
	    console.log(data);
	  },
	  error : function(data) {
	    console.log(data);
	  }
	});		

Notes	none		

Table 99 Change the password of a user

2.6 Error codes

Besides the http failure code, error results of an API call always return a dedicated 14-digit error code for better error handling. The first quad (4 digits) represents the exception type. The second quad (3 digits) represents the action which caused the exception. The third quad represents the realm in which the action was (or was supposed to be) executed. The forth quad (4 digits) is used for further differentiation required by the Server. The following lists show possible error codes and their according meaning:

Error Code (Exception)				Description
9000	xxx	xxx	xxxx	General failure
9001	xxx	xxx	xxxx	Internal Server failure
9002	xxx	xxx	xxxx	External failure or Java core failure
9100	xxx	xxx	xxxx	Indicates a requested entity was not found
9101	xxx	xxx	xxxx	Indicates a requested address was not found
9102	xxx	xxx	xxxx	Indicates a requested basicinfo was not found
9103	xxx	xxx	xxxx	Indicates a requested company was not found
9104	xxx	xxx	xxxx	Indicates a requested contact was not found
9105	xxx	xxx	xxxx	Indicates a requested organization contact was not found
9106	xxx	xxx	xxxx	Indicates a requested person contact was not found
9107	xxx	xxx	xxxx	Indicates a requested deal was not found
9108	xxx	xxx	xxxx	Indicates a requested device was not found
9109	xxx	xxx	xxxx	Indicates a requested group was not found
9110	xxx	xxx	xxxx	Indicates a requested interaction was not found
9111	xxx	xxx	xxxx	Indicates a requested note was not found
9112	xxx	xxx	xxxx	Indicates a requested social profile or instant messenger profile was not found
9113	xxx	xxx	xxxx	Indicates a requested role was not found
9114	xxx	xxx	xxxx	Indicates a requested tag was not found
9115	xxx	xxx	xxxx	Indicates a requested task was not found
9116	xxx	xxx	xxxx	Indicates a requested tasklist was not found
9117	xxx	xxx	xxxx	Indicates a requested user was not found
9118	xxx	xxx	xxxx	Indicates a requested basicinfo variation was not found
9119	xxx	xxx	xxxx	Indicates a requested contact picture was not found
9200	xxx	xxx	xxxx	Data validation failure
9201	xxx	xxx	xxxx	Indicates invalid request data
9202	xxx	xxx	xxxx	Indicates missing mandatory request data
9300	xxx	xxx	xxxx	Bad request data
9301	xxx	xxx	xxxx	Indicates an attempt to assign one basicinfo to multiple contacts

9302	xxx	xxx	xxxx	Indicates an attempt to register a device which is already registered
9303	xxx	xxx	xxxx	Indicates an attempt to assign an email address to multiple users
9304	xxx	xxx	xxxx	Indicates an attempt to update outdated data
9305	xxx	xxx	xxxx	Indicates an attempt to touch a contact picture from an external system
9306	xxx	xxx	xxxx	Indicates a requested report does not exist
9401	xxx	xxx	xxxx	Authentication failure
9402	xxx	xxx	xxxx	Indicates requesting user has no permission to execute request
9999	xxx	xxx	xxxx	Unhandled external failure or Java core failure

Table 100 Error codes (first quad)

Error Code (Action)				Description
xxxx	010	xxx	xxxx	View
xxxx	011	xxx	xxxx	Edit, Update
xxxx	012	xxx	xxxx	Create
xxxx	013	xxx	xxxx	Merge
xxxx	014	xxx	xxxx	Delete, Remove
xxxx	015	xxx	xxxx	Import
xxxx	016	xxx	xxxx	Share
xxxx	017	xxx	xxxx	Pull Synchronization Data
xxxx	018	xxx	xxxx	Bulk Push Synchronization Data
xxxx	019	xxx	xxxx	Open Websocket
xxxx	020	xxx	xxxx	Login
xxxx	021	xxx	xxxx	Logout
xxxx	022	xxx	xxxx	Erase, Reset, Wipe-Out, Zero

Table 101 Error codes (second quad)

Error Code (Realm)				Description
xxxx	xxx	000	xxxx	Activities
xxxx	xxx	040	xxxx	Authentication
xxxx	xxx	050	xxxx	Data Synchronization
xxxx	xxx	060	xxxx	Websocket
xxxx	xxx	090	xxxx	Data Cleanup
xxxx	xxx	100	xxxx	Contacts
xxxx	xxx	103	xxxx	Contact Relations
xxxx	xxx	104	xxxx	Contact Deals
xxxx	xxx	105	xxxx	Contact Interactions
xxxx	xxx	106	xxxx	Contact Notes

xxxx	xxx	107	xxxx	Contact Profiles
xxxx	xxx	108	xxxx	Contact Searchterms
xxxx	xxx	109	xxxx	Contact Tags
xxxx	xxx	110	xxxx	Contact Tasks
xxxx	xxx	111	xxxx	Contact Import
xxxx	xxx	200	xxxx	Deals
xxxx	xxx	201	xxxx	Deal Notes
xxxx	xxx	202	xxxx	Deal Tasks
xxxx	xxx	300	xxxx	Tasks
xxxx	xxx	301	xxxx	Task Notes
xxxx	xxx	302	xxxx	Tasklists
xxxx	xxx	400	xxxx	Notes
xxxx	xxx	500	xxxx	Tags
xxxx	xxx	600	xxxx	Devices
xxxx	xxx	601	xxxx	Trigger Devices
xxxx	xxx	700	xxxx	Administration
xxxx	xxx	701	xxxx	Company Administration
xxxx	xxx	702	xxxx	Role Administration
xxxx	xxx	703	xxxx	User Administration
xxxx	xxx	704	xxxx	Group Administration
xxxx	xxx	705	xxxx	Custom field administration
xxxx	xxx	750	xxxx	My Account
xxxx	xxx	800	xxxx	Basicinfos
xxxx	xxx	900	xxxx	Interactions

Table 102 Error codes (third quad)

For your information here is the complete list of possible values for the forth quad:

1001
2001 2002 2003 2004 2005
3003
4003 4004
5001 5002 5003
6001 6002
7001 7002

Table 103 Error codes (forth quad)

3 Index of tables

4 Attachments

- Javadoc