berliCRM

REST Webservices API Reference Manual

Version 1.5.3
Rev. from 27.09.2018

berliCRM: API Reference Manual

Trade Marks

For more Information about crm-now you may look up:

www.crm-now.com

Manual ID: 532-001-002

Table of Contents

1 Overview

This documentation describes the REST based application programming interface (the API) used by the berliCRM system. It is recommended that you go through the tutorial, provided in a different document, for a better understanding of this documentation.

Use this REST API Reference to learn about available endpoints and operations for accessing, creating, updating, or deleting CRM resources by GET and POST requests over HTTP.

The REST API is organized around data object resources. A resource is a single and named object stored by the CRM, such as a Contact, an Account or Lead. Resources can be linked with other resources.

Any submitted operation requests to the API return a response, all requests and responses are JSON encoded.

2 Formats

The following describes data formats used for all REST operations.

2.1 URL Format

All REST based web service calls are https calls with the following format:

https://your_CRM_url/webservice.php?operation=[**operationType**]&**sessionName**=[session Name]&[**operationSpecificParameters**]

operationType: see chapter 3 for a complete list

sessionName: The key that is used to identify the current session and is unique for every session. This must be part of every request.

operationSpecificParameters: Parameters which depend on the operation type as explained in the following.

2.2 The Response Format

Each REST call will get a response from the CRM system. All responses of the CRM have the following format.

Successfully processed:

```
Response{
        success:Boolean=true
        result: Data Object
}
```

The resulting data object depends on the operation type.

If an error occurred while processing the request, the following message will be presented:

```
Response{
        success:Boolean=false
        error:ErrorObject
}

ErrorObject{
        code: <String: string representation of the error type>
        message: <String: error message from the API>
}
```

2.3 Data Object

A data object is a map representing the full contents of a CRM entity based object including references to other data objects. A special id format is used to identify data objects.

2.3.1 Id Format

All IDs used by a web service have the following format:

objectTypeId 'x' **objectId**

objectTypeId - Id of the object type. This is generated uniquely for each entity supported by the web service API. You can obtain the objectTypeId as idPrefix field information with a describe operation.

objectId - id of object in the CRM's database. This is a unique id for all objects of the given entity.

2.3.2 Map

A map is an associative array of key value pairs.

Reference fields are links to other resources which have their own data objects. All reference fields are represented by using an id with type information. For instance, the Account information of a Contact data object are represented by an Account id.

```
Contact Data Object
[
        [id] => 4x2712
        [salutationtype] =>
        [firstname] => Richard
        [contact_no] => CON12123
        [phone] => 03039001800
        [lastname] => Meier
        [mobile] => 0178123456
        [account_id] => 3x12455
        ...
]
```

In the (incomplete) example of a Contact data object with id 4x2712 the reference to the account_id 3x12455 contains the object type id 3, and divided by an x the unique internal object id 12455 for the Account information.

2.4 TimeStamp

A long representation of the number of seconds since unix epoch. See https://en.wikipedia.org/wiki/Unix_time

3 Operations

The following describes all REST operations in detail.

3.1 Get Challenge

Get a challenge token from the server. This operation must always be executed before any login operation.

getchallenge(username:String): GetChallengeResult
Request Type: GET
username: a CRM username

Required URL format:
https://your_CRM_url/webservice.php?operation=getchallenge&username=[username]

Result:
An json object representing the response of a getchallenge operation.

```
{"success":true,
"result":{
        "token":"<string: challenge token from the server>",
        "serverTime":<time stamp: the current server time>,
        "expireTime":<time stamp: the time when the token expires >
        }
}
```

3.2 Login

Login to the server using the challenge token obtained in get challenge operation.

login(username:String, accessKey:String):LoginResult
Request Type:POST
username: A CRM username.
accessKey: An md5 of the concatenation of the challenge token and the user's web service access key.

Required URL format:
https://your_CRM_url/webservice.php?operation=login&username=[username]&accessKey=[accesskey]

Result:
An object representing the response of a login operation.

```
{"success":true,
"result":{
        "sessionName":"<string: unique identifier for the session>",
        "userId":"<string: the CRM id for the logged in user>",
        "version":"<string: the version of the web services API>",
        "vtigerVersion":"<string: the CRM version>",
        "tagVersion":"<string: the CRM tag version>"
        }
}
```

3.3 List Types

List the names of all the CRM objects available through the API for a specific user and provided additional information.

listtypes():Nothing
Request Type: GET
Returns a map containing the key 'types' with the value, being a list of names of CRM objects.

Required URL format:
https://your_CRM_url/webservice.php?operation=listtypes&sessionName=[session id]

Result:
An object representing the response of a listtypes operation.

```
{"success":true,
"result":{
        "types":[<array: list of CRM objects reachable>],
        "information":{
                "<object name>":{
                        "isEntity":<Boolean: marks entity modules>,
                        "label":"<string: object label>",
                        "singular":"<string: singular object name>"
                        }
                }
                ...
        }
}
```

3.4 Describe

Provides the type information about a given CRM object.

describe(elementType: String): object
Request Type: GET
elementType: The type name of the CRM object to describe.

Required URL format:
https://your_CRM_url/webservice.php?operation=describe&sessionName=[session id]&elementType=[elementType]

Result:
An object representing the response of a describe operation.

```
{"success":true,
"result":{
        "label":"<string: object label>",
        "name":"<string: internal object name>",
        "createable":<Boolean: set when create operation is permitted>,
        "updateable":<Boolean: set when update operation is permitted>,
        "deleteable":<Boolean: set when delete operation is permitted>,
        "retrieveable":<Boolean: set when retrieve operation is permitted>,
        "fields":[
                {"name":"<string: internal field name>",
```

```
                "label":"<string: field label in user's language>",
                "mandatory":<Boolean: set when field is mandatory>,
                "type":{
                        "name":<string: defines field type>",
                },
        "nullable":<Boolean: set when field can get set to NULL>,
        "editable":<Boolean: set when edit operation is permitted>,
        "fieldsize":"<string: internal field size>",
        "default":"<string: default value for create operation>",
        },
        ...
}
```

For a list of supported field types, see chapter 5.

3.5 CRUD Operations

These operations are able to **c**reate, **r**etrieve, **u**pdate or **d**elete CRM data objects.

3.5.1 Create

Create a new data entry at the CRM server.

create(element:Map, elementType:String): Data Object
Request Type: POST
element: Fields of the object to populate. Values for mandatory fields must be provided.
elementType: the object type

Required URL format:
https://your_CRM_url/webservice.php?operation=create&sessionName=[session id]&element=[object]&elementType=[object type]

Result:
A data object representing the created data object.

3.5.2 Retrieve

Retrieve an existing entry from the server.

retrieve(id: Id): Data Object
Request Type: GET
id: The Id of the object.

Required URL format:
https://your_CRM_url/webservice.php?operation=retrieve&session_name=[session id]&id=[object id]

Result:
A data object representing the retrieved data object.

3.5.3 Update

Update an existing entry on the CRM object.

update(object: Data Object): Data Object
Request Type: POST
object: The Data Object to update.

Required URL format:
https://your_CRM_url/webservice.php?operation=update&sessionName=[session id]&element=[object]

Result:
A data object representing the updated data object.

3.5.4 Delete

Delete an entry from the CRM system's GUI. That moves it to the recycling bin.

delete(id:Id):Nothing
Request Type: POST
id: The Id of the object to be deleted.
response: A map with one key status with value 'successful'

Required URL format:
https://your_CRM_url/webservice.php?operation=delete&sessionName=[session id]&id=[object id]

Result:
A map with one key status with value 'successful'.

```
{"success":true,
"result":{
        "message":"successfull"
        }
}
```

3.6 Revise

The revise operation updates individual fields of a data object. Compared to an update operation you send only the fields which needs to get changed.

However, the revise operation ignores unknown fields silently. A field could be unknown to a certain user due to lack of permissions or by a wrong field name.

update(object: Data Object): Data Object
Request Type: POST
object: The Data Object to update, containing the fields to update and the record id.

Required URL format:
https://your_CRM_url/webservice.php?operation=revise&sessionName=[session id]&element=[object]

Result:
A data object representing the updated data object.

3.7 Query

You can use the query operation to select data from CRM objects. The syntax is similar to SQL operations but limited to certain operations and independent from the CRM's data tables.

query(queryString : String): [Data Object]
Request Type: GET
queryString:The query to process.

Queries are currently limited to a single object. All query operations are currently supported for entity modules only.

Joins are not supported.

Query always limits its output to 100 records. If you want to obtain more than 100 records you need to loop with offset and limit.

The query format:

select * | <column_list> | <count(*)>
from <object> [where <conditionals>]
[order by <column_list>] [limit [<m>,]<n>];
The column list in the 'order by' clause can have at most two column names.

- column_list: comma separated list of field names.
- object: type name of the object.
- conditionals: condition operations or in clauses or like clauses separated by 'and' or 'or' operators these are processed from left to right. There is no grouping supported by bracket operators.
- conditional operators: <, >, <=, >=, =, !=.
- in clauses: in ().
- like clauses: like 'sqlregex'.
- value list: a comma separated list of values.
- m, n: integer values to specify the offset and limit respectively.

Required URL format:
https://your_CRM_url/webservice.php?operation=query&sessionName=[session id]&query=[query string]

Result:
A list of maps containing the fields selected fields.

3.8 Special Document operations

3.8.1 File Upload by extended create

To upload a file for a document you can use the create operation as described above withe an additional parameter. It requires that the file location type of the document's data object is set to Internal.

Create a new data entry at the CRM server.

create(element:Map, elementType:String, filename:String): Data Object
Request Type: POST
element: Fields of the object to populate. Values for mandatory fields must be provided.
elementType: the object type
filename: the file's type, path and name

Required URL format:
https://your_CRM_url/webservice.php?operation=create&sessionName=[session id]&element=[object]&elementType=[object type]& filename=[file information]

Result:
A data object representing the created data object.

3.8.2 File Download by retrievedocattachment

The retrievedocattachment operation retrieves a file attached to a document. This complements the retrieve operation for a document which only provides the document's master data.

retrievedocattachment (id: Id, returnfile: Boolean): Data Object
Request Type: GET
id: The Id of the document object.

Required URL format:
http://your_CRM_url/webservice.php?operation=retrievedocattachment&sessionName=[session id]&id=[document id]&returnfile=[true|false]

Result:
Returns the document file as a base64 encoded string if returnfile set to true. Returns the document file information only if returnfile is set to false.

3.9 Special CWC Operations

There are special document operations available if the CWC extension module is installed and active.

3.9.1 Get Document relations

The get_document_relations operation retrieves the ids from objects related to a document. This complements the retrieve operation for a document which only provides the document's master data.

get_document_relations (docids: Map): Data Object
Request Type: GET
docids: a json coded array of document object ids.

Required URL format:

http://your_CRM_url/webservice.php?operation=get_document_relations&sessionName=[session id]& docids=[document ids map]

Result:
Returns a map with key "found" of related entity ids.

3.9.2 Update Document relations

The update_document_relations operation sets a relationship between documents and other CRM entries. This complements the create operation for a document which only creates the document's master data.

update_document_relations (docid: String, relids: String, preserve: Boolean): Data Object
Request Type: POST
docid: is the document object id
relids: is a string of IDs object of objects to be related
preserve: decides whether existing document relations should be preserved

Required URL format:
http://your_CRM_url/webservice.php?operation=update_document_relations&sessionName=[session id]&docis=[document id]&relids=[string with object IDs]&preserve[true|false]

Result:
Returns a map with key "relids" of related entity ids.

3.9.3 Campaign operation

The get_campaign_entities operation provides the Contact, Leads and Account IDs related to a campaign object. It complements the retrieve operation for a Campaign which only returns the master data.

get_campaign_entities (campaignid: String, returnresults: String): Data Object
Request Type: GET
campaignid: is the Campaign object id
returnresults: empty or "Accounts", "Leads", "Contacts", if empty it returns all related IDs

Required URL format:
http://your_CRM_url/webservice.php?operation=get_campaign_entities&sessionName=[session id]& campaignid=[campaign id]& returnresults =[string module name]

Result:
Returns a map with comma separated strings of related IDs with module name as key.

3.10 Product operation

The update_product_relations operation provides a list of entitiy IDs related to a product object. It complements the retrieve operation for a Product which only returns the master data.

update_product_relations (productid: String, relids: String): Data Object
Request Type: POST
productid: is the Product object id
relids: a json coded array of entity ids.

Required URL format:
http://your_CRM_url/webservice.php?operation=update_product_relations&sessionName=[session id]&productid =[campaign id]& relids =[json encoded string of IDs array]

Result:
Returns a map with comma separated strings of related IDs with module name as key.

3.11 Related List operation

Most of the CRM relations between two entities are 1:n or n:1 and can get retrieved by a retrieve operation. In addition there are n:n relations, like the relation between Contacts and Products. The get_multi_relations operation covers this case and provides a list of related entitiy IDs for n:n relations.

get_multi_relations (id: String): Data Object
Request Type: GET
id: is the object id

Required URL format:
http://your_CRM_url/webservice.php?operation=get_multi_relations&sessionName=[session id]&id =[entity id]

Result:
Returns a map with comma separated strings of related IDs with module name as key.

3.12 Sync

Sync will return a map containing details of changes made to CRM objects after the modifiedTime.

sync(modifiedTime: Timestamp, elementType: String):SyncResult
Request Type: GET
modifiedTime: The time of the last synced modification.
elementType: This is an optional parameter, if specified the changes for that module after the given time otherwise changes to all user accessible module are returned.

Required URL format:
https://your_CRM_url/webservice.php?operation=sync&sessionName=[session id]&modifiedTime=[timestamp]&elementType=[elementType]

Result:
An object representing the response of a Sync operation.

SyncResult{
updated:[Object] //List of Objects created or modified.
deleted:[Id] //List of IDs of objects deleted.
lastModifiedTime:Timstamp //time of the latest change. This can be used in the next call to the Sync api to get all the latest changes that the client hasn't obtained.
}

3.13 Convert Lead

The convertlead operation provides a lead conversion. It creates a Potential, an Account or a Contacts by demand, transfers lead information to these entities and relates these entities to eachother.

The convertlead (leadid: String, element: Object): Data Object
Request Type: POST
leadid: is the Lead object id which should get converted
element: Fields of the objects to create. Values for mandatory fields must be provided.

Required URL format:
http://your_CRM_url/webservice.php?operation= convertlead&sessionName=[session id]& element =[Map]

Result:
Returns a map with a list of new IDs for Potentials and related Account and Contact entities.

3.14 Change User's Password

The changePassword operation sets a new password for a particular user.

The changePassword (id: String, oldPassword: String, newPassword: String, confirmPassword: String): Data Object
Request Type: POST
id: user object id
oldPassword: current user password
newPassword: new user password

confirmPassword: new user password

Required URL format:
http://your_CRM_url/webservice.php?operation= changePassword &sessionName=[session id]&id =[Object id] &oldPassword =[String]& newPassword =[String]& confirmPassword =[String]

Result:
Returns a map with success message.

3.15 Logout

A logout kills the current web services session. This leaves the web service session id invalid for further use. This should be used before a new login.

logout(): Nothing
Request Type: POST

Required URL format:
https://your_CRM_url/webservice.php?operation=logout&sessionName=[session id]

Result:
A map with one key status with value 'successful'.

```
{"success":true,
"result":{
        "message":"successfull"
        }
}
```

3.16 Extend Session

In the future this operation can be used to extend the valid time of a session id.

With a default PHP settings a session id is valid for app. 24 minutes. This time extends automatically with each operation for max. 24h. If you need to keep a session valid for more than 24h you can use the extendsession operation.

extendsession(username:String):LoginResult
Request Type: POST
username: A CRM username.

This has to be a POST request.

Required URL format:
https://your_CRM_url/webservice.php?operation=extendsession

4 CRM Entities

To get an actual List of possible CRM entities exposed by the API, to certain CRM user with a specific rule, run the listTypes operation.

5 Field Types

List of field types exposed by the API.

picklist

A field that can a hold one of a list of values, the map will contain two elements, picklistValues which is a list of possible values, and defaultValue which is the default value for the picklist.

Name	Description
picklistValues	Which is a list of possible values.
defaultValue	Specifies which value should be used as the default value for the picklist
name	Name of the field type.

reference

A field that shows a relation to another object, the field type map will contain another element called refersTo which is an array containing the name of modules of which the field can point to.

Name	**Description**
refersTo	Which is an array containing the name of modules to which the field can point to.
name	Name of the field type.

datetime

A string representing the date and time, the format is base on the user's settings date format.

date

A string representing a date, the field type map will contain another element called format which is the format in which the value of this field is expected, its based on the user's settings date format.

Name	**Description**
format	The format in which the value of this field is expected.
name	Name of the field type.

text

A multiline text field.

time

A string of the format hh:mm, format is based on the user's settings time format.

string

A one line text field.

boolean

A Boolean field, can have the values true or false.

integer

A none decimal number field.

owner

A field for defining the owner of a field which could be a group or individual user.

autogenerated

These are fields for which the values are generated automatically by the CRM, this is usually an object's id field.

email

A field for storing email ids.

phone

A field for storing phone numbers.

url

A field for storing URLs.

double

A field for floating point numbers.

file

A field for adding a file to the CRM.

Name	Description
maxUploadFileSize	Max allowed size of a file getting uploaded.
name	Name of the field type.

password

A field for storing passwords.

decimal

A field for floating point numbers.

skype

A field for storing skype ids or phone numbers.

multipicklist

A picklist field where multiple values are existent can be selected.

6 Known Issues

- Sync does not work on the users module and none entity modules like Currency, Groups etc.
- Query does not work on none-entity modules like Currency, Groups etc.

7 Acknowledgement

The very first API description was originally provided by vtiger™. With the friendly permission of vtiger™ it has been modified to meet the specific requirements of berliCRM.