LeadSuccess API for Microsoft Dynamics

The **LeadSuccess API for Microsoft Dynamics** provides access to server-side data structures and procedures via a RESTful API to enable 3rd party developers building customized Export interface from the LeadSuccess System to Microsoft Dynamics CRM system with standard web techniques and is covering the data created in the original LeadSuccess App in objects compatible to Microsoft Dynamics objects.

You can find more information on http://www.convey.de/ to learn about further components of the LeadSuccess system.

LeadSucces App

Introduction

The LeadSuccess App allows you to collect data and information about the visitors in an easy, fast and reliable way. It allows you to get the visitor´s data in various ways; in fact, it is possible to scan barcodes, QR-Codes, business card or enter information manually. In addition to this, it is also possible to fill out a customizable questionnaire, take notes and add pictures and sketches to every contact you collect.

Exhibitor Portal

Everything can be managed through the "Exhibitor Portal" which allows you to manage your App users, create your own event-specific questionnaire, check and edit leads and furthermore export your collected leads to different Excel formats.

You will need a valid LeadSuccess admin account to configure the event and app users. App user credentials configured in the "Exhibitor Portal" can be used either with the original LeadSuccess App or for requests with basic authentication to use the API.

You will find additional information about administration of LeadSuccess in the document "User Guide LeadSuccess Mobile – Online Portal"

LeadSuccess app structure

The LeadSuccess App user interface offers the following functional areas to the user:

- **Start Page:**
  Event name, user name, number of contacts, present state, new message flag
- **Capture Barcode**
  Create a new contact based on the visitor's badge barcode
- **Capture Business card**
  Create a new contact based on the visitor's business card photo and save the photo linked to the contact
- **Edit contact manually**
  Create a new contact or edit existing address fields of a selected visitor contact
- **Edit questionnaire**
  Offers a form to answer the questions of the questionnaire based on the questionnaire configuration (multiselect, single select, combo, rating, text notes, date picker, optional questions, mandatory questions). Each questionnaire is linked to a selected contact
- **Create notes**
  Create and edit sketches in SVG format, capture photos, capture audio messages. All notes are linked to a selected contact
- **Edit user state**
  Edit personal user data, toggle the present state, receive and send user messages and capture a user photo. The data is linked to the employee data of the app user and can be administrated in the Exhibitor Portal

Access protocol

To access this interface is used a subset of ODATA protocol. See ODATA org. ODATA server is based on SAP SQL Anywhere 17 OData implementation. Therefore currently only ODATA version 2.0 syntax can be used. See http://www.odata.org/documentation/odata-version-2-0

LeadSuccess API users will get the **<server-name>** and **<api-name>** to be used for your API-requests together with their login information from convey.

To access this API, HTTPS basic authentication is used. It means user and password has to be transmitted via HTTPS with each request. Since HTTP(S) is a stateless protocol, each request is computed in individual transactions. GET and POST requests return table data as result of the request in the specified format, e.g. JSON. If you use a XMLHTTPRequest alike JavaScript API, use JSON.parse(response.responseText) to transform the result in a JSON object that for automatic data-binding to UI elements within your preferred framework. Some requests will result in URLs to get further data. Due to technical limitations the provided address of these links is a local address and not your API address. Please replace that local address in these URLs to:

```
https://<server-name>/<api-name>/...
```

All parameter data must be transferred in JSON format to the ODATA server. For results the JSON format can be selected with the "$format=json" request option or with the HTTP header “Accept: application/json”, otherwise the server returns the results in XML format. All text data is UTF-8 encoded, except described otherwise for specific data fields. All date-time data is in UTC time zone.

Pay attention to the fact, that the OData request syntax is **case sensitive**!

A user can only keep one transaction active at a time. If several parallel transactions are to be supported, different users must be used for this!

The following request types are supported

Select data

Use requests of type: **GET** to select data from LeadSuccess relations. You should distinguish at least these different types of GET requests:

Select a list of rows
You can select lists of data from API objects with requests like:

GET https://<server-name>/<api-name>/LSA_<table-name>?$format=json

You can use the $filter=(<filter-list>) and $orderby=(<orderby-list>) options to specify restrictions and list order to the request.
Requests of this kind will return an array of rows, that could be handled like this:

```
function xhrSuccess(response) {
  var obj = JSON.parse(response.responseText);
  var results = obj && obj.d && obj.d.results;
  handleResults(results);
}
```

The result size maximum to be fetched by one request is limited to 100 rows. To fetch the next row-set, you can use the parameter $skiptoken(<primary-key-value>) in a following GET request to the same relation. To simplify that, you can use the following member in result JSON structure as described above:

```
obj.d.__next
```

Remember to replace the server path in the URL as described above, e.g.:

```
var getNextUrl = function (json) {
    var url = "";
    if (json && json.d) {
        var next = json.d.__next;
        if (next && typeof next === "string") {
            var viewNamePos = next.lastIndexOf("/");
            if (viewNamePos >= 0) {
                url = "https://<server-name>/<api_name>" +
                      next.substr(viewNamePos);
            }
        }
    }
    return url;
}
```

Fetch next list of rows

Use a nextURL given from earlier select request to fetch the next row-set. The returned result is similar to the one of the first select request, including results array and __next member for following row-set, if end of data isn't reached yet.

Select single row

You can use the primary key of each table to select a single row of data with the following request:

```
GET https://<server-name>/<api-name>/LSA_<table-name>(<primary-key-value>)?$format=json
```

Primary keys are always attributes of an integer type.

Requests of this kind will return one row, that could be handled like this:

```
function xhrSuccess(response) {
  var obj = JSON.parse(response.responseText);
  var result = obj && obj.d;
  handleResults(result);
}
```

Procedure call

Use requests of type: **GET** to call procedures. Parameters need to be placed URL-encoded within the URL.

```
var options = {
  type: "GET",
  user: <user-name>,
  password: <password>,
  url: "https://<server-name>/<api-name>/<procedure-name>?<parameters>& $format=json",
};
```

You can build the parameters list like this:

```
<paramaters> = <param1>=<value1>&<param2>=<value2>…;
```

CORS requirements

To support cross-site access-control requests, you should specify the `withCredentials` member of the `XMLHttpRequest` to `true`. You may need to add an option e.g. like this to add the value all your xhr requests, if supported by the `XMLHttpRequest` object:

```
var options = {
  //
  // other options depending from action, see above
  //
  //
  customRequestInitializer: function(req) {
    if (typeof req.withCredentials !== "undefined") {
      req.withCredentials = true;
    }
  }
};
```

Authorization header

To support server-side authorization propagation by some browser clients, like Google Chrome, you should add an “Authorization” header:

```
var options = {
  //
  // other options depending from action, see above
  //
  //
  headers: {
    "Authorization": "Basic " + btoa(<user> + ":" + <password>)
  }
};
```

Resources

Data categories

Based on the given function areas, the frontend needs to handle different data structures to offer the functionality of the LeadSuccess App. This data can be divided in:

- **Static data**
  to be loaded only once after app installation
- **Event data:**
  to be loaded at least once after app installation and updated after administrative changes in the LeadSuccess Exhibitor Portal
- **Application runtime data**
  to be loaded at least once after app installation and updated after interactive changes in the app or in the LeadSuccess Exhibitor Portal
- **Visitor data**
  to be created interactively in the app and to be retrieved via search and list panes

All data is organized in an object structure similar to Microsoft Dynamics objects. Object contain Microsoft Dynamics system fields like Id, CreatedDate and LastModifiedDate and object-specific fields. Each Object has a unique Id, that can be referenced as OwnerId field in another object. The LeadSuccess API for Microsoft Dynamics API offers views called WCE_<object-name> or WCE_<Procedure-name> to access the data.

Access rights

The LeadSuccess API for Microsoft Dynamics offers access rights on object-level to the provided views in order to their usage. For static application and runtime data only GET requests are offered. Accessing an object with an unsupported request option will return an exception.

The LeadSuccess API for Microsoft Dynamics offers access rights on row-level to the provided views. You can only access rows of data in the user context of the currently logged-in user.

Data schema

You can retrieve the full schema data with the following request:

```
GET https://<server-name>/<api-name>/$metadata
```

Static master data

WCE_Country

Select the entries of this view to receive s list of CountryCodes and Countries that can be referenced in LeadSuccess API for Microsoft Dynamics.

Name	Type	Description
LGNTINITLandVIEWID	Int32	Primary key value
CountryId	Single-Byte-String(2)	Two letter ISO code. Only single-byte characters are allowed in this field
DisplayName	String(255)	Display text for the country to select

You can use several reference columns to identify country selection for data export.

Application runtime data

Select this data to show information relevant for application runtime.

WCE_Event

Select the entries of this view to show information about the events referenced by collected data.

Name	Type	Description
VeranstaltungViewId	Int32	Primary key value.
EventId	UUID-String(36)	Universal Unique Identifier. Use this value as reference in objects referencing the **Event** object.
DisplayName	String(127)	Display text for the event shown in the app. The value can be edited in LeadSuccess portal
CreatedBy	Single-Byte-String(63)	Login name of the user who created the event object
CreatedOn	DateTime	Creation timestamp of the event object
ModifiedBy	Single-Byte-String(63)	Login name of the user who created the event object last recently
ModifiedOn	DateTime	Timestamp of last modification of the event object
StartDate	Date	Start date of event
EndDate	Date	End date of event
Type	String(255)	Name of event organizer, defaults to value “LeadSuccess”
EventSubtype	String(1000)	Title of the event, usually the name of a trade show
Description	String(255)	A description of the event. The value can be edited in the LeadSuccess portal

WCE_User

Select the entries of this view to show information about the users creating or modifying the collected data.

Name	Type	Description
MitarbeiterViewId	Int32	Primary key value
UserId	Single-Byte-String(63)	Login name of the user. Only single-byte characters are allowed in this field
DisplayName	String(320)	Display name of user
CreatedBy	Single-Byte-String(63)	Login name of the user who created the user object
CreatedOn	DateTime	Creation timestamp of the user object
ModifiedBy	Single-Byte-String(63)	Login name of the user who created the event object last recently
ModifiedOn	DateTime	Timestamp of last modification of the event object
FirstName	String(64)	First name of the user. Can be edited in LeadSuccess portal and app
LastName	String(255)	Last name of the user. Can be edited in LeadSuccess portal and app
Email	String(500)	Email address of the user. Can be edited in the LeadSuccess app
Phone	String(64)	Phone number of the user. Can be edited in the LeadSuccess app
MobilePhone	String(64)	Cellphone number of the user. Can be edited in the LeadSuccess app
Street	String(127)	Street address of the user. Can be edited in the LeadSuccess app
PostalCode	String(12)	Postal code / ZIP address of the user. Can be edited in the LeadSuccess app
City	String(127)	City address of the user. Can be edited in the LeadSuccess app
Country	String(255)	Country address of the user. Can be selected in the LeadSuccess app
CountryISOCode	String(2)	2-character ISO-code of the country address of the user.
Currentstatus	String(32)	Role of the user in LeadSuccess system, e.g. booth staff member or administrator
EventID	UUID-String(36)	Universal Unique Identifier. References the **Event** where the user is currently related to and is able to collect data for in the LeadSuccess app.

Visitor data

WCE_Lead

Select the entries of this view to retrieve visitor contact data collected or edited on LeadSuccess App Portal, Kiosk or Service devices or via LeadSuccess API.

Name	Type	Description
KontaktViewId	Int32	Primary key value
LeadId	UUID-String(36)	Universal Unique Identifier. Use this value as reference in objects referencing the **Lead** object.
CreatedBy	Single-Byte-String(63)	Reference to the **User** who created this **Lead**
CreatedOn	DateTime	Creation timestamp of the lead object.
ModifiedBy	Single-Byte-String(63)	Reference to the **User** who modified this **Lead** most recently
ModifiedOn	DateTime	Timestamp of last modification of the lead object
Salutation	String(32)	Salutation
Suffix	String(80)	Name suffix
FirstName	String(255)	First name
MiddleName	String(64)	Middle name
LastName	String(128)	Last name
CompanyName	String(1024)	Company name
JobTitle	String(80)	Job title
Address1_Telephone1	String(64)	Phone number
MobilePhone	String(64)	Cellphone number
Address1_Fax	String(64)	Fax number
EmailAddress1	String(500)	Email
WebsiteUrl	String(500)	Website
Address1_Line1	String(128)	Street address
Address1_PostalCode	String(12)	Postcode / ZIP of address
Address1_City	String(128)	Name of city
Address1_Country	String(255)	Name of country
Address1_CountryISOCode	Single-Byte-String(2)	Two letter ISO code. Only single-byte characters are allowed in this field
Address1_StateOrProvince	String(128)	Name of federal state
Description	String(4000)	User edited comments or automatically recognized other information that isn't related to the other fields
AttachmentIdList	LONG String: List of UUID-String(36)	Comma-separated list of UUID-Strings referencing **Attachment** objects related to the **Lead** object
SalesArea	String(4000)	User edited or automatically created text to describe the sales area where the lead might be related to. Sales area might be related to address regions via postal code or to specific questionnaire selection
RequestBarcode	String(1000)	Barcode identifier in case of the lead was collected via ticket barcode-scan and is delivered from trade show organizers visitor database
StatusMessage	String(255)	Error message from trade show organizers visitor database in case of failed lead retrieval by ticket barcode-scan

Name	Type	Description
DeviceId	Int32	Identifier of the local app database on the device where the lead was collected. Usually the Identifier of the users device. This number is shown in the LeadSuccess app and portal as the first number on “Contact list” or “Contact page”.
DeviceRecordId	Int32	Identifier of the lead on the local app database on the device where the lead was collected. This number is shown in the LeadSuccess app and portal as the second number on “Contact list” or “Contact page”.
ModifiedBySystemOn	DateTime	Timestamp of last modification of the lead object by automatic processing on the LeadSuccess server. Even if LastModifiedDate happened earlier on a device that had no online connection to the LeadSuccess server for a while, SystemModstamp will change after new or modified leads are being received from the LeadSuccess server. This value defaults to LastModifiedDate if no automatic processing occurred yet.
EventID	UUID-String(36)	UUID reference to the **Event** where the Lead was collected
Gender	String(32)	Gender selection from Salutation: Male, Female or null
Topic	String(127)	Event name
Newsletter	String(100)	Answer from questionnaire to first question containing the word ‘newsletter’
Department	String(100)	Single-Selection-answer from questionnaire to first question containing the word ‘department’
DisplayName	String (320)	Computed display name
IsReviewed	String(100)	Review status
DepartmentText	String (100)	Department description text

By adding a $filter query like the following to your GET request, you can do a call to query for only the newly created **Lead** objects that have been added since the last call or whether there are any **Lead** objects that have been processed on the server since a certain date-time because older data from devices has been received subsequently, because devices weren’t online all the time:

```
$filter=(ModifiedBysystemOn ge cast(‘2023-11-02T07:00:00Z’,’Edm.DateTime’))
```

given the certain date-time of 2. Nov. 2023 at 7:00 UTC when your last call happened. Remember to use greater-or-equal to get changes that happened in the same second of your previous call. You have to use the returned date-time in milliseconds to compare with previously received results.

Remember to call for data of all attachments in the returned AttachmentIdList and check the LastModifiedDate of each attachment in case of ModifiedBySystemOn has changed in the **Lead** object.

You can use the $orderby option for specific sort order of your result set:

```
$orderby=KontaktViewId
```

e.g. to order the result set by primary key value. Adding `desc` will order the results descending

Please remember to URL-encode spaces %20 and quotes %27 in your GET request.

WCE_AttachmentById

Call this procedure to select any Attachment related to a Lead object, like the PDF file of the questionnaire, the business card image, questionnaire-related or other attached photos, attached sketches or voice-notes. Attachments can be saved as document files of different file types specified via MIME type. Binary data of the file content is Base64 encoded.

Parameter	Type	Description
Id	UUID-String(36)	UUID of the attachment

The procedure will return **WCE_Attachment** data, like row select. Direct query string selection of **WCE_Attachment** isn't supported at the moment for performance reason.

WCE_Attachment

Name	Type	Description
AttachmentId	UUID-String(36)	Universal Unique Identifier. Use this value as reference in objects referencing the **Attachment** object.
CreatedBy	Single-Byte-String(63)	Reference to the **User** who created this **Attachment**
CreatedOn	DateTime	Creation timestamp of the attachment object.
ModifiedBy	Single-Byte-String(63)	Reference to the **User** who modified this **Attachment** most recently
ModifiedOn	DateTime	Timestamp of last modification of the attachment object
FileName	String(255)	File name of the attachment
Subject	String(4000)	Optional description of the attachment. Description is delivered for attachments containing data-protection policies, business card images or questionnaire-related photos.
MimeType	String(255)	MIME-type of the file data
Body	LONG String	Base64 encoded data of the file
FileSize	Int32	File size
EventId	UUID-String(36)	Reference to the **Lead** object where this **Attachment** object is related to

You can use the following GET-request to select a specific **Attachment** object with Id <attachment-id> from the AttachmentIdList of a **Lead** object.:

```
GET https://<server-name>/<api-name>/WCE_AttachmentById?Id=’<attachment-id>’&$format=json
```

Please remember to URL-encode quotes %27 in your GET request.

Error codes

For all requests, a result code and possibly a textual error message is returned.

The error message serves to describe the error for a developer as exactly as possible. Ideally, it should be saved in case of error in a log file. It is not intended to be displayed to an end user (exhibitor).

The result code is roughly based on the HTTP status codes and is intended as the basis for an automatic response to specific errors. The three-digit numeric error code can be followed by a detail error code with a dot.

The rough classification of the error is made possible by the first digit of the result code:

- "2" for success

- "4" for client-side errors, e.g. wrong or missing parameters

- "5" for server-side errors

Common error codes used by the API:

- "400" Bad Request: Incorrect request, e.g. missing mandatory parameter

- "404" Not Found: No matching record was found. If appropriate, the type of missing data record is specified as the detail error code:

- "500" Internal Server Error: There is a problem on the server, e.g. missing documents. Here a LeadSuccess administrator should be contacted. The plain text message helps to localize the problem!

Please note that in addition to the application error codes defined here, further error messages of the overlying protocol levels (ODATA, web server) can occur.

Version Overview:

Version Manual	**Version Database**	**Date**	**Changes**
0.1	8.2.24	2024-04-26	1st pre-release of LeadSuccess API for Microsoft Dynamics manual
			.
			.
			.