Evoko Home 2.0 API specification document

Content

I API specification

This document contains specification of endpoints that are available on Evoko Home.

1. Register Integration User

1.1. Overview

Method name: registerIntegrationUser
Parameters: groupToken of your clientGroup
Return value: Response user object if action successful, throws an error if action fails.
Description: This method is used to register a integration user related to clientGroup referenced with clientToken for successful login over DDP.

1.2. Parameter format

String

1.3. Example of paramter

string would be a token something like “ksdjafhZ_34923jff” which can be taken directly from the database from a specific clientGroup reading its groupToken

1.4. Return value

If the request was successful, the method will return a JSON response object with details of a integration user. Properties ‘username’ and ‘profile.groupToken’ should be used as username and password when logging in over DDP.

In the mongoDB there are details for defaultDevIntegrationUser which corresponds to DevAPIIntegration group, that could be used as username and password as well - if you wish to use that, there may be no need to register additional users.

Since version of 2.0 Alpha 19 you now have to be logged in to create API users and clientGroups. The default super-user for executing these actions can be the defaultDevIntegrationUser - for which you can find details in the mongoDb - a user would have to log in (see part 3) with credentials of that super-user to start creating specific clientGroups and associated users per need.

Meteor exception object has two attributes: error and reason.
The "error" attribute contains the error code and "reason" attribute contains an error message.
Because of internalization, service will return error key instead of error message. In new ERM, the device uses this key to find an appropriate message on language user has selected.

2. Register Client Group

Method name: registerClientGroup2x
Parameters: Group Name
Return value: Response group object if action successful with groupName and groupToken, throws an error if action fails.
Description: This method is used to register a client group so you can register API users related to this group using a groupToken for successful login over DDP.

2.2. Parameter format

String

2.3. Example of paramter

string would be a API client group name something like “MobilePhones”

2.4. Return value

If the request was successful, the method will return a JSON response object with details of a client group. Properties ‘groupName’ and ‘groupToken’ the latter of which can be used as password for users that are registered to that specific clientGroup.

In the mongoDB there are details for defaultDevIntegrationUser which corresponds to DevAPIIntegration group, that could be used as username and password as well - if you wish to use that, there may be no need to register additional users and additional client groups

Since version of 2.0 Alpha 19 you now have to be logged in to create API users and clientGroups. The default super-user for executing these actions can be the defaultDevIntegrationUser - for which you can find details in the mongoDb - a user would have to log in (see part 3) with credentials of that super-user to start creating specific clientGroups and associated users per need.

Meteor exception object has two attributes: error and reason.
The "error" attribute contains the error code and "reason" attribute contains an error message.
Because of internalization, service will return error key instead of error message. In new ERM, the device uses this key to find an appropriate message on language user has selected.

3. How to login over DDP using Meteor

This method is to be executed on DDP object.

3.1. Overview

Method name: loginWithPassword
Parameters: connection, JSON object with username, password, callback method
Return value: No response if action is successful. Throws an error if action fails.

Description: This method is used to send username and password for authentication with remote connection server over DDP. Username and password are retrieved within JSON response.

3.2. Parameter format

connection (return object of DDP.connect(IPaddress)), JSON object with username, password, callback method (callback method is optional)

Example of JSON object with username:
{username: ‘admin’ }

3.3. Example of JSON object

This method does not accept JSON object as single parameter.

3.4. Return value

If the request was successful the method returns no parameters, if the method fails it returns a Meteor exception.
Meteor exception object has two attributes: error and reason.
The "error" attribute contains the error code and "reason" attribute contains an error message.
Because of internalization, service will return error key instead of error message. In new ERM, the device uses this key to find an appropriate message on language user has selected.

4. Create Booking

4.1. Overview

Method name: createBooking
Parameters: JSON Object
Return value: Response object if success, throws an error if fail.
Description: This method is used to book a meeting in specified time slot.

4.2. Parameter format

Parameter must be a JSON object. All properties are of type string.
Properties that JSON object must have are:

- **roomId** – Document id of a room (_id field)
- **pin** – PIN number if PIN authentication is required. If PIN authentication is not required then the value of this property must be null.
- **rfid** – RFID ID when using RFID authentication. If RFID authentication is not used you can remove this field or set it to null.
- **startDate** – Start date of a meeting in format ISO 8601 (YYYY-MM-DDTHH:mm:ssZ).
- **endDate** – End date of a meeting. Minutes of this property must be set to 0, 15, 30 or 45 value. This is because of slot restriction to 15 minutes per slot. The format is the same as with start date.
- **confirmed** - true if the meeting has already started, otherwise false.
- **subject** - subject of created booking

• **metadata** – JSON object containing a single property "demoMode" that signal that application works in demo mode if set to "true", otherwise “false”.

4.3. Example of JSON object

```
{
        startDate: '2016-09-01T13:00:00+00:00',
        endDate: '2016-09-01T13:15:00+00:00',
        pin: '1832',
        roomId: 'AfLrwwy2hCnSw4reW',
        confirmed: false,
        subject: “Test subject”,
        metadata: { demoMode: false }
}
```

4.4. Return value

If the request was successful, the method will return a JSON response object with three attributes: type, message and body.
In case there was an error during execution of the method, the method will throw a Meteor exception. Meteor exception object has two attributes: error and reason.

The "error" attribute contains the error code and "reason" attribute contains the error message.

Because of internalization, service will return error key instead of error message. In new ERM, the device uses this key to find an appropriate message on language user has selected.

5. Delete meeting

5.1. Overview

Method name: deleteEvent
Parameters: JSON Object
Return value: Response object if success, throws an error if fail.
Description: This method is used to delete a meeting identified by its id.

5.2. Parameter format

Parameter must be a JSON object. All properties are of type string.
Properties that JSON object must have:

- **bookingId** – Document id of a meeting (_id field)
- **pin** – PIN number if PIN authentication is required. If PIN authentication is not required then value of this property must be null.
- **rfid** – Rfid ID when using rfid authentication. If rfid authentication is not used you can remove this field or set it to null.

- **metadata** – JSON object containing single property "demoMode" that signal that application works in demo mode if set to "true". If application is not working in demo mode then this property value should be "false".
- **ongoing** - true if meeting is in progress, otherwise false.

5.3. Example of JSON object

```
{
        pin: "1234",
        bookingId: "BmcHJpsxQYNavpyL4",
        ongoing: true,
        metadata: { demoMode: false }
}
```

5.4. Return value

If the request was successful, the method will return a JSON response object with three attributes: type, message and body. In case there was an error during execution of the method, the method will throw a Meteor exception.
Meteor exception object has two attributes: error and reason.
The "error" attribute contains error code and "reason" attribute contains error message.
Because of internalization, service will return error key instead of error message. In new ERM, device uses this key to find appropriate message on language user has selected.

6. Update a meeting

6.1. Overview

Method name: updateEvent
Parameters: JSON Object
Return value: Response object if success, throws an error if fail.
Description: This method is used to update a meeting identified by its id.

6.2. Parameter format

Parameter must be a JSON object. All properties are of type string.
Properties that JSON object must have:

- **_id** – Database id of the meeting (_id field)
- **id** – id of the meeting on booking system (id field)
- **pin** – PIN number if PIN authentication is required. If PIN authentication is not required then value of this property must be null.
- **rfid** – Rfid ID when using rfid authentication. If rfid authentication is not used you can remove this field or set it to null.
- **startDate** – Start date of a meeting in format ISO 8601 (YYYY-MM-DDTHH:mm:ssZ).
- **endDate** – End date of a meeting. Minutes of this property must

be set to 0, 15, 30 or 45 value. This is because of slot restriction to 15 minutes per slot. Format is the same as with start date.

- **metadata** – JSON object containing single property "demoMode" that signal that application works in demo mode if set to "true". If application is not working in demo mode then this property value should be "false".
- **ongoing** - true if meeting is in progress, otherwise false.

6.3. Example of JSON object

```
{
        _id: "lkjsdfbgliksdjfg",
        pin: "1234",
        id: "086090CFD39411298325802100364E63",
        endDate: "2016-09-01T13:45:00+00:00",
        startDate: "2016-09-01T13:00:00+00:00",
        ongoing:  true,
        metadata: { demoMode: false }
}
```

6.4. Return value

If the request was successful, the method will return a JSON response object with three attributes: type, message and body. In case there was an error during execution of the method, the method will throw a Meteor exception.
Meteor exception object has two attributes: error and reason.
The "error" attribute contains error code and "reason" attribute contains error message.
Because of internalization, service will return error key instead of error message. In new ERM, device uses this key to find appropriate message on language user has selected.

7. Confirm a meeting

7.1. Overview

Method name: confirmMeeting
Parameters: JSON Object
Return value: Response object if success, throws an error if fail.
Description: This method is used to check in a meeting. Check in is defined in settings for room. Check in option is available from five minutes before meeting starts to confirm duration time since meeting is started. Confirm duration time represents number of minutes after meeting has started that user is allowed to check in meeting. Confirm duration time is set in room settings on Evoko Home.

7.2. Parameter format

Parameter must be a JSON object. All properties are of type string.
Properties that JSON object must have:

- **bookingId** – Document id of a meeting (_id field).
- **pin** – PIN number if PIN authentication is required. If PIN

authentication is not required then value of this property must be null.

- **rfid** – Rfid ID when using rfid authentication. If rfid authentication is not used you can remove this field or set it to null.
- **metadata** – JSON object containing single property "demoMode" that signal that application works in demo mode if set to "true". If application is not working in demo mode then this property value should be "false".

7.3. Example of JSON object

```
{
        pin: null,
        bookingId: "BmcHJpsxQYNavpyL4",
        metadata: { demoMode: false }
}
```

7.4. Return value

If the request was successful, the method will return a JSON response object with three attributes: type, message and body. In case there was an error during execution of the method, the method will throw a Meteor exception.
Meteor exception object has two attributes: error and reason.
The "error" attribute contains error code and "reason" attribute contains error message.
Because of internalization, service will return error key instead of error message. In new ERM, device uses this key to find appropriate message on language user has selected.

8. Search for available rooms

8.1. Overview

Method name: getAvailableRoomsAdvancedNew
Parameters: JSON Object
Return value: Array that contains two arrays. First contain rooms that fully satisfy filter conditions, second contain rooms that partially satisfy filter conditions. Throws an error if fail.
Description: This method is used to get list of available rooms in specified timespan and selected filters.

8.2. Parameter format

Parameter must be a JSON object. Properties that JSON object must have:

- **isFiltered** – false if no filtering was done on user interface, otherwise true
- **room** - complete current room object from database
- **startTime** – Start date to filter in format ISO 8601 (YYYY-MM-DDTHH:mm:ssZ).
- **endTime** – End date to filter. Format is the same as with start date.
- **currentDate** - Current time in device device timezone. Format is the same as with start date.

- **equipment** – This field is used to filter rooms by their equipment. This is a JSON object that contains properties that represents various pieces of equipment. Supported equipment is: teleConference, videoConference, wifi, computer, whiteboard, information and projector. Property has value true if that piece of equipment is required by user, otherwise false. Rooms not fully matching filtering by equipment will be returned as partial matches.
- **metadata** – JSON object containing single property "demoMode" that signal that application works in demo mode if set to "true". If application is not working in demo mode then this property value should be "false".
- **seats** - Lower and upper boundary for seat filtering. Available options: “1-5”, “5-10”, “10-20”, “20-50”, “50+”
- **location** – Filter rooms by location. This is a JSON object that contains properties that represents various levels of hierarchy structures. If certain level of hierarchy is selected than property of this object has value equal to name of selected hierarchy, otherwise false. Available levels: country, city, building, floor.
- **customEquipment** - This field is used to filter by custom equipment. An array of JSON objects in format [{ _id: “fakeId”, name: “fakeName”, isChecked: true }] or [] if no custom equipment is selected.

8.3. Example of JSON object

Example of JSON object for this method where room has organisation structure and user wants to filter by date, equipment and number of seats:

{

isFiltered: false,

room:{"_id":"E56TjX4jpDh52kQyA","name":"Room test","mail":"roomtest@testdomain.com","address":"roomtest@testdomain.com","id":"roomtest","numberOfSeats":0,"alias":"Room test","isActive":true,"isDeleted":false,"equipment":{"lights":null,"projector":null,"computer":null,"teleConference":null,"wifi":null,"whiteboard":null,"videoConference":null,"display":null,"minto":null,"ac":null,"information":null},"assigned":true},

currentDate: "2018-06-12T14:18:19+02:00",

startTime: "2018-06-12T14:30:00+02:00",

endTime: "2018-06-12T14:45:00+02:00",

seats: "1-5",

location:{"country":"false","city":"false","building":"false","floor":"false"},

equipment:{"wifi":false,"whiteboard":false,"videoConference":false,"computer":false,"projector":false,"teleConference":false,"information":false,"minto":false,"display":false,"lights":false,"ac":false},

customEquipment: [{_id: “fakeId”, name: “fakeName”, isChecked: true }],

metadata: {"demoMode":false}

}

8.4. Return value

If the request was successful, the method will return an array that contains two arrays. First contain rooms that fully satisfy filter conditions, second contain rooms that partially satisfy filter conditions. Room that does not satisfy all filter condition but does not fail more than three filters will be returned as partial match. Timespan is mandatory field, rooms that are not available in specified timespan will not be returned regardless of other filters.
Meteor exception object has two attributes: error and reason.
The "error" attribute contains error code and "reason" attribute contains error message.
Because of internalization, service will return error key instead of error message. In new ERM, device uses this key to find appropriate message on language user has selected.

9. Update of room equipment

9.1. Overview

Method name: editRoomEquipmentReport
Parameters: JSON Object
Return value: Response object if success, throws an error if fail.
Description: This method is used to update list of room equipment. Equipment can be reported as broken or reported as repaired (Admin only). When equipment is reported as repaired, authorization is always necessary and it must be done by admin user.

9.2. Parameter format

Parameter must be a JSON object. Properties that JSON object must have:

- **adminReportFlag** – Flag indicating if admin is reporting that equipment is repaired. Only admin users can report that some piece of equipment is repaired.
- **pin** – PIN number if PIN authentication is required. If PIN authentication is not required then value of this property must be null.
- **rfid** – Rfid ID when using rfid authentication. If rfid authentication is not used you can remove this field or set it to null.
- **metadata** – JSON object containing single property "demoMode" that signal that application works in demo mode if set to "true". If application is not working in demo mode then this property value should be "false".
- **roomObject** – JSON object of room. This is complete room object with updated equipment status.

9.3. Example of JSON object

```
{
        pin: ‘’,
        roomObject: {
                _id: 'Xn3BrqZrZBrBo3c4F',
                name: 'Room 5',
                alias: 'Room 5',
                mail: 'room5@testdomain.com',
                address: 'room5@testdomain.com',
                id: 'room5',
                isActive: true,
                isDeleted: false,
                structureId: 'MwNEkFcjoQdowShmN',
                numberOfSeats: '10',
                equipment: {
                        wifi: true,
                        whiteboard: true,
                        videoConference: true,
                        computer: false,
                        projector: null,
                        teleConference: true,
                        information: null,
                        minto: true,
                        display: null,
                        lights: true,
                        ac: null
                },
        },
        adminReportFlag: false,
        metadata: { demoMode: false }
}
```

9.4. Return value

If the request was successful, the method will return a JSON response object with three attributes: type, message and body. In case there was an error during execution of the method, the method will throw a Meteor exception.
Meteor exception object has two attributes: error and reason.
The "error" attribute contains error code and "reason" attribute contains error message.
Because of internalization, service will return error key instead of error message. In new ERM, device uses this key to find appropriate message on language user has selected.

10. Store information about device with added information

10.1. Overview

Method name: storeInformationAboutDeviceWithIPAddresses
Parameters: JSON object
Return value: This method does not return value. Throws an error if fail.
Description: This method is used to store specific information about device on Evoko Home. This method is called only on device's startup.

10.2. Parameter format

This method accepts JSON object with following fields:
macAddress - Unique device address for ethernet interface.

macAddressWifi - Unique device address for wifi interface.
ipAddress - IP address for active network interface.
interfaceActive - Information about which network interface is active.
roomId – Id of a room on which device is currently subscribed.
evokeHomeIpAddress - IP address and port of Evoke Home Server.
lastRebootTime – Last reboot time of a device
firmwareVersion - Firmware version of the device
vncActive - set to true when VNC is activated, otherwise false.

10.3. Example of JSON object

```
{
        macAddress: "00:00:00:00:00:00",
        macAddressWifi: "00:00:00:00:00:00",
        ipAddress: "127.0.0.1",
        interfaceActive: "ethernet",
        roomid: "BuYuMCPc9RdsSZ8Nb",
        evokeHomeIpAddress: "127.0.0.1:3002",
        lastRebootTime: "2017-01-17T07:42:07+00:00",
        firmwareVersion: "2017_01_16_v1.25",
        vncActive: false
}
```

10.4. Example of request

call("storeInformationAboutDeviceWithIPAddresses", deviceData)

10.5. Return value

This method does not return a response.
In case there was an error during execution of the method, the method will throw a Meteor exception.
Meteor exception object has two attributes: error and reason.
The "error" attribute contains error code and "reason" attribute contains error message.
Because of internalization, service will return error key instead of error message. In new ERM, device uses this key to find appropriate message on language user has selected.

11. Receiving logs from device

11.1. Overview

Method name: logReceiver
Parameters: JSON object representing data sent from device
Return value: This method does not return value. Throws an error if fail.
Description: This method is used to receive logs from device. In case error happened during retrieval of log, error message can be sent in log field.

11.2. Parameter format

Parameter must be a JSON object. Properties that JSON object must have:

- **log** – This is a log data in string format.
- **lastModifiedUTC** – This represents a time in UTC when log file is last modified.
- **type** – Type of a log being sent. Type can be kernel, evoko, dmesg or monit depending on a log being sent.
- **macAddress** – Mac address of a device that is sending a log.

11.3. Example of JSON object

```
{
        log: "Some log text",
        lastModifiedUTC: "2016-12-09T15:01:49+00:00",
        type: "evoko",
        macAddress: "00:00:00:00:00:00"
}
```

II Subscription to a collection

1.1. Overview

Data from published collections can be read in Meteor by subscribing to that collection on the client, whereas the collections are published on the server. In order to get a list of meetings or book a meeting, subscription to "Bookings" collection is necessary. When booking a meeting, id of a room where user wants to book a meeting is necessary. In order to get list of all available rooms, subscription to "Rooms" collection must be established.
Identifier needed to subscribe to the "Bookings" collection is "allBookings".
Identifier needed to subscribe to the "Rooms" collection is "allRooms".
After an established connection with the service, a subscription should be done as stated above. No further steps should be necessary.

1.2. Connection details

Evoko Home and Evoko Liso communicate over secure socket layer (SSL). When connecting to Evoko Home you can use https or wss. It will work with both. Example:

```
https://ipaddress:port
wss://ipaddress:port
```

Port is a SSL port of Evoko Home that user has set during wizard process.

1.3. Collections

In order to fetch all relevant data from Evoko Home to ERM subscription to following collections must be initiated:

- **Rooms** – Key to subscribe to the Rooms collection is "allRooms". Parameters are a roomId (Document id) and page number. Depending of a value of roomId parameter, subscription will return data about single room or if null is passed data about all rooms. Page number is needed when roomId is null. In that case Evoko

Home will return all rooms so in order to not overload client in case when there is a huge number of rooms, page number is passed to enforce paging. Evoko Home will return up to 25 rooms per page. Page number values should be incremented/decremented by 25 every time new page is required (example: 0, 25, 50, 75, 100…). If page number is not passed, Evoko Home will use default value of 0 and it will return first 25 rooms.

- **Bookings** – Key to subscribe to the Bookings collection is "allBookings". Parameter is roomId (Document id of a room). Depending of a value of this parameter, subscription will return all booking for all rooms(not recommended for device) if value is "ALL", only booking for room identified by roomId or in case parameter is null it will not return any meeting.
- **Settings** – Key to subscribe to the Settings collection is "specificSettings", and it requires **roomId** (Document id of a room) parameter. Depending of value of this parameter, subscription will return all settings for this specific room.
- **Structures** – Key to subscribe to the Structures collection used in 1.x is "structures".
- **StructuresFlat** – Key to subscribe to the Structures collection used in 2.x is "structuresFlat".
- **Firmware** – Key to subscribe to Firmware collection is "firmwareData".
- **Users** – Key to subscribe to Users collection is "users".
  It should be noted that in the future some of this collections will publish restricted data not all.
- **LogNotifications** – This collection is used to notify device to perform various actions depending on request on Evoko Home. When for example upgrade request has been initiated for some device on Evoko Home, device will receive notification about upgrade and it should proceed with sending request to download image from Evoko Home and initiate upgrade process. Key to subscribe to LogNotifications collection is "allLogNotification". Parameter is mac address of a device which is subscribing to notifications from Evoko Home. Other notifications are related to various type of logs that device can send to the Evoko Home. Notifications can have values of "true" and "false". Value of "true" indicates that Evoko Home has requested that that device perform action which value is "true". Example of JSON object notification to upgrade device:

```
{
        "_id": "tGawLYw7bZbg9zrnB",
        "macAddress": "00:14:0b:84:bc:d5",
        "kernel": false,
        "monit": false,
        "dmesg": false,
        "evoko": false,
        "upgradeDevice": true,
        "upgradeDeviceTimeStampUTC": null
}
```

Time stamp value is used when scheduled upgrade is requested from Evoko Home and device should schedule upgrade at requested time. If all logs are requested then device will receive notification where value of kernel, monit, dmesg and evoko fields are set to true. Device should send logs to Evoko Home by calling **logReceiver** method. In case when all logs are requested then it should send separately every log in separate request to the Evoko Home.