OneMediaHub Version 14.5
Server API Developer's Guide

OneMediaHub Version 14.5 Server API Developer's Guide

Version 14.5.0 - Revision 180010 [2015-02-27]

Chapter 1. Get started

The OneMediaHub Server API utilizes a REST-style architecture (see [4]) over the HTTP protocol and all information is delivered in JSON format. If you are familiar with HTTP, REST API and JSON objects, you can skip ahead to the API definition in Chapter 3, *OneMediaHub Server API*.

Otherwise you might want to take a look first at the following sections:

- Section 6.4, “My first API call: examples” - examples in Java, PHP, and JavaScript

  Warning

  All API calls of type POST that require to pass data must accomplish this only by using a JSON object as request body. The use of the `data` parameter with data passed in encoded form in the URL is DEPRECATED.

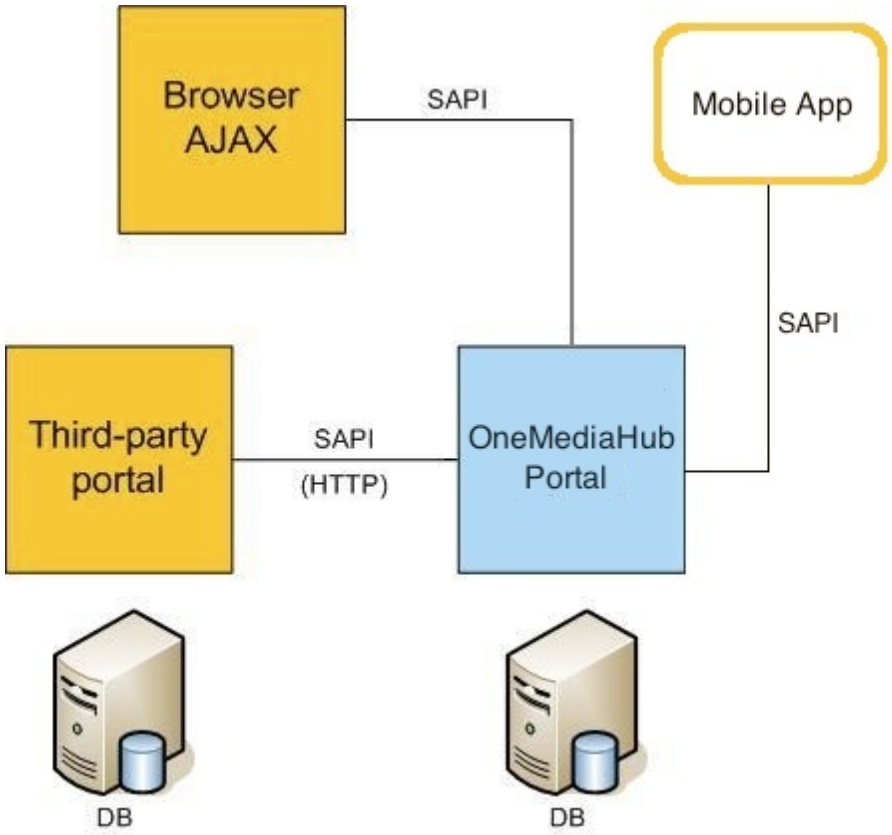
  In all API that require authentication, should be used a validation token to prevent a type of attack that allows a malicious user to impersonate another user. See Section 3.1.2, “Validation key and prevention of Cross Site Request Forgery attacks” for more details

- Section 6.5, “Tools for debugging and testing” - how to debug the UI XHR calls using Firebugs and other tools
- Chapter 6, *Tips and tools* - authentication options and much more
- Section 3.1.4, “Escaping special characters in JSON” - how to avoid to break JSON syntax
- Appendix A, *API List* - a handy list of all the available API

Chapter 2. Introduction

The OneMediaHub Server API (SAPI) enables customers, partners and professional services to easily integrate the main OneMediaHub functionalities into existing systems, applications and environments.

Figure 2.1. OneMediaHub Server API



Thanks to SAPI, it is possible to integrate OneMediaHub Portal functionalities into a preexisting customer portal without having to reference the OneMediaHub Portal database; it is also possible to access OneMediaHub Portal functionalities directly from an AJAX browser (see Figure 2.1, “OneMediaHub Server API”).

In the same way, client applications (e.g. apps) can profit of the same API to access the media synchronization capabilities provided by the OneMediaHub server, with no need to directly interact with the database behind it.

The main advantages of SAPI are:

- a highly interoperable API to access OneMediaHub Portal functionalities (e.g. user profile and account, supported devices, manual instructions, etc.)
- a highly decoupled architecture
- ease of use by developers
- easily integrated with AJAX based applications

This document describes the design principles and choices behind SAPI and serves as a specification document for SAPI implementation.

Chapter 3. OneMediaHub Server API

This chapter gives a definition of the APIs grouped by functionality.

3.1 General concepts

3.1.1 Security

Three security realms are defined:

public
: calls that can be performed without authentication

user
: calls that can be performed in the context of an authenticated portal user (i.e. data associated to the user can be accessed)

system
: calls that can be performed in the context of a super user or administrator (i.e. OTA settings provisioning, subscriptions' management, user profile information gathering)

For the **user** and **system** security realms, all calls must be authorized. The authentication can be performed at session level calling the login API, sending an *authorization* header according to an HTTP basic authentication like schema (see Section 3.2.3, “HTTP Basic Authentication like login”).

Also supported is the authentication protocol OAuth 2.0. Refer to [2], to Section 3.2.2, “OAuth login”, and to Section 5.32, “OAuth”.

Session tracking should be performed using the `jsessionid` cookie.

Note

Actions in the **user** security realm can usually be invoked from a **system** security realm as well. In this case, the caller can send administrative commands or retrieve profile informations belonging to any user, but can not access to user's personal data. The principle behind this choice is that when the API is used from an AJAX-based authentication, the client application must work in the **user** security realm, and therefore it must have access to public and personal information only. On the other hand, when SAPI is used to integrate with an external system, it must be possible to work at a system level, so that all not private user information can be accessed and updated. In this latter case the user must be specified in the call as *`userid`* parameter, and can be specified only by an administrator.

If the specified *`userid`* doesn't exist the PRO-1101 error code will be returned.

- If the given credentials are not authenticated, an HTTP status code 401 (unauthorized) is returned.
- If a call is not authorized for the given user, an HTTP status code 403 (forbidden) is returned.
- If the credentials (login and password) are provided as parameters or as an authorization header inside an open session, the error code SEC-1002 will be returned.
- If both the authentication parameters and authorization header are provided outside any session, the error code SEC-1004 will be returned.

Exceptions to these rules, if any, will be specified in the API description.

Block administrators connecting from untrusted IPs

To enhance security, it is possible to configure the OneMediaHub Portal to allow administrators to perform SAPI calls only from a set of trusted IPs. See [9] for more details on how to configure the OneMediaHub Server to support this feature. If this feature is enabled and an admin call is performed from an untrusted IP address, an HTTP status code 401 (unauthorized) is returned.

3.1.2 Validation key and prevention of Cross Site Request Forgery attacks

Cross-site Request Forgery (CSRF) is a type of attack that allows a malicious user to impersonate another user on a vulnerable site. The attack is possible when a site does not properly validate the origin of the request, and relies solely on the existence of a valid session identifier.

A possible solution is to introduce a mandatory validation token in all APIs that require authentication. The OneMediaHub server does support such measure.

The token is session and user dependent and is returned by the *login* API. Such token must be used for all subsequent requests for the logged user for the given session. If the token is not provided or is incorrect, a SEC-1003 error code is returned in the JSON error message.

The validation key is validated and used only for non-public APIs. For example, the requests for resetting the calendar described as

```
POST /sapi/calendar?action=reset
```

should be

```
POST /sapi/calendar?action=reset&validationkey=TXkkIjo1OUo+ImNLL0RPaA==
```

where `validationkey=TXkkIjo1OUo+ImNLL0RPaA==` is a key valid for the logged user for the given session only.

3.1.3 Conventions

OneMediaHub Server API (SAPI) is called invoking *actions*. Each group mentioned in the introduction of this chapter contains a list of all the available actions.

Each action is defined by:

- the HTTP request (method and URI) to be invoked
- request headers if any
- request parameters
- response
- errors that the request could return
- security realm
- example

Important

The only encoding supported by the SAPI is UTF-8.

Request

Any call that sends an object to the server must send a JSON object (see [1]) that contains the following property:

- `data`: the request payload

Note

Encapsulating the request payload in a `data` property allows more flexibility for future changes, meaning that the API can be changed in next versions maintaining the backward compatibility.

Warning

The usage of *`data`* as encoded parameter is now deprecated.

Response

All calls return a JSON object (see [1]) that can contain the following properties:

- *`data`*: the response payload
- *`error`*: error details, if any; the error payload contains an error code, an error message, a list of parameters and, if any, the original cause and the root cause.

Null fields are not returned in the response payload. Below is an example of response with just the payload:

```
{
   "data":{
      "roles":[
         {
            "name":"sync_administrator",
            "description":""
         }
      ]
   }
}
```

Below is an example of response with an error:

```
{
   "error":{
      "code":"PRO-1000",
      "message":"Unknown exception in profile handling",
      "parameters":[

      ],
      "cause":"com.funambol.portal.api.profile.ProfileException: OTA
 Message not sent for unknown reason",
      "rootcause":"com.funambol.framework.sms.SMSProviderException:
 Unable to retrieve the sms id (response: not authorized)"
   }
}
```

3.1.4 Escaping special characters in JSON

As for the JSON specification (see [3]), the object's key and value (when the value is a string) are a collection of zero or more Unicode characters, wrapped in double quotes, using backslash escapes. A character is represented as a single character string. There are two-character sequence escape representations of some popular characters so, for example, a string containing only a single backslash may be represented more compactly as "\\". When a single backslash is added in a string before regular characters, the behavior is not defined by the JSON specification and the `Json-lib` library (see [7]) used in the OneMediaHub Server API leaves the backslash alone (`a\bc` is the same as `abc`).

3.1.5 Server response time

Most of the time differences of a few minutes or seconds (in UTC) between the server time and the client time are irrelevant. Sometimes, it's important to know the exact server time of the response to be able to determine the correct time interval when performing subsequent requests. It's possible to ask the OneMediaHub server to add to the JSON response the "responsetime" (the time of the response) using the *`responsetime`* parameter and subsequent requests can be performed using the previous "responsetime" as reference in the new "from".

Example 1

Response time

Request:

```
POST /sapi/media/profile/changes?
action=get&from=20110220&responsetime=true
```

Response:

```
{
   "responsetime":1298477013293,
   "data":{

   }
}
```

3.1.6 Default ordering

Unless otherwise stated in the definition of a specific *GET* API call, the default ordering of the arrays in the JSON response is arbitrary. For example, no assumptions should be made on the default ordering of the contacts returned by the server. In the given scenario, the consumer of the given API should take care at client side of the ordering, if required.

3.1.7 Base URL

The OneMediaHub Server API (SAPI) works by default with the `/sapi` base URL. It is possible to configure OneMediaHub to support a different SAPI base URL. For more information, see Section 3.12.17, “How to configure the Server API base URL” in *OneMediaHub Version 14.5 Installation and Operation Guide*.

3.2 Login

These functions are used to authenticate the caller and obtain a session identifier to use in the subsequent API requests.

Note

The custom (not mandatory) HTTP header `X-devicename` provides, if present, the Base64 encoded device name and description that can be used to update the corresponding device information.

3.2.1 Login

Log the user into the system.

Security realm: public

Definition

```
POST /sapi/login?action=login
```

Parameters

- *`login`*: the username
- *`password`*: the password
- *`cred-info`*: (**optional**) if present with the value `captcha.token` it means that the server must also validate the CAPTCHA challenge. It can also be used to specify additional information about credentials (like their format.)
- *`rememberme`*: (**optional**) if present and `true`, the server will generate and return a persistent login cookie in the authentication response, if valid. If `false`, no persistent login cookie is returned. Default value is `false`.

Warning

For security reasons, username and password should be passed in the **body** of the request as parameters, and not in the request URL. This is to avoid that the user's credentials are shown in the access logs of the HTTP server.

Response

A JSON object containing a string with the session ID and a string containing the validation key to use in the subsequent API requests and the roles of the user. The JSON object can also contain the CAPTCHA URL necessary to build the next request. For more information about the *Role* JSON object, see Section 5.45, “Role”

Example 1

Roles for an authenticated portal user:

Request:

```
POST /sapi/login?action=login
```

Request body:

```
login=username&password=apiuser&cred-info=custom.account.auth
```

Response:

```
{
   "data":{
      "jsessionid":"ED8F3F87C5B0927222C913BD7889A690",
      "validationkey":"TXkkIjo1OUo+ImNLL0RPaA==",
      "roles":[
         {
            "name":"standard",
            "description":""
         },
         {
            "name":"sync_user",
            "description":""
         }
      ]
   }
}
```

Example 2

Roles for a portal administrator:

Request:

```
POST /sapi/login?action=login
```

Request body:

```
login=admin-user&password=password
```

Response:

```
{
   "data":{
      "jsessionid":"4B339C8F5437B7A9506D8C69901833BF",
      "validationkey":"TXkkIjo1OUo+ImNLL0RPaA==",
      "roles":[
         {
            "name":"sync_administrator",
            "description":""
         }
      ]
   }
}
```

Example 3

Login request, with CAPTCHA challenge enabled, after the number of max failed attempts is reached:

Request:

```
POST /sapi/login?action=login
```

Request body:

```
login=username&password=apiuser&cred-info=captcha.1234
```

Response:

```
{
   "data":{
      "jsessionid":"ED8F3F87C5B0927222C913BD7889A690",
      "validationkey":"TXkkIjo1OUo+ImNLL0RPaA==",
      "roles":[
         {
            "name":"standard",
            "description":""
         },
         {
            "name":"sync_user",
            "description":""
         }
      ]
   }
}
```

Errors

HTTP 400, HTTP 401, SEC-1002, SEC-1004, in addition to generic errors described in Section 4.1.3, “Errors across multiple Server APIs”.

3.2.2 OAuth login

Log in to a OneMediaHub server using an appropriate OAuth key. The OAuth key can be an OAuth authorization code or a JSON object encoded in Base64 containing an access token and a refresh token acquired previously from the client through the OAuth endpoint. See Section 5.32, “OAuth” for the specification of the JSON object.

Security realm: public

Definition

```
POST /sapi/login/oauth?action=login
```

Parameters

- *`key`*: authorization code, or JSON object containing OAuth tokens (encoded in Base64)
- *`keytype`*: can be one of
  - `authorizationcode`
  - `accesstoken`
- *`platform`*: the type of client. Valid values for supported clients are
  - `android`
  - `ios`
  - `windows`
  - `macos`

- `web`

- *`redirecturi`*: if specified, the server redirects the client to the specified URI instead

> **Note**
>
> If the OAuth authorization header is present, the parameter values will be taken from it.

Response

If no parameter *`redirecturi`* is specified, a string containing the session ID to be used in the subsequent API requests and the user's roles.

On the other hand, if the parameter *`redirecturi`* is specified, a client redirect is performed following the given redirect URI.

Example 1

Login with authorization code and no URI redirection:

Request

```
POST /sapi/login/oauth?
action=login&platform=web&key=TRHE3432IUM989ABCT567643RT6Y&keytype=au
thorizationcode
```

Response body

```
{
   "data":{
      "jsessionid":"ED8F3F87C5B0927222C913BD7889A690",
      "validationkey":"TXkkIjo1OUo+ImNLL0RPaA==",
      "roles":[
         {
            "name":"standard",
            "description":""
         },
         {
            "name":"sync_user",
            "description":""
         }
      ]
   }
}
```

Example 2

Login with authorization code and URI redirection:

Request

```
POST /sapi/login/oauth?
action=login&platform=web&key=TRHE3432IUM989ABCT567643RT6Y&keytype=au
thorizationcode&redirecturi=http%3A%2F%2Fonemediahub.com
```

Response

Redirection to the given redirect URI

Example 3

Login with an access token:

Request

```
POST /sapi/login/oauth?
action=login&keytype=accesstoken&key=eyJkYXRhIjp7ImFjY2Vzc3Rva2VuIjoi
QUNDRVNTX1RPS0VOIiwidmFsaWQiOiJ0cnVlIiwicGxhdGZvcm0iOiJ3aW5kb3dzIiwic
mVmcmVzaHRva2VuIjoiUkVGUkVTSF9UT0tFTiJ9fQ
```

Response body

```
{
   "data":{
      "jsessionid":"ED8F3F87C5B0927222C913BD7889A690",
      "validationkey":"TXkkIjo1OUo+ImNLL0RPaA==",
      "roles":[
         {
            "name":"standard",
            "description":""
         },
         {
            "name":"sync_user",
            "description":""
         }
      ]
   }
}
```

3.2.3 HTTP Basic Authentication like login

A user may be authenticated thanks to the authorization header in very similar way as the HTTP Basic Authentication schema (see [10]). The "basic" authentication scheme is based on the model that the client must authenticate itself with a login and a password for each realm. The server will service the request only if it can validate the login and password for the protected space of the Request-URI. There are no optional authentication parameters. To receive authorization, the client sends the login and password, separated by a single colon (":") character, within a Base64 encoded string in the credentials. Logins might be case sensitive. If the user agent wishes to send the login "Aladdin" and password "open sesame", for example, it would use the following header field:

```
Authorization: Basic QWxhZGRpbjpvcGVuIHNlc2FtZQ==
```

By the way, the above description is about the general HTTP basic authentication schema, there are some differences between the usual schema and what the Server API supports:

- Server API doesn't bind authentication info to any HTTP like realm
- Server API expects the user to send the authorization header inside the first HTTP request, without waiting for the server's response containing the `WWW-Authenticate` header
- if a user doesn't authenticate, an HTTP 401 status code (unauthorized) is returned to the client, but the server response does not contain a `WWW-Authenticate` header

Security realm: public

Headers

- Authorization: Basic xxx, where xxx is the login and the password separated by a colon ":" and encoded using the Base64 encoding.

Parameters

None.

Response

A string containing the session ID to use in the subsequent API requests and the roles of the user. For more information about the Role JSON object, see Section 5.45, “Role”

Example 1

Roles for an authenticated portal user:

Request:

```
POST /sapi/login?action=login
Authorization: Basic QWxhZGRpbjpvcGVuIHNlc2FtZQ==
```

Response:

```
{
   "data":{
      "jsessionid":"ED8F3F87C5B0927222C913BD7889A690",
      "validationkey":"TXkkIjo1OUo+ImNLL0RPaA==",
      "roles":[
         {
            "name":"standard",
            "description":""
         },
         {
            "name":"sync_user",
            "description":""
         }
      ]
   }
}
```

Example 2

Roles for a portal administrator:

Request:

```
POST /sapi/login?action=login
Authorization: Basic QWRtaW46YWRtaW4gcHdk
```

Response:

```
{
   "data":{
```

```
      "jsessionid":"4B339C8F5437B7A9506D8C69901833BF",
      "validationkey":"TXkkIjo1OUo+ImNLL0RPaA==",
      "roles":[
         {
            "name":"sync_administrator",
            "description":""
         }
      ]
   }
}
```

Errors

HTTP 400, HTTP 401, SEC-1002, SEC-1004 in addition to generic errors described in Section 4.1.3, “Errors across multiple Server APIs”

3.2.4 Logout

Log out from the system

Security realm: public

Definition

```
GET /sapi/login?action=logout
```

Parameters

None.

Response

The 200 HTTP code for a successful request.

Errors

Generic errors described in Section 4.1.3, “Errors across multiple Server APIs”.

3.3 Profile

These functions allow working with all the information about a user profile, such as personal details, email account configuration and device configuration.

3.3.1 Get changes

Retrieve all changes occurred to PIM data and media since the given *from* date. This API is the best way to monitor changes in user data since a given time. The API call returns the ID of *deleted*, *new*, and *updated* items.

Note

An item ID will be in an array only, and the status of an item reflects the latest status available: its latest change after the date passed with the `from` parameter. If several modifications occurred in that item, only the last one will be returned. For example, if the item has been added after the *from*

date and then deleted, the item ID will be in the `D` array. If the item has been added after the *from* date and then updated, the item ID will be in the `U` array. In media items (Pictures, Videos, Audio, and Files), the item IDs will be returned ordered by creation date.

Security realm: user

Definition

```
GET /sapi/profile/changes?action=get
```

Parameters

- *`from`*: (**mandatory**) date string in RFC-2445 (`yyyymmddThhmmss`) format or in UTC milliseconds.
- *`type`*: (**optional**) if not present, the changes related to any type of data source will be returned, otherwise only changes related to the given types. The value should be a list of comma-separated types, or the string `all` (which is the default behavior if this optional parameter is not present.) Currently valid types are
  - `contact`
  - `calendar`
  - `picture`
  - `video`
  - `file`
  - `audio`
  - `folder`
  - `family`

Response

A JSON object containing the updates. For each type affected by changes, the IDs of *new*, *updated*, and *deleted* items are returned.

Example 1

```
GET /sapi/profile/changes?action=get&from=20100815T000000
```

same as

```
GET /sapi/profile/changes?action=get&from=20100815T000000&type=all
```

same as

```
GET /sapi/profile/changes?action=get&from=1281844800000&type=all
```

Response:

```
{
   "data":{
      "picture":{
```

```
         "N":[
            12,
            34,
            432
         ],
         "U":[
            23,
            42,
            98
         ],
         "D":[
            10,
            232,
            123
         ]
      },
      "video":{
         "N":[
            67,
            21
         ],
         "U":[
            112
         ],
         "D":[
            11,
            321
         ]
      },
      "file":{
         "N":[
            64,
            2
         ],
         "U":[
            11
         ],
         "D":[
            1,
            31
         ]
      },
      "contact":{
         "N":[
            67,
            21
         ],
         "U":[
            112
         ],
         "D":[
            11,
            321
         ]
```

```
      },
      "calendar":{
         "N":[
            67,
            21
         ],
         "U":[
            112
         ],
         "D":[
            11,
            21
         ]
      },
      "audio":{
         "N":[
            76,
            77
         ],
         "U":[
            98
         ]
      },
      "folder":{
         "N":[
            67,
            21
         ]
      },
      "family":{
         "N":[
            456
         ],
         "U":[
            909,
            910
         ],
         "D":[
            187
         ]
      }
   }
}
```


Example 2

In a situation similar to the previous example, calling

```
GET /sapi/profile/changes?
action=get&from=20100815T000000&type=picture,contact
```

would have returned the following response


```
{
```

```
   "data":{
      "picture":{
         "N":[
            12,
            34,
            432
         ],
         "U":[
            231,
            42,
            98
         ],
         "D":[
            10,
            23,
            123
         ]
      },
      "contact":{
         "N":[
            67,
            21
         ],
         "U":[
            112
         ],
         "D":[
            11,
            21
         ]
      }
   }
}
```

Errors

COM-1008, COM-1011, PRO 1100.

3.3.2 Send OTA configuration

Send the OTA configuration message to a specific device, basing on its characteristics with the standard portal configuration.

Note

This API requires that a device is configured for the user. For more information, see Section 3.3.16, “Get device information for a user” to retrieve information about the configured device and Section 3.3.27, “Add a new device for a user” to add a new device.

Configuring the sources in the body of the JSON request, it is possible to limit the sync sources for the OTA configuration. To send a more customized OTA configuration or to send an OTA configuration to a user tho does not have a configured device in the OneMediaHub portal, please refer to Section 3.3.3, “Send generic OTA configuration”.

Security realm: user

Definition

```
POST /sapi/profile/ota?action=send-configuration
```

Parameters

- *`userid`*: this parameter can only be specified by a user logged in at system administrator level. It represents the user ID of the user to whom the SMS message is sent. In this case, the recipient of the SMS must be a registered user in the database. Users calling this API while not logged in at system administrator level can send the configuration only to themselves so the parameter is not required.
- *`userpassword`*: (**optional**) if specified, the user password in the database is ignored and the new one is used for the OTA message.

Response

The 200 HTTP code indicates a successful request.

Request body

The *Sources* JSON object described at Section 5.50, “Sources”.

If a sync source is not specified in the request body, the standard sync sources of the OneMediaHub (based on the model of the device associated to the user) will be configured, otherwise if one or more sync sources are specified in the request body, only those ones will.

Example 1

Request for sending the portal configuration to the logged in user (non-administrator user):

```
POST /sapi/profile/ota?action=send-configuration
```

Request body (if the data is empty, the body can be omitted):

```
{"data":{}}
```

Response:
The 200 HTTP code for a successful request.

Example 2

Request for sending the portal configuration to the logged in user just for the 'card' source (non-administrator user):

```
POST /sapi/profile/ota?action=send-configuration
```

Request body:

```
{
   "data":{
      "sources":[
         {
            "uri":"card"
         }
```

```
      ]
   }
}
```

Response:
The 200 HTTP code for a successful request.

Example 3

Request for sending the portal configuration to the specified user (administrator):

```
POST /sapi/profile/ota?action=send-configuration&userid=username
```

Request body:

```
{"data":{}}
```

Response:
The 200 HTTP code for a successful request.

Errors

PRO-1000, PRO-2001, PRO-2002, PRO-2003, PRO-2004, in addition to generic errors described in Section 4.1.3, “Errors across multiple Server APIs”.

3.3.3 Send generic OTA configuration

Send the OTA configuration message specifying the recipient and all the configuration parameters. The recipient may not be a OneMediaHub portal user, or may be a user who does not have a configured device in the OneMediaHub portal. To verify if the user has a configured device, please refer to Section 3.3.16, “Get device information for a user”.

Security realm: system

Definition

```
POST /sapi/profile/ota?action=send-generic-configuration
```

Parameters

- *`phonenumber:`* the phone number of the recipient. Note that the phone number should be encoded as a parameter in the URL, so the "+" should be replaced with "%2B".

Response

The 200 HTTP code indicates a successful request.

Request body

See Section 5.34, “OTA configuration”.

Note

The OTA profiles folder (`$FUNAMBOL_HOME/config/com/funambol/otacp/profiles-repository`) has a tree structure to provide more flexibility and to better support

different scenarios where the OTA configuration for a given manufacturer and model may or may not be known. For example:

1. **OTA configuration for a well known manufacturer and model**
   A known manufacturer/model is a device supported by OneMediaHub that has a specific OTA configuration (for example, Nokia e61). Providing the specific *`devicemanufacturer`* and *`devicemodel`*, the specific OTA configuration (as defined in `$FUNAMBOL_HOME/config/com/funambol/otacp/profiles-repository/nokia/e61/profile.properties`) can be used.

2. **OTA configuration for an unknown manufacturer and unknown model**
   The manufacturer and model of the user's device are unknown, or a specific OTA configuration is not available in OneMediaHub. A generic OTA configuration message (as defined in `$FUNAMBOL_HOME/config/com/funambol/otacp/profiles-repository/profile.properties`) can be used.

3. **OTA configuration for a known manufacturer but unknown model**
   This is the common scenario of a new phone for a supported manufacturer, where a generic OTA configuration message for the manufacturer is available but not a specific OTA configuration for the new device. For example, for a new Nokia device, a generic OTA configuration message for the manufacturer (as defined in `$FUNAMBOL_HOME/config/com/funambol/otacp/profiles-repository/nokia/profile.properties`) can be used.

Example

Request for sending a OTA configuration message to +3933332453532:

```
POST /sapi/profile/ota?action=send-generic-configuration&phonenumber=
%2B3933332453532
```

Request body:

```
{
   "data":{
      "accountusername":"username",
      "accountpassword":"nhoj",
      "devicemanufacturer":"nokia",
      "devicemodel":"e61",
      "userpin":"1234",
      "sources":[
         {
            "uri":"ex-cal",
            "name":"Calendar",
            "contenttype":"text/x-vcalendar",
            "contentversion":"1.0"
         },
         {
            "uri":"ex-card",
            "name":"Contact",
            "contenttype":"text/x-vcard",
            "contentversion":"2.1"
         }
```

```
        ]
    }
}
```

Response:
The 200 HTTP code for a successful request.

Errors

COM-1001, PRO-1000, PRO-1001, in addition to generic errors described in Section 4.1.3, “Errors across multiple Server APIs”.

3.3.4 Download SMS

Send a download SMS to a specific device. This is a text SMS containing the URL from where the OneMediaHub App can be downloaded.

Security realm: user

Definition

```
POST /sapi/profile/ota?action=download
```

Parameters

- *`userid`*: it can be specified only by an administrator user and represents the user ID of the registered user whom the SMS should be sent to.

Response

The 200 HTTP code indicates a successful request.

Example 1

Request for sending download SMS to the logged in user (non-administrator user):

```
POST /sapi/profile/ota?action=download
```

Example 2

Request for sending download SMS to the user 'username' (administrator):

```
POST /sapi/profile/ota?action=download&userid=username
```

Errors

COM-1005, PRO-1000, PRO-1104, PRO-1111, PRO-2002, SMS-1000, in addition to generic errors described in Section 4.1.3, “Errors across multiple Server APIs”.

3.3.5 Send the download link via SMS

Send the link of the download page via SMS as specified in the server configuration. See Chapter 3, *Installation and configuration* in *OneMediaHub Version 14.5 Installation and Operation Guide*.

Other than the one described in Section 3.3.4, “Download SMS”, this API call is public, so it can be used when the user has not yet been created. Moreover, the link sent is generic and not device dependent. In order to avoid SMS sending abuse, this API call needs a CAPTCHA token, like for the signup phase.

Security realm: public

Definition

```
POST /sapi/profile/download?action=send-download-link
```

Parameters

- *token*: the CAPTCHA code provided by the Portal
- *phonenumber*: the user's phone number to send the SMS to
- *language*: (**optional**) the language (ISO 639-2) to be used in the SMS

Examples

Request body example 1

```
{
   "data":{
      "phonenumber":"+3912312312",
      "token":"12345"
   }
}
```

Request body example 2

```
{
   "data":{
      "phonenumber":"+3912312312",
      "token":"12345",
      "language":"en"
   }
}
```

Response

The 200 HTTP status code indicates a successful request.

Errors

COM-1001, PRO-1001, PRO-1101, PRO-1104, PRO-1110, PRO-1122, PRO-1126, SMS-1000, in addition to generic errors described in Section 4.1.3, “Errors across multiple Server APIs”.

3.3.6 Send 'Download Mobile App' link to user per Email

Send an Email to the provided address with the link to download the mobile app, according to the type of device of the user.

If no Email address is provided, the address stored in the user profile will be used.

See Section 3.14.2, “Email counter configuration for messages containing the app download URL” in *OneMediaHub Version 14.5 Installation and Operation Guide* to know how to configure the server to limit the number of Email messages sent to every user.

Security realm: user

Definition

```
POST /sapi/profile/generic?action=send-download-link
```

Parameters

None.

Response

The 200 HTTP code indicates a successful request.

Example

Request:

```
POST /sapi/profile/generic?action=send-download-link
```

Request body example:

```
{
   "data":{
      "useremail":"frank@mail.com"
   }
}
```

Errors

PRO-1104, PRO-1122, in addition to generic errors described in Section 4.1.3, “Errors across multiple Server APIs”.

3.3.7 Get the download URL for the OneMediaHub App given a phone

Retrieve the download URL for the OneMediaHub App.

Security realm: system

Definition

```
GET /sapi/profile/download?action=get-phone-url
```

Parameters

- *`modelid`*: the ID of the model
- *`userid`*: the user ID of the registered user. It can only be specified by a user logged in at system administrator level.
- *`carrierid`*: the carrier ID.

- *pim*: (**optional**) boolean parameter that allows to download the OneMediaHub App for BlackBerry. Only supported value is `true`.

Note

The *modelid* parameter is always mandatory. The different scenarios are listed below:

- If the *modelid* is that of a BlackBerry device that supports the OneMediaHub App and the *pim* parameter is provided, the details for the App are provided.
- If the *modelid* corresponds to a device that does not support the OneMediaHub for BlackBerry, a PRO-1111 error is returned.
- If the optional *pim* parameter is provided but OneMediaHub or the device do not support the App, a COM-1005 error is returned.

Response

The download URL for the device of the logged in user. For more information about the Download URL JSON object, see Section 5.13, “Download URL”.

Example

Request for the download URL for a phone that supports the OneMediaHub for BlackBerry:

```
GET /sapi/profile/download?action=get-phone-url&modelid=1898&pim=true
```

Response:

```
{
   "data":{
      "downloadurl":{
         "portalurl":"https://my.server.com",
         "downloadpath":"/bb/plugin.jad"
      }
   }
}
```

Errors

COM-1003, COM-1005, PRO-1000, PRO-1001, PRO-1101, PRO-1104, PRO-1111, in addition to generic errors described in Section 4.1.3, “Errors across multiple Server APIs”.

3.3.8 Get the download URL for the OneMediaHub App for Windows

Retrieve the download URL for the OneMediaHub App for Windows.

Security realm: public

Definition

```
GET /sapi/profile/download?action=get-windows-url
```

Response

The download URL for the OneMediaHub App for Windows. For more information about the Download URL JSON object, see Section 5.13, “Download URL”

Example

Request for the download URL for the OneMediaHub for Windows:

```
GET /sapi/profile/download?action=get-windows-url
```

Response:

```
{
   "data":{
      "downloadurl":{
         "portalurl":"https://my.server.com",
         "downloadpath":"/me/onemediahub-for-windows.exe"
      }
   }
}
```

Errors

Generic errors described in Section 4.1.3, “Errors across multiple Server APIs”.

3.3.9 Get the download URL for the OneMediaHub App for Mac OS

Retrieve the download URL for the OneMediaHub App for Mac OS.

Security realm: public

Definition

```
GET /sapi/profile/download?action=get-macos-url
```

Response

The download URL for the OneMediaHub for Mac OS. For more information about the Download URL JSON object, see Section 5.13, “Download URL”.

Example

Request for the download URL for the OneMediaHub for Mac OS:

```
GET /sapi/profile/download?action=get-macos-url
```

Response:

```
{
   "data":{
      "downloadurl":{
         "portalurl":"https://my.server.com",
         "downloadpath":"/macos/onemediahub-for-macos"
      }
   }
}
```

Errors

Generic errors described in Section 4.1.3, “Errors across multiple Server APIs”.

3.3.10 Get the download URL for the OneMediaHub App given a user device

Retrieve the download URL for the App.

Security realm: user

Definition

```
GET /sapi/profile/download?action=get-device-url
```

Parameters

- *`userid`*: the ID of the registered user. It can be specified only by an administrator.
- *`phonenumber`*: the phone number. It can be specified only by an administrator. It is used to retrieve the user with the given phone number. Note that the phone number should be encoded as a parameter in the URL, so the "+" should be replaced with "%2B".
- *`deviceid`*: the device ID. It can be specified only by an administrator. It is used to retrieve the user with the given device ID.
- *`pim`*: boolean parameter that allows to download the OneMediaHub App for BlackBerry. Only supported value is `true`. If OneMediaHub or the device do not support the App, a COM-1005 error is returned.

> **Note**
>
> The administrator must specify only one of the *`userid`*, *`phonenumber`*, and *`deviceid`* parameters.

Response

The download URL for the device of the logged in user or according to the given parameters. For more information about the Download URL JSON object, see Section 5.13, “Download URL”.

Example 1

Request for the download URL for the logged (non-administrator) user:

```
GET /sapi/profile/download?action=get-device-url
```

Response:

```
{
   "data":{
      "downloadurl":{
         "portalurl":"https://my.server.com",
         "downloadpath":"/bb/plugin.jad"
      }
   }
}
```

Example 2

Request for the download URL for the user 'username':

```
GET /sapi/profile/download?action=get-device-url&userid=username
```

Response:

```
{
   "data":{
      "downloadurl":{
         "portalurl":"https://my.server.com",
         "downloadpath":"/bb/plugin.jad"
      }
   }
}
```

Errors

COM-1001, COM-1005, PRO-1000, PRO-1101, PRO-1104, PRO-1111, PRO-1118, in addition to generic errors described in Section 4.1.3, “Errors across multiple Server APIs”.

3.3.11 Get user profile

Retrieve profile information for a user: generic, phone, and Email.

Security realm: user

Definition

```
GET /sapi/profile?action=get
```

Parameters

- *`userid`*: it can be specified only by an administrator.

Response

A User object with all the user profile information. For more information, see Section 5.59, “User” (User JSON object), Section 5.22, “Generic user information” (Generic User Information JSON object), and Section 5.36, “Phone” (Phone JSON object).

Example

Request:

```
GET /sapi/profile?action=get
```

Response:

```
{
   "data":{
      "user":{
         "generic":{
            "firstname":"John",
            "lastname":"Dawson",
            "useremail":"john@dawson.com",
```

```
            "timezone":"Europe/Rome",
            "active":true,
            "userid":"dawson",
            "mailinglist":true,
            "birthday":20020901,
            "male":true,
            "photo":true
         },
         "phones":[
            {
               "deviceid":2,
               "phonenumber":"+393333333333",
               "modelid":1105,
               "carrierid":9,
               "countrya2":"IT",
               "active":true,
               "converttmz":1
            }
         ],
         "emails":[

         ]
      }
   }
}
```


Errors

PRO-1000, PRO-1101, in addition to generic errors described in Section 4.1.3, “Errors across multiple Server APIs”.

3.3.12 Get user account information

Retrieve all generic information (first name, last name, ...) for a user.

Security realm: user

Definition

```
GET /sapi/profile/generic?action=get
```

Parameters

- *`userid`*: it can be specified only by an administrator.

Response

A User object with all the generic information for a user. For more information about the Generic User Information JSON object, see Section 5.22, “Generic user information”.

Example

Request:

```
GET /sapi/profile/generic?action=get
```

Response:

```
{
   "data":{
      "user":{
         "generic":{
            "firstname":"John",
            "lastname":"Dawson",
            "useremail":"john@dawson.com",
            "timezone":"Europe/Rome",
            "userid":"dawson",
            "active":true,
            "mailinglist":true,
            "birthday":20020901,
            "male":true,
            "photo":false
         }
      }
   }
}
```

Errors

PRO-1000, PRO-1101, in addition to generic errors described in Section 4.1.3, “Errors across multiple Server APIs”.

3.3.13 Search for users

Find user profiles that exactly match all the search conditions. The search is case insensitive and returns a list.

If no user matching the search criteria is found, an empty list is returned.

Being the request of type GET, the parameters should be passed in the URL (with their content encoded as per RFC 1738).

Security realm: system

Definition

```
GET /sapi/profile?action=search
```

Parameters

> **Note**
>
> At least one parameter must be provided.

- *`firstname`*: the user's first name
- *`lastname`*: the user's last name
- *`userid`*: the user's username

- *phonenumber*: the user's phone number. Note that the phone number has to be encoded as a parameter in the URL, so the + sign must be replaced with %2B
- *useremail*: Email address stored in the profile. Note that the Email address has to be encoded as a parameter in the URL, so the @ sign must be replaced with %40
- *active*: boolean flag. If not present, a list of active and inactive users is returned; if true only the active ones, if false only the inactive

Example for *userid*

```
userid=username
```

Example for *phonenumber*

```
phonenumber=+393385949767
```

and, as URL-encoded *phonenumber*:

```
phonenumber=%2B393385949767
```

Example for *useremail*

```
useremail=user@email.com
```

and, as URL-encoded *useremail*:

```
useremail=user%40email.com
```

Response

An array of *User* JSON objects. A *User* object will contain all the user profile information. For more about this see Section 5.59, “User”, Section 5.22, “Generic user information”, and Section 5.36, “Phone”.

Example

Request

```
GET /sapi/profile?action=search&phonenumber=%2B393333333333
```

Response

```
{
   "data":{
      "users":[
         {
            "userid":"username",
            "firstname":"",
            "lastname":"",
            "useremail":"username@email.com",
            "timezone":"GMT",
            "active":true,
            "photo":false
         },
         {
            "username":"user2",
            "firstname":"",
```

```
            "lastname":"",
            "email":"user2@email.com",
            "timezone":"Europe/Rome",
            "active":true,
            "photo":false
         }
      ]
   }
}
```

Errors

PRO-1000, PRO-1001, in addition to generic errors described in Section 4.1.3, “Errors across multiple Server APIs”.

3.3.14 Send reset password URL to user

Send the URL that will allow the user to reset his password given an existing profile email, username or MSISDN.

Security realm: public

Definition

```
POST /sapi/profile/generic/password?action=reset &sendby=[email|sms]
```

Parameters

- *sendby*: (**optional**) the medium to which the communication will take place. Possible values are `email` or `sms`

Response

The 200 HTTP code indicates a successful request.

Example

Request:

```
POST /sapi/profile/generic/password?action=reset &sendby=[email|sms]
```

Request body example 1:

```
{
   "data":{
      "useremail":"frank@mail.com"
   }
}
```

Request body example 2:

```
{
   "data":{
      "userid":"homer"
   }
}
```

Request body example 3:

```
{
   "data":{
      "msisdn":"1234567890"
   }
}
```

Errors

COM-1016, PRO-1101, PRO-1110, PRO-1122, PRO-1129, PRO-1136, in addition to generic errors described in Section 4.1.3, “Errors across multiple Server APIs”.

3.3.15 Reset user password

Reset the password given a valid reset token and a valid new password.

Security realm: public

Definition

```
POST /sapi/profile/generic/password?action=save
```

Parameters

None.

Response

The 200 HTTP code indicates a successful request.

Example

Request:

```
POST /sapi/profile/generic/password?action=save
```

Request body example 1:

```
{
   "data":{
         "password":"mynewpassword",
         "resettoken":"safgsfaslkjfhs3u4pt13haskj"
      }
   }
}
```

Errors

PRO-1001, PRO-1115, PRO-1133, in addition to generic errors described in Section 4.1.3, “Errors across multiple Server APIs”.

3.3.16 Get device information for a user

Retrieve all the device information (devices, model, manufacturer, ...) for a given user.

Security realm: user

Definition

```
GET /sapi/profile/device?action=get
```

Parameters

- *`userid`*: it can be specified only by an administrator.

Response

A User object with the list of all the devices and their properties for the user. If no device is available for the given user, a PRO-1104 error is returned. For more information about the *Phone* JSON object, see Section 5.36, “Phone”.

Example 1

Request:

```
GET /sapi/profile/device?action=get
```

Response:

```
{
   "data":{
      "user":{
         "phones":[
            {
               "deviceid":2,
               "phonenumber":"+393333333333",
               "modelid":1105,
               "carrierid":9,
               "countrya2":"IT",
               "active":true,
               "converttmz":1
            }
         ]
      }
   }
}
```

Example 2

Request:

```
GET /sapi/profile/device?action=get
```

Response:

```
{
   "error":{
      "code":"PRO-1104",
      "message":"Device not found",
      "parameters":[
```

```
      ],

 "cause":"com.funambol.portal.api.profile.NoUserDevicesFoundException
"
   }
}
```

Errors

PRO-1000, PRO-1101, PRO-1104, in addition to generic errors described in Section 4.1.3, “Errors across multiple Server APIs”.

3.3.17 User signup

Allow a guest who provides a correct validation token to add a new user and a device. See Section 3.9.9, “Get a CAPTCHA image”. The following information has to be provided:

- **Generic user information**
  - username
  - password
  - first name (**optional**)
  - last name (**optional**)
  - email address
  - timezone (**optional**)
  - preferredcommunicationchannel (**optional**)
- **Device information**
  - device model ID (given the model and manufacturer)
  - device phone number
  - device carrier ID
  - device country A2 code
  - device time zone conversion option (**optional**)

> **Note**
>
> Providing a device in the signup is optional, but if the device is specified, some properties are mandatory (see below).

> **Note**
>
> If a timezone is not provided in the *Generic user information*, the value `GMT` will be used as default fallback. However, this timezone is not considered valid by the API call described at

Section 3.3.22, “Update an existing user”, which only accepts values as returned by the API call described at Section 3.9.5, “Get timezones”.

Important

In order to have a token in the session, this API must be called after having invoked the API described at Section 3.9.9, “Get a CAPTCHA image”.

Security realm: public

Definition

```
POST /sapi/profile?action=signup
```

Request body: user

A JSON object with information regarding the user. For more information about the *Generic User Information* JSON object, see Section 5.22, “Generic user information”.

Request body: phone

A JSON object with information regarding the phone. For more information about the *Phone* JSON object, see Section 5.36, “Phone”.

Parameters

- *`token`*: (**mandatory**) the validation token to match the value in the image returned by the API call described in Section 3.9.9, “Get a CAPTCHA image”.
- *`email`*: if `true`, a confirmation email is sent to the registered user using the email address provided during signup. Default is `false`.

Response

A JSON object that acknowledges if the new user is active or not.

Example

Request:

```
POST /sapi/profile?action=signup&token=2333
```

Request body:

```
{
   "data":{
      "user":{
         "generic":{
            "userid":"username",
            "password":"mypassword",
            "firstname":"John",
            "lastname":"Dawson",
            "useremail":"john@dawson.com",
```

```
            "timezone":"Europe/Rome",
            "l10n":"en"
            "preferredcommunicationchannel":"email"
         },
         "phone":{
            "phonenumber":"+393333333333",
            "modelid":1105,
            "carrierid":9,
            "countrya2":"IT"
         }
      }
   }
}
```

Response

```
{
   "data":{
      "user":{
         "active":true
      }
   }
}
```

Errors

COM-1006, COM-1008, PRO-1000, PRO-1001, PRO-1106, PRO-1107, PRO-1113, PRO-1115, PRO-1122, PRO-1126, PRO-1135, in addition to generic errors described in Section 4.1.3, “Errors across multiple Server APIs”.

3.3.18 Mobile user signup

Allow a guest user who provides a correct validation token to add a new user and a device. If the validation token is required, see Section 3.9.9, “Get a CAPTCHA image”. The following information has to be provided:

General user and device information

- phone number
- password
- platform
- userid (**optional**)
- email address (**optional**)
- timezone (**optional**)
- device manufacturer (**optional**)
- device model (**optional**)
- device carrier (**optional**)

- country A2 code (**optional**)
- l10n - The locale of the new user, for example `en-US` (**optional**)
- preferredcommunicationchannel (**optional**)

Note

1. If the country A2 code is not specified (for example 'US'), the country id will be retrieved from the (mandatory) phone number that must include the international code.
2. If the device manufacturer name and the model name matches one that is available in the OneMediaHub product, that specific model ID will be used. Otherwise the API will try to find a similar match in the database, if no similar match is found, a generic model ID for the given platform will be used.
3. If the carrier name matches one that is available in the OneMediaHub product the specific carrier ID will be stored. Otherwise the API will try to find a similar match in the database, if no similar match is found, a generic carrier ID for the given country will be used ('Other'.)
4. If the timezone id is not specified the default value for the given country is used.
5. When a new user is added, no confirmation email will be sent to the newly created user.
6. The 'phonenumber' is used as 'login' for the login if the field 'userid' is not specified

Security realm: public

Definition

```
POST /sapi/mobile?action=signup
```

Parameters

- *`token`*: (**mandatory** if CAPTCHA enabled server side, otherwise ignored) the validation token to match the value in the image returned by the API call described at Section 3.9.9, “Get a CAPTCHA image”.

Possible platform field values

Table 3.1. Platforms

Type	Component
bbpim	OneMediaHub for BlackBerry
bbpim_os47	OneMediaHub for BlackBerry (OS 4.7 and 5.0)
bbpim_os60	OneMediaHub for BlackBerry (OS 6.0)
android	OneMediaHub for Android
iphone	OneMediaHub for iPhone
native	Native devices

Request body: Mobile signup user

A JSON object with information regarding the mobile user. For more information about the *Mobile User Information* JSON object, see the example below.

Response

A user object with the status (active or not active) of the new user.

Example 1

Request:

```
POST /sapi/mobile?action=signup&token=23333
```

Request body:

```
{
   "data":{
      "user":{
         "phonenumber":"+1234300000",
         "password":"12345",
         "platform":"bbpim",
         "manufacturer":"Blackberry",
         "model":"8800",
         "carrier":"Verizon",
         "countrya2":"US",
         "l10n":"en-US"
         "preferredcommunicationchannel":"sms"
      }
   }
}
```

Response:

```
{
   "data":{
      "user":{
         "active":true
      }
   }
}
```

Example 2

Request:

```
POST /sapi/mobile?action=signup&token=23333&userid=username
```

Request body:

```
{
   "data":{
      "user":{
         "phonenumber":"+1234300000",
```

```
            "password":"12345",
            "platform":"bbpim",
            "manufacturer":"Blackberry",
            "model":"8800",
            "carrier":"Verizon",
            "countrya2":"US"
            "preferredcommunicationchannel":"sms"
         }
      }
}
```

Response:

```
{
   "data":{
      "user":{
         "active":true
      }
   }
}
```

Errors

COM-1001, COM-1006, COM-1008, PRO-1000, PRO-1001, PRO-1106, PRO-1113, PRO-1115, PRO-1122, PRO-1126, PRO-1128, PRO-1135, in addition to generic errors described in Section 4.1.3, “Errors across multiple Server APIs”.

3.3.19 Validate user credentials before signup

Allow a guest user to validate the specified credentials (user ID if specified, phone number and password) before signup.

The following validation rules are processed in the following order: phone number presence, phone number format, country code existence, userid format if specified, username availability, password format. The first failed validation step is returned.

Security realm: public

Definition

```
POST /sapi/profile?action=validate
```

Parameters

None.

Request body

A JSON object containing the mobile user information to be validated. See Section 5.60, “User credentials”.

Response

The object with successful messages in case of validation passed or the error object with the first failed validation step otherwise.

Example 1

Request:

```
POST /sapi/profile?action=validate
```

Request body:

```
{
   "data":{
      "user":{
         "phonenumber":"+1234300000",
         "password":"12345"
      }
   }
}
```

Response:

```
{
   "validated":true
}
```

Example 2

Request:

```
POST /sapi/profile?action=validate
```

Request body:

```
{
   "data":{
      "user":{
         "phonenumber":"+1234300000",
         "password":"incorrect;!:password"
      }
   }
}
```

Response:

```
{
   "error":{
      "code":"PRO-1115",
      "message":"The password is not valid. Only letters (a-z)
 and numbers (0-9) and dash are allowed. Minimum 4, maximum 16
 characters."
   }
}
```

Example 3

Request:

```
POST /sapi/profile?action=validate
```

Request body:

```
{
   "data":{
      "user":{
         "phonenumber":"8997778899",
         "password":"correct-password"
      }
   }
}
```

Response:

```
{
   "error":{
      "code":"COM-1017",
      "message":"The country code is not valid"
   }
}
```

Example 4

Request:

```
POST /sapi/profile?action=validate
```

Request body:

```
{
    "data":{
        "user":{
            "userid":"+1234345678",
            "phonenumber":"+1234300000",
            "password":"12345"
         }
    }
}
```

Response:

```
{
    "validated":true
}
```

Errors

COM-1017, PRO-1000, PRO-1113, PRO-1115, PRO-1128, in addition to generic errors described in Section 4.1.3, “Errors across multiple Server APIs”.

3.3.20 Get a download App page for the new user

Allow a guest user who opens the welcome page from their mobile device browser to download the correct corresponding OneMediaHub App for the automatically detected device. The call performs HTTP redirect to the download page with specifying the device platform and preferred language. The location of the download page is specified in the server configuration (see Chapter 3, *Installation and configuration* in *OneMediaHub Version 14.5 Installation and Operation Guide*).

Security realm: public

Definition

```
POST /sapi/mobile?action=welcome
```

Parameters

The following parameters are passed to the download page:

- *platform*: detected device platform (if successfully detected). See the Table 3.1, “Platforms” for possible values;
- *language*: language code (ISO 639-2) preferred by the device browser and which is supported by the portal instance.

Response

The HTTP redirect to the download page.

Example

Request:

```
POST /sapi/mobile?action=welcome
```

Response:

```
GET /js/clients.jsp?platform=bbpim_os47&language=it
```

Errors

COM-1008, in addition to generic errors described in Section 4.1.3, “Errors across multiple Server APIs”.

3.3.21 Add a new user

Add a new user. The following information has to be provided:

- username
- password
- first name (**optional**)
- last name (**optional**)
- email address (**optional**)
- timezone (**optional**)

The new user will be active.

Note

If a timezone is not provided, the value `GMT` will be used as default fallback. However, this timezone is not considered valid by the SAPI call described at Section 3.3.22, “Update an existing user”, which only accepts values as returned by the SAPI call described at Section 3.9.5, “Get timezones”.

Security realm: system

Definition

```
POST /sapi/profile/generic?action=add
```

Parameters

- *`email`* (**optional**): true to send the welcome email, false otherwise. The parameter is false by default.

Request body

A JSON object with information regarding the user. For more information about the *Generic User Information* JSON object, see Section 5.22, “Generic user information”.

Response

The 200 HTTP code indicates a successful request.

Example

Request:

```
POST /sapi/profile/generic?action=add
```

Request body:

```
{
   "data":{
      "user":{
         "generic":{
            "userid":"username",
            "password":"mypassword",
            "firstname":"John",
            "lastname":"Dawson",
            "useremail":"john@dawson.com",
            "timezone":"Europe/Rome"
         }
      }
   }
}
```

Errors

COM-1006, PRO-1000, PRO-1001, PRO-1106, PRO-1107, PRO-1113, PRO-1115, PRO-1122, in addition to generic errors described in Section 4.1.3, “Errors across multiple Server APIs”.

3.3.22 Update an existing user

Update an existing user.

Security realm: user

Definition

```
POST /sapi/profile/generic?action=update
```

Parameters

- *`userid`*: the user ID of the registered user to update. It can be specified only by an administrator.

Request body

A JSON object with information regarding the user. For more information about the *Generic User Information* JSON object, see Section 5.22, “Generic user information”.

Response

The 200 HTTP code indicates a successful request.

Note

Only the JSON object's properties specified in the request body will be updated. If the value is empty for a property, the value will be updated to empty. If a property is not provided in the request, the value will not be updated.

Example

Request:

```
POST /sapi/profile/generic?action=update
```

Request body:

```
{
   "data":{
      "user":{
         "generic":{
            "oldpassword":"myoldpassword",
            "password":"mynewpassword",
            "firstname":"John",
            "lastname":"",
            "active":true
         }
      }
   }
}
```

Errors

COM-1006, PRO-1000, PRO-1001, PRO-1101, PRO-1107, PRO-1115, PRO-1122, PRO-1131, PRO-1132, in addition to generic errors described in Section 4.1.3, “Errors across multiple Server APIs”.

3.3.23 Delete an existing user

Delete an existing user. An administrator can delete an existing user, or a user can delete himself.

Security realm: user

Definition

```
POST /sapi/profile/generic?action=delete
```

Parameters

- *userid*: it represents the user ID of the user to be deleted. This parameter can only be specified by a user logged in at system administrator level. A user not logged in at system administrator level can call this API only for deleting himself, so in this case the parameter is not required.
- *token*: it is **mandatory** when the logged user tries to delete himself.

Request body

See Section 5.61, “User password”.

Response

The `200` HTTP code indicates a successful request.

Example 1

Request:

```
POST /sapi/profile/generic?action=delete&userid=username
```

Example 2

Request:

```
POST /sapi/profile/generic?action=delete&token=12345
```

Request body:

```
{
   "data":{
      "userPwd":"mypassword"
   }
}
```

Errors

MED-1014, PRO-1000, PRO-1001, PRO-1101, PRO-1107, PRO-1112, PRO-1126, SEC-1001, in addition to generic errors described in Section 4.1.3, “Errors across multiple Server APIs”.

3.3.24 Insert/update profile photo

Insert or update a user's profile picture. This request must be performed as a multipart form post, because the body of the request will contain the file and the fields needed for the update.

Security realm: user

Definition

```
POST /sapi/profile/photo?action=photosave
```

Parameters

- *userid* (**optional** in the URL request): the user ID of the registered user for whom to add/update the profile picture. It can only be specified by a user logged in at system administrator level.

- *`input_photo_0`* (**required** in multipart form post): the user picture.
- *`callback`* (**optional** in multipart form post): a JavaScript method to be called in order to update the picture of the user in the interface. This is needed because the photo is posted using a form submit; it is a way of refreshing the interface without refreshing the entire page.

Response

An HTML response with success message and ID (or error code) and callback method. If the callback is not present, the response will be a JSON object only.

Example

callback: `window.parent.Zapatec.Widget.getWidgetById(74).serverSendOnLoad`

JSON response message: `{"success":"Photos saved successfully."}`

```
<html><body><script type='text/javascript'>
window.parent.Zapatec.Widget.getWidgetById(74).serverSendOnLoad({"succes
s":"Photos
saved successfully."})</script></body></html>
```

Errors

PRO-1000, PRO-1101, PRO-4002, in addition to generic errors described in Section 4.1.3, “Errors across multiple Server APIs”.

3.3.25 Delete profile picture

Delete the user's profile picture.

Security realm: user

Definition

```
POST /sapi/profile/photo?action=photodelete
```

Parameters

- *`userid`*: (**optional**) the user ID of the registered user for whom to delete the profile picture. It can only be specified by a user logged in at system administrator level.

Response

A JSON response with success message and ID or error code.

Example:

```
{
    "success":"Photo removed successfully"
}
```

Errors

PRO-1000, PRO-1101, PRO-4001, in addition to generic errors described in Section 4.1.3, “Errors across multiple Server APIs”.

3.3.26 Retrieve profile picture

Return the photo as it is stored on the server.

Security realm: user

Definition

```
GET /sapi/profile/photo?action=photodownload
```

Parameters

- *`userid`*: (**optional**) the user ID of the registered user for whom to retrieve the profile picture. It can only be specified by a user logged in at system administrator level.

Response

The user picture as stored in the server.

Errors

PRO-1000, PRO-1101, in addition to generic errors described in Section 4.1.3, “Errors across multiple Server APIs”.

3.3.27 Add a new device for a user

Add a new device for a given user. The following information has to be provided:

- model ID (given the model and manufacturer)
- phone number
- carrier ID
- country A2 code
- timezone conversion option (**optional**)
- device name (**optional**): a custom device name/description.

The new device will be active.

> **Note**
>
> Only one device can be associated with a user.

Security realm: user

Definition

```
POST /sapi/profile/device?action=add
```

Parameters

- *`userid`*: the user ID of the registered user. It can only be specified by a user logged in at system administrator level.

Request body

A JSON object with information regarding the phone. For more information about the *Phone* JSON object, see Section 5.36, “Phone”.

Response

A User object with the device ID of the new device.

Example

Request:

```
POST /sapi/profile/device?action=add&userid=username
```

Request body:

```
{
   "data":{
      "user":{
         "phone":{
            "phonenumber":"+393333333333",
            "modelid":2883,
            "carrierid":9,
            "countrya2":"IT",
            "devicename":"My Default Phone"
         }
      }
   }
}
```

Response:

```
{
   "data":{
      "user":{
         "phone":{
            "deviceid":129
         }
      }
   }
}
```

Errors

COM-1001, COM-1002, COM-1003, COM-1007, PRO-1000, PRO-1001, PRO-1101, PRO-1105, PRO-1114, PRO-1116, PRO-1118, PRO-1120, in addition to generic errors described in Section 4.1.3, “Errors across multiple Server APIs”.

3.3.28 Update an existing device

Update an existing device. One or more pieces of the following information have to be provided:

- model ID

- phone number
- carrier ID
- country A2 code
- time zone conversion option
- active option
- device name

Security realm: user

Definition

```
POST /sapi/profile/device?action=update
```

Parameters

- *`deviceid`*: the device ID of the device to update, or the device sync ID (if updating a device name/ description).

Request body

A JSON object with information regarding the phone. For more information about the *Phone* JSON object, see Section 5.36, “Phone”.

Note

Only the JSON object properties specified in the request body will be updated. If the value is empty for a property, the value will be updated to empty. If a property is not provided in the request, the value will not be updated.

Response

The 200 HTTP code indicates a successful request.

Example 1

Request:

```
POST /sapi/profile/device?action=update&deviceid=444
```

Request body:

```
{
   "data":{
      "user":{
         "phone":{
            "phonenumber":"+393333333333",
            "active":true,
            "devicename":"My Iphone 5C"
```

```
         }
      }
   }
}
```

Example 2

Request:

```
POST /sapi/profile/device?action=update&deviceid=fac-android12345
```

Request body:

```
{
   "data":{
      "user":{
         "phone":{
            "devicename":"Updated device name"
         }
      }
   }
}
```

Errors

COM-1001, COM-1002, COM-1003, COM-1007, PRO-1000, PRO-1101, PRO-1104, PRO-1108, PRO-1114, PRO-1118, PRO-1120, PRO-1134, in addition to generic errors described in Section 4.1.3, “Errors across multiple Server APIs”.

3.3.29 Delete a device

Delete a device given the device ID.

Security realm: user

Definition

```
POST /sapi/profile/device?action=delete
```

Parameters

- *`deviceid`*: the device ID of the device to be deleted. If the logged in user is not an administrator, only its device can be deleted.
- *`userid`*: the ID of the device's owner. It can only be specified by a user logged in at system administrator level.

An administrator user must specify one of the previous parameters. A non-administrator may specify a device ID. If a non-administrator user does not specify a device ID and the Carrier Edition version supports multiple devices per user, all her/his devices will be deleted.

Response

The 200 HTTP code indicates a successful request.

Example

Request

```
POST /sapi/profile/device?action=delete&deviceid=144
```

Errors

PRO-1000, PRO-1104, PRO-1108, in addition to generic errors described in Section 4.1.3, “Errors across multiple Server APIs”.

3.3.30 Get the last sync for a user

Retrieve the last sync time for all devices associated to the given user. This API call cannot be used to determine if a synchronization is running for the user.

Security realm: user

Definition

```
GET /sapi/profile?action=get-last-sync
```

Parameters

- *`userid`*: the user to be checked. It can be specified only by an administrator.

Response

A *User* JSON object with the array of devices synchronized (times are in UTC.) For more information about the *Last Sync* JSON object, see Section 5.26, “Last sync”.

Example

Request:

```
GET /sapi/profile?action=get-last-sync&userid=username
```

Response:

```
{
   "data":{
      "lastsynchronizations":[
         {
            "deviceid":"fwm-3519800106775801",
            "devicedescription":"",
            "devicetype":"Phone",
            "endsync":1213966828971,
            "startsync":1213966804556,
            "status":200,
            "syncsource":"mail",
            "synctype":200
         },
         {
            "deviceid":"fwm-3519800106775801",
```

```
            "devicedescription":"",
            "devicetype":"Phone",
            "endsync":1213886637201,
            "startsync":1213886605823,
            "status":200,
            "syncsource":"scal",
            "synctype":200
         },
         {
            "deviceid":"fwm-3519800106775801",
            "devicedescription":"",
            "devicetype":"Phone",
            "endsync":1213886637201,
            "startsync":1213886605823,
            "status":200,
            "syncsource":"scard",
            "synctype":200
         }
      ]
   }
}
```


Errors

PRO-1000, PRO-1101, in addition to generic errors described in Section 4.1.3, “Errors across multiple Server APIs”.

3.3.31 Send a push message to a user

Push a notification message to all the devices of the given user or to the specified device only.

Security realm: system

Definition

```
POST /sapi/profile/push?action=send
```

Parameters

- *`userid`*: the user ID of a registered user. It can only be specified by a user logged in at system administrator level.
- *`deviceid`*: the device ID of the device to notify
- *`phonenumber`*: the user's phone number. Note that the phone number should be encoded as a parameter in the URL, so the "+" should be replaced with "%2B".

The *`userid`* parameter is mandatory. If *`deviceid`* is specified, only the device with the given ID (if any) is notified. If the phone number is specified, the user device will be updated and the push notification will be sent to the new phone number, in this case the *`deviceid`* parameter is mandatory in the request.

Request body

See Section 5.42, “Push notification message”

Response

The 200 HTTP code indicates a successful request.

Example 1

Request to push the sources scard and scal for all the devices of the user 'username':

```
POST /sapi/profile/push?action=send&userid=username
```

Request body:

```
{
   "data":{
      "sources":[
         {
            "uri":"scard",
            "contenttype":"text/x-s4j-sifc",
            "synctype":"206"
         },
         {
            "uri":"scal"
         }
      ],
      "uimode":"1"
   }
}
```

Example 2

Request to push the source scal for the device 'fwm-35525701681145401' of the user 'username':

```
POST /sapi/profile/push?
action=send&userid=username&deviceid=fwm-35525701681145401
```

Request body:

```
{
   "data":{
      "sources":[
         {
            "uri":"scal"
         }
      ],
      "uimode":"1"
   }
}
```

Errors

PRO-1101, PRO-2004, PRO-2005, in addition to generic errors described in Section 4.1.3, “Errors across multiple Server APIs”.

3.3.32 Get the update info for the OneMediaHub App

Retrieve all the information needed by the OneMediaHub App to auto-update.

Security realm: public

Definition

```
GET /sapi/profile/client?action=get-update-info
```

Parameters

- *`component`*: (**mandatory**) the type of the component requesting the update. Possible values are listed in Table 3.2, “Components”.

Table 3.2. Components

Type	Component
bbpim	OneMediaHub for BlackBerry
bbpim_os47	OneMediaHub for BlackBerry (OS 4.7 and 5.0)
bbpim_os60	OneMediaHub for BlackBerry (OS 6.0)
macos	OneMediaHub for Mac OS
windows	OneMediaHub for Windows

- *`format`*: (**optional**) the response format. By default, the response format is a JSON object; if the value `properties` has been specified, a properties file is returned instead of a JSON file.

Response

An update device info object containing all the information needed by an App to auto-update.

- `version`: the latest version available for the App on the repository
- `type`: the update type can be 'mandatory', 'recommended' or 'optional'
- `activationdate`: the date since when the App will be officially adopted and, if the update is mandatory, the old version will not be supported anymore
- `downloadurl`: the download URL for the App file. If the format is 'json', the `dowloadurl` is composed of `portalurl` and `downloadpath`; otherwise, if the format is 'properties', the property label is `url`.
- `size`: the size of the App file

Example

Request for the update information for a phone that supports the OneMediaHub for BlackBerry:

```
GET /sapi/profile/client?action=get-update-info&component=bbpim
```

Response:

```
{
   "data":{
      "version":"10.0.9",
      "activationdate":"20120209",
      "downloadurl":{
```

```
         "portalurl":"https://my.server.com",
         "downloadpath":"/m"
      },
      "size":"1104380",
      "type":"optional"
   }
}
```

Errors

PRO-1000, PRO-1001, PRO-1130, in addition to generic errors described in Section 4.1.3, “Errors across multiple Server APIs”.

3.3.33 Add new comment

Add a new comment providing information about a given user's customer care call.

Security realm: system

Definition

```
POST /sapi/profile/comment?action=add
```

Parameters

- *`userid`*: the user ID of the registered user

Request body

A JSON object with the comment about a user customer care call:

Name	Mandatory	Type	Value
comment	Y	String	Comment about the user customer care call. The text must have less than 4000 characters.

Response

The 200 HTTP code indicates a successful request.

Example

Request:

```
POST /sapi/profile/comment?action=add&userid=jdawson
```

Request body:

```
{
 "data":{
 "comment":"the user had an issue with the Google contacts import
 feature"
 }
}
```

Errors

PRO-1000, PRO-1001, PRO-1101, PRO-1102, PRO-1103, PRO-5101 in addition to generic errors described in Section 4.1.3, “Errors across multiple Server APIs”.

3.3.34 Retrieve comments

Retrieve comments about a user's customer care call.

Security realm: system

Definition

```
GET /sapi/profile/comment?action=get
```

Parameters

- *`userid`*: the user ID of the registered user
- *`from`*: the date (in UTC milliseconds) starting from when the comments should be retrieved

Response

An array with all the comments for the given user. If there are no comments in the database for the given user, an empty array will be returned.

Response

The 200 HTTP code indicates a successful request.

Example

Request:

```
POST /sapi/profile/comment?action=get&userid=jdawson
```

Request body:

```
{
   "data":{
      "comments":[
         {
            "id":"2501",
            "comment":"<....>",
            "creationdate":1267046824780,
            "author":"admin1"
         },
         {
            "id":"3046",
            "comment":"<....>",
            "creationdate":1267046512635,
            "author":"admin2"
         }
      ]
   }
}
```

Errors

PRO-1000, PRO-1001, PRO-1101, PRO-1102, PRO-1103, in addition to generic errors described in Section 4.1.3, “Errors across multiple Server APIs”.

3.3.35 Roles management

The OneMediaHub provides the following roles:

Table 3.3. Roles

Role	Description
sync_user	This is the role to enable a user to the sync feature (PIM and Media)
sync_administrator	This is the role for the administrator
demo	This role grants 25 MB of space for the Media sync (PIM sync is not evaluated in the quota)
standard	This role grants 50 MB of space for the Media sync (PIM sync is not evaluated in the quota)
premium	This role grants 250 MB of space for the Media sync (PIM sync is not evaluated in the quota)
premiumplus	This role grants 1 GB of space for the Media sync (PIM sync is not evaluated in the quota)
ultimate	This role grants 2 GB of space for the Media sync (PIM sync is not evaluated in the quota)

For instance, a generic user who can sync PIM and Media entities and has no pre-payed subscription will have *sync_user* and *standard* roles.

3.3.35.1 Get roles

Retrieve the list of all the available roles.

Security realm: system

Definition

```
GET /sapi/profile/role?action=roles
```

Parameters

None.

Response

A JSON object containing an array `roles` (see Section 5.45, “Role”) with all the roles available.

Example

Request:

```
GET /sapi/profile/role?action=roles
```

Response:


```
{
   "data":{
      "roles":[
         {
            "name":"demo",
            "description":""
         },
         {
            "name":"premium",
            "description":""
         },
         {
            "name":"premiumplus",
            "description":""
         },
         {
            "name":"standard",
            "description":""
         },
         {
            "name":"sync_administrator",
            "description":""
         },
         {
            "name":"sync_user",
            "description":""
         },
         {
            "name":"ultimate",
            "description":""
         }
      ]
   }
}
```


Errors

PRO-1000, in addition to generic errors described in Section 4.1.3, “Errors across multiple Server APIs”.

3.3.35.2 Get user's roles

Retrieve the list of the user's roles.

Security realm: user

Definition

```
GET /sapi/profile/role?action=get
```

Parameters

- *`userid`*: the user ID of the registered user to get info about his roles. It can be specified only by an administrator.

Response

A JSON object containing an array `roles` (see Section 5.45, “Role”) with all the roles for the given user.

Example

Request:

```
GET /sapi/profile/role?action=get&userid=username
```

Response:

```
{
   "data":{
      "roles":[
         {
            "name":"standard",
            "description":""
         },
         {
            "name":"sync_user",
            "description":""
         }
      ]
   }
}
```

Errors

PRO-1000 and PRO-1101, in addition to generic errors described in Section 4.1.3, “Errors across multiple Server APIs”.

3.3.35.3 Update user's roles

Set the roles for an existing user.

Security realm: system

Definition

```
POST /sapi/profile/role?action=save
```

Parameters

`userid`: the user ID of the registered user whose roles have to be updated. It can be specified only by an administrator.

Request body

A JSON object with a property with all the roles separated by comma.

Response

A JSON object containing an array `roles` (see Section 5.45, “Role”) with all the new roles for the given user.

Example

Request:

```
POST /sapi/profile/role?action=save&userid=username
```

Request body:

```
{
   "data":{
      "roles":[
         {
            "name":"standard",
            "description":""
         },
         {
            "name":"sync_user",
            "description":""
         }
      ]
   }
}
```

Errors

PRO-1000 and PRO-1101, in addition to generic errors described in Section 4.1.3, “Errors across multiple Server APIs”

3.3.36 Handling properties

In some cases, a UI may need to store some custom properties that are not covered by any specific API (since it is possible to have many different UIs on top of SAPI).

To cover such specific cases, this API stores key-value pairs that are sent from client to server and that can be at any time retrieved, removed or updated.

In order to handle different parts of the UI, it may be useful to enforce a naming convention, such as for example: `<component>.<property>` (e.g. `calendar.startdayofweek`)

3.3.36.1 Get properties

Return all the key-value pairs stored for this user.

Security realm: user

Definition

```
POST /sapi/profile/properties?action=get
```

Parameters

- *`userid`*: the user ID of the account where the properties are stored. It can be specified only by an administrator.

Request body

See Section 5.40, “Properties (strings)”. The request body is optional. It should indicate the properties to return, if not present all properties should be returned.

Response

An object with the key-value pairs requested

Example

Request:

```
POST /sapi/profile/properties?action=get
```

Request body:

```
{
 "data":{
   "properties":[
     "calendar.timeformat",
     "calendar.firstdayofweek"]
  }
}
```

Response:

```
{
 "data":{
   "properties":[
     {"name":"calendar.timeformat","value":"hh:mm"},
     {"name":"calendar.firstdayofweek ","value":"MO"}]
   }
}
```

Errors

PRO-1000, in addition to generic errors described in Section 4.1.3, “Errors across multiple Server APIs”

3.3.36.2 Add/update properties

Add/update the key-value pairs for this user. If any of the specified properties is not present, then it is added; otherwise, it is updated.

Security realm: user

Definition

```
POST /sapi/profile/properties?action=set
```

Parameters

- *`userid`*: the user ID of the user owner of the account where the properties are stored. It can be specified only by an administrator.

Request body

See Section 5.39, “Properties (JSON objects)”. The request body must contain an array of JSON objects each representing a key-value pair.

Response

A HTTP 200 code if completed successfully.

Example

Request:

```
POST /sapi/profile/properties?action=set
```

Request body:

```
{
   "data":{
      "properties":[
         {
            "name":"calendar.timeformat",
            "value":"hh:mm"
         },
         {
            "name":"calendar.weekstartson",
            "value":"MO"
         },
         {
            "name":"calendar.startofworkday",
            "value":"MO"
         }
      ]
   }
}
```

Errors

PRO-1000, in addition to generic errors described in Section 4.1.3, “Errors across multiple Server APIs”

3.3.36.3 Remove properties

Remove a key-value pair for the specified user.

Security realm: user

Definition

```
POST /sapi/profile/properties?action=remove
```

Parameters

- *`userid`*: the user ID of the user owner of the account where the properties are stored. It can be specified only by an administrator.

Request body

See Section 5.40, “Properties (strings)”. The request body is required. It must indicate the properties to remove.

Response

A HTTP 200 code if completed successfully.

Example

Request:

```
POST /sapi/profile/properties?action=remove
```

Request body:

```
{
   "data":{
      "properties":[
         "calendar.timeformat"
      ]
   }
}
```

Errors

PRO-1000, in addition to generic errors described in Section 4.1.3, “Errors across multiple Server APIs”

3.3.36.4 Remove all properties

Remove all key-value pairs for the specified user.

Security realm: user

Definition

```
POST /sapi/profile/properties?action=removeall
```

Parameters

- *`userid`*: the user ID of the user owner of the account where the properties are stored. It can be specified only by an administrator.

Request body

None.

Response

A HTTP 200 code if completed successfully.

Example

Request:

```
POST /sapi/profile/properties?action=removeall
```

Errors

PRO-1000, in addition to generic errors described in Section 4.1.3, “Errors across multiple Server APIs”

3.3.37 Register cloud push token

Register a token of a cloud messaging service (like Apple's remote push notification service) for a given device.

Security realm: user

Definition

```
POST /sapi/profile/device/token?action=save
```

Parameters

- *deviceid*: (**mandatory**) the device ID of the device for which the token should be stored.

Request body

```
{
   "data":{

 "token":"af8d4430355965da1db893d634268535e97a7ffef95182eef4723e6c5f2d18
07"
   }
}
```

Response

```
{
   "data":{
      "success":"token saved successfully"
   }
}
```

Errors

COM-1005, COM-1011, PRO-1104 ,PRO-1000, in addition to generic errors described in Section 4.1.3, “Errors across multiple Server APIs”.

3.3.38 Get OMA SAN push message for a user

Return an array of SAN messages. Each message is related to one of the devices associated with the given user.

Security realm: system

Definition

```
POST /sapi/profile/push?action=getsan
```

Parameters

- *userid*: (**mandatory**) the user ID of a registered user.

Request body

A JSON object containing the push information about the sync sources to be pushed and the server recommendations as described in Section 5.41, “Push info”.

Response

The response contains an array of SAN messages. Every element of the array is related to a given device ID and the actual message is a Base64-encoded string of the SAN binary message:

```
{
   "data":{
      "messages":[
         {
            "deviceid":"IMEI:123ASDF123",
            "san":"......"
         },
         {
            "deviceid":"fwm-12312SDSDSU4354",
            "san":"......."
         }
      ]
   }
}
```

Example

Request to push the sync sources `card` and `cal` for all the devices of the user `username`:

```
POST /sapi/profile/push?action=getsan&userid=username
```

Request body

```
{
   "data":{
      "sources":[
         {
            "uri":"card",
            "contenttype":"text/vcard",
            "synctype":"206"
         },
         {
            "uri":"cal",
            "contenttype":"text/vcal",
            "synctype":"206"
         }
      ],
      "uimode":"1"
   }
}
```

Response body

```
{
   "data":{
```

```
      "messages":[
         {
            "deviceid":"IMEI:123ASDF123",

 "san":"QcYwG/4rylShgt5C1z5prwMYAAAAAAAIRnVuYW1ib2wgYAAAAARjYXJkYAAAA
ANjYWw="
         }
      ]
   }
}
```

Errors

PRO-1101, PRO-2004, PRO-2005, PRO-2006, in addition to generic errors described in Section 4.1.3, “Errors across multiple Server APIs”.

3.4 Subscription plans

This set of APIs is used to manage the user's subscription plan. A subscription plan entitles a user to benefit of an amount of storage within a certain period of time.

3.4.1 Get available subscription plans

Return the list of subscription plans available for the user that belong to the user's current subscription family.

Each subscription plan is assigned to a payment type. A payment type defines the way a payment for a subscription is handled. Clients should only request the list of subscription plans according to the payment type they support - so, for example, an iOS client should only request plans of payment type `apple`.

Security realm: user

Definition

```
GET /sapi/subscription/plan?action=get
```

Parameters

- *`paymenttype`*: the payment type of the subscription plans - if no value is specified, the server does use the payment type `default`
- *`name`*: the name of a subscription plan - if this parameter is used, *`paymenttype`* is ignored

Response

A JSON object containing the subscription plans for the current user (see Section 5.54, “Subscription plan”):

```
{
   "data":{
      "plans":[
         {
            "name":"standard",
            "price":"5",
```

```
            "currency":"USD",
            "period":"month",
            "quota":"1GB",
            "default":false,
            "displayname":"Standard",
            "description":"1GB for 5$ a month",
            "activateable":"false",
            "message":"You cannot downgrade to this plan because your
 quota exceeds the plan's",
            "messagecode":"NA-002"
         }
      ]
   }
}
```

Errors

SUB-1000, in addition to generic errors described in Section 4.1.3, “Errors across multiple Server APIs”.

3.4.2 Get subscription families

Return complete information about the subscription families defined in the system, with details about the subscription plans belonging to them.

Security realm: user

Definition

```
POST /sapi/subscription/families?action=get
```

Parameters

- *name*: (**optional**) the name of a subscription family

Response

```
{
   "data":{
      "families":[
         {
            "name":"default",
            "displayname":"Default Family",
            "plans":[
               {
                  "name":"standard",
                  "price":5,
                  "currency":"USD",
                  "period":"month",
                  "quota":"1GB",
                  "displayname":"Standard",
                  "description":"1GB for 5$ a month",
                  "activatable":false,
                  "message":"Your currently used storage exceeds 1GB"
               },
```

```
                ...
            ]
        }
    ]
  }
}
```

Errors

SUB-1000, in addition to generic errors described in Section 4.1.3, “Errors across multiple Server APIs”.

3.4.3 Save subscription plan

Change the subscription plan or the status of the current plan of the user.

> **Note**
>
> Being the procedure asynchronous, the new subscription plan might or might not be immediately active, migrated, or canceled after this API has been called.

Security realm: user

Definition

```
POST /sapi/subscription?action=save
```

Parameters

- *`userid`*: the user ID of the user whose subscription plan has to be saved. It can be specified only by an administrator.

Request body

A *Subscription request* JSON object (see Section 5.51, “Subscription request”) with information regarding the subscription plan and its status.

Response

A JSON object containing the current subscription plan (see Section 5.52, “Subscription response”).

Example 1

The current user's subscription plan should be migrated to the "premium" plan:

Request body

```
{
   "data":{
      "subscription":{
         "plan":"premium"
      }
   }
}
```

Response

```
{
   "data":{
      "subscription":{
         "plan":"standard",
         "activated":"20130221T163755",
         "nextrenewal":"20130321T163755",
         "laststatuschange":"20130221T163934",
         "migratetoplan":"premium",
         "status":"active"
      }
   }
}
```

Example 2

The current "standard" subscription plan should be canceled:

Request body

```
{
   "data":{
      "subscription":{
         "plan":"standard",
         "status":"canceled"
      }
   }
}
```

Response

```
{
   "data": {
       "subscription": {
          "plan": "standard",
          "activated": "20130221T163755",
          "nextrenewal": "20130321T163755",
          "laststatuschange": "20130221T163934",
          "status": "canceled",
          "deletedate": "20130404T163755"
       }
    }
}
```

Example 3

Set an expiration date to the subscription plan of the user ("standard" plan), when invoking the API from a *System* security realm (see Note):

Request body

```
{
   "data":{
      "subscription":{
```

```
         "plan":"standard",
         "expiredate":"20150331T113527"
      }
   }
}
```

Response

```
{
   "data":{
      "subscription":{
         "plan":"standard",
         "activated":"20130221T163755",
         "nextrenewal":"20130321T163755",
         "laststatuschange":"20130221T163934",
         "status":"active",
         "expiredate":"20150331T113527"
      }
   }
}
```

Example 4

Remove expiration date from the subscription plan of the user ("standard" plan), when invoking the API from a *System* security realm (see Note):

Request body

```
{
   "data":{
      "subscription":{
         "plan":"standard",
         "expiredate":""
      }
   }
}
```

Response

```
{
   "data":{
      "subscription":{
         "plan":"standard",
         "activated":"20130221T163755",
         "nextrenewal":"20130321T163755",
         "laststatuschange":"20130221T163934",
         "status":"active"
      }
   }
}
```

Errors

SUB-1000, SUB-1001, SUB-1003, SUB-1004, SUB-1005, SUB-1006 in addition to generic errors described in Section 4.1.3, “Errors across multiple Server APIs”.

3.4.4 Get current subscription

Return information about the status of the currently active subscription plan.

Security realm: user

Definition

```
GET /sapi/subscription?action=get
```

Parameters

None.

Response

As per following examples.

Warning

The dates are exposed in local time, but they are created by the server, so they are in UTC time.

Example 1

the subscription is active:

```
{
   "data":{
      "subscription":{
         "plan":"standard",
         "activated":"20120430T084622",
         "nextrenewal":"20120430T093122",
         "laststatuschange":"20120430T084622",
         "status":"active"
      }
   }
}
```

Example 2

the subscription should be migrated to the premium plan:

```
{
   "data":{
      "subscription":{
         "plan":"standard",
         "activated":"20120430T084622",
         "nextrenewal":"20120430T093122",
         "laststatuschange":"20120430T084622",
         "migratetoplan":"premium",
         "status":"active"
      }
   }
```

```
}
```

Example 3

the subscription is canceled:

```
{
    "data": {
       "subscription": {
          "plan": "standard",
          "activated": "20130221T163755",
          "nextrenewal": "20130321T163755",
          "laststatuschange": "20130221T163934",
          "status": "canceled",
          "deletedate": "20130404T163755"
       }
    }
}
```

Example 4

The subscription is active and has an expiration date

```
{
   "data":{
      "subscription":{
         "plan":"standard",
         "activated":"20120430T084622",
         "nextrenewal":"20120430T093122",
         "laststatuschange":"20120430T084622",
         "status":"active",
         "expiredate":"20150331T113527"
      }
   }
}
```

Statuses

The value of the *`status`* parameter can be one of the following:

- `active`
- `canceled`
- `migrated` (the user has been migrated from a subscription to an other one; the previous subscription is set as `migrated` and is no longer active. This status is used for retrieving the history of the user's subscriptions
- `payment_required`
- `unknown` (returned in case of error, when the status of the subscription is undefined for some reason)

Errors

SUB-1000, SUB-1002, in addition to generic errors described in Section 4.1.3, “Errors across multiple Server APIs”.

3.4.5 Cancel current subscription

Cancel the current subscription plan of the user.

Security realm: user

Definition

```
POST /sapi/subscription?action=cancel
```

Parameters

None.

Response

The 200 HTTP code indicates a successful request.

Important

The subscription plan might or might not be canceled immediately after this API has been called.

Errors

SUB-1000, in addition to generic errors described in Section 4.1.3, “Errors across multiple Server APIs”.

3.4.6 Get subscriptions history

Return information about the history of the subscriptions for the user.

Security realm: user

Definition

```
GET /sapi/subscription/history?action=get
```

Parameters

None.

Response

```
{
    "data": {
       "subscriptions": [
           {
              "plan": "demo",
              "activated": "20130222T081203",
              "nextrenewal": "21130222T081203",
              "laststatuschange": "20130222T081355",
              "migratetoplan": "premium",
              "status": "migrated"
           },
```

```
            {
               "plan": "premium",
               "activated": "20130222T081203",
               "nextrenewal": "20130322T081203",
               "laststatuschange": "20130222T081355",
               "status": "active"
            },
            {
               "plan": "premium",
               "activated": "20130221T163755",
               "nextrenewal": "20130321T163755",
               "laststatuschange": "20130221T163934",
               "status": "canceled",
               "deletedate": "20130404T163755"
            }
         ]
   }
}
```

Statuses

The value of the *`status`* parameter can be one of the following:

- `active`
- `canceled`
- `migrated`
- `suspended`
- `payment_required`
- `unknown`

Errors

SUB-1000, in addition to generic errors described in Section 4.1.3, “Errors across multiple Server APIs”.

3.4.7 Register a client side payment

Allow clients - such as an iPhone - to register payments handled by the client itself. The server verifies each payment and will activate the purchased subscription only if the verification succeeds.

Security realm: user

Definition

```
POST /sapi/subscription/payment?action=save
```

Parameters

- *`name`*: the name of the purchased subscription plan
- *`transactionid`*: a string containing the information necessary for the server to verify the payment - for iOS this is the Base64-encoded purchase receipt.

Response

The 200 HTTP status code indicates a successful request.

Warning

A successful request does not mean that the server verified the payment - only that it successfully registered the payment.

Errors

SUB-1000, in addition to generic errors described in Section 4.1.3, “Errors across multiple Server APIs”.

3.4.8 Get the list of payments

Return the list of registered payments for the user.

Security realm: user

Definition

```
GET /sapi/subscription/payment?action=get
```

Response

```
{
   "data":{
      "plans":[
         {
            "id":1,
            "plan":"standard",
            "status":"verified",
            "transactionid":"XXXXXX"
         },
         {
            "id":2,
            "plan":"premium",
            "status":"new",
            "transactionid":"YYYYY"
         }
      ]
   }
}
```

Errors

SUB-1000, in addition to generic errors described in Section 4.1.3, “Errors across multiple Server APIs”.

3.5 Media

These API calls are intended to manage media content like pictures, videos, audio, and other files.

Media functions allow to retrieve, delete, reset, refresh, count, download, update, rotate the thumbnails and get information like metadata about media for the user.

Note

The media URLs returned by the API calls should not be cached, since they might be only temporarily valid.

3.5.1 Get storage space information

Return the free storage space, used soft-deleted space, and storage space available quota for the user.

Security realm: user

Definition

```
GET /sapi/media?action=get-storage-space
```

Parameters

- *`userid`*: (**optional**) the ID of the registered user. It can be specified only by an administrator.
- *`softdeleted`*: (**optional**) if `true`, the used soft-deleted space is included in the response.

Response

The server will return a data object with fields for *quota* and *free space* in bytes which includes all media types.

Example 1

Request

```
GET /sapi/media?action=get-storage-space
```

Response

```
{
   "data":{
      "free":534123,
      "quota":1024000
   }
}
```

Example 2

Request

```
GET /sapi/media?action=get-storage-space?softdeleted=true
```

Response

```
{
   "data":{
      "free":534123,
      "softdeleted":12077,
      "quota":1024000
```

```
    }
}
```

Errors

PIC-1000, in addition to generic errors described in Section 4.1.3, “Errors across multiple Server APIs”.

3.5.2 Retrieve pictures

Retrieve pictures for the user.

Security realm: user

Definition

```
GET /sapi/media/picture?action=get
```

Parameters

- *`id`*: (**optional**) the ID of the picture or a JSON array with IDs to be retrieved. If an array of IDs is specified, the items will be returned in the same order they were requested.
- *`from`*: the date (in UTC milliseconds) starting from when the pictures should be retrieved
- *`limit`*: (**optional**) the maximum number of pictures retrieved starting from the latest ones and considering the offset. Cannot be used with the parameter *`id`*.
- *`offset`*: (**optional**) the offset (inclusive) of pictures to be considered in the response. Cannot be used with the parameter *`id`*.
- *`exported`*: if the value is `true`, then for every picture is returned information to be shared on external services.
- *`exif`*: it can be `none` or `all`. Default is `all`.
- *`sortby`*: (**optional**) specifies the field to use to order returned pictures. Allowed values are `date` (date of the last update), and `id`.
- *`sortorder`*: (**optional**) the order in which the *`sortby`* parameter is applied. Allowed values are `ascending` and `descending`. If a value for *`sortby`* is supplied and no *`sortorder`* is specified, the *`sortorder`* will default to `ascending`.
- *`shared`*: (**optional**) if the value is `true`, then a flag named `shared` will be returned for every picture, which is `true` if the picture is part of a media set. Cannot be used with the *`softdelete`* and *`onlysoftdelete`* parameters. Default value is `false`.
- *`favorite`*: (**optional**) if `true`, then only pictures previously added to the favorite pictures list will be returned. If not specified, pictures will be returned wether they are favorite or they are not.
- *`tag`*: (**optional**) if specified, defines the tag pictures should have assigned to be included in the results
- *`geoboundary`*: (**optional**) specifies a rectangle within which pictures have to be located. The boundary has to be of the format `ul-lat,ul-lon;lr-lat,lr-lon`.
  - `ul-lat` specifies the **latitude** of the **upper left corner** in degrees (`-90` to `90`)
  - `ul-lon` specifies the **longitude** of the **upper left corner** in degrees (`-180` to `180`)

- `lr-lat` specifies the **latitude** of the **lower right corner** in degrees (`-90` to `90`)
- `lr-lon` specifies the **longitude** of the **lower right corner** in degrees (`-180` to `180`)

All values will be truncated at 6 decimals. If *`geoboundary`* is not specified, pictures will be returned wether they have a geolocation assigned to or not. Only pictures which have the following Exif GPS attributes will be included in a search using the *`geoboundary`* parameter:

- `GPSLatitudeREF` (North or South latitude)
- `GPSLatitude`
- `GPSLongitudeREF` (East or West longitude)
- `GPSLongitude`

- *`softdelete`*: (**optional**) if `true`, pictures marked as soft deleted will be included in the response. Cannot be used with the parameters *`shared`* and *`onlysoftdelete`*. Default is `false`.
- *`onlysoftdelete`*: (**optional**) if `true`, only pictures marked as soft deleted will be included in the response. Cannot be used with the parameters *`shared`* and *`softdelete`*. Default is `false`.

IDs example:

```
id={"ids":[12,23,456]}
```

and, as encoded parameter:

```
id=%7B%22ids%22%3A%5B12%2C23%2C456%5D%7D
```

Response

The server URL where the pictures are stored (*`mediaserverurl`*) and an array with all the picture objects for the given user; or, if the *`limit`* parameter is specified, the latest *`limit`* for pictures (if there is at least a *`limit`* or a subset of a *`limit`* for pictures) from a given offset, if the *`offset`* parameter is also specified. If the *`from`* parameter is used at the same time as *`limit`* and *`offset`*, the pictures are first filtered by the *`from`* parameter, and only after that, the *`offset`* and *`limit`* are applied. If the *`softdelete`* parameter is `true`, both soft deleted and non soft deleted pictures are included. If the *`onlysoftdelete`* parameter is `true`, only soft deleted pictures are returned.If there are no pictures in the database for the given user or the offset is greater than the number of pictures, an empty array will be returned. For more information about the Picture JSON object, see Section 5.37, “Picture”.

Example code 1

```
GET /sapi/media/picture?action=get&limit=3
```

```
{
   "data":{
      "mediaserverurl":"https://my.server.com",
      "pictures":[
         {
            "id":"0",
            "url":"/picture/uq/rf/xo/fu/um/dq/ki/le/
dGVzdA==/1ugzzekm3nhrf-0.jpg",
            "viewurl":"/picture/uq/rf/xo/fu/um/dq/ki/le/
dGVzdA==/1ugzzekm3nhrf-0.jpg-ext/504.jpg",
```

```
            "date":1399985134672,
            "creationdate":1072958456000,
            "modificationdate":1161931347000,
            "size":722942,
            "labels":[
               {
                  "labelid":0,
                  "name":"US trip"
               }
            ],
            "name":"100_1924.jpg",
            "exif":{
               "Make":"'EASTMAN KODAK COMPANY'",
               "Model":"'KODAK CX7525 ZOOM DIGITAL CAMERA'",
               "Orientation":"1",
               "XResolution":"72",
               "Date Time Original":"'2004:01:01 12:00:56'",
               "Create Date":"'2004:01:01 12:00:56'",
               "Exif Image Width":"1920",
               "Exif Image Length":"2560"
            },
            "thumbnails":[
               {
                  "w":192,
                  "h":256,
                  "size":"176",
                  "url":"/picture/uq/rf/xo/fu/um/dq/ki/le/
dGVzdA==/1ugzzekm3nhrf-0.jpg-ext/176.jpg",
                  "etag":"6W9kgBeRnzrDNH47tgD0qQ=="
               },
               {
                  "w":576,
                  "h":768,
                  "size":"504",
                  "url":"/picture/uq/rf/xo/fu/um/dq/ki/le/
dGVzdA==/1ugzzekm3nhrf-0.jpg-ext/504.jpg",
                  "etag":"6W9kgBeRnzrDNH47tgD0qQ=="
               }
            ],
            "mediatype":"picture",
            "status":"U",
            "etag":"6W9kgBeRnzrDNH47tgD0qQ==",
            "folder":1,
            "softdeleted":false
         }
      ]
   }
}
```


Example code 2

```
GET /sapi/media/picture?action=get&exported=true
```


```
{
```

```
   "data":{
      "mediaserverurl":"https://my.server.com",
      "pictures":[
         {
            "id":"0",
            "url":"/picture/uq/rf/xo/fu/um/dq/ki/le/
dGVzdA==/1ugzzekm3nhrf-0.jpg",
            "viewurl":"/picture/uq/rf/xo/fu/um/dq/ki/le/
dGVzdA==/1ugzzekm3nhrf-0.jpg-ext/504.jpg",
            "date":1399985134672,
            "creationdate":1072958456000,
            "modificationdate":1161931347000,
            "size":722942,
            "labels":[
               {
                  "labelid":0,
                  "name":"US trip"
               }
            ],
            "name":"100_1924.jpg",
            "exif":{
               "Make":"'EASTMAN KODAK COMPANY'",
               "Model":"'KODAK CX7525 ZOOM DIGITAL CAMERA'",
               "Orientation":"1",
               "XResolution":"72",
               "Date Time Original":"'2004:01:01 12:00:56'",
               "Create Date":"'2004:01:01 12:00:56'",
               "Exif Image Width":"1920",
               "Exif Image Length":"2560"
            },
            "thumbnails":[
               {
                  "w":192,
                  "h":256,
                  "size":"176",
                  "url":"/picture/uq/rf/xo/fu/um/dq/ki/le/
dGVzdA==/1ugzzekm3nhrf-0.jpg-ext/176.jpg",
                  "etag":"6W9kgBeRnzrDNH47tgD0qQ=="
               },
               {
                  "w":576,
                  "h":768,
                  "size":"504",
                  "url":"/picture/uq/rf/xo/fu/um/dq/ki/le/
dGVzdA==/1ugzzekm3nhrf-0.jpg-ext/504.jpg",
                  "etag":"6W9kgBeRnzrDNH47tgD0qQ=="
               }
            ],
            "exported":{
               "picasa":{
                  "exporttime":1274192165
               },
               "flickr":{
                  "exporttime":1274345544
```

```
            }
         },
         "mediatype":"picture",
         "status":"U",
         "etag":"6W9kgBeRnzrDNH47tgD0qQ==",
         "folder":1,
         "softdeleted":false
      }
   ]
  }
}
```

Example code 3 (with encryption)

```
GET /sapi/media/picture?action=get&exported=true
```

```
{
   "data":{
      "mediaserverurl":"https://my.server.com",
      "pictures":[
         {
            "id":"2501",
            "url":"/sapi/download/picture?
action=get&filename=03nyna3a8cy5d",
            "viewurl":"/sapi/download/picture?
action=get&filename=/03nyna3a8cy5d-ext/504.jpg",
            "date":1267046824780,
            "size":1325816,
            "name":"smalix.jpg",
            "thumbnails":[
               {
                  "w":176,
                  "h":132,
                  "size":"176",
                  "url":"/sapi/download/picture?
action=get&filename=/03nyna3a8cy5d-ext/176.jpg",
                  "etag":"180aafqzchtgdvLsw7zhifZKg=="
               },
               {
                  "w":504,
                  "h":378,
                  "size":"504",
                  "url":"/sapi/download/picture?
action=get&filename=/03nyna3a8cy5d-ext/504.jpg",
                  "etag":"180aafqzchtgdvLsw7zhifZKg=="
               }
            ],
            "exif":{
               "Make":"'HTC'",
               "Model":"'HTC Touch 3G T3232'",
               "Orientation":"1",
               "XResolution":"72",
               "Create Date":"'2009: 01: 10 22: 37: 54'"
            },
```

```
            "exported":{
               "picasa":{
                  "exporttime":1274192165
               },
               "flickr":{
                  "exporttime":1274345544
               }
            },
            "mediatype":"picture",
            "status":"U",
            "etag":"aafqzchtgdvLsw7zhifZKg==",
            "folder":1,
            "softdeleted":false
         }
      ]
   }
}
```

Example code 4

```
GET /sapi/media/picture?
action=get&geoboundary=37.566535,126.977969;37.18439,127.20108&favori
te=true
```

The previous request would return favorite pictures taken in an area covering Seoul.

Example code 5 (soft deleted)

```
GET /sapi/media/picture?action=get&softdelete=true
```

```
{
   "data":{
      "mediaserverurl":"https://my.server.com",
      "pictures":[
         {
            "id":"0",
            "url":"/picture/uq/rf/xo/fu/um/dq/ki/le/
dGVzdA==/1ugzzekm3nhrf-0.jpg",
            "viewurl":"/picture/uq/rf/xo/fu/um/dq/ki/le/
dGVzdA==/1ugzzekm3nhrf-0.jpg-ext/504.jpg",
            "date":1399985134672,
            "creationdate":1072958456000,
            "modificationdate":1161931347000,
            "size":722942,
            "labels":[
               {
                  "labelid":0,
                  "name":"US trip"
               }
            ],
            "name":"100_1924.jpg",
            "exif":{
               "Make":"'EASTMAN KODAK COMPANY'",
               "Model":"'KODAK CX7525 ZOOM DIGITAL CAMERA'",
```

```
                "Orientation":"1",
                "XResolution":"72",
                "Date Time Original":"'2004:01:01 12:00:56'",
                "Create Date":"'2004:01:01 12:00:56'",
                "Exif Image Width":"1920",
                "Exif Image Length":"2560"
             },
             "thumbnails":[
                {
                   "w":192,
                   "h":256,
                   "size":"176",
                   "url":"/picture/uq/rf/xo/fu/um/dq/ki/le/
dGVzdA==/1ugzzekm3nhrf-0.jpg-ext/176.jpg",
                   "etag":"6W9kgBeRnzrDNH47tgD0qQ=="
                },
                {
                   "w":576,
                   "h":768,
                   "size":"504",
                   "url":"/picture/uq/rf/xo/fu/um/dq/ki/le/
dGVzdA==/1ugzzekm3nhrf-0.jpg-ext/504.jpg",
                   "etag":"6W9kgBeRnzrDNH47tgD0qQ=="
                }
             ],
             "mediatype":"picture",
             "status":"U",
             "etag":"6W9kgBeRnzrDNH47tgD0qQ==",
             "softdeleted":true
          }
       ]
    }
}
```


Errors

COM-1008, MED-1016, MED-1017, MED-1021, PIC-1003, PIC-1005, in addition to generic errors described in Section 4.1.3, “Errors across multiple Server APIs”.

3.5.3 Delete pictures

Delete pictures for the user.

Security realm: user

Definition

```
POST /sapi/media/picture?action=delete
```

Parameters

- *`softdelete`*: (**optional**): if `true`, the pictures will be marked as soft deleted, that is, in the trash. Default is `false`.

Request body

- *`data`*: the JSON object containing an array named *`pictures`* with the IDs of the pictures to be deleted.

Example:

```
{
    "data":{
       "pictures":[23508,23816]
    }
}
```

Response

The 200 HTTP code indicates a successful request.

Errors

COM-1013, MED-1016, MED-1022, PIC-1000, in addition to generic errors described in Section 4.1.3, “Errors across multiple Server APIs”.

3.5.4 Reset pictures

Delete all the pictures stored on the media storage provider.

Note

If a user performs a call to this API, no items will be returned in a subsequent call of the API described at Section 3.3.1, “Get changes”

Security realm: user

Definition

```
POST /sapi/media/picture?action=reset
```

Response

The 200 HTTP code indicates a successful request.

Errors

MED-1000 and MED-1014, in addition to generic errors described in Section 4.1.3, “Errors across multiple Server APIs”.

3.5.5 Count the pictures

Return the number of pictures for the user.

Security realm: user

Definition

```
GET /sapi/media/picture?action=count
```

Parameters

- *`onlysoftdelete`*: (**optional**): if `true`, the number of pictures marked as soft deleted will be returned.

Response

The number of pictures for the user.

Example:

```
{
    "data":{
       "success":"Count returned successfully",
        "count":30
    }
}
```

Errors

PIC-1000, in addition to generic errors described in Section 4.1.3, “Errors across multiple Server APIs”.

3.5.6 Add pictures to the list of favorites

Add a picture or a list of pictures to the list of favorite pictures. See Section 3.5.2, “Retrieve pictures” on how to get a list of favorite pictures.

Security realm: user

Definition

```
POST /sapi/media/picture/favorite?action=add
```

Example for the Request body

```
{
   "data":{
      "pictures":[
         1,
         2,
         3
      ]
   }
}
```

Errors

Generic errors described in Section 4.1.3, “Errors across multiple Server APIs”.

3.5.7 Remove pictures from the list of favorites

Remove a picture or a list of pictures from the list of favorite pictures. See Section 3.5.2, “Retrieve pictures” on how to get a list of favorite pictures.

Security realm: user

Definition

```
POST /sapi/media/picture/favorite?action=remove
```

Example for the Request body

```
{
   "data":{
      "pictures":[
         1,
         2,
         3
      ]
   }
}
```

Errors

Generic errors described in Section 4.1.3, “Errors across multiple Server APIs”.

3.5.8 Retrieve videos

Retrieve videos for the user.

Security realm: user

Definition

```
GET /sapi/media/video?action=get
```

Parameters

- *`id`*: (**optional**) the ID of the video or a JSON array of IDs to be retrieved. If an array of IDs is specified, the items will be returned in the same order they were requested.
- *`from`*: the date (in UTC milliseconds) starting from when the videos should be retrieved
- *`limit`*: (**optional**) the maximum number of videos retrieved starting from the latest ones and considering the offset. Cannot be used with the *`id`* parameter.
- *`offset`*: (**optional**) the offset (inclusive) of videos to be considered in the response. Cannot be used with the *`id`* parameter.
- *`exported`*: if the value is `true`, then for every video, information is returned to be shared on external services.
- *`sortby`*: (**optional**) specifies the field to use to order returned videos. Allowed values are `date` (date of the last update), and `id`.
- *`sortorder`*: (**optional**) the order in which the *`sortby`* parameter is applied. Allowed values are `ascending` and `descending`. If a value for *`sortby`* is supplied and no *`sortorder`* is specified, the *`sortorder`* will default to `ascending`.

- *shared*: (**optional**) if the value is true, then a flag named shared will be returned for every video, which is true if the video is part of a media set. Cannot be used with the *softdelete* and *onlysoftdelete* parameters. Default value is false.
- *metadata*: can be none or all. Default is all.
- *softdelete*: (**optional**) if true, videos marked as soft deleted will be included in the response. Cannot be used with the parameters *shared* and *onlysoftdelete*. Default is false.
- *onlysoftdelete*: (**optional**) if true, only videos marked as soft deleted will be included in the response. Cannot be used with the parameters *shared* and *softdelete*. Default is false.

IDs example:

```
id={"ids":[12,23,456]}
```

and, as encoded parameter:

```
id=%7B%22ids%22%3A%5B12%2C23%2C456%5D%7D
```

Response

The server URL where the videos are stored (*mediaserverurl*) and an array with all the video objects for the given user; or, if the limit parameter is specified, the latest limit for videos (if there is at least a limit or a subset of a limit for videos) from a given offset, if the offset parameter is also specified. If the from parameter is used at the same time as limit and offset, the videos are first filtered by the from parameter, and only after that, the offset and limit are applied. If the *softdelete* parameter is true, both soft deleted and non soft deleted videos are included. If the *onlysoftdelete* parameter is true, only soft deleted videos are returned. If there are no videos in the database for the given user or the offset is greater than the number of videos, an empty array will be returned. For more information about the Video JSON object, see Section 5.62, “Video”

Example code 1

```
GET /sapi/media/video?action=get&limit=3
```

```
{
   "data":{
      "mediaserverurl":"https://my.server.com",
      "videos":[
         {
            "id":"479",
            "url":"/video/uq/rf/xo/fu/um/dq/ki/le/
dGVzdA==/0yqm54u76daw7-479.mov",
            "viewurl":"/video/uq/rf/xo/fu/um/dq/ki/le/
dGVzdA==/0yqm54u76daw7-479.mov-ext/504.jpg",
            "date":1399994605688,
            "creationdate":1341997020000,
            "modificationdate":1226439062000,
            "size":2574169,
            "labels":[
               {
                  "labelid":100,
                  "name":"US trip"
               }
            ],
```

```
            "name":"100_3601.mov",
            "metadata":{
               "duration":12620,
               "bitrate":1631759,
               "codec":"mpeg4",
               "height":480,
               "width":640
            },
            "playbackurl":"",
            "thumbnails":[
               {
                  "w":176,
                  "h":132,
                  "size":"176",
                  "url":"/video/uq/rf/xo/fu/um/dq/ki/le/
dGVzdA==/0yqm54u76daw7-479.mov-ext/176.jpg",
                  "etag":"/Wv3khtmDYmoGq5D4bHE1w=="
               },
               {
                  "w":504,
                  "h":378,
                  "size":"504",
                  "url":"/video/uq/rf/xo/fu/um/dq/ki/le/
dGVzdA==/0yqm54u76daw7-479.mov-ext/504.jpg",
                  "etag":"/Wv3khtmDYmoGq5D4bHE1w=="
               }
            ],
            "mediatype":"video",
            "status":"U",
            "etag":"/Wv3khtmDYmoGq5D4bHE1w==",
            "softdeleted":false
         },
         {
            "id":"466",
            "url":"/video/uq/rf/xo/fu/um/dq/ki/le/
dGVzdA==/1dvhmkzjcbmtv-466.MOV",
            "viewurl":"/video/uq/rf/xo/fu/um/dq/ki/le/
dGVzdA==/1dvhmkzjcbmtv-466.MOV-ext/504.jpg",
            "date":1399994344920,
            "creationdate":1341997020000,
            "modificationdate":1225663414000,
            "size":138494,
            "name":"100_3248.MOV",
            "metadata":{
               "duration":476,
               "bitrate":2326813,
               "codec":"mpeg4",
               "height":480,
               "width":640
            },
            "playbackurl":"",
            "thumbnails":[
               {
                  "w":176,
```

```
                  "h":132,
                  "size":"176",
                  "url":"/video/uq/rf/xo/fu/um/dq/ki/le/
dGVzdA==/1dvhmkzjcbmtv-466.MOV-ext/176.jpg",
                  "etag":"dFmWIS3PWlyMVCPx2p2Q4w=="
               },
               {
                  "w":504,
                  "h":378,
                  "size":"504",
                  "url":"/video/uq/rf/xo/fu/um/dq/ki/le/
dGVzdA==/1dvhmkzjcbmtv-466.MOV-ext/504.jpg",
                  "etag":"dFmWIS3PWlyMVCPx2p2Q4w=="
               }
            ],
            "mediatype":"video",
            "status":"U",
            "etag":"dFmWIS3PWlyMVCPx2p2Q4w==",
            "folder":12,
            "softdeleted":false
         }
      ]
   }
}
```


Example code 2

```
GET /sapi/media/video?action=get&exported=true
```


```
{
   "data":{
      "mediaserverurl":"https://my.server.com",
      "videos":[
         {
            "id":"0",
            "url":"/video/af/ec/mm/od/ap/bm/ci/if/
bHVpZ2lh/0lfj9ach9mkdp-0.MOV",
            "viewurl":"/video/af/ec/mm/od/ap/bm/ci/if/
bHVpZ2lh/0lfj9ach9mkdp-0.MOV-ext/504.jpg",
            "date":1400055593649,
            "creationdate":1341997020000,
            "modificationdate":1225663414000,
            "size":138494,
            "name":"100_3248.MOV",
            "metadata":{
               "duration":476,
               "bitrate":2326813,
               "codec":"mpeg4",
               "height":480,
               "width":640
            },
            "playbackurl":"",
            "thumbnails":[
               {
```

```
                  "w":176,
                  "h":132,
                  "size":"176",
                  "url":"/video/af/ec/mm/od/ap/bm/ci/if/
bHVpZ2lh/0lfj9ach9mkdp-0.MOV-ext/176.jpg",
                  "etag":"UEx/gZrPY5Jr09P10UacQg=="
               },
               {
                  "w":504,
                  "h":378,
                  "size":"504",
                  "url":"/video/af/ec/mm/od/ap/bm/ci/if/
bHVpZ2lh/0lfj9ach9mkdp-0.MOV-ext/504.jpg",
                  "etag":"UEx/gZrPY5Jr09P10UacQg=="
               }
            ],
            "mediatype":"video",
            "status":"U",
            "etag":"UEx/gZrPY5Jr09P10UacQg==",
            "softdeleted":false,
            "exported":{
               "youtube":{
                  "exporttime":1400055593643
               }
            }
         },
         {
            "id":"1",
            "url":"/video/af/ec/mm/od/ap/bm/ci/if/
bHVpZ2lh/0e0kh1qo3jset-1.mov",
            "viewurl":"/video/af/ec/mm/od/ap/bm/ci/if/
bHVpZ2lh/0e0kh1qo3jset-1.mov-ext/504.jpg",
            "date":1400055451808,
            "creationdate":1341997020000,
            "modificationdate":1226439062000,
            "size":2574169,
            "name":"100_3601.mov",
            "metadata":{
               "duration":12620,
               "bitrate":1631759,
               "codec":"mpeg4",
               "height":480,
               "width":640
            },
            "playbackurl":"",
            "thumbnails":[
               {
                  "w":176,
                  "h":132,
                  "size":"176",
                  "url":"/video/af/ec/mm/od/ap/bm/ci/if/
bHVpZ2lh/0e0kh1qo3jset-1.mov-ext/176.jpg",
                  "etag":"TRpX8/rMOcURYbIosfvsvA=="
               },
```

```
               {
                  "w":504,
                  "h":378,
                  "size":"504",
                  "url":"/video/af/ec/mm/od/ap/bm/ci/if/
bHVpZ2lh/0e0kh1qo3jset-1.mov-ext/504.jpg",
                  "etag":"TRpX8/rMOcURYbIosfvsvA=="
               }
            ],
            "mediatype":"video",
            "status":"U",
            "etag":"TRpX8/rMOcURYbIosfvsvA==",
            "softdeleted":false
         }
      ]
   }
}
```

Example code 3 (with encryption)

```
GET /sapi/media/video?action=get&exported=true
```

```
{
   "data":{
      "mediaserverurl":"https://my.server.com",
      "videos":[
         {
            "id":"101",
            "url":"/sapi/download/video?
action=get&userid=fe173a2ab7b76058&id=101&key=313165346469677a7072643731
2d3130312e6d6f76",
            "viewurl":"/sapi/download/video?
action=get&userid=fe173a2ab7b76058&id=101&key=313165346469677a7072643731
2d3130312e6d6f762d6578742532463530342e6a7067",
            "date":1400057036647,
            "creationdate":1341997020000,
            "modificationdate":1226439062000,
            "size":2574169,
            "labels":[
               {
                  "labelid":0,
                  "name":"US trip"
               }
            ],
            "name":"100_3601.mov",
            "metadata":{
               "duration":12620,
               "bitrate":1631759,
               "codec":"mpeg4",
               "height":480,
               "width":640
            },
            "playbackurl":"",
            "thumbnails":[
```

```
                {
                   "w":176,
                   "h":132,
                   "size":"176",
                   "url":"/sapi/download/video?
action=get&userid=fe173a2ab7b76058&id=101&key=313165346469677a7072643731
2d3130312e6d6f762d6578742532463137362e6a7067",
                   "etag":"5VmvMBzge7TglttCmJUUDg=="
                },
                {
                   "w":504,
                   "h":378,
                   "size":"504",
                   "url":"/sapi/download/video?
action=get&userid=fe173a2ab7b76058&id=101&key=313165346469677a7072643731
2d3130312e6d6f762d6578742532463530342e6a7067",
                   "etag":"5VmvMBzge7TglttCmJUUDg=="
                }
             ],
             "mediatype":"video",
             "status":"U",
             "etag":"5VmvMBzge7TglttCmJUUDg==",
             "folder":12,
             "softdeleted":false
          },
          {
             "id":"100",
             "url":"/sapi/download/video?
action=get&userid=fe173a2ab7b76058&id=100&key=313974617a3968637a61356233
2d3130302e4d4f56",
             "viewurl":"/sapi/download/video?
action=get&userid=fe173a2ab7b76058&id=100&key=313974617a3968637a61356233
2d3130302e4d4f562d6578742532463530342e6a7067",
             "date":1400056934163,
             "creationdate":1341997020000,
             "modificationdate":1225663414000,
             "size":138494,
             "name":"100_3248.MOV",
             "metadata":{
                "duration":476,
                "bitrate":2326813,
                "codec":"mpeg4",
                "height":480,
                "width":640
             },
             "playbackurl":"",
             "thumbnails":[
                {
                   "w":176,
                   "h":132,
                   "size":"176",
                   "url":"/sapi/download/video?
action=get&userid=fe173a2ab7b76058&id=100&key=313974617a3968637a61356233
2d3130302e4d4f562d6578742532463137362e6a7067",
```

```
                  "etag":"jGsec/iweWG00CxTpQi8vw=="
               },
               {
                  "w":504,
                  "h":378,
                  "size":"504",
                  "url":"/sapi/download/video?
action=get&userid=fe173a2ab7b76058&id=100&key=313974617a3968637a61356233
2d3130302e4d4f562d6578742532463530342e6a7067",
                  "etag":"jGsec/iweWG00CxTpQi8vw=="
               }
            ],
            "mediatype":"video",
            "status":"U",
            "etag":"jGsec/iweWG00CxTpQi8vw==",
            "softdeleted":false,
            "exported":{
               "youtube":{
                  "exporttime":1400056933980
               }
            }
         }
      ]
   }
}
```

Example code 4 (with encryption)

This example contains the playback URL of the video transcoded successfully, and an empty playback URL for a video not transcoded because of an error.

```
GET /sapi/media/video?action=get&limit=2
```

```
{
   "data":{
      "mediaserverurl":"https://my.server.com",
      "videos":[
         {
            "id":"101",
            "url":"/sapi/download/video?
action=get&userid=fe173a2ab7b76058&id=101&key=313165346469677a7072643731
2d3130312e6d6f76",
            "viewurl":"/sapi/download/video?
action=get&userid=fe173a2ab7b76058&id=101&key=313165346469677a7072643731
2d3130312e6d6f762d6578742532463530342e6a7067",
            "date":1400057036647,
            "creationdate":1341997020000,
            "modificationdate":1226439062000,
            "size":2574169,
            "labels":[
               {
                  "labelid":0,
                  "name":"US trip"
               }
```

```
            ],
            "name":"100_3601.mov",
            "metadata":{
               "duration":12620,
               "bitrate":1631759,
               "codec":"mpeg4",
               "height":480,
               "width":640
            },
            "playbackurl":"/sapi/download/video?
action=get&userid=fe173a2ab7b76058&id=101&key=313165346469677a7072643731
2d3130312e6d6f762d657874253246706c61796261636b2e6d7034",
            "playbackcontenttype":"video/mp4",
            "thumbnails":[
               {
                  "w":176,
                  "h":132,
                  "size":"176",
                  "url":"/sapi/download/video?
action=get&userid=fe173a2ab7b76058&id=101&key=313165346469677a7072643731
2d3130312e6d6f762d6578742532463137362e6a7067",
                  "etag":"5VmvMBzge7TglttCmJUUDg=="
               },
               {
                  "w":504,
                  "h":378,
                  "size":"504",
                  "url":"/sapi/download/video?
action=get&userid=fe173a2ab7b76058&id=101&key=313165346469677a7072643731
2d3130312e6d6f762d6578742532463530342e6a7067",
                  "etag":"5VmvMBzge7TglttCmJUUDg=="
               }
            ],
            "mediatype":"video",
            "status":"U",
            "etag":"5VmvMBzge7TglttCmJUUDg==",
            "folder":12,
            "softdeleted":false
         },
         {
            "id":"100",
            "url":"/sapi/download/video?
action=get&userid=fe173a2ab7b76058&id=100&key=313974617a3968637a61356233
2d3130302e4d4f56",
            "viewurl":"/sapi/download/video?
action=get&userid=fe173a2ab7b76058&id=100&key=313974617a3968637a61356233
2d3130302e4d4f562d6578742532463530342e6a7067",
            "date":1400056934163,
            "creationdate":1341997020000,
            "modificationdate":1225663414000,
            "size":138494,
            "name":"100_3248.MOV",
            "metadata":{
               "duration":476,
```

```
                "bitrate":2326813,
                "codec":"mpeg4",
                "height":480,
                "width":640
             },
             "playbackurl":"",
             "thumbnails":[
                {
                   "w":176,
                   "h":132,
                   "size":"176",
                   "url":"/sapi/download/video?
action=get&userid=fe173a2ab7b76058&id=100&key=313974617a3968637a61356233
2d3130302e4d4f562d6578742532463137362e6a7067",
                   "etag":"jGsec/iweWG00CxTpQi8vw=="
                },
                {
                   "w":504,
                   "h":378,
                   "size":"504",
                   "url":"/sapi/download/video?
action=get&userid=fe173a2ab7b76058&id=100&key=313974617a3968637a61356233
2d3130302e4d4f562d6578742532463530342e6a7067",
                   "etag":"jGsec/iweWG00CxTpQi8vw=="
                }
             ],
             "mediatype":"video",
             "status":"U",
             "etag":"jGsec/iweWG00CxTpQi8vw==",
             "softdeleted":false,
             "exported":{
                "youtube":{
                   "exporttime":1400056933980
                }
             }
          }
       ]
    }
}
```


Errors

COM-1008, EXT-SRV-1000, MED-1003, MED-1005, MED-1016, MED-1017, MED-1021, in addition to generic errors described in Section 4.1.3, “Errors across multiple Server APIs”.

3.5.9 Delete videos

Delete videos for the user.

Security realm: user

Definition

```
POST /sapi/media/video?action=delete
```

Request body

- *`data`*: the JSON object containing an array named `videos` with the IDs of the videos to be deleted.
- *`softdelete`*: (**optional**) if `true`, the videos will be marked as soft deleted, that is, in the trash. Default is `false`.

Example:

```
{
   "data":{
      "videos":[
         23408,
         23814
      ]
   }
}
```

Response

The 200 HTTP code indicates a successful request.

Errors

COM-1013, MED-1000, MED-1016, MED-1022, in addition to generic errors described in Section 4.1.3, “Errors across multiple Server APIs”.

3.5.10 Reset videos

Delete all the videos stored on the media storage provider.

> **Note**
>
> If a user performs a call to this API, no items will be returned in a subsequent call of the API described at Section 3.3.1, “Get changes”)

Security realm: user

Definition

```
POST /sapi/media/video?action=reset
```

Response

The 200 HTTP code indicates a successful request.

Errors

MED-1000, MED-1014, in addition to generic errors described in Section 4.1.3, “Errors across multiple Server APIs”.

3.5.11 Count the videos

Return the number of videos for the user.

Security realm: user

Definition

```
GET /sapi/media/video?action=count
```

Parameters

- *`onlysoftdelete`*: (**optional**) if `true`, the number of videos marked as soft deleted will be returned.

Response

The number of videos for the user.

Example:

```
{
   "data":{
      "success":"Count returned successfully",
      "count":30
   }
}
```

Errors

MED-1000, in addition to generic errors described in Section 4.1.3, “Errors across multiple Server APIs”.

3.5.12 Update transcoding job status

Update the transcoding job status.

Security realm: public

Definition

```
POST sapi/media/video?action=set-transcoding-status
```

Request Body

Note

The request body is defined by the service used for video transcoding.

The following examples are based on the notification sent by the Amazon Elastic Transcoder service.

Example 1

This is the request body of the confirmation message:

```
{
   "Type":"SubscriptionConfirmation",
   "MessageId":"8cdcc43e-0297-4542-83cf-55067f45b9d5",

 "Token":"2336412f37fb687f5d51e6e241d164b14a5cd0e6f68910b18fef14d8ba8e3
991792d8eafe15ef0e2ac87386b3f5b37706657a8fefb09feed30f2b2b277147c18d7c1a
```

```
eb3f9dcad79d3a6402042068295811feaf7de82dfadc4882138a143b23ead48f1c1e5071
e64eaa8758590abc3fa922fc78782e7535ecf0f8244c63fb909",
   "TopicArn":"arn:aws:sns:eu-west-1:440740892053:transcoding-fun",
   "Message":"You have chosen to subscribe to the topic arn:aws:sns:eu-
west-1:440740892053:transcoding-fun.\nTo confirm the subscription, visit
 the SubscribeURL included in this message.",
   "SubscribeURL":"https://sns.eu-west-1.amazonaws.com/?
Action=ConfirmSubscription&TopicArn=arn:aws:sns:eu-
west-1:440740892053:transcoding-
fun&Token=2336412f37fb687f5d51e6e241d164b14a5cd0e6f68910b18fef14d8ba8e39
91792d8eafe15ef0e2ac87386b3f5b37706657a8fefb09feed30f2b2b277147c18d7c1ae
b3f9dcad79d3a6402042068295811feaf7de82dfadc4882138a143b23ead48f1c1e5071e
64eaa8758590abc3fa922fc78782e7535ecf0f8244c63fb909",
   "Timestamp":"2013-04-19T14:00:54.569Z",
   "SignatureVersion":"1",
   "Signature":"R8GBremkLvlVCuVamliqGQb/+ACMHtuJwO98ZM6BoLJ3Lz+q7/
UHiV0M0TLjp/
MHoNib890P1HvJqrNpXgqjFsToozzV5b0Ug59+eQY6Q5DsRMLRVPj7I027gaHN4MSYsRbMOy
zenNhhfvZeZdNdduxYdfAefu+15EhmR3sHHno=",
   "SigningCertURL":"https://sns.eu-west-1.amazonaws.com/
SimpleNotificationService-f3ecfb7224c7233fe7bb5f59f96de52f.pem"
}
```

Example 2

This is the request body when the job is created:

```
{
   "Type":"Notification",
   "MessageId":"b6ddd261-24c4-5739-b9e8-9b8a604387da",
   "TopicArn":"arn:aws:sns:eu-west-1:440740892053:transcoding-fun",
   "Subject":"Amazon Elastic Transcoder has scheduled job
 1366377029384-520b87 for transcoding.",
   "Message":"{\n  \"state\" : \"PROGRESSING\",\n
 \"jobId\" : \"1366377029384-520b87\",\n  \"pipelineId\" :
 \"1365069402727-74e0f3\"\n}",
   "Timestamp":"2013-04-19T13:10:31.430Z",
   "SignatureVersion":"1",
   "Signature":"mJchyM1GPxMqJusZL7RIkL7lXIzlUIMcCbAfTe3Rt3cDEGYgMcvQa
+COzz3G6K//GCiegCkqXtMDNN8UafXfm7FlK3nQhcx+sBjuoky7mq+BjfRWVbrh/
KsG73nRlFrXi+QfqdaPmEgTBVgrpgrF/zhxMTZXRx6EykcbGfqhck=",
   "SigningCertURL":"https://sns.eu-west-1.amazonaws.com/
SimpleNotificationService-f3ecfb7224c7233fe7bb5f59f96de52f.pem",
   "UnsubscribeURL":"https://sns.eu-west-1.amazonaws.com/?
Action=Unsubscribe&SubscriptionArn=arn:aws:sns:eu-
west-1:440740892053:transcoding-fun:489140f4-4383-4033-
bb02-681130f6ab77"
}
```

Example 3

This is the request body when the job has been completed successfully:

```
{
   "Type":"Notification",
```

```
   "MessageId":"1f99fa78-11f8-5fd0-bda9-b393561466b7",
   "TopicArn":"arn:aws:sns:eu-west-1:440740892053:transcoding-fun",
   "Subject":"Amazon Elastic Transcoder has finished transcoding job
 1366377890966-8ca7cc.",
   "Message":"{\n  \"state\" : \"COMPLETED\",\n  \"jobId
\" : \"1366377890966-8ca7cc\",\n  \"pipelineId\" :
 \"1365069402727-74e0f3\"\n}",
   "Timestamp":"2013-04-19T13:25:00.506Z",
   "SignatureVersion":"1",
   "Signature":"GW9iHHf0VISq/hgK0YmBhdaKYbyTEuQTFJ1Gb01ewhzpnkyXGCs/
q2GnqABo7GocGYjTGx82f7RUfB4/kmkzg+Vz2G4rb6ZvwxGszyuMSe/
fyEBjn2CMnNNKZo7N8jaRdnLYNBAxVGv5iK+VmDbCuyNLUV/ruFwlvtlcgqDPh+g=",
   "SigningCertURL":"https://sns.eu-west-1.amazonaws.com/
SimpleNotificationService-f3ecfb7224c7233fe7bb5f59f96de52f.pem",
   "UnsubscribeURL":"https://sns.eu-west-1.amazonaws.com/?
Action=Unsubscribe&SubscriptionArn=arn:aws:sns:eu-
west-1:440740892053:transcoding-fun:489140f4-4383-4033-
bb02-681130f6ab77"
}
```

Example 4

This is the request body when the job has been completed with error:

```
{
   "Type":"Notification",
   "MessageId":"9f53bf8b-2447-5f5d-b8e9-f66060ce7a78",
   "TopicArn":"arn:aws:sns:eu-west-1:440740892053:transcoding-fun",
   "Subject":"The Amazon Elastic Transcoder job 1366377029384-520b87 has
 failed.",
   "Message":"{\n  \"state\" : \"ERROR\",\n  \"jobId\" :
 \"1366377029384-520b87\",\n  \"pipelineId\" : \"1365069402727-74e0f3\",
\n  \"errorCode\" : 3002,\n  \"messageDetails\" : \"3002
 b92cca51-8a39-4e24-a661-0173b57e5a77: The specified object could
 not be saved in the specified bucket because an object by that name
 already exists: bucket=fdo-container-fun, key=af/ec/mm/od/ap/bm/ci/if/
bHVpZ2lh/0x7nwcuc9xj9f-600.xcoding.MOV.\"\n}",
   "Timestamp":"2013-04-19T13:10:33.569Z",
   "SignatureVersion":"1",
   "Signature":"mpv3bPlVAObitTgJaQhVN5DG/
V3GDwjy8YvrEXzI03NGsBNsVeBSddTr8SxvBkp+wp
+X5+QhHprytGXKRj18ckDIHm6Qx5vzgq9Sd0pyc4HA0kywE0y2+F2ql1QEXvvUrAHgdAibxb
ctmg9V85Gkebkh6T031vc/wxNDIogzi3M=",
   "SigningCertURL":"https://sns.eu-west-1.amazonaws.com/
SimpleNotificationService-f3ecfb7224c7233fe7bb5f59f96de52f.pem",
   "UnsubscribeURL":"https://sns.eu-west-1.amazonaws.com/?
Action=Unsubscribe&SubscriptionArn=arn:aws:sns:eu-
west-1:440740892053:transcoding-fun:489140f4-4383-4033-
bb02-681130f6ab77"
}
```

Errors

MED-1019, in addition to generic errors described in Section 4.1.3, “Errors across multiple Server APIs”.

3.5.13 Retrieve files

Retrieve files for the user.

Security realm: user

Definition

```
GET /sapi/media/file?action=get
```

Parameters

- *id*: (**optional**) the ID of the file or a JSON array of IDs to be retrieved. If an array of IDs is specified, the items will be returned in the same order they were requested.
- *from*: the date (in UTC milliseconds) starting from when the files should be retrieved
- *limit*: (**optional**) the maximum number of files retrieved starting from the latest ones and considering the offset. Cannot be used with the *id* parameter.
- *offset*: (**optional**) the offset (inclusive) of files to be considered in the response. Cannot be used with the *id* parameter.
- *sortby*: (**optional**) specifies the field to use to order returned files. Allowed values are `date` (date of the last update), and `id`.
- *sortorder*: (**optional**) the order in which the *sortby* parameter is applied. Allowed values are `ascending` and `descending`. If a value for *sortby* is supplied and no *sortorder* is specified, the *sortorder* will default to `ascending`.
- *shared*: (**optional**) if the value is `true`, then a flag named `shared` will be returned for every file, which is `true` if the file is part of a media set. Cannot be used with the *softdelete* and *onlysoftdelete* parameters. Default value is `false`.
- *softdelete*: (**optional**) if `true`, files marked as soft deleted will be included in the response. Cannot be used with the parameters *shared* and *onlysoftdelete*. Default is `false`.
- *onlysoftdelete*: (**optional**) if `true`, only files marked as soft deleted will be included in the response. Cannot be used with the parameters *shared* and *softdelete*. Default is `false`.

IDs example:

```
id={"ids":[12,23,456]}
```

and, as encoded parameter:

```
id=%7B%22ids%22%3A%5B12%2C23%2C456%5D%7D
```

Response

The server URL where the files are stored (*mediaserverurl*) and an array with all the file objects for the given user; or, if the *limit* parameter is specified, the latest limit for files (if there is at least a limit or a subset of a limit for files) from a given offset, if the *offset* parameter is also specified. If the *from* parameter is used at the same time as *limit* and *offset*, the files are first filtered by the *from*

parameter, and only after that, the offset and limit are applied. If the *softdelete* parameter is true, both soft deleted and non soft deleted files are included. If the *onlysoftdelete* parameter is true, only soft deleted files are returned. If there are no files in the database for the given user or the offset is greater than the number of files, an empty array will be returned. For more information about the File JSON object, see Section 5.19, “File”.

Example code 1

```
GET /sapi/media/file?action=get&limit=3
```

```
{
   "data":{
      "mediaserverurl":"https://my.server.com",
      "files":[
         {
            "id":"2501",
            "url":"/file/aq/jt/si/ma/gi/ug/ka/ee/
username/03nyna3a8cy5d",
            "date":1267046824780,
            "size":1325816,
            "name":"testx.doc",
            "mediatype":"file",
            "labels":[
               {
                  "id":1,
                  "name":"US trip"
               }
            ],
            "status":"U",
            "etag":"aafqzchtgdvLsw7zhifZKg==",
            "folder":123,
            "softdeleted":false
         }
      ]
   }
}
```

Example code 2

```
GET /sapi/media/file?action=get&exported=true
```

```
{
   "data":{
      "mediaserverurl":"https://my.server.com",
      "files":[
         {
            "id":"2501",
            "url":"/file/aq/jt/si/ma/gi/ug/ka/ee/
username/03nyna3a8cy5d",
            "date":1267046824780,
            "size":1325816,
            "name":"testx.doc",
            "mediatype":"file",
            "exported":{
```

```
               "picasa":{
                  "exporttime":1274192165
               }
            },
            "labels":[
               {
                  "id":1,
                  "name":"US trip"
               }
            ],
            "status":"U",
            "etag":"aafqzchtgdvLsw7zhifZKg==",
            "folder":123,
            "softdeleted":false
         }
      ]
   }
}
```

Example code 3 (with encryption)

```
GET /sapi/media/file?action=get&exported=true
```

```
{
   "data":{
      "mediaserverurl":"https://my.server.com",
      "files":[
         {
            "id":"2501",
            "url":"/sapi/download/file?
action=get&filename=03nyna3a8cy5d",
            "date":1267046824780,
            "size":1325816,
            "name":"testx.doc",
            "mediatype":"file",
            "exported":{
               "picasa":{
                  "exporttime":1274192165
               }
            },
            "labels":[
               {
                  "id":1,
                  "name":"US trip"
               }
            ],
            "status":"U",
            "etag":"aafqzchtgdvLsw7zhifZKg==",
            "folder":123,
            "softdeleted":false
         }
      ]
   }
}
```

Errors

COM-1008, EXT-SRV-1000, MED-1003, MED-1005, MED-1016, MED-1017, MED-1021, in addition to generic errors described in Section 4.1.3, “Errors across multiple Server APIs”.

3.5.14 Delete files

Delete files for the user.

Security realm: user

Definition

```
POST /sapi/media/file?action=delete
```

Request body

- *`data`*: the JSON object containing an array named `files` with the IDs of the files to be deleted.
- *`softdelete`*: (**optional**) if `true`, the files will be marked as soft deleted, that is, in the trash. Default is `false`.

Example:

```
{
   "data":{
      "files":[
         23408,
         23814
      ]
   }
}
```

Response

The 200 HTTP code indicates a successful request.

Errors

COM-1013, MED-1000, MED-1016, MED-1022, in addition to generic errors described in Section 4.1.3, “Errors across multiple Server APIs”.

3.5.15 Reset files

Delete all the files stored on the media storage provider.

> **Note**
>
> If a user performs a call to this API, no items will be returned in a subsequent call of the API described at Section 3.3.1, “Get changes”)

Security realm: user

Definition

```
POST /sapi/media/file?action=reset
```

Response

The 200 HTTP code indicates a successful request.

Errors

MED-1000, MED-1014, in addition to generic errors described in Section 4.1.3, “Errors across multiple Server APIs”.

3.5.16 Count the files

Return the number of files for the user.

Security realm: user

Definition

```
GET /sapi/media/file?action=count
```

Parameters

- *`onlysoftdelete`*: (**optional**) if `true`, the number of files marked as soft deleted will be returned.

Response

The number of files for the user.

Example:

```
{
   "data":{
      "success":"Count returned successfully",
      "count":30
   }
}
```

Errors

MED-1000, in addition to generic errors described in Section 4.1.3, “Errors across multiple Server APIs”.

3.5.17 Retrieve audio

Retrieve audio data for the user.

Security realm: user

Definition

```
GET /sapi/media/audio?action=get
```

Parameters

- *`id`*: (**optional**) the ID of the audio file or a JSON array with IDs to be retrieved. If an array of IDs is specified, the items will be returned in the same order they were requested

- *from*: the creation date of the audio file (in UTC milliseconds)
- *limit*: (**optional**) the maximum number of tracks retrieved starting from the latest ones and considering the offset. Cannot be used with the parameter *id*.
- *offset*: (**optional**) the offset (inclusive) of tracks to be considered in the response. Cannot be used with the parameter *id*.
- *metadata*: it can be none or all. Default is all.
- *sortby*: (**optional**) it specifies the field to use to order returned audio data. Allowed fields are `date` (date of the last update), and `id`.
- *sortorder*: (**optional**) the order in which the *sortby* parameter is applied. Allowed values are `ascending` and `descending`. If a value for *sortby* is specified but no *sortorder* is specified, the *sortorder* will be set by default to `ascending`
- *softdelete*: (**optional**) if `true`, audio items marked as soft deleted will be included in the response. Cannot be used with the *onlysoftdelete* parameter. Default is `false`.
- *onlysoftdelete*: (**optional**) if `true`, only files marked as soft deleted will be included in the response. Cannot be used with the *softdelete* parameter. Default is `false`.

IDs example:

```
id={"ids":[12,23,456]}
```

and, as encoded parameter:

```
id=%7B%22ids%22%3A%5B12%2C23%2C456%5D%7D
```

Response

The server URL where the audio files are stored (*mediaserverurl*) and an array with all the audio objects for the given user; or, if the *limit* parameter is specified, the latest limit for audio files (if there is at least a limit or a subset of a limit for audio data) from a given offset, if the *offset* parameter is also specified. If the *from* parameter is used at the same time as *limit* and *offset*, the audio files are first filtered by the *from* parameter, and only after that the offset and limit are applied. If the *softdelete* parameter is `true`, both soft deleted and non soft deleted files are included. If there are no audio items in the database for the given user or the offset is greater than the number of audio items, an empty array will be returned. For more information about the *Audio* JSON object, see Section 5.6, “Audio”.

Example code 1

```
GET /sapi/media/audio?action=get&limit=3
```

```
{
   "data":{
      "mediaserverurl":"https://my.server.com",
      "audios":[
         {
            "id":"2501",
            "url":"/audio/aq/jt/si/ma/gi/ug/ka/ee/
username/03nyna3a8cy5d",
            "viewurl":"/audio/album/cover.jpg",
            "date":1267046824780,
```

```
            "size":1325816,
            "name":"alba chiara.mp3",
            "metadata":{
               "album":"non siamo mica gli americani",
               "artist":"vasco rossi",
               "tracknumber":"7"
            },
            "status":"U",
            "etag":"aafqzchtgdvLsw7zhifZKg==",
            "folder":143,
            "softdeleted":false
         }
      ]
   }
}
```

Example code 2 (with encryption)

```
GET /sapi/media/audio?action=get
```

```
{
   "data":{
      "mediaserverurl":"https://my.server.com",
      "audios":[
         {
            "id":"2501",
            "url":"/sapi/download/audio?
action=get&filename=03nyna3a8cy5d",
            "viewurl":"audio/album/cover.jpg",
            "date":1267046824780,
            "size":1325816,
            "name":"alba chiara.mp3",
            "metadata":{
               "album":"non siamo mica gli americani",
               "artist":"vasco rossi",
               "tracknumber":"7"
            },
            "status":"U",
            "etag":"aafqzchtgdvLsw7zhifZKg==",
            "folder":143,
            "softdeleted":false
         }
      ]
   }
}
```

Errors

COM-1008, COM-1019, MED-1000, MED-1003, MED-1016, MED-1017, MED-1021, in addition to generic errors described in Section 4.1.3, “Errors across multiple Server APIs”.

3.5.18 Delete audio

Delete audio files for the user.

Security realm: user

Definition

```
POST /sapi/media/audio?action=delete
```

Parameters

- *`softdelete`*: (**optional**) if `true`, the audio items will be marked as soft deleted, that is, in the trash. Default is `false`.

Request body

- *`data`*: the JSON object containing an array named `audios` with the IDs of the tracks to be deleted

Example:

```
{
    "data":{
       "audios":[23508,23816]
    }
}
```

Response

The 200 HTTP code indicates a successful request.

Errors

COM-1013, MED-1000, MED-1016, MED-1022, in addition to generic errors described in Section 4.1.3, “Errors across multiple Server APIs”.

3.5.19 Reset audio

Delete all the audio items stored on the media storage provider.

> **Note**
>
> If a user performs a call to this API, no items will be returned in a subsequent call of the API described at Section 3.3.1, “Get changes”

Security realm: user

Definition

```
POST /sapi/media/audio?action=reset
```

Response

The 200 HTTP code indicates a successful request.

Errors

MED-1000 and MED-1014, in addition to generic errors described in Section 4.1.3, “Errors across multiple Server APIs”

3.5.20 Count the audio items

Return the number of audio items for the user.

Security realm: user

Definition

```
GET /sapi/media/audio?action=count
```

Parameters

- *`onlysoftdelete`*: (**optional**) if `true`, the number of audio items marked as soft deleted will be returned.

Response

The number of audio items for the user.

Example:

```
{
   "data":{
      "success":"Count returned successfully",
      "count":30
   }
}
```

Errors

MED-1000, in addition to generic errors described in Section 4.1.3, “Errors across multiple Server APIs”

3.5.21 Create a media set

Create a media set, which is a bundle of media items having a unique URL. The unique URL points to a page which displays the media items and allows to download them. No authentication is required for visiting the landing page.

Security realm: user

Definition

```
POST /sapi/media/set?action=save
```

Request body

A JSON object with information regarding the media set.

Example

Request body:

```
{
```

```
   "data":{
      "set":{
         "description":"Visit to Vienna",
         "type":"picture",
         "items":[
            1,
            2
         ]
      }
   }
}
```

Response

A JSON response with success message, ID and URL, or error code.

Example

Response:

```
{
   "success:":"Media set saved successfully",
   "id":1,
   "url":"https://my.server.com/share/1123456"
}
```

A unique key for the media set is encoded in the media set URL - the format of the URL may be changed using server settings - the default format is that the key is following the last forward slash of the URL.

Errors

COM-1013 and MED-1000, in addition to generic errors described in Section 4.1.3, “Errors across multiple Server APIs”

3.5.22 Retrieve a media set

Return a previously created media set.

Security realm: public

Definition

```
GET /sapi/media/set?action=get
```

Parameters

- *`id`*: the ID of the media set. May only be used in security realm *user*.
- *`key`*: the unique key of the media set. No authentication is required to use the API when called in this way.
- *`items`*: (**optional**) if `true` the list of items contained in the media set will be returned (`false` by default).

Example 1

Request

```
GET /sapi/media/set?action=get&id=1&items=true
```

Response


```
{
   "data":{
      "sets":[
         {
            "description":"Visit to Vienna",
            "type":"picture",
            "id":1,
            "url":"https://my.server.com/share/1123456",
            "items":[
               {
                  "name":"image_1.jpg",
                  "viewurl":"https://my.server.com/picture/aq/jt/si/
ma/gi/ug/ka/ee/username/1-ext/504x504_pic.jpg",
                  "url":"https://my.server.com/picture/aq/jt/si/ma/
gi/ug/ka/ee/username/1.jpg"
               },
               {
                  "name":"image_2.jpg",
                  "viewurl":"https://my.server.com/picture/aq/jt/si/
ma/gi/ug/ka/ee/username/2-ext/504x504_pic.jpg",
                  "url":"https://my.server.com/picture/aq/jt/si/ma/
gi/ug/ka/ee/username/2.jpg"
               }
            ]
         }
      ],
      "user":{
         "generic":{
            "firstname":"John",
            "lastname":"Doe",
            "userid":"johndoe"
         }
      }
   }
}
```


Example 2

Request

```
GET /sapi/media/set?action=get&id=1&items=true
```

Response


```
{
   "data":{
      "user":{
```

```
         "generic":{
            "firstname":"John",
            "lastname":"Doe",
            "userid":"johndoe"
         }
      },
      "sets":[
         {
            "id":1,
            "description":"My shared set",
            "url":"https://my.server.com/share/1123456",
            "type":"video",
            "items":[
               {
                  "name":"htc_legend.3gp",
                  "url":"https://my.server.com/video/aq/jt/si/ma/gi/
ug/ka/ee/username/114ja7w4ftoyv-3.3gp",
                  "viewurl":"https://my.server.com/video/aq/jt/si/ma/
gi/ug/ka/ee/username/114ja7w4ftoyv-3.3gp-ext/504.jpg",
                  "playbackurl":"https://my.server.com/video/aq/jt/
si/ma/gi/ug/ka/ee/username/114ja7w4ftoyv-3.3gp-ext/playback.mp4"
               },
               {
                  "name":"movie.3gp",
                  "url":"https://my.server.com/video/aq/jt/si/ma/gi/
ug/ka/ee/username/114ja7w4ftoaa-4.3gp",
                  "viewurl":"https://my.server.com/video/aq/jt/si/ma/
gi/ug/ka/ee/username/114ja7w4ftoaa-4.3gp-ext/504.jpg",
                  "playbackurl":"https://my.server.com/video/aq/jt/
si/ma/gi/ug/ka/ee/username/114ja7w4ftoaa-4.3gp-ext/playback.mp4"
               }
            ]
         }
      ]
   }
}
```


Errors

MED-1011, COM-1011, MED-1012, PRO-1101, and MED-1000, in addition to generic errors described in Section 4.1.3, “Errors across multiple Server APIs”

3.5.23 Remove media from sets

Remove media items from all the media sets they are contained in.

Security realm: user

Definition

```
POST /sapi/media?action=remove-from-sets
```

Parameters

- `id`: the ID of the media items or a JSON array of IDs to be removed from sets

IDs example:

```
id={"ids":[12,23,456]}
```

and, as encoded parameter:

```
id=%7B%22ids%22%3A%5B12%2C23%2C456%5D%7D
```

Response

The 200 HTTP code indicates a successful request.

Errors

COM-1011 and MED-1000, in addition to generic errors described in Section 4.1.3, “Errors across multiple Server APIs”.

3.5.24 Create a folder

Create a folder or update it, in case the folder already exists. In case the *`items`* parameter is used, if one or more items are not present on the server, or one or more of them cause an error, they will be skipped (no error returned to the caller.) An *offline folder* must always be a child of the default watch folder (the latter also known as the *magic folder*.)

Security realm: user

Definition

```
POST /sapi/media/folder?action=save
```

Request body

A JSON object with information regarding the folder (see Section 5.21, “Folder”).

Response

A JSON object containing information related to the created folder.

Example 1 (new folder creation)

Request:

```
POST /sapi/media/folder?action=save
```

Request body:

```
{
   "data":{
      "mediatypes": ["picture"],
      "items": [1, 2, 3],
      "name": "abcfolder",
      "devicename":"pcdevice",
      "magic":false,
      "offline":false,
      "parentid":12
```

```
   }
}
```

Response:

```
{
   "success":"Folder saved successfully",
   "id":"13",
   "lastupdate":1373357109924
}
```

Example 2 (update of an existing folder)

Request:

```
POST /sapi/media/folder?action=save
```

Request body:

```
{
   "data":{
      "id":12,
      "mediatypes": ["picture"],
      "items": [1, 2, 3],
      "name": "abcfolder",
      "devicename":"pcdevice",
      "magic":false,
      "offline":false,
      "parentid":11
   }
}
```

Response:

```
{
   "success":"Folder saved successfully",
   "id":"12",
   "lastupdate":1373987654321
}
```

Errors

COM-1011, FOL-1000 , FOL-1002, FOL-1003, FOL-1004, FOL-1009, FOL-1010, FOL-1012, FOL-1013, FOL-1014, in addition to generic errors described in Section 4.1.3, “Errors across multiple Server APIs”.

3.5.25 Get folders

Allow the user to retrieve one or more folders.

Security realm: user

Definition

```
GET /sapi/media/folder?action=get
```

Parameters

- *id*: (**optional**) the ID of the folder or a JSON array with IDs to be retrieved. If an array of IDs is specified, the items will be returned in the same order they were requested in.
- *limit*: (**optional**) the maximum number of folders retrieved starting from the latest ones and considering the value of *offset*. Cannot be used with the parameter *id*.
- *offset*: (**optional**) the offset (inclusive) of folders to be considered in the response. Cannot be used with the parameter *id*.

IDs example

```
id = {"ids":[12,23,456]}
```

and, as an encoded parameter

```
id=%7B%22ids%22%3A%5B12%2C23%2C456%5D%7D
```

Response

A JSON array containing the JSON object of kind *Folder* (see Section 5.21, “Folder”) with the required folder info. If the *limit* or *offset* parameters are specified, the array will contain a subset of folders, limited by the values of those parameters. The returned array of folders is guaranteed to be ordered in such a way that the parent folder comes always before any of its children. This means that the hierarchy is output in a way that any folder having a parent folder ID (*parentid*) comes after its parent.

Example

Request

```
GET /sapi/media/folder?action=get&limit=3
```

Response

```
{
   "data":{
      "folders":[
         {
            "id":2500,
            "name":"onemediahub",
            "magic":true,
            "devicename":"mac-personal",
            "status":"U",
            "date":1231323524524
         },
         {
            "id":"2501",
            "name":"pictures",
            "devicename":"win-personal",
            "mediatypes":[
               "picture"
            ],
            "status":"U",
```

```
            "parentid":2500,
            "date":1231323524524
         }
      ]
   }
}
```

Errors

COM-1021, FOL-1000, FOL-1011, in addition to generic errors described in Section 4.1.3, “Errors across multiple Server APIs”.

3.5.26 Add media to a folder

Add one or more media items to a given folder.

> **Note**
>
> After this API invocation the media items will be considered updated, but not the folder.

Security realm: user

Definition

```
POST /sapi/media/folder?action=add-item
```

Request body

A JSON object containing the information to add media items to the given existing folder in the following parameters:

- *`items`*: (**mandatory**) a JSON array containing the list of IDs of the media items. It must not be empty. It must respect the folder type, if it exists.
- *`folderid`*: (**mandatory**) the ID of the folder.

Response

The 200 HTTP code indicates a successful request.

Example

Request

```
POST sapi/media/folder?action=add-item
```

Request body

```
{
   "data":{
      "items":[
         1,
         2,
```

```
            3
         ],
         "folderid":1
      }
}
```

Errors

COM-1011, FOL-1000, FOL-1004, in addition to generic errors described in Section 4.1.3, “Errors across multiple Server APIs”.

3.5.27 Remove media from a folder

Allow the user to remove media from an existing folder. The folder, also if empty, will not be deleted. The media item will not be deleted but considered as without folder.

Note

After this request the media items will be considered updated, but not the folder.

Security realm: user

Definition

```
POST /sapi/media/folder?action=remove-item
```

Request body

A JSON object containing the information to remove media items from the given folder in the following parameters:

- *`items`*: list of media item IDs. One between *`item`* and *`items`* is **mandatory**
- *`folderid`*: (**mandatory**) the ID of the folder

Response

The `200` HTTP response code indicates a successful request

Example 1

Request

```
POST sapi/media/folder?action=remove-item
```

Request body

```
{
   "data":{
      "items":[
         1,
         2,
         3
```

```
      ],
      "folderid":1
   }
}
```

Errors

COM-1011, FOL-1000, FOL-1004, in addition to generic errors described in Section 4.1.3, “Errors across multiple Server APIs”.

3.5.28 Delete folders

Let a user delete an existing folder. Folder deletion implies that all the subfolders contained in the deleted one will be deleted as well. All the items included in the folder or in its subfolders will be considered as not belonging to any folder.

Security realm: user

Definition

```
POST /sapi/media/folder?action=delete
```

Request body example

- *`data`*: the JSON object containing an array named `folders` with the IDs of the folders to be deleted:

```
{
   "data":{
      "folders":[
         100,
         95
      ]
   }
}
```

Response

The `200` HTTP code indicates a successful request.

Errors

COM-1011, FOL-1000, in addition to generic errors described in Section 4.1.3, “Errors across multiple Server APIs”.

3.5.29 Reset folders

Delete all the folders of the user.

Note

If a user performs a call to this API, no items will be returned in a subsequent call of the *Get changes* API (see Section 3.3.1, “Get changes”)

Security realm: user

Definition

```
POST /sapi/media/folder?action=reset
```

Response

The `200` HTTP code indicates a successful request.

Errors

FOL-1000, in addition to generic errors described in Section 4.1.3, “Errors across multiple Server APIs”.

3.5.30 Create a label

Allow the user to create a label (also empty) with a given name, or to modify an existing one by renaming it or adding a group of items.

Security realm: user

Definition

```
POST /sapi/label?action=save
```

Request body

A JSON object with information regarding the label (see Section 5.24, “Label”).

Response

A JSON object with the ID of the created label.

Example

Request:

```
POST /sapi/label?action=save
```

Request body:

```
{
   "data":{
      "items": [1, 2, 3],
      "name": "USA 2011"
   }
}
```

Response:

```
{
   "data":{
      "labelid": "1",
      "success": "Label saved successfully"
```

```
   }
}
```

Example 2

Request body:

```
{
   "data":{
      "items": [1, 2, 3],
      "name": "USA 2011",
      "type": "picture"
   }
}
```

Response:

```
{
   "data":{
      "labelid": "1",
      "success": "Label saved successfully"
   }
}
```

Example Rename Label

The label already exists on the server, with name *USA Trip 2011*:

Request body:

```
{
   "data":{
      "labelid":1,
      "name": "USA 2011"
   }
}
```

Response:

```
{
   "data":{
      "labelid": "1",
      "success": "Label saved successfully"
   }
}
```

Errors

LAB-1000, LAB-1001, LAB-1003, LAB-1004, LAB-1006, LAB-1007, in addition to generic errors described in Section 4.1.3, “Errors across multiple Server APIs”.

3.5.31 Add media to a label

Allow the user to mark one or more media items with an existing label.

Security realm: user

Definition

```
POST /sapi/label?action=add-item
```

Request body

A JSON object containing the information to mark media items with the given label in the following parameters:

- *`items`*: (**mandatory**) JSON array containing the list of IDs of the media items. It must not be empty.
- *`labelid`*: (**mandatory**) the ID of the label.

Response

The 200 HTTP code indicates a successful request.

Example

Request:

```
POST sapi/label?action=add-item
```

Request body:

```
{
   "data":{
      "items": [1, 2, 3],
      "labelid": 1
   }
}
```

Errors

LAB-1000, LAB-1001, LAB-1003, LAB-1005, LAB-1007, in addition to generic errors described in Section 4.1.3, “Errors across multiple Server APIs”.

3.5.32 Remove media from label

Allow the user to remove a media item from an existing label.

Security realm: user

Definition

```
POST /sapi/label?action=remove-item
```

Request body

A JSON object containing the information to remove media items from the given label in the following parameters:

- *`item`*: the ID of the media item.

- *`items`*: list of media item IDs. One between `item` and `items` is mandatory
- *`labelid`*: (**mandatory**) the ID of the label.

Response

The 200 HTTP code indicates a successful request.

Example 1

Request:

```
POST sapi/label?action=remove-item
```

Request body:

```
{
   "data":{
      "item": 1,
      "labelid": 1
   }
}
```

Example 2

Request body:

```
{
   "data":{
      "items": [1, 2],
      "labelid": 1
   }
}
```

Errors

LAB-1000, LAB-1001, LAB-1002, LAB-1003, in addition to generic errors described in Section 4.1.3, “Errors across multiple Server APIs”.

3.5.33 Delete label

Allow the user to delete an existing label.

Security realm: user

Definition

```
POST /sapi/label?action=delete
```

Parameters

- *`labelid`*: (**mandatory**) the ID of the label.

Response

The 200 HTTP code indicates a successful request.

Example

Request:

```
POST sapi/label?action=delete&labelid=1
```

Errors

LAB-1000, LAB-1001, LAB-1003, in addition to generic errors described in Section 4.1.3, “Errors across multiple Server APIs”.

3.5.34 Get label

Allow the user to retrieve one or more existing labels.

Security realm: user

Definition

```
POST /sapi/label?action=get
```

Request body

A JSON object containing in the following parameters the information about the label to retrieve:

- *`labelids`*: the list of label IDs. If no label IDs are specified, all the labels owned by the user will be returned.
- *`type`*: specify the desired label type.

Response

A JSON array containing the JSON object of kind *Label* (see Section 5.24, “Label”) with the required label info.

Example 1

Request:

```
POST /sapi/label?action=get
```

Request body:

```
{
   "data":{
      "labelids": [1]
   }
}
```

Response:

```
{
   "data":{
      "labels": [
```

```
                {
                  "labelid":1,
                  "name":"USA TRIP",
                  "items": [1, 2, 3]
                }
               ]
   }
}
```

Example 2

Request body:

```
{
   "data":{
      "labelids": [1, 2]
   }
}
```

Response:

```
{
   "data":{
      "labels": [
                {
                  "labelid":1,
                  "name":"USA TRIP",
                  "items": [1, 2, 3]
                },
                {
                  "labelid":2,
                  "name":"UK TRIP",
                  "items": [4, 1, 3]
                }
              ]
   }
}
```

Example 3

Request body:

```
{
   "data":{
   }
}

or empty body
```

Response:

```
{
   "data":{
      "labels": [
                {
```

```
                    "labelid":1,
                    "name":"USA TRIP",
                    "items": [1, 2, 3]
                  },
                  {
                    "labelid":2,
                    "name":"UK TRIP",
                    "items": [4, 1, 3]
                  }
                ]
   }
}
```

Example 4

Request:

```
POST /sapi/label?action=get&type=picture
```

Request body:

```
{
   "data":{
      "labelids": [1, 2]
   }
}
```

Response:

```
{
   "data":{
      "labels": [
            {
              "labelid":1,
              "name":"USA TRIP",
              "items": [1, 2, 3]
            },
            {
              "labelid":2,
              "name":"UK TRIP",
              "items": [4, 1, 3]
            }
         ]
   }
}
```

Errors

LAB-1000, in addition to generic errors described in Section 4.1.3, “Errors across multiple Server APIs”.

3.5.35 Change validation status

Allow the administrator to change the validation status of an item. This API call can be used for the asynchronous content validation, and not on a partial uploaded item (metadata only or partial binary.)

Security realm: system

Definition

```
POST /sapi/media?action=change-validation-status
```

Parameters

- *id*: (**mandatory**) the item ID of which the administrator want to change the status
- *status*: (**mandatory**) the new status of the item (allowed values are: U (uploaded), C (copyrighted, means not sharable), I (Illicit, means not available at all), V (not yet validated, not available until other status will be set)
- *userid*: (**mandatory**) the owner of the item specified in the *id* parameter

Request example

```
sapi/media?action=change-validation-
status&id=12345&status=U&userid=username
```

Response

The 200 HTTP code indicates a successful request.

Errors

MED-1005, MED-1013, MED-1015, in addition to generic errors described in Section 4.1.3, “Errors across multiple Server APIs”.

3.5.36 Delete all soft deleted items

Delete all soft deleted items for the user, regardless of their media type.

Security realm: user

Definition

```
POST /sapi/media?action=emptytrash
```

Parameters

None.

Response

The `200` HTTP code indicates a successful request.

Response body

```
{
  "success:":"Media removed successfully"
}
```

Errors

COM-1013, MED-1000, MED-1014, MED-1016, MED-1017, in addition to generic errors described in Section 4.1.3, “Errors across multiple Server APIs”.

3.5.37 Delete all soft deleted pictures

Delete all soft deleted pictures for the user.

Security realm: user

Definition

```
POST /sapi/media/picture?action=emptytrash
```

Parameters

None.

Response

The `200` HTTP code indicates a successful request.

Response body

```
{
  "success:":"Pictures removed successfully"
}
```

Errors

COM-1013, MED-1000, MED-1014, MED-1016, MED-1017, in addition to generic errors described in Section 4.1.3, “Errors across multiple Server APIs”.

3.5.38 Delete all soft deleted videos

Delete all soft deleted videos for the user.

Security realm: user

Definition

```
POST /sapi/media/video?action=emptytrash
```

Parameters

None.

Response

The `200` HTTP code indicates a successful request.

Response body

```
{
```

```
  "success:":"Videos removed successfully"
}
```

Errors

COM-1013, MED-1000, MED-1014, MED-1016, MED-1017, in addition to generic errors described in Section 4.1.3, “Errors across multiple Server APIs”.

3.5.39 Delete all soft deleted files

Delete all soft deleted files for the user.

Security realm: user

Definition

```
POST /sapi/media/file?action=emptytrash
```

Parameters

None.

Response

The `200` HTTP code indicates a successful request.

Response body

```
{
  "success:":"Files removed successfully"
}
```

Errors

COM-1013, MED-1000, MED-1014, MED-1016, MED-1017, in addition to generic errors described in Section 4.1.3, “Errors across multiple Server APIs”.

3.5.40 Delete all soft deleted audio items

Delete all soft deleted audio items for the user.

Security realm: user

Definition

```
POST /sapi/media/audio?action=emptytrash
```

Parameters

None.

Response

The `200` HTTP code indicates a successful request.

Response body

```
{
  "success:":"Audio removed successfully"
}
```

Errors

COM-1013, MED-1000, MED-1014, MED-1016, MED-1017, in addition to generic errors described in Section 4.1.3, “Errors across multiple Server APIs”.

3.5.41 Restore soft deleted items

Restore soft deleted items for the user, regardless of their media type.

Security realm: user

Definition

```
POST /sapi/media?action=restore
```

Parameters

None.

Request body

- *`data`*: the JSON object containing an optional array named `pictures`, an optional array named `videos`, an optional array named `files`, an optional array named `audios`, with the IDs of the items to be restored. At least one array must be defined.

Example

Request body

```
{
   "data":{
      "pictures":[
         1,
         2
      ],
      "videos":[
         3,
         4
      ],
      "files":[
         5,
         6
      ],
      "audios":[
         7,
         8
      ]
   }
}
```

Response

A JSON object containing a success message or an error code.

Example response

```
{
   "success:":"Media restored successfully"
}
```

Errors

COM-1011, COM-1016, COM-1021, MED-1000, MED-1005, MED-1013, MED-1016, MED-1017, MED-1023, in addition to generic errors described in Section 4.1.3, “Errors across multiple Server APIs”.

3.5.42 Restore soft deleted pictures

Restore soft deleted pictures for the user.

Security realm: user

Definition

```
POST /sapi/media/picture?action=restore
```

Parameters

None.

Request body

- *`data`*: the JSON object containing an array named `pictures` with the IDs of the pictures to be restored.

Example

Request body

```
{
   "data":{
      "pictures":[
         1,
         2
      ]
   }
}
```

Response

A JSON object containing a success message or an error code.

Example response

```
{
   "success:":"Pictures restored successfully"
```

```
}
```

Errors

COM-1011, COM-1016, COM-1021, MED-1000, MED-1005, MED-1013, MED-1016, MED-1017, MED-1023, in addition to generic errors described in Section 4.1.3, “Errors across multiple Server APIs”.

3.5.43 Restore soft deleted videos

Restore soft deleted videos for the user.

Security realm: user

Definition

```
POST /sapi/media/video?action=restore
```

Parameters

None.

Request body

- *`data`*: the JSON object containing an array named `videos` with the IDs of the videos to be restored.

Example

Request body

```
{
   "data":{
      "videos":[
         3,
         4
      ]
   }
}
```

Response

A JSON object containing a success message or an error code.

Example response

```
{
   "success:":"Videos restored successfully"
}
```

Errors

COM-1011, COM-1016, COM-1021, MED-1000, MED-1005, MED-1013, MED-1016, MED-1017, MED-1023, in addition to generic errors described in Section 4.1.3, “Errors across multiple Server APIs”.

3.5.44 Restore soft deleted files

Restore soft deleted files for the user.

Security realm: user

Definition

```
POST /sapi/media/file?action=restore
```

Parameters

None.

Request body

- *`data`*: the JSON object containing an array named `files` with the IDs of the files to be restored.

Example

Request body

```
{
   "data":{
      "files":[
         5,
         6
      ]
   }
}
```

Response

A JSON object containing a success message or an error code.

Example response

```
{
   "success:":"Files restored successfully"
}
```

Errors

COM-1011, COM-1016, COM-1021, MED-1000, MED-1005, MED-1013, MED-1016, MED-1017, MED-1023, in addition to generic errors described in Section 4.1.3, “Errors across multiple Server APIs”.

3.5.45 Restore soft deleted audio items

Restore soft deleted audio items for the user.

Security realm: user

Definition

```
POST /sapi/media/audio?action=restore
```

Parameters

None.

Request body

- *`data`*: the JSON object containing an array named `audios` with the IDs of the audio items to be restored.

Example

Request body

```
{
   "data":{
      "audios":[
         7,
         8
      ]
   }
}
```

Response

A JSON object containing a success message or an error code.

Example response

```
{
   "success:":"Audios restored successfully"
}
```

Errors

COM-1011, COM-1016, COM-1021, MED-1000, MED-1005, MED-1013, MED-1016, MED-1017, MED-1023, in addition to generic errors described in Section 4.1.3, “Errors across multiple Server APIs”.

3.5.46 Upload binary files

These API calls can be used to upload a binary file to the server.

3.5.46.1 Direct upload

Allow a direct upload in a single request (using multipart) of the binary content and the information about the file, as size, encoding, name, etc.

Security realm: user

Definition

```
POST /sapi/upload?action=save
```

The following API calls are **DEPRECATED**:

```
POST /sapi/upload/picture?action=save
```

or

```
POST /sapi/upload/video?action=save
```

or

```
POST /sapi/upload/file?action=save
```

Parameters

- *type*: (**optional**) the type of the uploaded item. It may be `picture`, `video`, `audio` or `file`.

Request

Parameters in multipart request body as described in Section 5.57, “Upload binary files”.

Formats

See Appendix D, *Media Formats*.

Response

A JSON object with a success message, ID, type, status, ETag, and all the metadata (like thumbnails, Exif, URL for the download and so on) about the uploaded media item.

Automatic content type and type detection

All uploaded content has to have a content type. If no content type has been specified, the server will detect it based on the uploaded file's extension. The server uses the list of MIME mappings specified either in the `ROOT` web application's `web.xml` file or in the server `web.xml` file. If the server is not able to find a matching content type for a file extension, it will use the content type '`application/octet-stream`'.

If no type request parameter has been specified, the type of the uploaded item will be based on the item's file extension.

The items with one of the following file extensions will have type *picture*:

```
.gif
.jfif
.jif
.jpe
.jpeg
.jpg
.png
```

The items with one of the following file extensions will have type *video*:

```
.3g2
.3gp
.asf
.avi
.flv
.m4u
.m4v
.mov
.movie
.mp2
.mp4
.mpa
```

```
.mpe
.mpeg
.mpg
.wmv
```

The items with one of the following file extensions will have type *audio*:

```
.aac
.m4a
.mp3
```

Items with all other file extensions will have type *file*.

As sync sources may not be present (the various types map to sync sources of the same name) the server will try to map video, picture and audio to the type '`file`' if video or file or audio have been deactivated and no *`type`* parameter has been specified.

Examples

Example of the JSON parameter:

```
{
   "data":{
      "name":"pic45634.jpg",
      "creationdate":"20110101T012030Z",
      "modificationdate":"20110101T012030Z",
      "contenttype":"image/jpeg",
      "size":23453,
      "folderid":1
   }
}
```

Example of complete request body:

```
POST /sapi/upload?action=save HTTP/1.1
Host: my.server.com
Authorization: Basic Z3Vlc3Q6Z3Vlc3Q=
...
Content-Type:multipart/form-
data;boundary=--------------------------130661695710023513016786454 82
Content-Length: 48028

----------------------------130661695710023513016786454 82
Content-Disposition: form-data; name="data"

{
    "data":{
      "name":"pic45634.jpg",
      "creationdate":20110101T012030Z,
      "modificationdate":20110101T012030Z,
      "contenttype":"image/jpeg",
      "size":23453,
      "folderid":1
    }
}
```

```
----------------------------130661695710023513016786454 82
Content-Disposition: form-data; name="callback"

window.parent.Zapatec.Widget.getWidgetById(74).serverSendOnLoad({

----------------------------130661695710023513016786454 82
Content-Disposition: form-data; name="file"; filename="gasoleo.jpg"
Content-Type: image/jpeg

<file content here>

----------------------------130661695710023513016786454 82--
```

Example response without callback:

```
{
   "success":"Media uploaded successfully",
   "id":"410919",
   "type":"picture",
   "status":"U",
   "etag":"aafqzchtgdvLsw7zhifZKg==",
   "metadata":{
      "mediaserverurl":"https://my.server.com",
      "pictures":[
         {
            "id":"1100",
            "url":"/picture/tn/at/zn/px/gn/ig/gm/id/
cGFwaXVzZXIx/09cvkqyb1rq8n-1100.jpg",
            "viewurl":"/picture/tn/at/zn/px/gn/ig/gm/id/
cGFwaXVzZXIx/09cvkqyb1rq8n-1100.jpg-ext/504.jpg",
            "date":1353584263002,
            "creationdate":1205409301000,
            "modificationdate":1205409301000,
            "size":43598,
            "name":"pic45634.jpg",
            "exif":{
               "Make":"'Panasonic'",
               "Model":"'DMC-FX3'",
               "Orientation":"0",
               "XResolution":"72",
               "YResolution":"72",
               "Resolution Unit":"2",
               "Software":"'Ver.1.0'",
               "Date Time":"'2008:03:13 11:55:01'",
               "YCbCr Positioning":"2",
               "Exif Offset":"418",
               "Print IM":"80, 114, 105, 110, 116, 73, 77, 0, 48, 50,
 53, 48, 0, 0, 14, 0, 1, 0, 22, 0, 22, 0, 2, 0, 0, 0, 0, 0, 3, 0,
 100, 0, 0, 0, 7, 0, 0, 0, 0, 0, 8, 0, 0, 0, 0, 0, 9, 0, 0, 0, 0...
 (208)",
               "Exposure Time":"10/100 (0.1)",
               "FNumber":"28/10 (2.8)",
               "Exposure Program":"2",
               "ISO":"200",
```

```
               "Exif Version":"48, 50, 50, 49",
               "Date Time Original":"'2008:03:13 11:55:01'",
               "Create Date":"'2008:03:13 11:55:01'",
               "Components Configuration":"1, 2, 3, 0",
               "Compressed Bits Per Pixel":"4",
               "Exposure Compensation":"0",
               "Max Aperture Value":"3",
               "Metering Mode":"5",
               "Light Source":"0",
               "Flash":"16",
               "Focal Length":"58/10 (5.8)",
               "Maker Note":"80, 97, 110, 97, 115, 111, 110, 105,
 99, 0, 0, 0, 42, 0, 1, 0, 3, 0, 1, 0, 0, 0, 2, 0, 0, 0, 2, 0, 7,
 0, 4, 0, 0, 0, 0, 1, 0, 8, 3, 0, 3, 0, 1, 0, 0, 0, 1, 0, 0, 0, 7...
 (5702)",
               "Flashpix Version":"48, 49, 48, 48",
               "Color Space":"1",
               "Exif Image Width":"2816",
               "Exif Image Length":"2112",
               "Interop Offset":"6630",
               "Sensing Method":"2",
               "File Source":"3",
               "Scene Type":"1",
               "Custom Rendered":"0",
               "Exposure Mode":"0",
               "White Balance":"0",
               "Digital Zoom Ratio":"0",
               "Focal Length In 3 5mm Format":"35",
               "Scene Capture Type":"0",
               "Gain Control":"1",
               "Contrast":"0",
               "Saturation":"1",
               "Sharpness":"1",
               "Interop Index":"'R98'",
               "Interop Version":"48, 49, 48, 48",
               "Compression":"6",
               "Jpg From Raw Start":"7060",
               "Jpg From Raw Length":"6944"
            },
            "thumbnails":[
               {
                  "size":"176",
                  "url":"/picture/tn/at/zn/px/gn/ig/gm/id/
cGFwaXVzZXIx/09cvkqyb1rq8n-1100.jpg-ext/176.jpg",
                  "etag":"fjeYc9kO4kQmOe8ePBlPrA=="
               },
               {
                  "size":"504",
                  "url":"/picture/tn/at/zn/px/gn/ig/gm/id/
cGFwaXVzZXIx/09cvkqyb1rq8n-1100.jpg-ext/504.jpg",
                  "etag":"fjeYc9kO4kQmOe8ePBlPrA=="
               }
            ],
            "mediatype":"picture",
```

```
            "status":"U",
            "etag":"fjeYc9kO4kQmOe8ePBlPrA==",
            "folderid":1,
            "softdeleted":false
         }
      ]
   }
}
```

Example response with callback:

```
<html><body><script type='text/javascript'>
window.parent.Zapatec.Widget.getWidgetById(74).serverSendOnLoad(
{"success":"Media uploaded successfully","id":"410919","status":
 "U","etag":"aafqzchtgdvLsw7zhifZKg=="}
)</script></body></html>
```

Errors

COM-1020, FOL-1004, FOL-1006, MED-1007, MED-1016, MED-1017, MED-1020 in addition to generic errors described in Section 4.1.3, “Errors across multiple Server APIs”

3.5.46.2 Resumable upload

Perform a resumable upload. It can be used as a way to recover from a failure. Using this method you need to perform two requests to the server: upload of metadata, and upload of binary data. The first request will return the ID of the file on the server, and using this ID the client can start the upload.

Security realm: user

Definition of the first request

```
POST /sapi/upload/picture?action=save-metadata
```

or

```
POST /sapi/upload/video?action=save-metadata
```

or

```
POST /sapi/upload/file?action=save-metadata
```

or

```
POST /sapi/upload/audio?action=save-metadata
```

Example of the body, a JSON object with the upload item as per Section 5.58, “Upload item”:

```
{
   "data":{
      "name":"pic45634.jpg",
      "creationdate":"20110101T012030Z",
      "modificationdate":"20110101T012030Z",
      "contenttype":"image/jpeg",
      "size":23453,
```

```
      "folderid":1
   }
}
```

Response

A JSON object with the ID of the new media item.

For example:

```
{
   "success":"Media metadata saved successfully",
   "id":"410919",
   "status": "U"
}
```

Definition of the second request

```
POST /sapi/upload/picture?action=save
```

or

```
POST /sapi/upload/video?action=save
```

or

```
POST /sapi/upload/file?action=save
```

or

```
POST /sapi/upload/audio?action=save
```

Note

The API calls listed above are not yet deprecated in this context. They are deprecated only in the context described in Section 3.5.46.1, “Direct upload”

For the request, the following HTTP headers are required:

- `x-funambol-file-size`: the file size
- `x-funambol-id`: the ID of the item

In case `Content-Range` (the range of bytes to upload) is specified as /filesize, the `x-funambol-file-size` is not mandatory.

Example 1

```
POST /sapi/upload/picture?action=save
Host: my.server.com
Authorization: Basic Z3Vlc3Q6Z3Vlc3Q=
x-funambol-id:  410919
x-funambol-file-size:44351
```

```
Content-Length: 81047
Content-Type: image/jpeg


... file contents here ...
```

Example 2

The following request describes an upload with resume, the client is resuming the upload from byte 2345 onwards:

```
POST /sapi/upload/picture?action=save
Host: my.server.com
Authorization: Basic Z3Vlc3Q6Z3Vlc3Q=
x-funambol-id:  410919
x-funambol-file-size:44351
Content-Type: image/jpeg
Content-Range: bytes 2345-44350/44351
Content-Lenght: 42006

kjhhihaaèmpègfoar.......
```

Consider that the Content-Length header contains the number of bytes actually exchanged during the current post. Besides, it must be omitted if you specify the Transfer-Encoding header.

Example response:

```
{
   "success":"Media uploaded successfully",
   "id":"410919",
   "status":"U",
   "etag":"aafqzchtgdvLsw7zhifZKg==",
   "metadata":{
      "mediaserverurl":"https://my.server.com",
      "pictures":[
         {
            "id":"1100",
            "url":"/picture/tn/at/zn/px/gn/ig/gm/id/
cGFwaXVzZXIx/09cvkqyb1rq8n-1100.jpg",
            "viewurl":"/picture/tn/at/zn/px/gn/ig/gm/id/
cGFwaXVzZXIx/09cvkqyb1rq8n-1100.jpg-ext/504.jpg",
            "date":1353584263002,
            "creationdate":1205409301000,
            "modificationdate":1205409301000,
            "size":43598,
            "name":"pic45634.jpg",
            "exif":{
               "Make":"'Panasonic'",
               "Model":"'DMC-FX3'",
               "Orientation":"0",
               "XResolution":"72",
               "YResolution":"72",
               "Resolution Unit":"2",
               "Software":"'Ver.1.0'",
               "Date Time":"'2008:03:13 11:55:01'",
```

```
               "YCbCr Positioning":"2",
               "Exif Offset":"418",
               "Print IM":"80, 114, 105, 110, 116, 73, 77, 0, 48, 50,
53, 48, 0, 0, 14, 0, 1, 0, 22, 0, 22, 0, 2, 0, 0, 0, 0, 0, 3, 0,
100, 0, 0, 0, 7, 0, 0, 0, 0, 0, 8, 0, 0, 0, 0, 0, 9, 0, 0, 0, 0...
(208)",
               "Exposure Time":"10/100 (0.1)",
               "FNumber":"28/10 (2.8)",
               "Exposure Program":"2",
               "ISO":"200",
               "Exif Version":"48, 50, 50, 49",
               "Date Time Original":"'2008:03:13 11:55:01'",
               "Create Date":"'2008:03:13 11:55:01'",
               "Components Configuration":"1, 2, 3, 0",
               "Compressed Bits Per Pixel":"4",
               "Exposure Compensation":"0",
               "Max Aperture Value":"3",
               "Metering Mode":"5",
               "Light Source":"0",
               "Flash":"16",
               "Focal Length":"58/10 (5.8)",
               "Maker Note":"80, 97, 110, 97, 115, 111, 110, 105,
99, 0, 0, 0, 42, 0, 1, 0, 3, 0, 1, 0, 0, 0, 2, 0, 0, 0, 2, 0, 7,
0, 4, 0, 0, 0, 0, 1, 0, 8, 3, 0, 3, 0, 1, 0, 0, 0, 1, 0, 0, 0, 7...
(5702)",
               "Flashpix Version":"48, 49, 48, 48",
               "Color Space":"1",
               "Exif Image Width":"2816",
               "Exif Image Length":"2112",
               "Interop Offset":"6630",
               "Sensing Method":"2",
               "File Source":"3",
               "Scene Type":"1",
               "Custom Rendered":"0",
               "Exposure Mode":"0",
               "White Balance":"0",
               "Digital Zoom Ratio":"0",
               "Focal Length In 3 5mm Format":"35",
               "Scene Capture Type":"0",
               "Gain Control":"1",
               "Contrast":"0",
               "Saturation":"1",
               "Sharpness":"1",
               "Interop Index":"'R98'",
               "Interop Version":"48, 49, 48, 48",
               "Compression":"6",
               "Jpg From Raw Start":"7060",
               "Jpg From Raw Length":"6944"
            },
            "thumbnails":[
               {
                  "size":"176",
                  "url":"/picture/tn/at/zn/px/gn/ig/gm/id/
cGFwaXVzZXIx/09cvkqyb1rq8n-1100.jpg-ext/176.jpg",
```

```
                  "etag":"fjeYc9kO4kQmOe8ePBlPrA=="
               },
               {
                  "size":"504",
                  "url":"/picture/tn/at/zn/px/gn/ig/gm/id/
cGFwaXVzZXIx/09cvkqyb1rq8n-1100.jpg-ext/504.jpg",
                  "etag":"fjeYc9kO4kQmOe8ePBlPrA=="
               }
            ],
            "mediatype":"picture",
            "status":"U",
            "etag":"fjeYc9kO4kQmOe8ePBlPrA==",
            "folderid":1,
            "softdeleted":false
         }
      ]
   }
}
```

Example 3

The following request describes a query to understand how much of a resource has already been uploaded to the server:

```
POST /sapi/upload/picture?action=save
Host: my.server.com
x-funambol-id:  410919
Authorization: Basic Z3Vlc3Q6Z3Vlc3Q=
Content-Range: bytes */44351
Content-Lenght: 0
```

In this case, the server will respond as follows:

```
HTTP/1.1 308 Resume Incomplete
Range: 0-3234
Content-Lenght: 0
```

Example 4

The following request describes a query to understand how much of a resource has already been uploaded to the server:

```
POST /sapi/upload/picture?action=save
Host: my.server.com
x-funambol-id:  410919
Authorization: Basic Z3Vlc3Q6Z3Vlc3Q=
Content-Range: bytes */44351
Content-Lenght: 0
```

In this case, the upload ended successfully, so there's no need to resume it:

```
HTTP/1.1 200 OK
Range: 0-44351
Content-Lenght: 0
```

Errors

COM-1008, COM-1011, COM-1014, COM-1015, FOL-1004, FOL-1006, MED-1000, MED-1001, MED-1002, MED-1006, MED-1007, MED-1016, MED-1017, MED-1020 in addition to generic errors described in Section 4.1.3, “Errors across multiple Server APIs”

3.5.46.3 Direct upload for file content update

Allow a direct upload in a single request using multipart of the binary content and the information of the file, as size, encoding, name, etc. in order to update the binary content of an existing file on the server. In this case, the term "file" means a picture, video, audio, or a generic file.

Security realm: user

Definition

```
POST /sapi/upload/file?action=save
```

Important

This API call is **not yet deprecated in this context**. It is deprecated only in the context described in Section 3.5.46.1, “Direct upload”

Request

Parameters in multipart request body as described in Section 5.57, “Upload binary files”.

Response

A JSON object with a success message, ID, type, status, ETag, and all the metadata (like thumbnails, Exif, URL for the download and so on) about the uploaded media item.

Examples

Example of the JSON parameter:

```
{
   "data":{
      "id":"1234",
      "name":"pic45634.jpg",
      "creationdate":"20110101T012030Z",
      "modificationdate":"20110101T012030Z",
      "contenttype":"image/jpeg",
      "size":23453,
      "folderid":1
   }
}
```

Example of complete request body:

```
POST /sapi/upload/picture?action=add HTTP/1.1
Host: my.server.com
Authorization: Basic Z3Vlc3Q6Z3Vlc3Q=
```

```
...
Content-Type:multipart/form-
data;boundary=--------------------------130661695710023513016786454 82
Content-Length: 48028

----------------------------130661695710023513016786454 82
Content-Disposition: form-data; name="data"

{
    "data":{
        "id":"1234",
        "name":"pic45634.jpg",
        "creationdate":20110101T012030Z,
        "modificationdate":20110101T012030Z,
        "contenttype":"image/jpeg",
        "size":23453,
        "folderid":1
    }
}
----------------------------130661695710023513016786454 82
Content-Disposition: form-data; name="callback"
```

```
window.parent.Zapatec.Widget.getWidgetById(74).serverSendOnLoad({
```

```
----------------------------130661695710023513016786454 82
Content-Disposition: form-data; name="file"; filename="gasoleo.jpg"
Content-Type: image/jpeg

<file content here>

----------------------------130661695710023513016786454 82--
```

Example response without callback:

```
{
   "success":"Media uploaded successfully",
   "id":"410919",
   "status":"U",
   "etag":"aafqzchtgdvLsw7zhifZKg==",
   "metadata":{
      "mediaserverurl":"https://my.server.com",
      "pictures":[
         {
            "id":"1100",
            "url":"/picture/tn/at/zn/px/gn/ig/gm/id/
cGFwaXVzZXIx/09cvkqyb1rq8n-1100.jpg",
            "viewurl":"/picture/tn/at/zn/px/gn/ig/gm/id/
cGFwaXVzZXIx/09cvkqyb1rq8n-1100.jpg-ext/504.jpg",
            "date":1353584263002,
            "creationdate":1205409301000,
            "modificationdate":1205409301000,
            "size":43598,
            "name":"pic45634.jpg",
            "exif":{
```

```
                "Make":"'Panasonic'",
                "Model":"'DMC-FX3'",
                "Orientation":"0",
                "XResolution":"72",
                "YResolution":"72",
                "Resolution Unit":"2",
                "Software":"'Ver.1.0'",
                "Date Time":"'2008:03:13 11:55:01'",
                "YCbCr Positioning":"2",
                "Exif Offset":"418",
                "Print IM":"80, 114, 105, 110, 116, 73, 77, 0, 48, 50,
53, 48, 0, 0, 14, 0, 1, 0, 22, 0, 22, 0, 2, 0, 0, 0, 0, 0, 3, 0,
100, 0, 0, 0, 7, 0, 0, 0, 0, 0, 8, 0, 0, 0, 0, 0, 9, 0, 0, 0, 0...
(208)",
                "Exposure Time":"10/100 (0.1)",
                "FNumber":"28/10 (2.8)",
                "Exposure Program":"2",
                "ISO":"200",
                "Exif Version":"48, 50, 50, 49",
                "Date Time Original":"'2008:03:13 11:55:01'",
                "Create Date":"'2008:03:13 11:55:01'",
                "Components Configuration":"1, 2, 3, 0",
                "Compressed Bits Per Pixel":"4",
                "Exposure Compensation":"0",
                "Max Aperture Value":"3",
                "Metering Mode":"5",
                "Light Source":"0",
                "Flash":"16",
                "Focal Length":"58/10 (5.8)",
                "Maker Note":"80, 97, 110, 97, 115, 111, 110, 105,
99, 0, 0, 0, 42, 0, 1, 0, 3, 0, 1, 0, 0, 0, 2, 0, 0, 0, 2, 0, 7,
0, 4, 0, 0, 0, 0, 1, 0, 8, 3, 0, 3, 0, 1, 0, 0, 0, 1, 0, 0, 0, 7...
(5702)",
                "Flashpix Version":"48, 49, 48, 48",
                "Color Space":"1",
                "Exif Image Width":"2816",
                "Exif Image Length":"2112",
                "Interop Offset":"6630",
                "Sensing Method":"2",
                "File Source":"3",
                "Scene Type":"1",
                "Custom Rendered":"0",
                "Exposure Mode":"0",
                "White Balance":"0",
                "Digital Zoom Ratio":"0",
                "Focal Length In 3 5mm Format":"35",
                "Scene Capture Type":"0",
                "Gain Control":"1",
                "Contrast":"0",
                "Saturation":"1",
                "Sharpness":"1",
                "Interop Index":"'R98'",
                "Interop Version":"48, 49, 48, 48",
                "Compression":"6",
```

```
                "Jpg From Raw Start":"7060",
                "Jpg From Raw Length":"6944"
             },
             "thumbnails":[
                {
                   "size":"176",
                   "url":"/picture/tn/at/zn/px/gn/ig/gm/id/
cGFwaXVzZXIx/09cvkqyb1rq8n-1100.jpg-ext/176.jpg",
                   "etag":"fjeYc9kO4kQmOe8ePBlPrA=="
                },
                {
                   "size":"504",
                   "url":"/picture/tn/at/zn/px/gn/ig/gm/id/
cGFwaXVzZXIx/09cvkqyb1rq8n-1100.jpg-ext/504.jpg",
                   "etag":"fjeYc9kO4kQmOe8ePBlPrA=="
                }
             ],
             "mediatype":"picture",
             "status":"U",
             "etag":"fjeYc9kO4kQmOe8ePBlPrA==",
             "folderid":1,
             "softdeleted":false
          }
       ]
    }
}
```


Example response with callback:


```
<html><body><script type='text/javascript'>
window.parent.Zapatec.Widget.getWidgetById(74).serverSendOnLoad(
{"success":"Media uploaded successfully","id":"410919","status":
 "U","etag":"aafqzchtgdvLsw7zhifZKg=="}
)</script></body></html>
```


Errors

FOL-1004, FOL-1006, MED-1005, MED-1016, MED-1017, MED-1020 in addition to generic errors described in Section 4.1.3, “Errors across multiple Server APIs”

3.5.46.4 File update with resumable upload

Update the binary content or the associated metadata of an exiting media item. The media item could be a picture, a video, an audio file, or a generic file. Using this API call it is possible to update the metadata of the items on the server. If the update is supposed to include also the binary data, then two requests are needed: upload of the metadata, and upload of the binary data. If the upload of the binary data is needed, then the JSON object sent to server must include the new size – this will delete the old binary data and indicate to the server that it should expect the second request with the data.

Security realm: user

Definition of the first request

```
POST /sapi/upload/file?action=save-metadata
```

Parameters

- *`lastupdate`*: if `true` the last update timestamp will be included in the response

Note

The file extension cannot be changed by this API, so if the filename is sent without extension or with a different one, it will be replaced by the old extension.

Request body

Examples of the body, a JSON object with the *Upload item* as per Section 5.58, “Upload item”, follow.

Only the metadata is updated:

```
{
   "data":{
      "id":"1234",
      "name":"pic45634.jpg",
      "creationdate":"20110101T012030Z",
      "modificationdate":"20110101T012030Z",
      "contenttype":"image/jpeg"
   }
}
```

Updating the metadata and the binary data (size of the new binary data must be included):

```
{
   "data":{
      "id":"1234",
      "name":"pic45634.jpg",
      "creationdate":"20110101T012030Z",
      "modificationdate":"20110101T012030Z",
      "contenttype":"image/jpeg",
      "size":23453,
      "folderid":1
   }
}
```

Response in both cases

For example:

```
{
   "success":"Media metadata saved successfully",
   "id":"1234",
   "status": "U"
}
```

Definition of the second request, to upload the binary data (when the binary update is required)

```
POST /sapi/upload/file?action=save
```

Note

This API call is not yet deprecated in this context. It is deprecated only in the context described in Section 3.5.46.1, “Direct upload”

For the request, the following HTTP headers are required:

- `x-funambol-file-size`: the file size
- `x-funambol-id`: the ID of the item

In case `Content-Range` (the range of bytes to upload) is specified as /filesize, the `x-funambol-file-size` is not mandatory.

Example 1

```
POST /sapi/upload/picture?action=save
Host: my.server.com
Authorization: Basic Z3Vlc3Q6Z3Vlc3Q=
x-funambol-id:  1234
x-funambol-file-size: 44351
Content-Length: 81047
Content-Type: image/jpeg

... file contents here ...
```

Example 2

The following request describes an upload with resume, the client is resuming the upload from byte 2345 onwards:

```
POST /sapi/upload/picture?action=save
Host: my.server.com
Authorization: Basic Z3Vlc3Q6Z3Vlc3Q=
x-funambol-id:  1234
x-funambol-file-size: 44351
Content-Type: image/jpeg
Content-Range: bytes 2345-44350/44351
Content-Lenght: 42006

kjhhihaaèmpègfoar.......
```

Consider that the `Content-Length` header contains the number of bytes actually exchanged during the current post. Besides, it must be omitted if you specify the `Transfer-Encoding` header.

Example response:

```
{
   "success":"Media uploaded successfully",
   "id":"1234",
   "status":"U",
   "etag":"aafqzchtgdvLsw7zhifZKg==",
   "metadata":{
      "mediaserverurl":"https://my.server.com",
      "pictures":[
```

```
         {
            "id":"1100",
            "url":"/picture/tn/at/zn/px/gn/ig/gm/id/
cGFwaXVzZXIx/09cvkqyb1rq8n-1100.jpg",
            "viewurl":"/picture/tn/at/zn/px/gn/ig/gm/id/
cGFwaXVzZXIx/09cvkqyb1rq8n-1100.jpg-ext/504.jpg",
            "date":1353584263002,
            "creationdate":1205409301000,
            "modificationdate":1205409301000,
            "size":43598,
            "name":"pic45634.jpg",
            "exif":{
               "Make":"'Panasonic'",
               "Model":"'DMC-FX3'",
               "Orientation":"0",
               "XResolution":"72",
               "YResolution":"72",
               "Resolution Unit":"2",
               "Software":"'Ver.1.0'",
               "Date Time":"'2008:03:13 11:55:01'",
               "YCbCr Positioning":"2",
               "Exif Offset":"418",
               "Print IM":"80, 114, 105, 110, 116, 73, 77, 0, 48, 50,
 53, 48, 0, 0, 14, 0, 1, 0, 22, 0, 22, 0, 2, 0, 0, 0, 0, 0, 3, 0, 100,
 0, 0, 0, 7, 0, 0, 0, 0, 0, 8, 0, 0, 0, 0, 0, 9, 0, 0, 0, 0... (208)",
               "Exposure Time":"10/100 (0.1)",
               "FNumber":"28/10 (2.8)",
               "Exposure Program":"2",
               "ISO":"200",
               "Exif Version":"48, 50, 50, 49",
               "Date Time Original":"'2008:03:13 11:55:01'",
               "Create Date":"'2008:03:13 11:55:01'",
               "Components Configuration":"1, 2, 3, 0",
               "Compressed Bits Per Pixel":"4",
               "Exposure Compensation":"0",
               "Max Aperture Value":"3",
               "Metering Mode":"5",
               "Light Source":"0",
               "Flash":"16",
               "Focal Length":"58/10 (5.8)",
               "Maker Note":"80, 97, 110, 97, 115, 111, 110, 105, 99, 0,
 0, 0, 42, 0, 1, 0, 3, 0, 1, 0, 0, 0, 2, 0, 0, 0, 2, 0, 7, 0, 4, 0, 0,
 0, 0, 1, 0, 8, 3, 0, 3, 0, 1, 0, 0, 0, 1, 0, 0, 0, 7... (5702)",
               "Flashpix Version":"48, 49, 48, 48",
               "Color Space":"1",
               "Exif Image Width":"2816",
               "Exif Image Length":"2112",
               "Interop Offset":"6630",
               "Sensing Method":"2",
               "File Source":"3",
               "Scene Type":"1",
               "Custom Rendered":"0",
               "Exposure Mode":"0",
               "White Balance":"0",
```

```
                "Digital Zoom Ratio":"0",
                "Focal Length In 3 5mm Format":"35",
                "Scene Capture Type":"0",
                "Gain Control":"1",
                "Contrast":"0",
                "Saturation":"1",
                "Sharpness":"1",
                "Interop Index":"'R98'",
                "Interop Version":"48, 49, 48, 48",
                "Compression":"6",
                "Jpg From Raw Start":"7060",
                "Jpg From Raw Length":"6944"
             },
             "thumbnails":[
                {
                   "size":"176",
                   "url":"/picture/tn/at/zn/px/gn/ig/gm/id/
cGFwaXVzZXIx/09cvkqyb1rq8n-1100.jpg-ext/176.jpg",
                   "etag":"fjeYc9kO4kQmOe8ePBlPrA=="
                },
                {
                   "size":"504",
                   "url":"/picture/tn/at/zn/px/gn/ig/gm/id/
cGFwaXVzZXIx/09cvkqyb1rq8n-1100.jpg-ext/504.jpg",
                   "etag":"fjeYc9kO4kQmOe8ePBlPrA=="
                }
             ],
             "mediatype":"picture",
             "status":"U",
             "etag":"fjeYc9kO4kQmOe8ePBlPrA==",
             "folderid":1,
             "softdeleted":false
          }
       ]
    }
}
```


Errors

COM-1008, COM-1010, COM-1011, COM-1014, COM-1015, FOL-1004, FOL-1006, MED-1000, MED-1001, MED-1002, MED-1005, MED-1006, MED-1007, MED-1016, MED-1017, MED-1020 in addition to generic errors described in Section 4.1.3, “Errors across multiple Server APIs”

3.5.47 Get timeline

Retrieve a list of JSON objects containing, for each period, the media IDs present on it, ordered by descending date. The periods, without specifying a granularity, will be returned in this way: each "busy" day for the current month, each "busy" month for the current year, each "busy" year for the remaining part.

Security realm: user

Definition

```
GET /sapi/media/timeline?action=get
```

Parameters

- *limit*: (**optional**) the maximum number of families retrieved starting from the latest ones and considering the offset.
- *offset*: (**optional**) the offset (inclusive) of families to be considered in the response.
- *granularity*: (**optional**) the granularity of the periods (can be year, month, day).
- *from*: (**optional**) the date (in UTC milliseconds) starting from which the media should be retrieved.
- *to*: (**optional**) the date (in UTC milliseconds) until the media should be retrieved.
- *type*: (**optional**) the media type you want to retrieve.
- *source*: (**optional**) the type of items returned by the response, the possible values are: `media` (for media items), `family` (for items shared in the user wall) and `folder` (for user folders). Sources `family` and `folder` cannot be used with *type* parameter. Default value is `media`.

Response

A JSON array containing for each period the IDs posted.

Example 1 (all the items in the digital life – API call performed on August 3rd, 2014)

Request

```
GET /sapi/media/timeline?action=get
```

Response

```
{
   "data":{
      "periods":[
         {
            "period":"2014-08-03",
            "ids": [135, 141345]
         },
         {
            "period":"2014-08-01",
            "ids": [1345, 1435]
         },
         {
            "period":"2014-07",
            "ids": [2375, 355]
         },
         {
            "period":"2014-06",
            "ids": [1, 56]
         },
         {
            "period":"2014-03",
            "ids": [134, 124, 8593, 12354, 125, 153, 129]
         },
         {
```

```
            "period":"2013",
            "ids": [5567, 7468, 5647, 8467]
         },
         {
            "period":"2012",
            "ids": [7456]
         }
      ]
   }
}
```

Example 2 (all the pictures in the digital life - daily *granularity*)

Request

```
GET /sapi/media/timeline?action=get&type=picture&granularity=daily
```

Response

```
{
   "data":{
      "periods":[
         {
            "period":"2014-07-30",
            "ids": [2375]
         },
         {
            "period":"2014-06-08",
            "ids": [1]
         },
         {
            "period":"2014-03-21",
            "ids": [134, 124, 8593, 12354, 125, 153, 129]
         },
         {
            "period":"2013-01-08",
            "ids": [5647]
         }
      ]
   }
}
```

Example 3 (all the items shared in the user wall – api called on August 3rd, 2014)

Request

```
GET /sapi/media/timeline?action=get&source=family
```

Response

```
{
   "data":{
      "periods":[
         {
            "period":"2014-08-01",
```

```
            "ids": [1345, 1435]
         }
      ]
   }
}
```

Example 4 (all the pictures in the digital life – monthly *granularity* – *from* parameter)

Request

```
GET /sapi/media/timeline?
action=get&type=picture&granularity=monthly&from=1388561251000
```

Response

```
{
   "data":{
      "periods":[
         {
            "period":"2014-07",
            "ids": [2375]
         },
         {
            "period":"2014-06",
            "ids": [1]
         },
         {
            "period":"2014-03",
            "ids": [134, 124, 8593, 12354, 125, 153, 129]
         }
      ]
   }
}
```

Example 5 (all the folders in the digital life – monthly *granularity* – *limit* parameter)

Request

```
GET /sapi/media/timeline?
action=get&granularity=monthly&limit=3&source=folder
```

Response

```
{
   "data":{
      "periods":[
         {
            "period":"2013-07",
            "ids": [5]
         },
         {
            "period":"2012-06",
            "ids": [6]
```

```
         },
         {
            "period":"2012-03",
            "ids": [23]
         }
      ]
   }
}
```

Errors

COM-1012, FAM-1008, MED-1000, MED-1010, MED-1024, PRO-1101, in addition to generic errors described in Section 4.1.3, “Errors across multiple Server APIs”

3.5.48 Get generic media

Retrieve media or family items information for the user, ordered by descending date. Note that this is a generic media API, not related with a given media type.

Security realm: user

Definition

```
POST /sapi/media?action=get
```

Parameters

- *`limit`*: (**optional**) the maximum number of items retrieved, starting from the latest ones and considering the offset. Cannot be used listing the IDs in the request body.
- *`offset`*: (**optional**) the offset (inclusive) of items to be considered in the response. Cannot be used listing the IDs in the request body.
- *`family`*: (**optional**) `true` or `false`. When present and `true`, the API will return only the items shared in the user's family. If not present or `false`, the API will return only the user media items.

Note

If *`limit`* parameter is not added to the API call, the server will reply with a limited number of items (100).

Note

If a list of IDs is specified in the body, the API response may contain the user media items and items shared with her in the family. The items will be returned in the same order they were requested.

Request body (optional)

A JSON object containing the information about the media to retrieve and/or the list of fields expected in the response.

- *`ids`*: (**optional**) a JSON array containing the list of IDs of the media items.

- *fields*: (**optional**) a JSON array containing the list of fields expected in the response. Keep in mind some fields will be always returned by the server (like *id*, *mediatype*, *status*, *date*, *userid*). The optional fields are: *name*, *url*, *viewurl*, *creationdate*, *modificationdate*, *size*, *labels*, *exif*, *playbackurl*, *etag*, *thumbnails*, *folderid*, *videometadata*, *audiometadata*, *exported*, *shared*, and *postingdate*.

Response

A JSON object with all the requested fields about the requested media items.

Note

The *url* will contain also the server (no *mediaserverurl* field in the JSON).

Note

The API response will contain a parameter *more* that will specify if there are more available items on server than the ones returned in the response (this parameter is not available when the items are requested by list of IDs).

Example 1 (no IDs, full list of fields requested – Note this kind of call will return no more than 100 items)

Request

```
POST sapi/media?action=get
```

Response

```
{
   "data":{
      "media":[
         {
            "id":"0",
            "url":"https://my.server.com/picture/uq/rf/xo/fu/um/dq/
ki/le/dGVzdA==/1ugzzekm3nhrf-0.jpg",
            "viewurl":"https://my.server.com/picture/uq/rf/xo/fu/um/
dq/ki/le/dGVzdA==/1ugzzekm3nhrf-0.jpg-ext/504.jpg",
            "date":1399985134672,
            "creationdate":1072958456000,
            "modificationdate":1161931347000,
            "size":722942,
            "labels":[
               {
                  "labelid":0,
                  "name":"US trip"
               }
            ],
            "name":"100_1924.jpg",
            "exif":{
               "Make":"'EASTMAN KODAK COMPANY'",
               "Model":"'KODAK CX7525 ZOOM DIGITAL CAMERA'",
```

```
                "Orientation":"1",
                "XResolution":"72",
                "Date Time Original":"'2004:01:01 12:00:56'",
                "Create Date":"'2004:01:01 12:00:56'",
                "Exif Image Width":"1920",
                "Exif Image Length":"2560"
             },
             "thumbnails":[
                {
                   "w":192,
                   "h":256,
                   "size":"176",
                   "url":"https://my.server.com/picture/uq/rf/xo/fu/
um/dq/ki/le/dGVzdA==/1ugzzekm3nhrf-0.jpg-ext/176.jpg",
                   "etag":"6W9kgBeRnzrDNH47tgD0qQ=="
                },
                {
                   "w":576,
                   "h":768,
                   "size":"504",
                   "url":"https://my.server.com/picture/uq/rf/xo/fu/
um/dq/ki/le/dGVzdA==/1ugzzekm3nhrf-0.jpg-ext/504.jpg",
                   "etag":"6W9kgBeRnzrDNH47tgD0qQ=="
                }
             ],
             "mediatype":"picture",
             "status":"U",
             "etag":"6W9kgBeRnzrDNH47tgD0qQ==",
             "folder":1,
             "userid":"user1"
          }
       ]
   },
   "more":false
}
```


Example 2 (filtering on fields *url*, *creationdate*, *etag* – Note this kind of call will return no more than 100 items)

Request

```
POST sapi/media?action=get
```

Request body

```
{
   "data":{
      "fields":["url","creationdate","etag"]
   }
}
```

Response


```
{
   "data":{
```

```
      "media":[
         {
            "id":"0",
            "url":"https://my.server.com/picture/uq/rf/xo/fu/um/dq/
ki/le/dGVzdA==/1ugzzekm3nhrf-0.jpg",
            "date":1399985134672,
            "creationdate":1072958456000,
            "mediatype":"picture",
            "status":"U",
            "etag":"6W9kgBeRnzrDNH47tgD0qQ==",
            "userid":"user1"
         }
      ]
   }
}
```

Example 3 (no fields requested – Note this kind of call will return no more than 100 items)

Request

```
POST sapi/media?action=get
```

Request body

```
{
   "data":{
      "fields":[]
   }
}
```

Response

```
{
   "data":{
      "media":[
         {
            "id":"10",
            "date":1399985134672,
            "mediatype":"picture",
            "status":"U",
            "userid":"user1"
         },
         {
            "id":"2",
            "date":1288874023561,
            "mediatype":"file",
            "status":"N",
            "userid":"user2"
         }
         {
            "id":"7",
            "date":1177763912450,
            "mediatype":"video",
            "status":"U",
```

```
            "userid":"user1"
         }
      ]
   },
   "more":false
}
```

Example 4 (specifying IDs, no fields requested)

Request

```
POST sapi/media?action=get
```

Request body

```
{
   "data":{
      "ids":[7,2],
      "fields":[]
   }
}
```

Response

```
{
   "data":{
      "media":[
         {
            "id":"2",
            "date":1288874023561,
            "mediatype":"file",
            "status":"N",
            "userid":"user2"
         }
         {
            "id":"7",
            "date":1177763912450,
            "mediatype":"video",
            "status":"U",
            "userid":"user1"
         }
      ]
   }
}
```

Errors

COM-1021, MED-1000, MED-1003, in addition to generic errors described in Section 4.1.3, “Errors across multiple Server APIs”

3.5.49 Import media from family

Allow the user to import into her Digital Life given media content already shared in the family of the user. The media item will be a copy of the original one, but owned by the user. The new item will be part of the user's quota.

Security realm: user

Definition

```
POST /sapi/media?action=import
```

Parameters

- *id*: (**mandatory**) the ID of the media item shared via Family Cloud to be imported into the user's Digital Life.
- *folderid*: (**optional**) the ID of the folder where the item should be placed after the import.

Response

A JSON object with a success message, ID, type, status, ETag, and all metadata (like thumbnails, Exif, URL for the download, and so on) about the imported media item.

Example

Request

```
POST sapi/media?action=get
```

Response


```
{
   "responsetime":1410789788422,
   "data":{
      "success":"Media imported successfully",
      "id":"607",
      "status":"N",
      "etag":"N2h92oCrtR1qW6eZdV2gFQ==",
      "media":{
         "id":"607",
         "date":1410789788105,
         "mediatype":"picture",
         "status":"N",
         "userid":"user2",
         "url":"https://my.server.com/picture/lv/pd/nk/in/ed/nn/be/
ux/cGFwaXVzZXIy/04jarqhv09y1z.jpg",
         "viewurl":"https://my.server.com/picture/lv/pd/nk/in/ed/nn/
be/ux/cGFwaXVzZXIy/04jarqhv09y1z.jpg-ext/504.jpg",
         "creationdate":1205409301000,
         "modificationdate":1387811719000,
         "size":43598,
         "name":"photo.jpg",
         "exif":{
            "Make":"Panasonic",
            "Model":"DMC-FX3",
            "Orientation":"0",
            "XResolution":"72",
            "Date Time Original":"2008:03:13 11:55:01",
            "Create Date":"2008:03:13 11:55:01",
```

```
            "Exif Image Width":"2816",
            "Exif Image Length":"2112"
         },
         "thumbnails":[
            {
               "w":141,
               "h":106,
               "size":"176",
               "url":"https://my.server.com/picture/lv/pd/nk/in/ed/
nn/be/ux/cGFwaXVzZXIy/04jarqhv09y1z.jpg-ext/176.jpg",
               "etag":"0N2h92oCrtR1qW6eZdV2gFQ=="
            },
            {
               "w":141,
               "h":106,
               "size":"504",
               "url":"https://my.server.com/picture/lv/pd/nk/in/ed/
nn/be/ux/cGFwaXVzZXIy/04jarqhv09y1z.jpg-ext/504.jpg",
               "etag":"0N2h92oCrtR1qW6eZdV2gFQ=="
            }
         ],
         "etag":"N2h92oCrtR1qW6eZdV2gFQ=="
      }
   }
}
```


Errors

COM-1011, FAM-1003, FAM-1004, FAM-1006, FAM-1009, MED-1005, MED-1007, in addition to generic errors described in Section 4.1.3, “Errors across multiple Server APIs”

3.6 External services

This API calls can be used for working with remote services such as social networks, photo sharing sites, etc.

3.6.1 Get services

Return a list of services to which it is possible to export videos and picture media files.

Security realm: user

Definition

```
GET /sapi/externalservice?action=get
```

Parameters

- *`servicename`*: (**optional**) represents the name of the service. A service can be returned only if this parameter is present.
- *`devicetype`*: (**optional**) represents the type of device that is requesting the external services. The server will return the best authorization page for each type of device basing on this parameter. Accepted values are `mobile` and `desktop`, default is `desktop`.

Response body

JSON object described at Section 5.18, “External service”.

Examples

An example with servicename:

```
{
   "data":{
      "servicename":"picasa",
      "displayname":"Picasa Web Albums",
      "authurl":"https://www.picasa.com?key=xpto",
      "authorized":true,
      "authtype":"native",
      "icon":"https://my.server.com/export/icons/picasa.png",
      "hasalbums":true,
      "supportrecipients":false,
      "albumprivacy":[
         "public",
         "private",
         "protected"
      ],
      "itemattributes":[
         "title",
         "description",
         "location",
         "latitude",
         "longitude"
      ],
      "albumattributes":[
         "title",
         "description"
      ],
      "accountname":"john.doe",
      "lastusedalbum":"9475638663638858465",
      "sources":[
         "picture"
      ]
   }
}
```

An example without servicename

```
{
   "data":{
      "services":[
         {
            "servicename":"picasa",
            "displayname":"Picasa Web Albums",
            "authurl":"https://www.picasa.com?key=xpto",
            "authorized":true,
            "authtype":"native",
            "icon":"https://my.server.com/export/icons/picasa.png",
```

```
            "hasalbums":true,
            "supportrecipients":false,
            "albumprivacy":[
               "public",
               "private",
               "protected"
            ],
            "itemattributes":[
               "name",
               "description",
               "location",
               "latitude",
               "longitude"
            ],
            "accountname":"john.doe",
            "lastusedalbum":"9475638663638858465",
            "sources":[
               "picture"
            ]
         },
         {
            "servicename":"facebook",
            "displayname":"Facebook",
            "apikey":"1234123efd1231244",
            "authorized":true,
            "authtype":"native",
            "hasalbums":true,
            "supportrecipients":false,
            "albumprivacy":[
               "EVERYONE",
               "ALL_FRIENDS",
               "NETWORKS_FRIENDS",
               "FRIENDS_OF_FRIENDS",
               "SELF"
            ],
            "icon":"https://my.server.com/export/icons/facebook.png",
            "albumattributes":[
               "name",
               "description"
            ],
            "accountname":"John Doe",
            "sources":[
               "picture"
            ]
         },
         {
            "servicename":"youtube",
            "displayname":"YouTube",
            "authorized":true,
            "authtype":"native",
            "authurl":"https://www.youtube.com?key=xpto",
            "hasalbums":false,
            "supportrecipients":false,
            "itemprivacy":[
```

```
                "public",
                "private"
            ],
            "icon":"https://my.server.com/export/icons/youtube.png",
            "itemattributes":[
                "name",
                "description",
                "location",
                "latitude",
                "longitude"
            ],
            "accountname":"johndoe",
            "lastuseditemprivacy":"private",
            "sources":[
                "video"
            ]
        },
        {
            "servicename":"twitter",
            "displayname":"Twitter",
            "authorized":true,
            "authtype":"native",
            "authurl":"https://my.server.com/sapi/externalservice/
twitter?action=authorize",
            "hasalbums":false,
            "supportrecipients":false,
            "itemprivacy":[

            ],
            "icon":"https://my.server.com/export/icons/twitter.png",
            "itemattributes":[
                "description"
            ],
            "accountname":"johndoe",
            "sources":[
                "picture"
            ],
            "exportmultiple":true
        }
    ]
  }
}
```


Attributes description

Attribute name	Attribute type	Description	Possible values
title	String	Title of the item or album	any string
description	String	Description or summary of the item or album	any string
tags	Array	Array of strings containing the tag names	any number of string tags

Attribute name	Attribute type	Description	Possible values
location	String	Geographic location of the item or album	any string
safety_level	Number	Safety level (Flickr-specific)	`1` for *Safe*, `2` for *Moderate*, or `3` for *Restricted*
content_type	Number	Type of the item (Flickr-specific)	`1` for *Photo*, `2` for *Screenshot*, or `3` for *Other*
hidden	Boolean	Should the item appear in global search results (Flickr-specific)?	`false` to show, `true` to hide
set_id	Number	ID of the set to share	ID of an existing media set

Returning landing page URL

For each service that requires external authorization, this API call returns a field `authurl`, containing the URL where the client should go in order to authorize the service.

This authorization flow will return a token to the server with the result of the process.

Once the server receives the result, it will then send back the HTTP status code `302` (redirect) to the client, together with the corresponding landing page URL for the specified service.

Note

During the authorization process there could be several redirects done by the external service. These redirects can change at any time and are not controlled by the API, although the API guarantees that the end URL (landing page) will contain the parameter needed to understand if the authorization was successful or not.

This landing page URL will have an extra parameter named *`success`* with the values of `true` or `false`, depending on the authorization result.

Some examples:

```
facebook_landing_success.html?servicename=facebook&success=true
facebook_landing_error.html?servicename=facebook&success=false
```

```
picasa_landing_failure.jsp?servicename=picasa&success=false
picasa_landing_success.jsp?servicename=picasa&success=true
```

By inspecting the value of the redirect URL *`success`* parameter, the client is able to understand the URL for successful and failing authentication.

Note that this API call already returns the authentication status as value of the `authorized` field (`true`/`false`)

Errors

COM-1008, EXT-SRV-1000.

3.6.2 Get albums

Return a list of albums for a specific service.

Security realm: user

Definition

```
GET /sapi/externalservice/album?action=get-albums&servicename=<service
 name>
```

Parameters

- *`servicename`*: (**required**) represents the name of the service, if this parameter is present only then the service can be returned

Response

A success response body is a JSON object containing the mandatory "data" JSON object. The "data" JSON object contains the "albums" JSON array. Each item of the "albums" array is a JSON object representing a single album. See album fields description below in Section 5.4, “Album”. If the current user doesn't have any albums, the empty "albums" array is returned. If the service doesn't support albums, the empty "data" object is returned.

An error response body is the standard OneMediaHub SAPI error response.

Example

```
{
   "data":{
      "albums":[
         {
            "albumid":"1",
            "name":"album1",
            "privacy":"public"
         },
         {
            "albumid":"2",
            "name":"album2",
            "privacy":"private"
         },
         {
            "albumid":"3",
            "name":"album3",
            "privacy":"public"
         }
      ]
   }
}
```

Errors

EXT-SRV-1002, EXT-SRV-1003.

3.6.3 Create album

Create an album in a specific service.

Security realm: user

Definition

```
POST /sapi/externalservice/album?action=create-album
```

Parameters

None.

Request body

See Section 5.5, “Album to be created”.

Example request:

```
{
   "data":{
      "servicename":"picasa",
      "albumprivacy":"FRIENDS_OF_FRIENDS",
      "attributes":[
         {
            "title":"this is the album name"
         },
         {
            "description":"this is an album description"
         }
      ]
   }
}
```

Response

```
{
   "success":"Album created successfully",
   "id":"3303426962747520536"
}
```

Errors

EXT-SRV-1000, EXT-SRV-1001, EXT-SRV-1005.

3.6.4 Export media

Export a media item (picture or video) to a given album.

Security realm: user

Definition

```
POST /sapi/media/picture?action=export
```

or

```
POST /sapi/media/video?action=export
```

Parameters

None.

Request body

See Section 5.28, “Media export”.

Note

Services whose *`exportmultiple`* parameter is `false`, will return an error if the `items` array contains more then one item

Example request:

```
sapi/media/picture?action=export
```

```
{
   "data":{
      "servicename":"flickr",
      "albumid":"album2",
      "items":[
         123
      ],
      "itemprivacy":"is_private",
      "itemattributes":[
         {
            "name":"this is the picture name"
         },
         {
            "description":"this is a picture description"
         },
         {
            "tags":[
               "tag1",
               "tag2"
            ]
         }
      ]
   }
}
```

Example response

```
{
   "success":"Picture Uploaded successfully"
}
```

Errors

EXT-SRV-1000, EXT-SRV-1004, EXT-SRV-1006, EXT-SRV-1009, EXT-SRV-1011, EXT-SRV-1012.

3.6.5 Import Facebook profile pictures

Update the contacts of the user by adding pictures of Facebook friends.

Security realm: user

Definition

```
POST /sapi/externalservice/facebook/friend?action=get
```

Parameters

- *`field`*: (**mandatory**) the Facebook field that has to be imported. The only current supported value is *photo*

Response

A JSON object with the number of updated contacts.

```
{
   "data":{
      "updated":0
   }
}
```

Example

Request

```
sapi/externalservice/facebook/friend?action=get&field=photo
```

Response

```
{
   "data":{
      "updated":0
   }
}
```

Errors

EXT-SRV-1000,EXT-SRV-1004, EXT-SRV-1100

3.6.6 Handle authorization

3.6.6.1 Receive authorization tokens

Handle the authorization tokens delivered by external services.

Security realm: user

Definition

```
GET/POST /sapi/externalservice/<servicename>
```

This API URL should be the one specified in any external service where a key is registered as the callback URL. This is the URL where external services will deliver session tokens, and since those services might use POST or GET methods, both must be supported. Each service will deliver the session token in a different way, so each added service will need to be handled as a separate case.

Errors

COM-1011, EXT-SRV-1002, in addition to generic errors described in Section 4.1.3, “Errors across multiple Server APIs”

3.6.6.2 Revoke authorization

Handle the authorization tokens delivered by external services when the authorization has to be revoked.

Security realm: user

Definition

```
GET/POST /sapi/externalservice/<servicename>?action=revoke
```

Errors

COM-1011, EXT-SRV-1002, in addition to generic errors described in Section 4.1.3, “Errors across multiple Server APIs”

3.6.7 Save Authorization Token

Request the authorization token for a given service and user ID, and save or update it directly on the server. If the associated token already exists on the server, the information is just updated.

Security realm: user

Definition

```
POST /sapi/externalservice/authorization?action=save
```

Request body:

```
{
    "data":{
          "token": "1212EF234232AB34C345345ABSGSJAK2627",
          "expiretime": 123453463445,
          "servicename":"facebook",
          "accountname": "user xyz"
         }
  }
```

Response

The 200 HTTP code for a successful request.

Errors

EXT-SRV-1000 and EXT-SRV-1003 in addition to generic errors described in Section 4.1.3, “Errors across multiple Server APIs”

3.7 Contacts

These API calls are used to retrieve, add, update or delete contacts for a user.

3.7.1 Get contacts

Retrieve all available contacts (or a single contact) for a user.

Security realm: user

Definition

```
POST /sapi/contact?action=get
```

Parameters

- *`contactid`*: (**optional**) the ID of the contact
- *`ids`*: (**optional**) the JSON array with IDs to be retrieved
- *`offset`*: (**optional**) the number of contacts to be skipped. It must not be used together with the *`contactid`* parameter
- *`limit`*: (**optional**) the maximal number of contacts to retrieve. It must not be used together with the *`contactid`* parameter
- *`filtervalue`*: (**optional**) the value to be used for filtering the contacts. The filter considers the fields `diplay_name` and `company` from the database table `fnbl_pim_contact`, and all the types of phone numbers described in the table `fnbl_pim_contact_item`. These fields will be filtered using the criterion '*field contains* *`filtervalue`*'. Comparison will be case-insensitive.

  Warning

  The parameter *`filtervalue`* must not be used together with the parameters *`contactid`* and *`ids`*.

  Note

  If *`filtervalue`* has been specified, *`limit`* and *`offset`* will operate on the filtered and sorted contacts

- *`sortby`*: (**optional**) allowed values are
  - `id`
  - `name`
  - `date`

  If no value is specified, contacts will be ordered by their ID.
- *`sortorder`*: (**optional**) the order in which the *`sortby`* parameter is applied. Allowed values are

- `ascending`
- `descending`

If a value for the *`sortby`* parameter is supplied and no *`sortorder`* parameter is specified, *`sortorder`* will default to `ascending`.

Important

`sortorder` is valid only if the *`filtervalue`* parameter has been specified.

IDs example:

```
ids={"ids":[12,23,456]}
```

and, as encoded parameter:

```
ids=%7B%22ids%22%3A%5B12%2C23%2C456%5D%7D
```

Response

A JSON object with an array that includes all available contacts or a single contact if the contact ID was provided. For more information about the Contact JSON object, see Section 5.11, “Contact”.

Example:

```
{
   "contacts":[
      {
         "id":"800",
         "firstname":"1",
         "middlename":"1",
         "lastname":"1",
         "prefix":"",
         "displayname":"1, 1 1",
         "orgs":[
            {
               "title":""
            }
         ],
         "email":[
            {
               "v":"11111111111111111111@alcatel-lucent.com",
               "t":"main"
            }
         ],
         "phone":[
            {
               "v":"121231233",
               "t":"work"
            }
         ],
```

```
        "notes":[
           {
              "v":""
           }
        ],
        "categories":""
     },
     {
        "id":"801",
        "firstname":"2",
        "middlename":"",
        "lastname":"2",
        "prefix":"",
        "displayname":"2, 2",
        "orgs":[
           {
              "title":""
           }
        ],
        "email":[
           {
              "v":"asdasd@awdasd.asd",
              "t":"main"
           }
        ],
        "phone":[
           {
              "v":"11312313223",
              "t":"work"
           }
        ],
        "notes":[
           {
              "v":""
           }
        ],
        "categories":"",
        "extension":[
           {
              "t":"x-accountname",
              "v":"acc-desc1"
           },
           {
              "t":"x-accounttype",
              "v":"acc-x"
           }
        ],
        "impp":[
           {
              "v":"AOL;CHARSET=UTF-8:aim@aim.com"
           }
        ]
     }
  ]
```

```
}
```

Errors

COM-1008, PIM-1200, PIM-1202, PIM-1205, in addition to generic errors described in Section 4.1.3, “Errors across multiple Server APIs”

3.7.2 Add or update a contact

Add a new contact or update an existing contact for a user, given the standard information of a vCard.

Security realm: user

Definition

```
POST /sapi/contact?action=contactsave
```

Request body

- The JSON object representing the new contact or the contact to be updated. If you want to create a new contact, pass an empty string as value of *`id`* property. For more information about the Contact JSON object, see Section 5.11, “Contact”. If you want to update an existing contact, pass its ID as value of *`id`* property.

  Example:

```
{
   "firstname":"John",
   "lastname":"Doe",
   "phone":[
      {
         "v":"1111111111111",
         "t":"work"
      }
   ],
   "address":[
      {
         "street_address":"street 3",
         "city":" LA",
         "region":"",
         "zip":"",
         "country":"",
         "t":"home"
      }
   ]
}
```

Response

A JSON object containing a success message and an ID, or an error code.

Example:

```
{
    "success":"Contact inserted successfully",
```

```
    "id":"23508"
}
```

Errors

PIM-1200, PIM-1201, in addition to generic errors described in Section 4.1.3, “Errors across multiple Server APIs”

3.7.3 Delete a contact

Delete a contact from the user's contact list.

Security realm: user

Definition

```
POST /sapi/contact?action=contactsdelete
```

Request body

- The JSON object containing an array named "contacts" with the IDs of the contacts to be deleted.

  Example:

```
{
   "contacts":[
      23508,
      23816
   ]
}
```

Response

A JSON object containing a success message and an ID, or an error code.

Example:

```
{
    "success":"Contacts removed successfully"
}
```

Errors

PIM-1200, PIM-1201, in addition to generic errors described in Section 4.1.3, “Errors across multiple Server APIs”

3.7.4 Reset contacts

Reset all contacts for the user.

Security realm: user

Definition

```
POST /sapi/contact?action=reset
```

Parameters

None.

Response

A HTTP 200 code if completed successfully.

Errors

PIM-1200, in addition to generic errors described in Section 4.1.3, “Errors across multiple Server APIs”

3.7.5 Add or update a contact picture

Add a new picture for the selected contact. If the ID is already present, the operation is considered an update.

> **Note**
>
> All pictures will be resized to a maximum size of 300 pixels per side, keeping aspect ratio.

Security realm: user

Definition

```
POST /sapi/contact?action=photosave
```

The `photosave` action must be called using a form submit, in order to send the picture itself and the needed parameters.

Parameters

- *`uid`*: the contact on the server side
- *`input_photo_0`*: the contact picture
- *`callback`*: a JavaScript method to be called in order to update the picture of the contact in the interface. This is needed because the photo is posted using a form submit; it is a way of refreshing the interface without refreshing the entire page.

Response

An HTML response with success message and ID (or error code) and callback method, or a JSON response message if no callback is present.

Example

callback: `window.parent.Zapatec.Widget.getWidgetById(74).serverSendOnLoad({`

JSON response message: `{"success":"Photos saved successfully."}`

```
<html><body><script type='text/javascript'>
window.parent.Zapatec.Widget.getWidgetById(74).serverSendOnLoad({"succes
s":"Photos
```

```
saved successfully."})</script></body></html>
```

Errors

PIM-1104, PIM-1200, in addition to generic errors described in Section 4.1.3, “Errors across multiple Server APIs”

3.7.6 Delete contact pictures

Delete a list of contact pictures.

Security realm: user

Definition

```
POST /sapi/contact?action=photodelete
```

Parameters

None.

Request body

The JSON object containing an array named *`photos`* with the IDs of the contact pictures to be deleted.

Example:

```
{
    "photos":[23508,23816]
}
```

Response

A JSON response with success message and ID or error code.

Example:

```
{
    "success":"Contacts removed successfully"
}
```

Errors

PIM-1200, in addition to generic errors described in Section 4.1.3, “Errors across multiple Server APIs”

3.7.7 Count the contacts

Return the number of contacts for the user.

Security realm: user

Definition

```
GET /sapi/contact?action=count
```

Parameters

- *`filtervalue`*: (**optional**) the value to be used for filtering the contacts. The filter considers the fields `display_name`, `company`, `account_name`, and `account_type` from the database table `fnbl_pim_contact`, and all the types of phone numbers described in the table `fnbl_pim_contact_item`. These fields will be filtered using the criterion '*field contains* *`filtervalue`*'. Comparison will be case-insensitive.

Response

The number of contacts for the user.

Example:

```
{
    "success":"Count returned successfully",
    "count":250
}
```

Errors

PIM-1200, in addition to generic errors described in Section 4.1.3, “Errors across multiple Server APIs”

3.7.8 Retrieve contact's picture

Retrieve the contact's picture as it is stored on the server.

Security realm: user

Definition

```
GET /sapi/contact?action=photodownload&id=<contactid>
```

Parameters

- *`id`*: the contact's ID.

Response

The contact picture as is stored in the server.

Errors

PIM-1200, in addition to generic errors described in Section 4.1.3, “Errors across multiple Server APIs”

3.7.9 Get the list of deleted contacts

Return the list of last dates when at least one contact was deleted.

Security realm: user

Definition

```
POST /sapi/contact?action=contactgetdeleted
```

Parameters

- *limit*: (**optional**) the number of the dates to retrieve.
- *userid*: (**optional**) the ID of the registered user. It can be specified only by an administrator.

Response

The array of JSON objects containing the date string in RFC-2445 (*yyyymmddThhmmss*) format with the number of contacts deleted on the corresponding date. All dates are in the user's timezone.

Deleted contacts are grouped by one-minute intervals, so the date always has the seconds part equal to zero.

Example:

```
{
    "dates": [
      {
       "date" : "20090215T124300",
       "count" : 1
      }, {
       "date" : "20090216T081200",
       "count" : 215
      }
    ]
}
```

Errors

PIM-1200, COM-1008, in addition to generic errors described in Section 4.1.3, “Errors across multiple Server APIs”

3.7.10 Restore deleted contacts

Restore contacts that were deleted on the specified date.

Security realm: user

Definition

```
POST /sapi/contact?action=contactrestore
```

Parameters

- *date*: (**required**) the date in RFC-2445 (*yyyymmddThhmmss*) format. The date is in the user's timezone. As deleted events are grouped by one-minute intervals, the date always has the seconds part equal to zero. Otherwise, the PIM-1106 error code is returned.
- *userid*: (**optional**) the ID of the registered user. It can be specified only by an administrator.

Example:

```
/sapi/contact?action=contactrestore&date=20091013T125400
```

Response

A HTTP 200 code if completed successfully.

Errors

PIM-1106, PIM-1107, PIM-1200, in addition to generic errors described in Section 4.1.3, “Errors across multiple Server APIs”

3.7.11 Export contacts

Export contacts in comma-separated values (CSV) format compatible with Microsoft Outlook.

Security realm: user

Definition

```
GET /sapi/contact?action=export
```

The contacts are returned with **content-type** `text/csv` and **content-disposition** `attachment`. The filename is `contacts.csv`.

Mapping between the Outlook CSV format and the Contact JSON object can be found at Appendix E, *Outlook CSV format and Funambol JSON equivalent*.

3.7.12 Import contacts

Import contacts from the service account set in the request URI. The service account is mapped to an existing external service with the same name.

Security realm: user

Definition

```
POST /sapi/contact/import/service?action=import
```

Parameters

- *`userid`*: the user ID of the user owner of the account where the contacts are stored. It can be specified only by an administrator.

Request body

The request body is not mandatory.

Response

An object with the number of contacts imported.

Example

Request:

```
POST /sapi/contact/import/google?action=import&userid=username
```

Response:

```
{
```

```
   "data":{
      "count":345
   }
}
```

Errors

EXT-SRV-1003, EXT-SRV-1004, PRO-1000, PRO-1101, PRO-3001, in addition to generic errors described in Section 4.1.3, “Errors across multiple Server APIs”.

3.8 Calendar

These API calls are used to retrieve, add, update, or delete calendar events for a user.

3.8.1 Get events

Retrieve all events for a user in a defined time frame.

The range is truncated between 1980 and 2050, and is limited to a 90-days time frame.

You can use the mutually exclusive parameters *`eventid`* and *`ids`* to filter by event ID the events in the time frame indicated by the parameter *`range`*.

Being the request of type GET, the parameters should be passed in the URL (with their content URL encoded when needed as per RFC 1738, i.e., in this case, when containing JSON objects), not as body of the request.

Security realm: user

Definition

```
POST /sapi/calendar?action=get
```

Parameters

- *`range`*: (**mandatory**) a JSON object indicating the time frame. Dates are indicated in the user's local time. To get all events for the same day, the *`from`* and *`to`* parameters of the *Range* JSON object must have the same value. The range is truncated between 1980 and 2050, and is limited to a 90-days time frame.
- *`eventid`*: the ID of an event
- *`ids`*: a JSON array indicating the IDs of the events to be retrieved
- *`filtervalue`*: (**optional**) the value to be used for filtering the events. The filter considers the fields `account_name` and `account_type` from the database table `fnbl_pim_calendar`. These fields will be filtered using the criterion '*field contains* *`filtervalue`*'. Comparison will be case-insensitive.

Important

`range` is not mandatory, but if one between *`ids`* and *`eventid`* is not present, *`range`* must be set. In the same way, *`ids`* and *`eventid`* are mutually exclusive. One of these three parameters must always be specified.

Example for *ids*:

```
ids={"ids":[12,23,456]}
```

and, as URL encoded parameter:

```
ids=%7B%22ids%22%3A%5B12%2C23%2C456%5D%7D
```

Example for *range*:

```
range={"from":"20090309","to":"20090316"}
```

and, as URL encoded parameter:

```
range=%7b%22from%22%3a%2220090309%22%2c%22to%22%3a%2220090316%22%7d
```

Response

A calendar containing all the events for the user in the given time frame with start and end time according to the user timezone specified in the response object.

If the *eventid* parameter is specified, the single event with the given *eventid* with date and time in the server timezone specified in the response object.

For more information about the *Event* JSON object, see Section 5.15, “Event”. For the *Calendar* JSON object, see Section 5.9, “Calendar”.

Example:


```
{
   "calendars":[
      {
         "id":"username",
         "title":"calendar",
         "cn":"",
         "email":"",
         "color":"#B1440E",
         "accessLevel":"owner",
         "summary":"",
         "where":"",
         "selected":true,
         "defaultRole":"none",
         "acl":[

         ],
         "events":[
            {
               "id":"24537",
               "calendarId":"paulo",
               "dtstart":"20090312T180000",
               "tzdtstart":"Europe/London",
               "dtend":"20090312T190000",
               "tzdtend":"Europe/London",
               "floating":false,
               "summary":"dinner with Jane",
               "where":" fift avenue",
```

```
            "transp":"transparent",
            "privacy":"private",
            "rrule":{
               "FREQ":"MONTHLY",
               "BYMONTHDAY":12,
               "INTERVAL":1,
               "DTSTART":"20090312T180000",
               "DTEND":"20090312T190000",
               "TZDTSTART":"Europe/London",
               "TZDTEND":"Europe/London"
            },
            "attendees":[
               {
                  "id":"1342",
                  "uri":"mailto:albert.rtiz@email.com",
                  "role":"req-participant",
                  "cn":"Albert Rtiz"
               },
               {
                  "id":"2423",
                  "uri":"mailto:john.smith@email.com",
                  "role":"chair",
                  "cn":"John Smith"
               }
            ]
         },
         {
            "id":"24605",
            "calendarId":"paulo",
            "dtstart":"20090313T180000",
            "tzdtstart":"Europe/London",
            "dtend":"20090313T200000",
            "tzdtend":"Europe/London",
            "floating":false,
            "summary":"pay phone bill",
            "transp":"transparent",
            "privacy":"private",
            "where":"Rome",
            "latitude":"41.54",
            "longitude":"12.29",
            "extension":[
               {
                  "t":"x-accounttype",
                  "v":"acc-x"
               },
               {
                  "t":"x-accountname",
                  "v":"acc-x"
               }
            ]
         }
      ]
   }
],
```

```
   "timezone":"Europe/London"
}
```

Errors

PIM-1100, PIM-1101, PIM-1103, PIM-1108, in addition to generic errors described in Section 4.1.3, “Errors across multiple Server APIs”

3.8.2 Create a new event

Add a new event to the user's default calendar.

Security realm: user

Definition

```
POST /sapi/calendar?action=eventcreate
```

Parameters

- *`event`*: a JSON object representing the new event.

Request body

A JSON object with information regarding the new event. For more information about the *Event* JSON object, see Section 5.15, “Event”

Example:

```
{
   "dtstart":"20090310T160000",
   "dtend":"20090310T170000",
   "tzdtstart":"Europe/London",
   "tzdtend":"Europe/London",
   "summary":"have a drink",
   "where":"Moe's bar",
   "calendarId":"paulo",
   "alarms":[
      {
         "action":"alert",
         "duration":15
      }
   ],
   "privacy":"public"
}
```

Example with Attendees 1

```
{
   "floating":false,
   "dtstart":"20090310T160000",
   "dtend":"20090310T170000",
   "tzdtstart":"Europe/London",
   "tzdtend":"Europe/London",
   "summary":"have a drink",
```

```
   "where":"Moe's bar",
   "calendarId":"paulo",
   "alarms":[
      {
         "action":"alert",
         "duration":15
      }
   ],
   "attendees":[
      {
         "id":"1",
         "uri":"mailto:albert.rtiz@email.com",
         "role":"req-participant",
         "cn":"Albert Rtiz"
      },
      {
         "id":"2",
         "uri":"mailto:john.smith@email.com",
         "role":"chair",
         "cn":"John Smith"
      }
   ],
   "transp":"opaque",
   "privacy":"public"
}
```

Example with Attendees 2

```
{
   "dtstart":"20081214",
   "dtend":"20081215",
   "calendarId":"username",
   "summary":"meeting e C",
   "organizer":{
      "cn":"",
      "email":""
   },
   "attendees":[
      {
         "uri":"mailto:john.smith@smith.com",
         "role":"non-participant",
         "cn":"John Smith",
         "valuetype":"VALUE=URL",
         "valuetypeformat":"TYPE=VCARD",
         "rsvp":"true",
         "status":"accepted",
         "expect":"request",
         "kind":"group"
      },
      {
         "uri":"mailto:carl.red@email.com",
         "role":"chair",
         "cn":"Carl Red",
         "valuetype":"VALUE=URL",
```

```
            "valuetypeformat":"TYPE=VCARD",
            "rsvp":"true",
            "status":"accepted",
            "expect":"request",
            "kind":"group"
         }
      ],
      "transp":"opaque",
      "alarms":[
         {
            "action":"alert",
            "duration":60
         }
      ],
      "iCalendarId":"0",
      "iEventId":0,
      "locked":true,
      "latitude":"44.54",
      "longitude":"15.50",
      "extension":[
         {
            "t":"x-accounttype",
            "v":"acc-y"
         },
         {
            "t":"x-accountname",
            "v":"acc-y"
         }
      ]
}
```

Response

A JSON response with success message and ID or error code.

Example:

```
{
     "success":"New event added successfully",
     "id":"23510"
}
```

Errors

PIM-1100, in addition to generic errors described in Section 4.1.3, “Errors across multiple Server APIs”

3.8.3 Modify an event

Update an event in the user's default calendar.

Security realm: user

Definition

```
POST /sapi/calendar?action=eventmodify
```

Request body

A JSON object with information regarding the event to be modified. For more information about the *Event* JSON object, see Section 5.15, “Event”

Example:

```
{
   "dtstart":"20090310T160000",
   "dtend":"20090310T170000",
   "tzdtstart":"Europe/London",
   "tzdtend":"Europe/London",
   "summary":"gym2",
   "where":"gym's    place",
   "calendarId":"paulo",
   "description":"this is a note",
   "alarms":[
      {
         "action":"alert",
         "duration":15
      }
   ],
   "privacy":"public",
   "id":"23510"
}
```

Example with Attendees

```
{
 "dtstart":"20081214",
 "dtend":"20081215",
 "calendarId":"username",
 "summary":"meeting e C",
 "organizer":
 {
  "cn":"",
  "email":""
 },
 "attendees":
 [
  {
   "uri":"mailto:john.smith@smith.com",
   "role":"non-participant","cn":"John Smith",
   "valuetype":"VALUE=URL",
   "valuetypeformat":"TYPE=VCARD",
   "rsvp":"true",
   "status":"accepted",
   "expect":"request",
   "kind":"group"
  },
  {
   "uri":"mailto:carl.red@email.com",
   "role":"chair","cn":"Carl Red",
   "valuetype":"VALUE=URL",
```

```
   "valuetypeformat":"TYPE=VCARD",
   "rsvp":"true",
   "status":"accepted",
   "expect":"request",
   "kind":"group"
  }
 ],
 "transp":"opaque",
 "alarms":
 [
  {
   "action":"alert",
   "duration":60
  }
 ],
 "iCalendarId":"0",
 "iEventId":0,
 "locked":true,
 "latitude":"44.54",
 "longitude":"15.50",
 "extension":
 [
  {
   "t":"x-accounttype",
   "v":"acc-y"
  },
  {
   "t":"x-accountname",
   "v":"acc-y"
  }
 ]
}
```

Note

The ID (*id*) of the event to be changed is mandatory.

Note

When sending an update for an event, the whole event information should be included, not only what was updated. For example, it is not possible to send the ID (*id*) of the event with only an updated recurrent rule.

Response

A JSON response with success message and ID or error code.

Example:

```
{
    "success":"Event updated successfully",
    "id":"23510"
```

```
}
```

Errors

PIM-1100, PIM-1105, in addition to generic errors described in Section 4.1.3, “Errors across multiple Server APIs”

3.8.4 Modify an instance of a recurring event

Update an instance of a recurring event in the user's default calendar.

Security realm: user

Definition

```
POST /sapi/calendar?action=eventmodifyone
```

Request body

- A JSON object for the event containing the date of the exception to be created as `exdate` (date to be removed from the recurrence pattern), the date `dtstart` representing the positive exception (added as a new event in OneMediaHub backend) and the calendar ID. For more information about the *Event* JSON object, see Section 5.15, “Event”.

Example:

```
{
   "id":"23511",
   "calendarId":"paulo",
   "dtstart":"20090310T160000",
   "dtend":"20090310T170000",
   "tzdtstart":"Europe/London",
   "tzdtend":"Europe/London",
   "summary":"recurrent2",
   "alarms":[
      {
         "action":"alert",
         "duration":15
      }
   ],
   "privacy":"private",
   "rrule":{
      "FREQ":"DAILY",
      "UNTIL":"20090313",
      "DTSTART":"20090310T160000",
      "DTEND":"20090310T170000",
      "TZDTSTART":"Europe/London",
      "TZDTEND":"Europe/London"
   },
   "exdate":"20090310T160000"
}
```

Response

A JSON response with success message and ID or error code.

Example:

```
{
    "success":"Event updated successfully",
    "id":"23511"
}
```

Errors

PIM-1000, PIM-1100, in addition to generic errors described in Section 4.1.3, “Errors across multiple Server APIs”

3.8.5 Delete an event

Delete an event in the user's default calendar.

Security realm: user

Definition

```
POST /sapi/calendar?action=eventdelete
```

Request body

- A JSON object for the event (providing the event's ID is sufficient).

Example:

```
{
    "id":"23513"
}
```

Response

A JSON response with success message or error code.

Example:

```
{
     "success":"Event deleted successfully",
     "id":"23513"
}
```

Errors

PIM-1100, in addition to generic errors described in Section 4.1.3, “Errors across multiple Server APIs”

3.8.6 Delete an instance of a recurring event

Delete an instance of a recurring event in the user's default calendar.

Security realm: user

Definition

```
POST /sapi/calendar?action=eventdeleteone
```

Request body

- A JSON object for the event, given the ID and *`dtstart`* of the event.

Example:

```
{
    "id":"23514",
    "dtstart":"20090310T100000"
}
```

Response

A JSON response with success message or error code.

```
{
    "success":"Event deleted successfully",
    "id":"23514"
}
```

Errors

PIM-1000, in addition to generic errors described in Section 4.1.3, “Errors across multiple Server APIs”

3.8.7 Get the busy dates

Return all the dates that have at least an event for the user in the indicated date range.

The range is truncated between 1980 and 2050, and is limited to a 90-days time frame.

Security realm: user

Definition

```
POST /sapi/calendar?action=busydatesget
```

Parameters

- *`range`*: a JSON object for the range. The range should be passed as an escaped parameter, not as body of the request. The range is truncated between 1980 and 2050, and is limited to a 90-days time frame.

Example:

```
{
    "from":"20090309",
    "to":"20090316"
}
```

and as encoded parameter:

```
%7b%22from%22%3a%2220090309%22%2c%22to%22%3a%2220090316%22%7d
```

Response

A JSON array of array of strings in RFC-2445 (`yyyymmdd`) format

```
{
   "busydates":[
      "20090215",
      "20090216",
      "20090217",
      "20090218"
   ]
}
```

Errors

PIM-1100, PIM-1101, in addition to generic errors described in Section 4.1.3, “Errors across multiple Server APIs”

3.8.8 Reset the calendar

Reset all events and tasks for the user.

Security realm: user

Definition

```
POST /sapi/calendar?action=reset
```

Parameters

None.

Response

A HTTP 200 code if completed successfully.

Errors

PIM-1100, in addition to generic errors described in Section 4.1.3, “Errors across multiple Server APIs”

3.8.9 Get the list of deleted events

Return the list of last dates when at least one event was deleted.

Security realm: user

Definition

```
GET /sapi/calendar?action=eventgetdeleted
```

Parameters

- *`limit`*: (**optional**) the number of the dates to retrieve.
- *`userid`*: (**optional**) the ID of the registered user; can be specified only by an administrator.

Response

The array of JSON objects containing the date string in the RFC-2445 format (*`yyyymmddThhmmss`*) with the number of events deleted on the corresponding date. All dates are in the user's timezone.

Deleted events are grouped by one-minute intervals, so the date always has the seconds part (ss) equal to zero.

Example:

```
{
   "dates":[
      {
         "date":"20091210T142300",
         "count":1
      },
      {
         "date":"20091013T125400",
         "count":215
      }
   ]
}
```

Errors

PIM-1100, COM-1008, in addition to generic errors described in Section 4.1.3, “Errors across multiple Server APIs”

3.8.10 Restore deleted events

Restores events which were deleted on the specified date.

Security realm: user

Definition

```
POST /sapi/calendar?action=eventrestore
```

Parameters

- *`date`*: (**required**) the date in the RFC-2445 format (*`yyyymmddThhmmss`*).

  Note

  The date is in the user's timezone. As deleted events are grouped by one-minute intervals, the date always has the seconds part (`ss`) equal to zero. Otherwise, the PIM-1106 error code is returned.

- *`userid`*: (**optional**) the ID of the registered user. It can be specified only by an administrator.

Example:

```
/sapi/calendar?action=eventrestore&date=20091013T125400
```

Response

A HTTP 200 code if completed successfully.

Errors

PIM-1100, PIM-1106, PIM-1107, in addition to generic errors described in Section 4.1.3, “Errors across multiple Server APIs”

3.8.11 Import calendar

Import calendar from the service account set in the request URI. The service account is mapped to an existing external service with the same name.

Security realm: user

Definition

```
POST /sapi/calendar/import/service?action=import
```

Parameters

- *`userid`*: the user ID of the user owner of the account where the calendar is stored. It can be specified only by an administrator.

Request body

The request body is not mandatory.

Response

An object with the number of calendar events imported (new and updated).

Example

Request:

```
POST /sapi/calendar/import/google?action=import&userid=username
```

Response:

```
{
    "data":{
        "count":20
    }
}
```

Errors

EXT-SRV-1003, EXT-SRV-1004, PRO-1000, PRO-5001, PRO-5100, in addition to generic errors described in Section 4.1.3, “Errors across multiple Server APIs”.

3.9 System

These API calls allow to access more general data such as supported timezones, carriers, countries, and devices.

3.9.1 Get manufacturers

Retrieve all available phone manufacturers.

Security realm: public

Definition

```
GET /sapi/system/manufacturer?action=get
```

Parameters

None.

Response

An array with all the available phone manufacturers. For more information about the Manufacturer JSON object, see Section 5.27, “Manufacturer”.

Example

Request:

```
GET /sapi/system/manufacturer?action=get
```

Response:

```
{
   "data":{
      "manufacturers":[
         {
            "id":107,
            "name":"Nokia",
            "popular":true
         },
         {
            "id":105,
            "name":"Motorola",
            "popular":true
         },
         {
            "id":111,
            "name":"Sony Ericsson",
            "popular":true
         },
         ......
      ]
   }
}
```

Errors

SYS-1000, in addition to generic errors described in Section 4.1.3, “Errors across multiple Server APIs”

3.9.2 Get phone models by manufacturer

Retrieve all phone models by the specific manufacturer ID.

Security realm: public

Definition

```
GET /sapi/system/model?action=get
```

Parameters

- *`manufacturerid`*: the manufacturer ID according to the ID retrieved by the Get Manufacturer API.

Response

An array with all the models for the given manufacturer ID. More details on the fields returned in the model object and how the phone capabilities can be used to send OTA messages or download SMS for Apps are discussed in Section 5.30, “Model”.

Example

Request:

```
GET /sapi/system/model?action=get&manufacturerid=106
```

Response:


```
{
   "data":{
      "models":[
         {
            "id":1040,
            "name":"228",
            "otasupport":true,
            "utcsupport":0,
            "utf8support":true,
            "fnblplugin":0,
            "jam":0,
            "syncml":true,
            "pimpush":false,
            "maxemails":0,
            "imagefileid":"handset135",
            "infofileid":"OtherMC4",
            "syncfileid":"OtherMS5",
            "active":true
         },
         {
            "id":1041,
            "name":"313",
            "otasupport":true,
            "utcsupport":0,
            "utf8support":true,
            "fnblplugin":0,
            "jam":0,
            "syncml":true,
            "pimpush":false,
            "maxemails":0,
```

```
            "imagefileid":"handset136",
            "infofileid":"OtherMC4",
            "syncfileid":"OtherMS5",
            "active":true
         },
         .....
      ]
   }
}
```

Errors

HTTP 400, SYS-1002, COM-1003, COM-1004, in addition to generic errors described in Section 4.1.3, “Errors across multiple Server APIs”

3.9.3 Get carriers

Retrieve all available carriers worldwide, the carriers for a specific country, or a specific carrier.

Security realm: public

Definition

```
GET /sapi/system/carrier?action=get
```

Parameters

- *`countryid`*: (**optional**) the two digit country code as for ISO 3166-1 alpha-2 standard (for example, "IT" for Italy). If *`countryid`* or *`carrierid`* are not provided, all the carriers available worldwide will be returned. The parameter is case insensitive.
- *`carrierid`*: (**optional**) the numeric value identifying the carrier. If *`countryid`* or *`carrierid`* are not provided, all the carriers available worldwide will be returned. The parameter is case insensitive.

Response

An array with all the carrier objects for the given country (i.e. all the carriers), or a unique carrier if the parameter *`carrierid`* is specified. For more information about the Carrier JSON object, see Section 5.10, “Carrier”.

If there are no carriers in the database for the given country, an empty array will be returned.

If there is no carrier corresponding to the specified ID, the COM-1007 error ("Carrier not found") is returned.

Example

Request for all Mexican carries:

```
GET /sapi/system/carrier?action=get&countryid=MX
```

Response:

```
{
```

```
   "data":{
      "carriers":[
         {
            "id":137,
            "name":"Telcel",
            "countryid":"144",
            "active":true,
            "otasupport":true,
            "trustedjam":true,
            "syncmlsupport":true
         },
         {
            "id":136,
            "name":"Telefonica (Movistar)",
            "countryid":"144",
            "active":true,
            "otasupport":true,
            "trustedjam":true,
            "syncmlsupport":true
         }
      ]
   }
}
```


Errors

COM-1002, COM-1007, in addition to generic errors described in Section 4.1.3, “Errors across multiple Server APIs”

3.9.4 Get phone capabilities

Retrieve the phone capabilities for a given phone using its model ID.

Security realm: public

Definition

```
GET /sapi/system/model?action=get
```

Parameters

- *`modelid`*: the phone model ID.

Response

A Model object with the phone capabilities for the given phone by the model ID. More details on the fields returned in the model object and how the phone capabilities can be used to send OTA messages or download SMS for Apps are discussed in Section 5.30, “Model”.

Example

Request for model ID 1101 (is the model ID of the E61):

```
GET /sapi/system/model?action=get&modelid=1101
```

Response:

```
{
   "data":{
      "model":{
         "id":1101,
         "name":"E61",
         "manufacturerid":107,
         "otasupport":true,
         "utcsupport":1,
         "utf8support":true,
         "fnblplugin":0,
         "jam":9,
         "syncml":true,
         "pimpush":true,
         "maxemails":50,
         "imagefileid":"NokiaE61",
         "infofileid":"NokiaMC4",
         "syncfileid":"NokiaMS7",
         "active":true
      }
   }
}
```

Errors

HTTP 400, SYS-1002, COM-1003, COM-1004, in addition to generic errors described in Section 4.1.3, “Errors across multiple Server APIs”

3.9.5 Get timezones

Retrieve all available timezones.

Security realm: public

Definition

```
GET /sapi/system/timezone?action=get
```

Parameters

- *`countryid`*: (**deprecated**) the uppercase two digit country code as for ISO 3166-1 alpha-2 standard (for example, "IT" for Italy).

Response

An array with all the carrier objects for the given country (or all the carriers). For more information about the Timezone JSON object, see Section 5.56, “Timezone”.

If there are no timezones in the database for the given country, an empty array will be returned.

Note

If no *`countryid`* is specified, the JSON array of timezones that is returned is the one from the JDK currently used to run OneMediaHub.

If a *countryid* is specified, the JSON array of timezones is populated according to an internal table that maps a subset of the timezones to the *countryid* to be able to create a more user friendly UI or a list of timezones that cover the entire country.

Note that in the full timezone list many timezones, for example for the U.S., have similar uses. For example, "US/Pacific" can be used whenever "America/Los_Angeles" is used.

Example

Request for all Mexican timezones:

```
GET /sapi/system/timezone?action=get&countryid=MX
```

Response:

```
{
   "data":{
      "timezones":[
         {
            "id":"America/Tijuana",
            "description":"America/Tijuana (GMT-08:00)",
            "default":false
         },
         {
            "id":"America/Mexico_City",
            "description":"America/Mexico-City (GMT-07:00)",
            "default":true
         },
         .....
      ]
   }
}
```

Errors

COM-1002 and generic errors described in Section 4.1.3, “Errors across multiple Server APIs”

3.9.6 Get countries

Retrieve all available countries.

Security realm: public

Definition

```
GET /sapi/system/country?action=get
```

Parameters

None.

Response

An array with all the available countries and their properties, such as country prefix (e.g. "39" for Italy) and A2 country code (e.g. "IT" for Italy). For more information about the Country JSON object, see Section 5.12, “Country”.

Example

Request:

```
GET /sapi/system/country?action=get
```

Response:

```
{
   "data":{
      "countries":[
         {
            "id":"20",
            "name":"Afghanistan",
            "a2":"AF",
            "phoneprefix":"093",
            "active":true
         },
         {
            "id":"21",
            "name":"Albania",
            "a2":"AL",
            "phoneprefix":"355",
            "active":true
         },
         {
            "id":"22",
            "name":"Algeria",
            "a2":"DZ",
            "phoneprefix":"213",
            "active":true
         },
         .....
      ]
   }
}
```

Errors

COM-1002, in addition to generic errors described in Section 4.1.3, “Errors across multiple Server APIs”

3.9.7 Get picture, info, and sync URL for a device

Retrieve the URL of the picture, the URL of the manual configuration instructions, and the URL of the synchronization instructions of a given device.

Security realm: public

Definition

```
GET /sapi/system/model?action=get-url
```

Parameters

- *`modelid`*: the phone model ID.

- *l10n*: (**optional**) the localization code for the language, for example en-US.

Note

To allow maximum flexibility the localization code is not validated so any string is acceptable as a *l10n* parameter.

Response

The portal URL and the paths to use to build the complete URLs for the given device by the model ID. For more information about the Model URL JSON object, see Section 5.31, “Model URL”.

Example 1

Request for model ID 1101 (is the model ID of the E61):

```
GET /sapi/system/model?action=get-url&modelid=1101
```

Response:

```
{
   "data":{
      "modelurl":{
         "portalurl":"https://my.server.com",
         "imagepath":"/html/devices/images/NokiaE61.gif",
         "infopath":"/html/devices/setup/NokiaMC4.html",
         "syncpath":"/html/devices/sync/NokiaMS7.html"
      }
   }
}
```

Example 2

Request for model ID 1101 (is the model ID of the E61) localized for American English:

```
GET /sapi/system/model?action=get-url&modelid=1101&l10n=en-US
```

Response:

```
{
   "data":{
      "modelurl":{
         "portalurl":"https://my.server.com",
         "imagepath":"/html/en-US/devices/images/NokiaE61.gif",
         "infopath":"/html/en-US/devices/setup/NokiaMC4.html",
         "syncpath":"/html/en-US/devices/sync/NokiaMS7.html"
      }
   }
}
```

Errors

HTTP 400, COM-1003, in addition to generic errors described in Section 4.1.3, “Errors across multiple Server APIs”

3.9.8 Get location from IP or `Accept-Language` header

Translate an IP into a country location.

Security realm: public

Definition

```
GET /sapi/system/location?action=get
```

Parameters

- *`language`*: `true` if the `Accept-Language` header should be used to decode location when the IP is not decoded. `false` is default

Response

The client's IP address and the two digit country code as for ISO 3166-1 alpha-2 standard (for example, "IT" for Italy). The country ID will be returned empty if it is not possible to translate the IP address into *`countryid`*.

Errors

SYS-1000, in addition to generic errors described in Section 4.1.3, “Errors across multiple Server APIs”

Example 1

Request for IP 89.97.143.243:

```
GET /sapi/system/location?action=get
```

Response:

```
{
    "data":{
        "ip":"89.97.143.243",
        "countryid":"IT"
    }
}
```

Example 2

Request for IP:

```
GET /sapi/system/location?action=get&language=true
```

Response:

```
{
    "data":{
        "ip":"127.0.0.l",
        "acceptlanguage":"IT-it",
        "countryid":"IT"
    }
}
```

3.9.9 Get a CAPTCHA image

Return the URL of an image that represents a token required to validate the session. This will ensure that the next POST request is not automatically generated by a computer.

Note

Multiple calls of the CAPTCHA image generate new token values in the session; only the latest one is available for validation.

Security realm: public

Definition

```
GET /sapi/system/captcha?action=get-url
```

Parameters

- *`mobile`*: (**optional**) when this parameter is set to `true` the mobile version of the CAPTCHA is returned.

Response

The portal URL, the path details that is necessary to build the complete URL for the CAPTCHA image and the status (active or not active).

Example 1

Request for a CAPTCHA image

```
GET /sapi/system/captcha?action=get-url
```

Response:

```
{
   "data":{
      "captchaurl":{
         "portalurl":"https://my.server.com:8080",
         "imagepath":"/Captcha.jpg",
         "active":true
      }
   }
}
```

Example 2

Request for a CAPTCHA image

```
GET /sapi/system/captcha?action=get-url&mobile=true
```

Response

```
{
   "data":{
      "captchaurl":{
```

```
         "portalurl":"https://my.server.com:8080",
         "imagepath":"/mcaptcha.jpg",
         "active":true
      }
   }
}
```

Errors

Generic errors described in Section 4.1.3, “Errors across multiple Server APIs”

3.9.10 Get server information

Retrieve information about the OneMediaHub server.

Security realm: public

Definition

```
GET /sapi/system/information?action=get
```

Parameters

None.

Response

A JSON object representing all the information about the OneMediaHub server.

Example

Request:

```
GET /sapi/system/information?action=get
```

Response:

```
{
   "portalurl":"https://my.server.com",
   "mobileurl":"https://my.server.com/m",
   "downloadurl":"https://my.server.com/d",
   "sapiversion":"14.0.0-SNAPSHOT",
   "man":"Funambol",
   "mod":"DS Server CarEd",
   "hwv":"-",
   "fwv":"-",
   "oem":"-",
   "devid":"funambol",
   "devtyp":"server",
   "verdtd":"1.2",
   "utc":"true",
   "supportlargeobjs":"true",
   "supportnumberofchanges":"true",
   "exts":"X-funambol-smartslow,X-funambol-media-http-upload,X-
funambol-msu"
```

```
}
```

Errors

Generic errors described in Section 4.1.3, “Errors across multiple Server APIs”

3.9.11 Get overall storage usage

Return the overall storage usage.

Security realm: system

Definition

```
GET /sapi/system/media?action=get-used-storage
```

Parameters

None.

Response

The total size in bytes of the storage used for media binaries.

Example

Request:

```
GET /sapi/system/media?action=get-used-storage
```

Response:

```
{
   "data":{
      "size":20000000
   }
}
```

Errors

SYS-1003, in addition to generic errors described in Section 4.1.3, “Errors across multiple Server APIs”

3.10 SMS

API to send text messages and to retrieve statistics on SMS usage.

3.10.1 Send generic text SMS

Send a generic text SMS to a specific device.

Security realm: system

Definition

```
POST /sapi/sms?action=send-text
```

Parameters

At least one of the following parameters should be provided:

- *`userid`*: the user ID of the registered user whom the SMS should be sent to. It can only be specified by a user logged in at system administrator level.
- *`phonenumber`*: the phone number of the recipient (for example, *`+39333333333`*). If this parameter is specified, it will be used instead of the phone number on record. Note that the phone number should be encoded as a parameter in the URL, so the "+" should be replaced with "%2B". For example, `+39333333333` should become `%2B39333333333.`

Request body

See Section 5.48, “SMS text message”.

Response

The 200 HTTP code indicates a successful request.

Example 1

Request:

```
POST /sapi/sms?action=send-text&userid=username
```

Request body:

```
{
   "data":{
      "message":"Hello John!"
   }
}
```

Example 2

Request:

```
POST /sapi/sms?action=send-text&phonenumber=%2B39333333333
```

Request body:

```
{
   "data":{
      "message":"Hello John!"
   }
}
```

Errors

COM-1001, HTTP 400, PRO-1001, PRO-1101, PRO-1104, SMS-1000, SMS-1101, SMS-1102, SMS-1103, in addition to generic errors described in Section 4.1.3, “Errors across multiple Server APIs”

3.10.2 Send user activation link

Send the user activation link via SMS to a specific device.

Security realm: system

Definition

```
POST /sapi/sms?action=send-activation-link
```

Parameters

- *`userid`*: (**mandatory**) the user ID of the registered user whom the SMS should be sent to

Response

The 200 HTTP code indicates a successful request.

Example

Request

```
POST/sapi/sms?action=send-activation-link&userid=username
```

Errors

PRO-1001, PRO-1104, SMS-1003, SMS-1102, in addition to generic errors described in Section 4.1.3, “Errors across multiple Server APIs”

3.11 Last activities

Note

For each user, only the last activity done on a specific data source is tracked.

3.11.1 Get last activities

Return a list containing the last activity per every data source for a given device ID, if specified.

Security realm: user

Definition

```
GET /sapi/activity?action=get
```

Parameters

- *`deviceid`*: (**optional**) the device ID to get last activities of.

Response

Example passing the parameter *`deviceid`*

```
{
   "data":{
      "deviceid":"fac-123456789",
      "starttime":1234567890,
      "endtime":1334567890,
      "status":"200",
```

```
      "activities":[
         {
            "activitytype":"add",
            "source":"picture",
            "sent":3,
            "received":2,
            "aborted":2,
            "rejected":2
         },
         {
            "activitytype":"add",
            "source":"video",
            "sent":3,
            "received":2,
            "aborted":2,
            "rejected":1
         },
         {
            "activitytype":"delete",
            "source":"video",
            "sent":3,
            "received":2,
            "aborted":2,
            "rejected":1
         },
         {
            "activitytype":"update",
            "source":"video",
            "sent":3,
            "received":2,
            "aborted":0,
            "rejected":1
         }
      ]
   }
}
```


Example without passing the parameter *`deviceid`*


```
{
   "data":{
      "devices":[
         {
            "deviceid":"fac-358150041339400",
            "starttime":1347526958593,
            "endtime":1347526961651,
            "status":"200",
            "activities":[
               {
                  "activitytype":"add",
                  "source":"picture",
                  "sent":0,
                  "received":1,
                  "aborted":0,
```

```
                  "rejected":1
               }
            ]
         },
         {
            "deviceid":"fac-359144045593699",
            "starttime":1347526919940,
            "endtime":1347526922235,
            "status":"200",
            "activities":[
               {
                  "activitytype":"add",
                  "source":"picture",
                  "sent":0,
                  "received":1,
                  "aborted":0,
                  "rejected":1
               }
            ]
         }
      ]
   }
}
```


For more information about the Last Activity JSON object, see Section 5.25, “Last activities”.

Errors

ACT-1000.

3.11.2 Save last activities

Save last activities for a given device.

Security realm: user

Definition

```
POST /sapi/activity?action=save
```

Parameters

None.

Example

Request

```
POST /sapi/activity?action=save
```


```
{
   "data":{
      "deviceid":"fac-123456789",
      "starttime":1234567890995,
      "endtime":1334567890999,
```

```
      "status":"200",
      "activities":[
         {
            "activitytype":"add",
            "source":"picture",
            "sent":3,
            "received":2,
            "aborted":0,
            "rejected":1
         },
         {
            "activitytype":"add",
            "source":"video",
            "sent":3,
            "received":2,
            "aborted":1,
            "rejected":1
         },
         {
            "activitytype":"delete",
            "source":"video",
            "sent":3,
            "received":2,
            "aborted":0,
            "rejected":1
         },
         {
            "activitytype":"update",
            "source":"video",
            "sent":3,
            "received":2,
            "aborted":2,
            "rejected":0
         }
      ]
   }
}
```


Response


```
{
   "success":"Activity saved successfully",
   "id":212
}
```


For more information about the Last Activity JSON object, see Section 5.25, “Last activities”.

Errors

ACT-1000, COM-1011.

3.12 EMAIL

API to send email messages.

3.12.1 Send Email

Send an email message to a specific user or email address of a user.

Security realm: system

Definition

```
POST /sapi/email?action=send
```

Parameters

At least one of the following parameters should be provided:

- *`userid`*: the id of the user which will receive the email.
- *`email`*: the email address to which the message will be sent.

Request body

A JSON object with information regarding the email subject and body Response.

Response

The 200 HTTP code indicates a successful request.

Example 1

Request

```
POST /sapi/email?action=send&userid=username
```

Request Body

```
{
   "data":{
      "subject": "Hello",
      "body": "Hello John!"
   }
}
```

Example 2

Request

```
POST /sapi/email?action=send&email=emailAddress
```

Request Body

```
{
   "data":{
      "subject": "Hello",
      "body": "Hello John!"
   }
}
```

Errors

MAIL-1000, MAIL-1001, MAIL-1002, MAIL-1003, PRO-1101, PRO-1110, PRO-1122, PRO-1129, COM-1016, COM-1021, in addition to generic errors described in Section 4.1.3, “Errors across multiple Server APIs”

3.12.2 Send activation link

Send an email message with the activation link to a specific user email address.

Security realm: system

Definition

```
POST /sapi/email?action=send-activation-link
```

Parameters

- *`userid`*: (**mandatory**) the id of the user which will receive the email

Response

The 200 HTTP code indicates a successful request.

Example 1

Request

```
POST /sapi/email?action=send-activation-link&userid=username
```

Errors

COM-1021, MAIL-1000, MAIL-1003, PRO-1001, PRO-1101, PRO-1110, in addition to generic errors described in Section 4.1.3, “Errors across multiple Server APIs”

3.13 Family

These functions allow administrator to manage user families in terms of group of users of the same family.

3.13.1 Save a family

Create a new family of users, or rename an existing one if the ID is provided within the JSON object carried with the request. An empty family cannot exist.

Security realm: system

Definition

```
POST /sapi/family?action=save
```

Request body

A JSON object that describes the family (see Section 5.65, “Family”)

Response

A JSON object that describes the created family. If a user in the request has not the privileges to be added to the family, she will be not added, but the family will be created with the other users, if any.

Example (creation)

Request

```
POST sapi/family?action=save
```

Request Body

```
{
  "data":{
    "family": {
       "name":"family cloud",
       "externalid":"thisisanexternalid",
       "users":[
          {
            "userid":"userid1"
          },
          {
            "userid":"userid2"
          }
       ]
    }
  }
}
```

Response

```
{
 "data":{
  "family":{
    "id":2500,
    "name":"family cloud",
    "externalid":"thisisanexternalid",
    "users":[
      {
        "userid":"userid1"
      },
      {
        "userid":"userid2"
      }
    ]
  }
 }
}
```

Example (rename)

Request

```
POST sapi/family?action=save
```

Request Body

```
{
  "data":{
    "family": {
       "name":"family cloud renamed",
       "id":2500
    }
  }
}
```

Response

```
{
 "data":{
  "family":{
    "id":2500,
    "name":"family cloud renamed"
  }
 }
}
```

Errors

COM-1011, FAM-1000, FAM-1001, FAM-1004 and FAM-1005, in addition to generic errors described in Section 4.1.3, “Errors across multiple Server APIs”

3.13.2 Delete a family

Delete an existing family.

Security realm: system

Definition

```
POST /sapi/family?action=delete
```

Request body

A JSON array containing the IDs of the families to be deleted.

Response

The 200 HTTP code indicates a successful request.

Example

Request

```
POST /sapi/family?action=delete
```

Request Body

```
{
```

```
  "data": {
     "families": [1, 2, 3]
  }
}
```

Errors

COM-1011, COM-1013 and FAM-1000, in addition to generic errors described in Section 4.1.3, “Errors across multiple Server APIs”

3.13.3 Retrieve a (set of) family(es)

Retrieve a set of families with all the information related to the families inside it. Non-admin users will be able to get information regarding only their family.

Security realm: user

Definition

```
POST /sapi/family?action=get
```

Parameters

- *`userid`*: (**optional**) the user you want to retrieve all the families.
- *`limit`*: (**optional**) the maximum number of families retrieved starting from the latest ones and considering the offset. Cannot be used with the body with IDs.
- *`offset`*: (**optional**) the offset (inclusive) of families to be considered in the response. Cannot be used with the body with IDs. Please note that if *`limit`* and *`offset`* parameters will not be used, the response from the server can be NOT exaustive, since the JSON array in the result can be limited to the maximum number the server can manage.

Request body (optional)

A JSON array containing the IDs or the external IDs of the families you want to retrieve. The request body is optional.

Response

A JSON array containing all the family objects (see Section 5.65, “Family”) that satisfy the request.

Example 1 (no IDs or *`userid`* parameters)

Request

```
POST /sapi/family?action=get
```

Response

```
{
  "data": {
    "families": [
       {
         "id":2500,
```

```
        "name":"family cloud",
        "externalid":"thisisanexternalid",
        "users":[
          {
            "userid":"userid1"
          },
          {
            "userid":"userid2"
          }
        ]
      },
      {
        "id":2501,
        "name":"family cloud",
        "users":[
          {
            "userid":"userid100"
          },
          {
            "userid":"userid200"
          }
        ]
      }
    ]
  }
}
```

Example 2 (*userid* parameter)

Request

```
POST /sapi/family?action=get&userid=userid100
```

Response

```
{
  "data": {
    "families": [
       {
        "id":2500,
        "name":"family cloud",
        "externalid":"thisisanexternalid",
        "users":[
          {
            "userid":"userid1"
          },
          {
            "userid":"userid2"
          }
        ]
      }
    ]
  }
}
```

Example 3 (IDs in the body)

Request

```
POST /sapi/family?action=get
```

Request Body

```
{
  "data": {
     "ids": [2501]
  }
}
```

Response

```
{
  "data": {
    "families": [
       {
         "id":2501,
         "name":"family cloud",
         "users":[
           {
             "userid":"userid100"
           },
           {
             "userid":"userid200"
           }
         ]
       }
    ]
   }
}
```

Example 4 (external IDs in the body)

Request

```
POST /sapi/family?action=get
```

Request Body

```
{
  "data": {
     "externalids": ["thisisanexternalid"]
  }
}
```

Response

```
{
  "data": {
    "families": [
       {
```

```
        "id":2500,
        "name":"family cloud",
        "externalid":"thisisanexternalid",
        "users":[
          {
            "userid":"userid1"
          },
          {
            "userid":"userid2"
          }
        ]
      }
    ]
  }
}
```

Example 5 (user realm)

Request

```
POST /sapi/family?action=get
```

Response

```
{
  "data": {
    "families": [
       {
        "id":2500,
        "name":"family cloud",
        "externalid":"thisisanexternalid",
        "users":[
          {
            "userid":"userid1",
            "firstname":"firstname",
            "lastname":"lastname"
         },
          {
            "userid":"userid2",
            "phonenumber":"+39123456789",
            "emailaddress":"mail@mail.ma"
          }
        ]
      }
    ]
  }
}
```

Errors

FAM-1000, in addition to generic errors described in Section 4.1.3, “Errors across multiple Server APIs”

3.13.4 Add a user to a family

Add a user to an existing family.

Security realm: system

Definition

```
POST /sapi/family/user?action=add
```

Request body

A JSON object containing the ID of the family and the *`userid`* to add.

Response

The 200 HTTP code indicates a successful request.

Example

Request

```
POST /sapi/family/user?action=add
```

Request Body

```
{
  "data": {
     "familyid": 2500,
     "userid":"user201"
  }
}
```

Errors

PRO-1101, COM-1011, COM-1013, FAM-1002 and FAM-1005, in addition to generic errors described in Section 4.1.3, “Errors across multiple Server APIs”

3.13.5 Remove a user from a family

Remove a user from an existing family.

Security realm: system

Definition

```
POST /sapi/family/user?action=remove
```

Request body

A JSON object containing the ID of the family and the *`userid`* to remove.

Response

The 200 HTTP code indicates a successful request.

Example

Request

```
POST /sapi/family/user?action=remove
```

Request Body

```
{
  "data": {
     "familyid": 2500,
     "userid":"user201"
  }
}
```

Errors

COM-1011, PRO-1101, COM-1013, FAM-1001, FAM-1003 and FAM-1005, in addition to generic errors described in Section 4.1.3, “Errors across multiple Server APIs”

3.13.6 Share media on the family hub

Allows a family member to share some media content on her family hub.

Security realm: user

Definition

```
POST /sapi/family/media?action=share
```

Request body

A JSON object containing the IDs of the items to be added.

Response

The 200 HTTP code indicates a successful request.

Example

Request

```
POST /sapi/family/media?action=share
```

Request Body

```
{
  "data": {
     "fdoids": [1176, 1546]
  }
}
```

Errors

COM-1011, COM-1013, FAM-1000, FAM-1005, in addition to generic errors described in Section 4.1.3, “Errors across multiple Server APIs”

3.13.7 Unshare media from the family hub

Allows a family member to unshare some media content from her family hub.

Security realm: user

Definition

```
POST /sapi/family/media?action=unshare
```

Request body

A JSON object containing the IDs of the items to be removed.

Response

The 200 HTTP code indicates a successful request.

Example

Request

```
POST /sapi/family/media?action=unshare
```

Request Body

```
{
  "data": {
     "fdoids": [1176, 1546]
  }
}
```

Errors

COM-1011, COM-1013, FAM-1000, FAM-1005, in addition to generic errors described in Section 4.1.3, “Errors across multiple Server APIs”

Chapter 4. Error definitions

This section lists all errors that the SAPI can return.

4.1 Error handling

4.1.1 JSON errors

Any JSON error code is composed by a module code and a numeric value (four digits); for example: COM-1002 if a request with an invalid parameter *countryid* is performed, 'ITT'. The HTTP status code is 200.

```
{
   "error":{
      "code":"COM-1002",
      "message":"Country not found",
      "parameters":[]
      "cause":"com.funambol.portal.api.system.CountryNotFoundException"
   }
}
```

The *parameters* array in the JSON code is optional and may include specific information to better clarify the error scenario and help the user of the API. For example:

```
{
   "error":{
      "code":"PRO-1001",
      "message":"Missing the following mandatory parameter(s)",
      "parameters":[{"param":"modelid"}]
      "cause":""
   }
}
```

4.1.2 HTTP errors

HTTP status codes are defined in RFC2616. The most common ones used in the OneMediaHub Server API are HTTP 200 (OK), HTTP 400 (400 Bad Request), HTTP 401 (Unauthorized) and HTTP 403 (Forbidden). For example, if the request requires user authentication the response is:

```
HTTP Status 401 -
type Status report
message
description This request requires HTTP authentication ().
```

4.1.3 Errors across multiple Server APIs

Most or all the Server APIs share some error codes:

1. All Server API requests, regardless of the REALM, can return the following error codes: PAPI-0000, SEC-1002,SEC-1004, HTTP 400.

2. All Server API requests with realm SYSTEM and USER can return the following HTTP error codes: HTTP 401, HTTP 402, HTTP 403, HTTP 405.

Note

HTTP 402 is used as Payment Required error code.

3. All Server API requests with realm SYSTEM and USER when performed as SYSTEM can return the error SEC-1001 if the user ID is not specified.
4. All Server API requests with realm SYSTEM and USER can return the error SEC-1003 if the server is configured to validate any authenticated SAPI request using a session unique validation key.

4.1.4 Modules

The following table reports the module codes:

Code	Module
COM	Common
MED	Media
PIC	Pictures (Pictures are a subset of Media)
PIM	Contact/Calendar
PRO	Profile
SEC	Security/login
SMS	SMS
SUB	Subscription
SYS	System

4.2 Generic error

Code	Message
PAPI-0000	Unrecognized error

4.3 Common errors

Code	Message
COM-1001	The phone number provided is not valid
COM-1002	Country not found
COM-1003	The model ID is not valid. It must be an integer
COM-1004	The manufacturer ID is not valid. It must be an integer
COM-1005	Unsupported operation
COM-1006	Invalid timezone
COM-1007	Carrier not found
COM-1008	Invalid data type or data format is not as expected
COM-1009	Invalid operating system
COM-1010	Invalid filename

Code	Message
COM-1011	Missing required parameter
COM-1012	Malformed interval
COM-1013	You must specify a non-empty data parameter
COM-1014	The provided LUID is not valid
COM-1015	The provided device ID is not valid
COM-1016	Parameters cannot be used at the same time
COM-1017	The country code is not valid
COM-1018	Sort order is not valid
COM-1020	Missing required HTTP header
COM-1021	Invalid parameter value

4.4 Security/login errors

Code	Message
SEC-1000	Unknown exception in security handling.
SEC-1001	The administrator must specify the user ID to perform the action.
SEC-1002	A session is already open. To provide new credentials please logout first.
SEC-1003	Invalid mandatory validation key
SEC-1004	Both header and parameter credentials provided, please use only one authentication schema.

4.5 Profile errors

Code	Message
PRO-1000	Unknown exception in profile handling.
PRO-1001	Missing the following mandatory parameter(s).
PRO-1101	User not found.
PRO-1102	You must specify the userid.
PRO-1104	Device not found.
PRO-1105	There is already a device for the specified user.
PRO-1106	The userid is not valid. Only letters (a-z), numbers (0-9), and periods (.) are allowed. Userid must be less than 16 characters and include at least one letter.
PRO-1107	You can not specify an existing e-mail address.
PRO-1108	The device ID is not valid. It must be an integer.
PRO-1110	E-mail account not found.
PRO-1111	The phone does not support any OneMediaHub App.

Code	Message
PRO-1112	The logged in user can not delete himself.
PRO-1113	You can not specify an existing username.
PRO-1114	The carrierid is from a different country from the country code provided.
PRO-1115	The password is not valid. Only letters (a-z) and numbers (0-9) and dash are allowed. Minimum 4, maximum 16 characters.
PRO-1116	The administrator cannot add a device to himself or another administrator.
PRO-1118	There is more than one user for the specified phone number.
PRO-1120	The convert timezone is not valid. Only numbers (0,1,2) are allowed.
PRO-1122	The e-mail address is not valid.
PRO-1126	Invalid CAPTCHA token
PRO-1128	The phone user ID must be at least 9 characters and at most 15 characters long, and can only have the symbols `(`, `)`,`-`, `+`
PRO-1129	This Email is used in more than one account.
PRO-1130	The component is invalid.
PRO-1131	The old password is incorrect.
PRO-1132	Unable to retrieve user information.
PRO-1133	Invalid reset token
PRO-1134	Forbidden phone number update since it is used for login
PRO-1135	Invalid channel specified, accepted values are: ['email', 'sms'].
PRO-1136	This msisdn is used in more than one account.
PRO-2001	OTA configuration not supported.
PRO-2002	No SMS left to send the OTA message.
PRO-2003	No device to configure.
PRO-2004	Sync source(s) not found.
PRO-2005	Uimode not valid. It must be `0`, `1`, `2`, or `3`.
PRO-2006	Sync type not valid. It must be `206`, `207`, `208`, `209`, or `210`.
PRO-3001	Import contacts not supported.
PRO-4001	User profile picture not found
PRO-4002	Unable to parse the user profile picture
PRO-5001	Import calendar not supported
PRO-5100	Import calendar not authorized

4.6 SMS errors

Code	Message
SMS-1000	Unknown exception in SMS handling.
SMS-1101	The text SMS is empty or too long.
SMS-1102	Unable to send the text SMS message.
SMS-1003	Incorrect delivery service configuration.

4.7 System errors

Code	Message
SYS-1000	Unknown exception in system handling.
SYS-1002	Model not found.
SYS-1003	Undefined used storage size

4.8 PIM errors

Code	Message
PIM-1000	Unknown exception in PIM handling
PIM-1101	Malformed range of dates
PIM-1100	Unknown exception in calendar handling
PIM-1200	Unknown exception in contacts handling
PIM-1102	Malformed interval of dates
PIM-1103	You must specify the range or the ID of the event
PIM-1104	Unable to parse the contact picture
PIM-1105	Missed the `dtstart` or the `dtend` required parameters
PIM-1106	Malformed date
PIM-1107	Missing required parameter
PIM-1108	Non-existing event
PIM-1201	No or empty data parameter passed
PIM-1202	Contact ID should not be used with limit or offset
PIM-1203	Item with the specified ID doesn't exist
PIM-1204	Contact does not belong to the user
PIM-1205	Filtervalue should not be used with contact ID
PIM-1230	Field not allowed in sorting

4.9 Media errors

Code	Message
MED-1000	Unknown exception in media handling

Code	Message
MED-1001	The size declared in the header does not match the one declared in the metadata.
MED-1002	The size of the uploading media does not match the one declared
MED-1003	Media ID shouldn't be used with limit or offset
MED-1005	Unable to retrieve media with the given ID
MED-1006	Invalid Content-Range
MED-1007	User quota reached
MED-1008	Unable to retrieve media file
MED-1011	Invalid key
MED-1012	Access to set denied
MED-1013	The provided status is not valid
MED-1014	Operation forbidden due some media are not available
MED-1015	Change status on existing item (which status is `N` or `P`) not allowed
MED-1016	Operation forbidden on a media not validated for illicit content
MED-1017	Operation forbidden on a media not yet validated
MED-1019	Generic transcoding error
MED-1020	File size exceeds configured max file size limit
MED-1021	Operation forbidden on a media item not validated due to being infected by a virus
MED-1022	File already soft deleted
MED-1023	File not soft deleted
MED-1024	Invalid granularity

4.10 Picture errors

Code	Message
PIC-1000	Generic error code for pictures
PIC-1001	No pictures specified
PIC-1003	Picture ID should not be used with limit or offset
PIC-1004	Invalid rotation amount
PIC-1005	Unable to retrieve picture
PIC-1006	Maximum user quota has been reached
PIC-1007	File is too large

4.11 External services errors

Code	Message
EXT-SRV-1000	General external service error
EXT-SRV-1001	Service name is not specified
EXT-SRV-1002	Service is not found
EXT-SRV-1003	Service is not available
EXT-SRV-1004	User is not authorized for external service
EXT-SRV-1005	Album is not specified
EXT-SRV-1006	Generic upload Error
EXT-SRV-1007	Service doesn't support albums
EXT-SRV-1008	Service doesn't support privacy settings
EXT-SRV-1009	Upload limit reached
EXT-SRV-1010	Album does not exist on external service
EXT-SRV-1011	Service does not support export of multiple items
EXT-SRV-1012	Request rate too high
EXT-SRV-1100	Unknown exception in Facebook friends handling

4.12 Last activities errors

Code	Message
ACT-1000	General activities error

4.13 Label errors

Code	Message
LAB-1000	Unknown exception in label handling
LAB-1001	The provided label doesn't exists
LAB-1002	Missing mandatory field item
LAB-1003	Missing mandatory parameter labelid
LAB-1004	Missing mandatory parameter name
LAB-1005	Missing mandatory array items
LAB-1006	There is already a label with this name for the specified user
LAB-1007	The specified label type is invalid or mismatch with the stored label type

4.14 Subscription errors

Code	Message
SUB-1000	General subscriptions error

Code	Message
SUB-1001	You must specify a valid plan name paramater
SUB-1002	User's subscriptions not found
SUB-1003	You must specify a valid status parameter
SUB-1004	You can migrate a canceled plan to an active plan only
SUB-1005	You can migrate an active plan to another with status `active` or `payment_required` only
SUB-1006	You can not migrate a plan in `migrated` or `payment_required` status

4.15 Folder errors

Code	Message
FOL-1000	Unknown exception in folder handling
FOL-1002	Unknown parent folder ID
FOL-1003	Unknown media type
FOL-1004	Unknown folder ID
FOL-1005	Missing mandatory parameter *`folderid`*
FOL-1006	Media type unsupported
FOL-1007	Missing mandatory parameter *`itemid`*
FOL-1008	Missing mandatory parameter *`folderids`*
FOL-1009	Duplicated *magic folder*
FOL-1010	Cycling redundancy using the given parent ID
FOL-1012	Duplicated offline folder
FOL-1013	Folder cannot be *magic* and offline at the same time
FOL-1014	Offline folder must be a child of a *magic folder*

4.16 Family errors

Code	Message
FAM-1000	Generic error saving family
FAM-1001	Cannot create an empty family
FAM-1002	User already in a family
FAM-1003	User not present in the family
FAM-1004	External ID already present, must be unique
FAM-1005	Family not found
FAM-1006	Family ID shouldn't be used with limit or offset
FAM-1008	Parameter *`type`* cannot be used with `family` or `folder` *`source`*
FAM-1009	Item was shared by this user to the family

Chapter 5. JSON parameters and objects

This section lists the JSON objects common to multiple APIs and their syntax.

5.1 Address

Name	Mandatory	Type	Value
city	N	String	City of the address
country	N	String	Country of the address
extended_address	N	String	Extended address of the address
po_box	N	String	Postal office box of the address
region	N	String	Region of the address
street_address	N	String	Street of the address
t	Y	String	The type of the address (`home` | `work` | `other`)
zip	N	String	Zip code of the address

5.2 Alarm

Name	Mandatory	Type	Value
action	Y	String	It must be 'alert'
duration	Y	Number	How far in advance the alarm for the event should be triggered (in minutes)

5.3 Alarm response

Name	Mandatory	Type	Value
description	Y	String	The description of the alert
time	N	String in RFC 2445 format	The time when the event is going to happen

5.4 Album

Field	Type	Description
albumid	String	External ID of the album
name	String	Human-readable title of the album
privacy	String	Privacy level of the album (in external service format)

5.5 Album to be created

Name	Type	Value
albumprivacy	String	Privacy value
attributes	Array	Array of JSON objects containing parameters and values with fields to be set for the created album. (*Not mandatory in portal, but most services require the title attribute for new albums*)
servicename	String	The service where the album should be created

5.6 Audio

Name	Mandatory	Type	Value
id	Y	String	Unique ID of the item
url	Y	String	URL of the binary file of the item
viewurl	N	String	URL of the album cover
creationdate	Y	Number	The creation date of the audio track, retrieved from metadata or from server time
modificationdate	Y	Number	The modification date of the audio track, retrieved from metadata or from server time
date	Y	Number	Date of the last update of the audio track that can be propagated during a synchronization. For example, a rename operation updates the date field
size	Y	Number	Dimension of the file in bytes
name	Y	String	Name of the audio item
metadata	N	JSON object	Metadata if available in the audio item. See Section 5.7, “Audio Metadata”
exported	N	JSON array	The services where the audio track has been exported to (for example, Flickr) and the date and time when the audio was exported

Name	Mandatory	Type	Value
etag	Y	String	The ETag of the audio item, in order to understand if it is changed
softdeleted	Y	Boolean	Specifies if the audio item is soft deleted, that is, in the trash bin
postingdate	N	Number	Date of posting to the Family Hub

5.7 Audio Metadata

Name	Mandatory	Type	Value
artist	N	String	Author of the song
album	N	String	Album in which the song is included
track_number	N	Number	The track number of the song inside the album
title	N	String	The title of the song
album_cover	N	String	The url of the album art of this song
genre	N	String	Genre of the song
codecname	N	String	The name of the codec used to encode the audio data - examples are: aac, mp3, alac
duration	N	String	The duration of the track in milliseconds
formatname	N	String	The name of the container format - examples are: aac, mp3, m4a
bitrate	N	String	The bitrate of the song

5.8 Attendee

Name	Mandatory	Type	Value
cn	N	String	Common Name of the attendee
expect	N	String	One of • `fyi` • `require` • `request`

Name	Mandatory	Type	Value
			• `immediate`
id	Y	String	Unique identifier of the attendee in the Funambol database
kind	N	String	One of • `individual` • `group` • `resource` • `room`
role	N	String	For vCalendar, one of • `attendee` • `delegate` • `organizer` • `owner` For iCalendar, one of • `req-participant` • `opt-participant` • `non-participant` • `chair`
rsvp	N	String	For vCalendar, one of • `yes` • `no` For iCalendar, one of • `true` • `false`
status	N	String	One of • `needs-action` • `accepted` • `declined` • `sent` • `delegated`

Name	Mandatory	Type	Value
			• `tentative` • `completed` • `confirmed`
uri	Y	String	Reference to the attendee. It can have the following values: • `mailto:john.smith@acme.com` (iCalendar) • `John Smith <john.smith@acme.com>` (vCalendar) • `https://www.xyz.com/yourvcard.vcf` (reference to a vCard object. See fields `valuetype` and `valuetypeformat`)
valuetype	N	String	e.g. `VALUE=URL`
valuetypeformat	N	String	e.g. `TYPE=VCARD`

5.9 Calendar

Note

The *Calendar* object is the container for events. It contains reserved values that are not fully supported and may be removed or be subject to changes in future versions of the API.

Name	Value
accessLevel	Reserved for future use
acl	Reserved for future use
cn	Reserved for future use
color	Reserved for future use
defaultRole	Reserved for future use
email	Reserved for future use
id	Reserved for future use
selected	Reserved for future use
summary	Reserved for future use
title	Reserved for future use
where	Reserved for future use

5.10 Carrier

Name	Mandatory	Type	Value
active	Y	Boolean	Is the carrier still active?
countryid	Y	String	The country ID (note that this is the number, for example 144, not the A2 code) for the carrier
id	Y	Number	Unique carrier ID
name	Y	String	Carrier name, for example Telcel.
otasupport	Y	Boolean	Does the carrier support OTA configuration?
syncmlsupport	Y	Boolean	Does the carrier allow SyncML synchronization?
trustedjam	Y	Boolean	Does the carrier support trusted JAM applications? (Not used yet)

5.11 Contact

Name	Mandatory	Type	Value
address	N	JSON array	Addresses (every address is a String in vCard format)
Anniversary	N	String	(**DEPRECATED**)
Birthday	N	String	(**DEPRECATED**) The birthday in the format `yyyy-mm-dd`
anniversaries	N	Array	List of JSON objects containing anniversaries
birthdays	N	Array	List of JSON objects containing birthdays in the format `yyyy-mm-dd`
categories	N	String	Categories
children	N	String	
displayname	N	String	Display name
email	N	JSON array	Emails
events	N	Array	Array of events
extension	N	Array	Extended fields. Valid types are: • `x-accountname`

Name	Mandatory	Type	Value
			• `x-accounttype` • `x-adrcustom` • `x-emailcustom` • `x-favorite` • `x-telcustom`
firstname	N	String	First name
folder	N	String	
gender	N	String	
hobbies	N	String	
id	Y (for update)	String	Contact ID in the database
im	N	JSON array	Instant messaging
importance	N	Number	
impp	N	Array	List of IMPP URIs. An IMPP URI can also have a *preferred* flag: `p`
initials	N	String	
languages	N	String	
lastname	N	String	Last name
mileage	N	String	
nickname	N	String	
notes	N	JSON array	Notes
org	N	Object	The organization of the contact (see Section 5.33, “Orgs”)
orgs	N	JSON array	Organization
phone	N	JSON array	Phone numbers
sensitivity	N	String	Valid values are: • `normal` • `personal` • `private` • `confidential`
spouse	N	String	
subject	N	String	
suffix	N	String	
titles	N	Array	Array of JSON objects

Name	Mandatory	Type	Value
webpage	N	JSON array	Webpages

The image submission is done separately as multi-part.

Name	Mandatory	Type	Value
t	Y	String	Type
v	Y	String	Value

Supported types for the parameters are the following:

`email`
: home, main, work

`im`
: other

`phone`
: company, home, home2, homefax, mobile, other, other2, otherfax, pager, work, work2, workfax

`webpage`
: main, other, work

5.12 Country

Name	Mandatory	Type	Value
a2	Y	String	The country A2 ID, for example `US`
active	Y	String	Is the country still existent? For example Yugoslavia is marked with ***`active`*** = `false`.
id	Y	String	The country unique ID
name	Y	String	The country name, for example United States
phoneprefix	Y	String	The country calling code. The Caribbean nations in zone 1 includes the area codes.

5.13 Download URL

Name	Mandatory	Type	Value
downloadpath	Y	String	The relative path of the App to be downloaded
portalurl	Y	String	The URL of the Portal. For example `https://my.server.com`. Combining `portalurl` and `downloadpath`,

Name	Mandatory	Type	Value
			the full URL of the App is available.

5.14 Email provider account

Name	Mandatory	Type	Value
accounttype	Y	String	Email account server type
login	Y	String	Username of the account on the mail server
password	Y	String	Password of the account on the mail server (encoded as Base64 string)

5.15 Event

Name	Mandatory	Type	Value
alarms	N	Array	The array containing the alarms associated to this event (see Section 5.2, “Alarm”)
attendees	N	Array	The array containing the attendees who will join this event
calendarId	Y	String	Calendar ID in the database. As OneMediaHub currently supports one calendar per user, it should default to username
description	N	String	Description
dtend	Y	String in RFC 2445 format	End date. Date is exclusive
dtstart	Y	String in RFC 2445 format	Start date. Date is inclusive
exdate	N	String in RFC 2445 format	Exception date, used to specify the date of the exception to be created in the database. It applies to updates only.
extension	N	Array	Extended fields. Valid types are: • `x-accounttype` • `x-accountname`

Name	Mandatory	Type	Value
id	Y (for update)	String	Event ID in the database. The event ID is mandatory for an update but should not be provided when adding an event
latitude	N	Number	Latitude of the location associated with the calendar event
longitude	N	Number	Longitude of the location associated with the calendar event
privacy	N	String	Access classification, can be: • `trdefault` • `public` • `private`
rrule	N	JSON object	Recurrence rule. See Section 5.44, "Recurrent Rule" for more information.
summary	N	String	Event summary
tzdtend	N	String	Timezone for the end date. If not provided, the user timezone is used. Applies to new events only
tzdtstart	N	String	Timezone for the start date. If not provided, the user timezone is used. Applies to new events only
where	N	String	Location of the event

5.16 Exif

Name	Mandatory	Type	Value
Date Time Original	N	String	The date and time when the image data was generated
Exif Image Width	N	String	Width of the image
Exif Image Length	N	String	Length of the image
Make	N	String	Manufacturer of the device

Name	Mandatory	Type	Value
Model	N	String	Model of the device
Orientation	N	String	The orientation of the camera relative to the scene, when the image was captured
XResolution	N	String	Display resolution of image
GPS Latitude	N	String	GPS Latitude
GPS Longitude	N	String	GPS Longitude

5.17 Export media

Name	Mandatory	Type	Value
albumid	N	String	The ID of the album where the item should be uploaded
itemattributes	N	Array	Array of JSON objects containing parameters and values with fields to be uploaded together with the media item. Supported properties are: **`title`** The media item name **`description`** Description of the media item **`location`** Location of the media item **`longitude`** Longitude information to geocode the media item **`latitude`** Latitude information to geocode the media item **`tags`** An array of tags for the item

Name	Mandatory	Type	Value
itemprivacy	N	String	Specifies the privacy type for the media item. If the service supports only album privacy, no privacy should be specified since the `albumid` is already part of the request
items	Y	Number	The IDs of the items to be uploaded
servicename	Y	String	The service where the item should be created

5.18 External service

Name	Type	Value
accountname	String	Name of the user's account at the external service.
albumattributes	Array	Array of album attributes supported by the service. Supported properties are: **`title`** Title of the album **`description`** Description of the new album
albumprivacy	Array	Specifies privacy options per album.
apikey	String	Service API key, for the cases were the API is needed at client side.
authorized	Boolean	Specifies if a service is available on the server. It is `true` only when the user has a valid session for this service. If the service cannot be used (e.g. no API key), then the service should not be returned at all.
authurl	String	URL to which the user should be redirected in order to get authorization tokens.
displayname	String	Human-readable name of the service.
exportmultiple	Boolean	Specifies if the service supports the upload of multiple items at once
hasalbums	Boolean	Specifies if a service supports albums.
hasprivacy	Boolean	Specifies if a service supports privacy settings.
icon	String	URL with the service logo.
itemattributes	Array	Array of JSON objects containing parameters and values with fields to be uploaded together with the media item. Supported properties are: **`title`** The media item name

Name	Type	Value
		`description` Description of the media item **`location`** Location of the media item **`longitude`** Longitude information to geocode the media item **`latitude`** Latitude information to geocode the media item **`tags`** An array of tags for the item
itemprivacy	Array	Specifies privacy options per item (picture or other media).
lastusedalbum	String	Specifies the last used album for an external service (**optional**).
lastuseditemprivacy	String	Specifies the last used item privacy for an external service (**optional**).
servicename	String	Internal name of the service.
sources	Array	Specifies the sources that can be shared using this external service. Possible values in the array are: `picture`, `video`, or `file`).
supportrecipients	boolean	Specifies if a service supports sharing to recipients
authtype	String	Specifies the authentication type for the given service. `Native` means that the server ignores the authentication type which will be responsability of the service itself.

5.19 File

Name	Mandatory	Type	Value
creationdate	Y	Number	The creation date of the file
modificationdate	Y	Number	The modification date of the file, retrieved from metadata or from server time
date	Y	Number	Date of the last update of the file that can be propagated during a synchronization
id	Y	String	Unique ID of the file
name	Y	String	Name of the file
size	Y	Number	Dimension of the file in bytes

Name	Mandatory	Type	Value
url	Y	String	URL of the binary file of the item or Server API to be called to get the decrypted binary file of the item
mediatype	Y	String	The media type of the item. For files it could be set to "file"
status	Y	String	One between `U`, `C`, `V`, `I`. The content uploaded status server side
etag	Y	String	The ETag of the file item, in order to understand if it is changed
softdeleted	Y	Boolean	Specifies if the file is soft deleted, that is, in the trash bin
postingdate	N	Number	Date of posting to the Family Hub

5.20 File update

Name	Mandatory	Type	Value
id	Y	String	The ID of the file to be updated
name	Y	String	The new name for the file
status	Y	String	One between `U`, `C`, `V`, `I`. The content uploaded status server side

5.21 Folder

Name	Mandatory	Type	Value
id	N	Number	The ID of the folder. Mandatory in case of folder update (if missing in the request it means an add will be performed) and in the response
mediatypes	N	JSON Array	The list of the items belonging to the given folder. In case of update the specified items will be added to the existing ones

Name	Mandatory	Type	Value
name	N	String	The name of the folder, mandatory in case of folder creation and in the response
items	N	JSON Array	The list of the items belonging to the given folder. In case of update the specified items will be added to the existing ones
devicename	N	String	The name of the device the folder belongs to
magic	N	Boolean	Specifies if the folder is the unique *magic folder* (default `false`)
parentid	N	Number	Specifies the ID of the existing folder the folder belongs to
status	N	String	The status of the folder at server side. Mandatory in the response
date	N	Number	Date of the last update of the file that can be propagated during a synchronization. Mandatory in the response
offline	N	Boolean	Specifies if the folder is the unique offline folder (default `false`)

5.22 Generic user information

Name	Mandatory for a new user	Type	Value
birthday	N	String	Date of birth (YYYYMMDD). Default is `19000101`
firstname	N	String	The first name of the new user
l10n	N	String	The locale of the new user, for example `en-US`.

Name	Mandatory for a new user	Type	Value
			Note To allow maximum flexibility of the localization code, this value is not validated. So any string is acceptable as a `l10n` parameter.
lastname	N	String	The last name of the new user
mailinglist	N	Boolean	Subscribe to the Portal mailing list (also known as the "keep me informed" flag)
male	N	Boolean	Gender of the user (male or female). Default is male
password	Y	String	The plain password of the new user
oldpassword	N	String	In case of change of password, the previous one. **Important** This field is **mandatory** when the `password` field is passed together to change the authenticated user password.
timezone	N	String	The timezone of the new user. Supported values are available calling the SAPI described at Section 3.9.5, “Get timezones”.

Name	Mandatory for a new user	Type	Value
useremail	Y	String	The email address of the new user
userid	Y	String	The user ID of the new user. It is mandatory for a new user but it cannot be specified during an update. Note that it is not possible to change the username once defined.
preferredcommunicationchannel	N	String	The preferred communication channel for receiving user communications. Possible values are “email” and “sms”.

5.23 Interval

Name	Mandatory	Type	Value
from	Y	String in RFC 2445 format	Start date of the interval. The date is inclusive
to	Y	String in RFC 2445 format	End date of the interval. The date is exclusive

5.24 Label

Name	Mandatory	Type	Value
labelid	Y	Number	The label ID in the database **Note** This field is not mandatory for the API call described at Section 3.5.30, “Create a label”
name	Y	String	The label name
items	N	Array	JSON array containing the IDs of the items marked with this label
type	N	String	The media type of the items of this label

5.25 Last activities

Name	Mandatory	Type	Value
activitytype	Y	String	The type of the activity to be saved. This should be generic to accept any type of activity but for *add*, *update*, and *delete* operations `add`, `update`, and `delete` should be used
deviceid	Y	String	ID of the device that executed the activity
endtime	Y	Number	GMT date, in milliseconds, when the changes were terminated
received	N	Number	Number of items received by the device
sent	N	Number	Number of items sent by the device
source	Y	String	Name of the data source where the changes occurred
starttime	Y	Number	GMT date, in milliseconds, when the changes were initiated
status	N	String	The status of the last sync. Error code instead, if something went wrong at client side
aborted	N	Number	The number of errors during the upload from the client to the server
rejected	N	Number	The number of errors during the download from the server to the client

Note

For backward compatibility, the parameters *`aborted`* and *`rejected`* are not mandatory.

5.26 Last sync

Name	Mandatory	Type	Value
devicedescription	N	String	The device's description

Name	Mandatory	Type	Value
deviceid	Y	String	The device ID; for example `fwm-3519800106775801`
devicetype	Y	String	The device type; for example `phone`
endsync	Y	Number	Ending time of the synchronization
startsync	Y	Number	Starting time of the synchronization
status	Y	Number	Synchronization Status: • `200` – OK • `224` – SUSPEND
syncsource	Y	String	Source URI
synctype	Y	Number	Synchronization type

5.27 Manufacturer

Name	Mandatory	Type	Value
id	Y	Number	Manufacturer ID
name	Y	String	Name of the manufacturer, for example `Nokia`
popular	Y	Boolean	Is this one of the most popular manufacturers? Default is false. The attribute is used to easily select the favorite manufacturers

5.28 Media export

Name	Type	Value
albumid	String	The ID of the album where the item should be uploaded.
itemattributes	Array	Array of JSON objects containing parameters and values with fields to be uploaded together with the pictures or videos.
itemid	String	The ID of the item to be uploaded (**DEPRECATED**)
itemprivacy	String	Specifies the privacy type for the media item. If the service supports only album privacy, no privacy

Name	Type	Value
		should be specified since the *albumid* is already part of the request.
items	Array	Array of picture or video IDs. For services which support export of more then one media item this array may contain more then one ID.
servicename	String	The service where the media file should be created.
recipients	Array	Array of recipients to share the items with (not used by current services)

5.29 Mobile signup user

Name	Mandatory for a new user	Type	Value
carrier	N	String	The carrier name
countrya2	N	String	The A2 country code of the new device
manufacturer	N	String	The manufacturer name
model	N	String	The model name
password	Y	String	The password in plain text of the new user
phonenumber	Y	String	This is going to be the username used by the new user to login.
platform	Y	String	The OneMediaHub platform, mandatory for OneMediaHub Apps
timezone	N	String	The timezone of the new user
useremail	N	String	The email address of the new user
l10n	N	String	The locale of the new user, for example `en-US`. **Note** To allow maximum flexibility of the localization code, this value

Name	Mandatory for a new user	Type	Value
			is not validated. So any string is acceptable as a `l10n` parameter.

5.30 Model

Name	Mandatory	Type	Value
active	Y	Boolean	The model is supported in OneMediaHub. An older device might not be supported anymore, but still be available as active=false to support existing customers and users.
fnblplugin	Y	Number	Specifies which OneMediaHub App is supported by the model. • `0`: no OneMediaHub App • `1`: OneMediaHub for Windows Mobile (Smartphone, no touchscreen) • `2`: OneMediaHub for Windows Mobile (Pocket PC, touchscreen) • `3`: OneMediaHub for BlackBerry (BBOS previous to 4.7) • `4`: OneMediaHub for iPhone • `5`: OneMediaHub for Symbian (S60 3rd Ed. and S60 3rd Ed. FP1 → SYMB1) • `6`: OneMediaHub for Symbian (S60 3rd Ed. FP2 and S60 5th Ed. → SYMB2)

Name	Mandatory	Type	Value
			• 7: OneMediaHub for BlackBerry (BBOS 4.7) • 8: OneMediaHub for Android > 2.0 • 9: OneMediaHub for BlackBerry (BBOS 6, BBOS 7 and following) • 10: OneMediaHub for Windows Phone If no (legacy) Java ME Email client is available (*jam* = 0) and no OneMediaHub App is available (*fnblplugin* = 0) any call to the method detailed in Section 3.3.7, "Get the download URL for the OneMediaHub App given a phone" will return a PRO-1111 error code.
id	Y	String	The internal model ID in the OneMediaHub database
imagefileid	Y	String	The name of the image file to be displayed. The full path is returned by the method detailed in Section 3.9.7, "Get picture, info, and sync URL for a device". Get picture, info and sync URL for a device Server API.
infofileid	Y	String	The name of the HTML file (without extension) that provides the configuration information for the device. The full path is returned by the method detailed in Section 3.9.7, "Get picture, info, and sync URL for a device". Get picture, info and

Name	Mandatory	Type	Value
			sync URL for a device Server API.
jam	Y	Number	Specifies which type of the (legacy) Java ME Email client is supported by the model: • `0`: no (legacy) Java ME Email client available • `1`: NokiaS602 • `2`: Moto_Low • `3`: NokiaS603 • `4`: Nokia6630 • `5`: NokiaN80 • `6`: NokiaS603FP1 • `7`: SymUIQ • `8`: SE • `9`: NokiaE61 • `10`: NokiaN73 • `11`: Moto_High • `12`: MotoLinux • `13`: BB • `14`: LG • `15`: LGWideScreen • `16`: SamsungHighRes • `17`: SamsungLowTech • `18`: SamsungS60 If no (legacy) Java ME Email client is available (*`jam`* = `0`) and no OneMediaHub App is available (*`fnblplugin`* = `0`) any call to the method detailed in Section 3.3.7, “Get the

Name	Mandatory	Type	Value
			download URL for the OneMediaHub App given a phone" will return a PRO-1111 error code.
manufacturerid	Y	Number	The manufacturer ID in the OneMediaHub database
maxemails	Y	Number	Specifies the maximum number of emails the device can handle using the (legacy) Java ME Email client.
name	Y	String	The model name
otasupport	Y	Boolean	Specifies if the model supports OTA settings. Android, BlackBerry, iPhone and Windows Mobile devices, for example, do not support this field.
pimpush	Y	Boolean	Specifies if the device supports PIM push natively. If the OneMediaHub App is used, the information is ignored.
syncfileid	Y	String	The name of the HTML file (without extension) that provides the synchronization information for the device. The full path is returned by the method detailed in Section 3.9.7, "Get picture, info, and sync URL for a device". Get picture, info and sync URL for a device Server API.
syncml	Y	Boolean	Specifies if the model supports a native client. If `pluginrequired` > `0`, this parameter might not be relevant as you may want to use the OneMediaHub App for the given device.

Name	Mandatory	Type	Value
utcsupport	Y	Number	Specifies if the native `SyncML` client supports UTC correctly. If not, events are sent to the device in local time. The possible values are: • `0`: unknown – it is unknown whether the device supports UTC • `1`: true - the device supports UTC • `2`: false - the device does not support UTC
utf8support	Y	Boolean	Specifies if the model supports UTF8; for recent smartphones, it is generally set to `true`. Related only to the device's native PIM sync client, not to the OneMediaHub App.

5.31 Model URL

Name	Mandatory	Type	Value
imagepath	Y	String	The relative path of the image of the phone. For example, `/html/devices/images/NokiaE61.gif`.
infopath	N	String	The relative path of the HTML fragment with the setup information for the phone. For example, `/html/devices/setup/NokiaMC4.html`.
portalurl	Y	String	The URL of the Portal (for example, `https://my.server.com`). Combining `portalurl` and the image or HTML path, the full URL is defined.
syncpath	N	String	The relative path of the HTML fragment with

Name	Mandatory	Type	Value
			the synchronization information for the phone. For example, `/html/devices/sync/NokiaMS7.html`.

5.32 OAuth

Name	Mandatory	Type	Value
accesstoken	Y	String	OAuth 2 Access Token
valid	Y	Boolean	Indicates if the token is still valid
platform	N	String	Platform of the client that required the token. Can be one of • `windows` • `ios` • `macos` • `android` • `web`
refreshtoken	N	String	OAuth 2 Refresh Token

5.33 Orgs

Name	Mandatory	Type	Value
department	N	String	The department of the contact
org	N	String	The company of the contact
title	N	String	The job title of the contact

5.34 OTA configuration

Name	Mandatory	Type	Value
accountpassword	Y	String	The password to set in the account
accountusername	Y	String	The username to set in the account
contenttype	Y	String	The content type of the source (e.g. `text/x-vcard`)
contentversion	Y	String	The version of the content type (e.g. `2.1`)

Name	Mandatory	Type	Value
devicefwv	N	String	The device firmware version. It is used only if the manufacturer, the model, the software version and the hardware version are specified.
devicehwv	N	String	The device hardware version. It is used only if the manufacturer, the model and the software version are specified.
devicemanufacturer	N	String	The device manufacturer, for example `nokia`. Note that this is not the unique manufacturer ID as for Section 3.9.1, "Get manufacturers" but the manufacturer name as in the OTA profiles folder. If no *`devicemanufacturer`* is specified, the generic OTA configuration *`profile.properties`* is used as the OTA profiles folder has a tree structure (`$FUNAMBOL_HOME/config/com/funambol/otacp/profiles-repository`).
devicemodel	N	String	The device model, for example the device e61 (`nokia \e61\profile.properties`). Note that this is not the unique model ID as for Section 3.9.2, "Get phone models by manufacturer", but the model name as in the OTA profiles folder. It is used only if the manufacturer is specified. If no *`devicemodel`* is specified, the default OTA configuration for the given *`devicemanufacturer`* is used (for example `nokia \profile.properties`) as the OTA profiles folder has a tree structure (`$FUNAMBOL_HOME/config/com/funambol/otacp/profiles-repository`).
deviceswv	N	String	The device software version. It is used only if the manufacturer and the model are specified.
name	Y	String	The name of the sync source as required by some phones. Usually one between: • `Contact` • `Calendar`
sources	Y	Array	Array of the sync sources to configure as defined in this table (see parameters

Name	Mandatory	Type	Value
			uri, *name*, *contenttype*, and *contentversion* here below)
uri	Y	String	The URI of the source
userpin	Y	String with only digits	The user pin to use in sending the OTA message (see [5])

5.35 Password reset

Name	Mandatory	Type	Value
password	Y	String	The new password for the user
resettoken	Y	String	A valid reset token for this account

5.36 Phone

Name	Mandatory for a new phone	Type	Value
carrierid	Y	Number	The carrier ID of the new device
converttmz	N	Number	The UTC timezone conversion option of the device: • `2`: convert date • `1`: do not convert date • `0`: not specified If not specified, in the synchronization phase the device capabilities will be used
countrya2	Y	String	The A2 country code of the new device
modelid	Y	Number	The model ID of the new device
phonenumber	Y	String	The phone number of the new device

5.37 Picture

Name	Mandatory	Type	Value
creationdate	Y	Number	The creation date of the picture, retrieved from Exif or from metadata or from server time

Name	Mandatory	Type	Value
date	Y	Number	Date of the last update of the picture that can be propagated during a synchronization. For example, a rename operation updates the date field while a rotate or view does not.
exif	N	JSON object	Exif data if available in the picture as for Sanselan documentation
exported	N	JSON array	The services where the picture has been exported to (for example, Flickr) and the date and time when the picture was exported
id	Y	String	Unique ID of the picture
modificationdate	Y	Number	The modification date of the picture, retrieved from Exif or from metadata or from server time
name	Y	String	Name of the picture
size	Y	Number	Dimension of the file in bytes
thumbnails	N	JSON array	Thumbnails if available according to server configuration. See Section 5.55, “Thumbnail” for more information
url	Y	String	URL of the binary file of the picture
viewurl	Y	String	URL of a reduced-size copy of the picture for display purposes. This image can be rotated as well. It can be equal to the URL specified in the configuration of the server if the original image is already small
mediatype	Y	String	The media type of the item. For pictures it is set to "picture"

Name	Mandatory	Type	Value
status	Y	String	One between `U`, `C`, `V`, `I`. The content uploaded status server side
etag	Y	String	The ETag of the picture item, in order to understand if it is changed
labels	N	JSON array	The labels for this picture. See Section 5.24, “Label”
softdeleted	Y	Boolean	Specifies if the picture is soft deleted, that is, in the trash bin
postingdate	N	Number	Date of posting to the Family Hub

5.38 Picture update

Name	Mandatory	Type	Value
id	Y	String	The ID of the picture to be updated
name	Y	String	The new name for the picture
status	Y	String	One between `U`, `C`, `V`, `I`. The content uploaded status server side

5.39 Properties (JSON objects)

Name	Mandatory	Type	Value
properties	N	Array JSON objects	List of properties to be added/updated

5.40 Properties (strings)

Name	Mandatory	Type	Value
properties	N	Array of strings	List of properties to be returned

5.41 Push info

Name	Mandatory	Type	Value
sources	Y	Array	Array of the sync sources to be pushed, as defined in Section 5.49, “Source”

Name	Mandatory	Type	Value
uimode	N	Number	Server recommendations: whether the session should be executed in the background or a notification should be displayed to the user. The possible values are: **0** Not specified. The server does not have a recommendation. **1** Background. The server recommends that the sync should be run in background. **2** Informative. The server recommends that a notification is displayed on the client. **3** User interaction. The server recommends that an informative notification is displayed to the user.

5.42 Push notification message

Name	Mandatory	Type	Value
contenttype	N	String	The content type to be specified in the push message (e.g. `text/x-vcard`). If not specified, the content type is retrieved from the source definition. If the specified source URI does not match any of the server sync sources and the content

Name	Mandatory	Type	Value
			type is not specified, the PRO-2004 error is returned.
sources	Y	Array	Array of the sync sources to be pushed.
synctype	N	Number	The requested sync type. One between: • `206`: two-way (default value) • `207`: one-way from client to server • `208`: refresh from client • `209`: one-way from server to client • `210`: refresh from server
uimode	N	Number	This field specifies the server recommendations: whether the session should be executed in the background or a notification should be displayed to the user. The possible values are: • `0`: not specified. The server does not have a recommendation. • `1`: background. The server recommends that the sync should be run in background. Default value. • `2`: informative. The server recommends that a notification is displayed on the client. • `3`: user interaction. The server recommend that an informative notification is displayed to the user.

Name	Mandatory	Type	Value
uri	Y	String	The URI of the source (e.g. `card`)

5.43 Range

Name	Mandatory	Type	Value
from	Y	String in YYYYMMDD format	Start date of the range. The date is inclusive (from 00:00:00 in GMT time)
to	Y	String in YYYYMMDD format	End date of the range. The date is exclusive (until 00:00:00 in GMT time)

Note

if the request is for a user in a timezone different than GMT, the request might need to add an extra day to the from/to of the range.

5.44 Recurrent Rule

Name	Mandatory	Type	Value
BYDAY	N	Object	Applicable for *`FREQ`* == `WEEKLY` only. Array of days when the event occurs; for example: • `[TU, TH]` - every Tuesday and Thursday. The first recurrence is the one after the *`DTSTART`*.
BYDAY	N	String	Applicable for *`FREQ`* == `MONTHLY` only. The week number followed by day abbreviation. The first recurrence is the one after the *`DTSTART`*.
BYMONTHDAY	N	String	Applicable for *`FREQ`* == `MONTHLY` only. The day of the month when the event occurs; for example: • `4` - every 4th day in month

Name	Mandatory	Type	Value
DTEND	Y	String in RFC 2445 format	End date of the first instance of the recurrence. This matches the dtend of the event.
DTSTART	Y	String in RFC 2445 format	Start date of recurrence. This matches the dtstart of the event.
FREQ	Y	String	Frequency, can be: • `DAILY` • `WEEKLY` • `MONTHLY` • `YEARLY`
INTERVAL	N	Number	Repeat interval
TZDTEND	Y	String in RFC 2445 format	Timezone of the recurrence pattern, this usually matches tzdtend
TZDTSTART	Y	String in RFC 2445 format	Timezone of the recurrence pattern, this usually matches tzdtstart
UNTIL	N	String in RFC 2445 format	Date until which the event should be triggered

5.45 Role

Name	Mandatory	Type	Value
description	N	String	Description of the role (not used)
name	Y	String	Name of the OneMediaHub role, for example `standard` or `sync_user`

5.46 Save authorization token

Name	Mandatory	Type	Value
token	Y	String	This is the authentication token retrieved from the external service
expiretime	Y	Number	This is the token expiration date in miliseconds
servicename	Y	String	This is the external service name (e.g. facebook, picasa)

Name	Mandatory	Type	Value
accountname	Y	String	This is the account name for the corresponding external service (could be any arbitrary name)

5.47 Server information

Name	Mandatory for a new user	Type	Value
devid	Y	String	Device ID of the server
devtyp	Y	String	Device type of the server
exts	Y	String	External properties that could be used by the client
fwv	Y	String	Firmware version of the machine
hwv	Y	String	Hardware version of the machine
man	Y	String	Manufacturer of the server
mobileurl	Y	String	OneMediaHub Portal Mobile URL
mod	Y	String	Model of the server
oem	Y	String	Original Equipment Manufacturer
portalurl	Y	String	OneMediaHub Portal URL
sapiversion	Y	String	Version of the SAPI `.jar` file of the portal
supportlargeobjs	Y	Boolean	If the server supports large objects
supportnumberofchanges	Y	Boolean	If the server supports the `NumberOfChanges` tag
utc	Y	Boolean	If the server supports UTC time
verdtd	Y	String	DTD Version

5.48 SMS text message

Name	Type	Value
message	String	The content of the SMS message. Max allowed length 512 chars

5.49 Source

Name	Mandatory	Type	Value
contenttype	N	String	The content type to be specified in the push message (e.g. `text/x-vcard`). If not specified, the content type is retrieved from the source definition. If the specified source URI does not match any of the server's sync sources and the content type is not specified, the PRO-2004 error is returned.
synctype	N	Number	The requested sync type. One between: **`206`** Two-way (default) **`207`** One-way from client to server **`208`** Refresh from client **`209`** One-way from server to client **`210`** Refresh from server
uri	Y	String	The URI of the source (e.g. `card`)

5.50 Sources

Name	Mandatory	Type	Value
sources	N	JSON array	Array of the URI of the sync sources to configure, for example *`card`*. All the specified sources must be available in the server, otherwise

Name	Mandatory	Type	Value
			error code PRO-2004 will be returned.

5.51 Subscription request

Name	Mandatory	Type	Value
plan	Y	String	The name of the subscription plan
status	N	String	The status of the subscription plan. Allowed values are • `active`: the user can access and manage all his content in the OneMediaHub service • `canceled`: the user is not related to any subscription, but he can still access and download the content from the OneMediaHub service • `payment_required`: the user is requested to pay for a given subscription, but he can still access and download his content from the OneMediaHub service
expiredate	N	String in RFC 2445 format	The expiration date of the subscription to be saved. Valid only when invoking the API from a *System* security realm

5.52 Subscription response

Name	Mandatory	Type	Value
plan	Y	String	The name of the subscription plan
activated	N	String in RFC 2445 format	The time when the subscription has been activated
nextrenewal	N	String in RFC 2445 format	The time when the next renewal should be done

Name	Mandatory	Type	Value
laststatuschange	N	String in RFC 2445 format	The last time when the status has been changed
migratetoplan	N	String	The name of the subscription plan to which the current plan should be migrated
status	Y	String	The status of the subscription plan. Returned values are • `active`: the user can access and manage all his content in the OneMediaHub service • `canceled`: the user is not related to any subscription, but he can still access and download the content from the OneMediaHub service • `payment_required`: the user is requested to pay for a given subscription, but he can still access and download his content from the OneMediaHub service • `unknown`: returned in case of error, when the status of the subscription is undefined for some reason
expiredate	N	String in RFC 2445 format	The expiration date of the subscription, if present

5.53 Subscription payment

Name	Mandatory	Type	Value
id	Y	Number	The subscription payment identifier
plan	Y	String	The name of the subscription plan
status	Y	String	The status of the payment

Name	Mandatory	Type	Value
transactionid	Y	String	The transaction identifier

5.54 Subscription plan

Name	Mandatory	Type	Value
name	Y	String	The name of the subscription plan
price	Y	Number	The price of the subscription plan
currency	Y	String	The currency of the subscription plan
period	Y	String	The time used for the next renewal of the subscription plan
quota	Y	String	The storage associated to the subscription plan
default	Y	Boolean	`true` if the plan is the default, `false` otherwise
displayname	Y	String	The long name of the subscription plan
description	Y	String	A description of the subscription plan
activateable	Y	Boolean	`true` = the user can activate this subscription plan
message (**DEPRECATED**)	N	String	(**DEPRECATED**) The reason why the subscription plan cannot be activated
messagecode	N	String	The code of the reason why the subscription plan cannot be activated: **`NA-000`** The selected plan cannot be activated because it is the current plan **`NA-001`** You cannot downgrade to the free plan **`NA-002`** You cannot downgrade to this

Name	Mandatory	Type	Value
			plan because your quota exceeds the plan's **`NA-003`** You cannot change your plan due to missing payment **`NA-004`** VIP users cannot change their plan

5.55 Thumbnail

Name	Mandatory	Type	Value
size	Y	String	Dimension in pixels of the thumbnail as for server configuration (`170` = 170 × 170 pixels)
url	Y	String	URL of the binary thumbnail file of the picture or Server API to be called to get the decrypted binary thumbnail file.
etag	Y	String	The ETag of the specific thumbnail
w	N	Number	Width of the thumbnail
h	N	Number	Height of the thumbnail

5.56 Timezone

Name	Mandatory	Type	Value
default	Y	Boolean	Is this the country's main timezone? Only one timezone is marked as default in every country.
description	Y	String	The timezone description, for example `US/Pacific (GMT-08:00)`
id	Y	String	The timezone ID, for example `US/Pacific`

5.57 Upload binary files

Name	Mandatory	Type	Value
callback	N	String	A JavaScript method to be called in order to let the interface know the picture was already uploaded. This is needed because the item is posted using a form submit; it is a way of refreshing the interface without refreshing the entire page.
data	Y	String	An *Upload item* object with the information for a file. For more information about the *Upload item* JSON object, see Section 5.58, “Upload item”.
file	Y	String	The binary content of the file

5.58 Upload item

Name	Mandatory	Type	Value
contenttype	Y	String	Content type of the binary data
creationdate	N	Date	Local date string in RFC 2445 (`yyyymmddThhmmss`) format
id	N	String	The ID of the file to be uploaded; Media ID in the database. The media item ID is mandatory for an update, but should not be provided when adding a new item
modificationdate	N	Date	Local date string in RFC 2445 (`yyyymmddThhmmss`) format
name	Y	String	The name of the file to be uploaded
size	Y	Number	File size

Name	Mandatory	Type	Value
folderid	N	Number	The folder ID in the database

5.59 User

Name	Mandatory	Type	Value
emails	N	JSON array	An array of email accounts for the user. Only one email account per user is currently supported.
generic	Y	JSON object	A JSON object with the generic information about the user. For more information, see Section 5.22, “Generic user information”.
phones	N	JSON array	An array of phones for the user. Only one phone per user is currently supported. For more information, see Section 5.36, “Phone”.

5.60 User credentials

Name	Mandatory	Type	Value
userid	N	String	The user ID to be validated
phonenumber	Y	String	The phone number to be validated
password	Y	String	The password to be validated

5.61 User password

Name	Mandatory	Type	Value
userPwd	Only if a non-administrator user tries to delete himself	String	The user password

5.62 Video

Name	Mandatory	Type	Value
creationdate	Y	Number	The creation date of the video, retrieved from

Name	Mandatory	Type	Value
			metadata or from server time
modificationdate	Y	Number	The modification date of the video, retrieved from metadata or from server time
date	Y	Number	Date of the last update of the video, that can be propagated during a synchronization
exported	N	JSON array	The services where the video has been exported to (YouTube) and the date and time when the video was exported
id	Y	String	Unique ID of the video
name	Y	String	Name of the video
size	Y	Number	Dimension of the file in bytes
url	Y	String	The URL of the binary content of the video, or of the API call for retrieving the decrypted binary content of the video
metadata	N	JSON object	The video metadata of the given video item. See Section 5.63, “Video Metadata”
mediatype	Y	String	The media type of the item. For videos it is set to `video`
status	Y	String	One between `U`, `C`, `V`, `I`. The content uploaded status server side
etag	Y	String	The ETag of the video item, in order to understand if it is changed
playbackurl	N	String	The URL of the transcoded binary content of the video, or of the API call for retrieving the decrypted transcoded binary content of the video

Name	Mandatory	Type	Value
playbackcontenttype	N	String	The content type of the transcoded video
softdeleted	Y	Boolean	Specifies if the video is soft deleted, that is, in the trash bin
thumbnails	N	JSON object	Thumbnails if available according to server configuration. See Section 5.55, “Thumbnail” for more information
postingdate	N	Number	Date of posting to the Family Hub

5.63 Video Metadata

Name	Mandatory	Type	Value
duration	N	String	The duration of the video
bitrate	N	String	The bitrate of the video
codec	N	String	The coded used by the video
height	N	String	The height of the frame
width	N	String	The width of the frame

5.64 Video update

Name	Mandatory	Type	Value
id	Y	String	The ID of the video to be updated
name	Y	String	The new name for the video
status	Y	String	One between `U`, `C`, `V`, `I`. The content uploaded status server side

5.65 Family

Name	Mandatory	Type	Value
name	Y	String	The name of the family
id	N	Number	The internal ID of the family
externalid	N	String	External ID (i.e. defined in the environment of the caller) of the family

Name	Mandatory	Type	Value
users	N	JSON Array	The users in the family – mandatory adding a family. See Section 5.66, “Family user”

5.66 Family user

Name	Mandatory	Type	Value
userid	Y	String	The user ID of the user
firstname	N	String	The first name of the given user (available only for user realm)
lastname	N	String	The last name of the given user (available only for user realm)
phonenumber	N	String	The phone number of the given user (available only for user realm)
emailaddress	N	String	The email address of the given user (available only for user realm)

Chapter 6. Tips and tools

This section includes examples on how to make a simple call to OneMediaHub Server API from an external application. Examples of URL encoding and examples in Java, JavaScript and PHP are listed below for quick reference. A review of useful tools for debugging and testing purposes is included as well.

6.1 Format of the request

The URL of every Server API request indicates the protocol, the server address, Server API path (`sapi` by default) and the component it refers to. Some of the possible components are *calendar*, *picture*, *system*, etc.

```
<protocol>://<address:port>/<sapi location>/<component>
```

6.2 Response and XSS vulnerabilities

To minimize the risk of cross-site scripting (XSS) vulnerabilities, is up to the Server API consumer to escape values returned in the JSON response before using them to be rendered in a browser.

6.3 Session handling

6.3.1 Unauthenticated requests

These are the type of calls that can be made under the **public** realm. Below is an example:

Request:

```
GET https://my.server.com/sapi/system/carrier?action=get&countryid=PT
```

Response:

```
{
   "data":{
      "carriers":[
         {
            "id":2,
            "name":"Optimus",
            "countryid":"12",
            "active":true,
            "otasupport":true,
            "trustedjam":true,
            "syncmlsupport":true
         },
         …
      ]
   }
}
```

6.3.2 Authenticated requests

In order to authenticate a user and have a valid session, you need to perform the login request. Below is an example:

Request:

```
POST https://my.server.com/sapi/login?action=login
```

Request body:

```
login=username&password=<password>
```

Response:

```
{
   "data":{
      "jsessionid":"EE2003AE8BEED4E2342B8C732FB30450",
      "roles":[
         {
            "name":"standard",
            "description":""
         },
         {
            "name":"sync_administrator",
            "description":""
         }
      ]
   }
}
```

As shown, the session ID needed for subsequent authenticated calls is returned in the JSON body, in addition to being returned as a cookie. The `jsessionid` in the JSON body can be used for example in client platforms that do not support cookies.

In summary, session tracking is performed using the `jsessionid` cookie or alternatively specifying the `jsessionid` value in the URL; for example:

```
https://my.server.com/sapi/ota;jsessionid=1235435324?param1=value1
```

6.3.3 Retrieving user data without a session ID

There are two ways of accessing user data without having a valid session ID for the user:

1. sending the user's login information in every request. This approach is used when you do not want to handle session ID management or when you want to retrieve the data in one request. Example:

   Request:

   ```
   https://<address:port>/sapi/profile?action=get
   ```

   Headers:

   ```
   Authorization: Basic dXNlcm5hbWU6cGFzc3dvcmQ=
   ```

2. adding the username by means of the *`userid`* parameter and using the administrator's session ID. This request is used when the call has to be performed by an administrator, e.g. for support. Example:

   Request:

   ```
   https://<address:port>/sapi/profile?action=get
   ```

Headers:

```
Authorization: Basic YWRtaW46cGFzc3dvcmQ=
```

Request body:

```
userid=username
```

By using either way of accessing user data, the result is the same as if the user were logged in with a valid session ID.

6.4 My first API call: examples

This section presents some basic examples in Java, JavaScript and PHP on how to perform a simple request and retrieve contacts for a registered user.

If you are looking for information on how to parse JSON data in other programing languages, refer to [1].

6.4.1 Java

Below is an example of how to retrieve a user's contacts using Java:

```
HttpClient client = new HttpClient();

PostMethod post = new PostMethod("https://my.server.com/sapi/login?
action=login");

GetMethod get = new GetMethod("https://my.server.com/sapi/contact?
action=get");

try {
     //login in
     NameValuePair[] namevalue = new NameValuePair[]{
     new NameValuePair("login", "username"),
     new NameValuePair("password", "password”)};
     post.setRequestBody(namevalue);
     int statusCode = client.executeMethod(post);
     byte[] responseBody = post.getResponseBody();
     System.out.println(" login:"+new String(responseBody));

     //counting the contacts on server
     statusCode = client.executeMethod(get);
     String responseBodyStr = get.getResponseBodyAsString();
     JSONObject obj = JSONObject.fromObject(responseBodyStr);
     System.out.println("Count " + obj.getString("count")+" Contacts.");
} catch (HttpException e) {
     e.printStackTrace();
} catch (IOException e) {
     e.printStackTrace();
} finally {
     post.releaseConnection();
     get.releaseConnection();
}
```

To add a picture using multipart post:

```
PostMethod filePost = new PostMethod(<server address>);

        String metadata = "{\"data\":{"
                + "\"name\":\"file.jpg\","
                + "\"creationdate\":20110209T221236Z,"
                + "\"modificationdate\":20110209T221236Z,"
                + "\"contenttype\":\"image/jpg\","
                + "\"size\": 50000 "}}";

        File file = new File("c:\file.jpg");

        Part[] parts = {
            new StringPart("data", metadata),
            new FilePart("file", file)
        };

        filePost.setRequestEntity(
                new MultipartRequestEntity(parts,
                filePost.getParams()));

        HttpClient client = new HttpClient();

        int status = client.executeMethod(filePost);

        System.out.println(filePost.getResponseBodyAsString());
```

To log in:

```
HttpClient client = new HttpClient();

PostMethod post = new PostMethod("http:/my.server.com/sapi/login?
action=login");

String username="username";
String password="password";

String credentials = username.concat(":").concat(password);
byte[] encodedCredentials = Base64.encode(credentials.getBytes());

try {
    Header header = new Header();
    header.setName("Authorization");
    header.setValue("Basic " + new String(encodedCredentials));
    post.setRequestHeader(header);

    //log in
    int statusCode = client.executeMethod(post);
    byte[] responseBody = post.getResponseBody();
    System.out.println(" login:"+new String(responseBody));

} catch (HttpException e) {
    e.printStackTrace();
```

```
} catch (IOException e) {
    e.printStackTrace();
} finally {
    post.releaseConnection();
}
```

The best way to access the Server API using a Java standalone applications is using Apache HttpClient (see [6]); this example uses v3.1.

In order to parse the JSON response, the library used is `json-lib` (see [7]).

6.4.2 JavaScript

Below is an example of a similar call performed in an AJAX UI using JavaScript.

Note

Due to cross domain security limitations, an *XMLHTTPRequest* can only be performed to a server inside the same domain. See [8] for more info on JSON parsing.

```
<script type="text/javascript">
    var http = false;
    var params = "login=username&password=password";

    if (navigator.appName == "Microsoft Internet Explorer") {
        http = new ActiveXObject("Microsoft.XMLHTTP");
    } else {
        http = new XMLHttpRequest();
    }
    function handle() {
        if (http.readyState == 4) {
            document.getElementById('response').innerHTML ="response:"+
 http.responseText;
            countcontacts();
        }
    }
    function login() {
        http.onreadystatechange=handle;
        http.setRequestHeader("Content-type", "application/x-www-form-
urlencoded");
        http.setRequestHeader("Content-length", params.length);
        http.setRequestHeader("Connection", "close");
        http.open("POST", "/sapi/login?action=login", true);
        http.send(params);
    }
    function countcontacts() {
        http.onreadystatechange = function(){
            if (http.readyState == 4 && http.status == 200) {
                the_object = eval('(' + http.responseText + ')')
                document.getElementById('response').innerHTML = "Found "
 + the_object.count + " contacts.";
            }
        }
```

```
        http.open("POST", "/sapi/contact?action=count", true);
        http.send(null);
    }
</script>
<p><a href="javascript:login()">execute</a></p>

------------------------

<div id="response">
    -response-
</div>
```

6.4.3 PHP

The same example on how to retrieve the contacts of a user is now provided using PHP:

```
<?php
       $username = "username";
       $password = "password";

       $curl = curl_init();
       curl_setopt($curl, CURLOPT_HEADER, false);
       curl_setopt($curl, CURLOPT_RETURNTRANSFER, true);
       curl_setopt($curl, CURLOPT_POST, true);
       curl_setopt($curl, CURLOPT_URL, "https://my.server.com/sapi/
login?action=login");
       curl_setopt($curl, CURLOPT_POSTFIELDS, "login=$username&password=
$password");
       $result = curl_exec($curl);
       // now parse JSON without any libraries. Any JSON parser can be
 used instead
       if (preg_match("/\"jsessionid\"\:\"([^\"]*)\"/", $result,
 $matches)) {
              $jsession_id = $matches[1];
              curl_setopt($curl, CURLOPT_COOKIE, "JSESSIONID=
$jsession_id;COOKIE_SUPPORT=true");
              curl_setopt($curl, CURLOPT_URL, "https://my.server.com/
sapi/contact?action=count");
              $result = curl_exec($curl);
              if (preg_match("/\"count\"\:(\d*)/", $result, $matches))
 {
                  echo "Found ".$matches[1]." contacts.<br>\n";
              }
              curl_setopt($curl, CURLOPT_URL, "https://my.server.com/
sapi/contact?action=get");
              $result = curl_exec($curl);
              echo "JSON response, to be parsed:<br>";
              echo "<pre>";
              echo $result;
              echo "</pre>";

       } else {
              echo "SESSION ID NOT FOUND";
       }
```

```
        curl_close($curl);
?>
```

6.5 Tools for debugging and testing

In order to debug the Server API protocol and verify that the client-side implementation is working correctly, there are a few tools that can be used. Many of these tools are Mozilla Firefox and/or Google Chrome extensions and can also be used to see how the OneMediaHub AJAX Portal performs Server API calls.

6.5.1 Tools for Mozilla Firefox and/or Google Chrome

Firebug and HTTPFox are two Firefox extensions that can help developing Server API applications; both extensions can be installed as regular Firefox extensions.

1. **Firebug** (https://addons.mozilla.org/en-US/firefox/addon/1843) is a Firefox extension used to debug a web page, and is particularly useful for debugging *XMLHTTPRequests* (i.e. AJAX requests.) When developing a web client to the Server API, Firebug will allow to keep track of all the XHR requests made to the server, and for each request it will track parameters, HTTP headers, the response, along with other useful data that can help in tracking issues during web application development. A version for Google Chrome is available under http://getfirebug.com/wiki/index.php/Chromebug.

2. While Firebug is more suitable for *XMLHTTPRequests*, **HTTPFox** (https://addons.mozilla.org/en-US/firefox/addon/6647) is much better for handling other HTTP calls such as for example picture downloads (contact and picture module.)

3. **LiveHTTPHeaders** (https://addons.mozilla.org/en-US/firefox/addon/3829) is useful for debugging headers and, in particular, to analyze cookies sent by a remote site.

4. **Poster** (https://addons.mozilla.org/en-US/firefox/addon/2691) is a development tool used to interact with web services and other web resources; it lets you send HTTP requests, and set the entity's body and content type, allowing to interact with web services and inspect the results. It is the easiest way to perform an example of Server API POST call. A version for Google Chrome is available under http://code.google.com/p/chrome-poster.

6.5.2 Tools for JSON

Many libraries are available online to parse a JSON file written in Java or any other language. The OneMediaHub Server API relies on the `json-lib` library (see [7]). More libraries can be found at htt://www.json.org. To validate a JSON object pasting the content directly online, you can use http://www.jsonlint.com.

Appendix A. API List

The following tables provide a quick overview of the available API calls grouped by functionality.

Definition	Security Realm	Description
Login		
`POST /sapi/login?action=login`	PUBLIC	Log the user into the system
`POST /sapi/login/oauth?action=login`	PUBLIC	Log in to a OneMediaHub server using an appropriate OAuth key
`GET /sapi/login?action=logout`	PUBLIC	Log out from the system

Definition	Security Realm	Description
Profile		
`GET /sapi/profile/changes?action=get`	USER	Retrieve all changes occurred to PIM data and media since the given from date
`POST /sapi/profile/ota?action=send-configuration`	USER	Send the OTA configuration message to a specific device, basing on its characteristics with the standard portal configuration
`POST /sapi/profile/ota?action=send-generic-configuration`	SYSTEM	Send the OTA configuration message specifying the recipient and all the configuration parameters
`POST /sapi/profile/ota?action=download`	USER	Send a download SMS to a specific device
`POST /sapi/profile/download?action=send-download-link`	PUBLIC	Send the link of the download page via SMS as specified in the server configuration
`POST /sapi/profile/generic?action=send-download-link`	USER	Send an Email to the provided address with the link to download the mobile app, according to the type of device of the user
`GET /sapi/profile/download?action=get-phone-url`	SYSTEM	Retrieve the download URL for the OneMediaHub App

Definition	Security Realm	Description
`GET /sapi/profile/download?action=get-windows-url`	PUBLIC	Retrieve the download URL for the OneMediaHub App for Windows
`GET /sapi/profile/download?action=get-macos-url`	PUBLIC	Retrieve the download URL for the OneMediaHub App for Mac OS
`GET /sapi/profile/download?action=get-device-url`	USER	Retrieve the download URL for the App
`GET /sapi/profile?action=get`	USER	Retrieve profile information for a user
`GET /sapi/profile/generic?action=get`	USER	Retrieve all generic information (first name, last name, ...) for a user
`GET /sapi/profile?action=search`	SYSTEM	Find user profiles that exactly match all the search conditions
`POST /sapi/profile/generic/password?action=reset &sendby=[email|sms]`	PUBLIC	Send the URL that will allow the user to reset his password given an existing profile email, username or MSISDN
`POST /sapi/profile/generic/password?action=save`	PUBLIC	Reset the password given a valid reset token and a valid new password
`GET /sapi/profile/device?action=get`	USER	Retrieve all the device information (devices, model, manufacturer, ...) for a given user
`POST /sapi/profile?action=signup`	PUBLIC	Allow a guest who provides a correct validation token to add a new user and a device
`POST /sapi/mobile?action=signup`	PUBLIC	Allow a guest user who provides a correct validation token to add a new user and a device
`POST /sapi/profile?action=validate`	PUBLIC	Allow a guest user to validate the specified credentials (user ID if specified, phone number and password) before signup
`POST /sapi/mobile?action=welcome`	PUBLIC	Allow a guest user who opens the welcome page

Definition	Security Realm	Description
		from their mobile device browser to download the correct corresponding OneMediaHub App for the automatically detected device
`POST /sapi/profile/generic?action=add`	SYSTEM	Add a new user
`POST /sapi/profile/generic?action=update`	USER	Update an existing user
`POST /sapi/profile/generic?action=delete`	USER	Delete an existing user
`POST /sapi/profile/photo?action=photosave`	USER	Insert or update a user's profile picture
`POST /sapi/profile/photo?action=photodelete`	USER	Delete the user's profile picture
`GET /sapi/profile/photo?action=photodownload`	USER	Return the photo as it is stored on the server
`POST /sapi/profile/device?action=add`	USER	Add a new device for a given user
`POST /sapi/profile/device?action=update`	USER	Update an existing device
`POST /sapi/profile/device?action=delete`	USER	Delete a device given the device ID
`GET /sapi/profile?action=get-last-sync`	USER	Retrieve the last sync time for all devices associated to the given user
`POST /sapi/profile/push?action=send`	SYSTEM	Push a notification message to all the devices of the given user or to the specified device only
`GET /sapi/profile/client?action=get-update-info`	PUBLIC	Retrieve all the information needed by the OneMediaHub App to auto-update
`POST /sapi/profile/comment?action=add`	SYSTEM	Add a new comment providing information about a given user's customer care call
`GET /sapi/profile/comment?action=get`	SYSTEM	Retrieve comments about a user's customer care call
`POST /sapi/profile/device/token?action=save`	USER	Register a token of a cloud messaging service (like Apple's remote push

Definition	Security Realm	Description
		notification service) for a given device
`POST /sapi/profile/push?action=getsan`	SYSTEM	Return an array of SAN messages

Definition	Security Realm	Description
Roles management		
`GET /sapi/profile/role?action=roles`	SYSTEM	Retrieve the list of all the available roles
`GET /sapi/profile/role?action=get`	USER	Retrieve the list of the user's roles
`POST /sapi/profile/role?action=save`	SYSTEM	Set the roles for an existing user

Definition	Security Realm	Description
Handling properties		
`POST /sapi/profile/properties?action=get`	USER	Return all the key-value pairs stored for this user
`POST /sapi/profile/properties?action=set`	USER	Add/update the key-value pairs for this user
`POST /sapi/profile/properties?action=remove`	USER	Remove a key-value pair for the specified user
`POST /sapi/profile/properties?action=removeall`	USER	Remove all key-value pairs for the specified user

Definition	Security Realm	Description
Subscription plans		
`GET /sapi/subscription/plan?action=get`	USER	Return the list of subscription plans available for the user that belong to the user's current subscription family
`POST /sapi/subscription/families?action=get`	USER	Return complete information about the subscription families defined in the system, with details about the subscription plans belonging to them

Definition	Security Realm	Description
`POST /sapi/subscription?action=save`	USER	Change the subscription plan or the status of the current plan of the user
`GET /sapi/subscription?action=get`	USER	Return information about the status of the currently active subscription plan
`POST /sapi/subscription?action=cancel`	USER	Cancel the current subscription plan of the user
`GET /sapi/subscription/history?action=get`	USER	Return information about the history of the subscriptions for the user
`POST /sapi/subscription/payment?action=save`	USER	Allow clients - such as an iPhone - to register payments handled by the client itself
`GET /sapi/subscription/payment?action=get`	USER	Return the list of registered payments for the user

Definition	Security Realm	Description
Media		
`GET /sapi/media?action=get-storage-space`	USER	Return the free storage space, used soft-deleted space, and storage space available quota for the user
`GET /sapi/media/picture?action=get`	USER	Retrieve pictures for the user
`POST /sapi/media/picture?action=delete`	USER	Delete pictures for the user
`POST /sapi/media/picture?action=reset`	USER	Delete all the pictures stored on the media storage provider
`GET /sapi/media/picture?action=count`	USER	Return the number of pictures for the user
`POST /sapi/media/picture/favorite?action=add`	USER	Add a picture or a list of pictures to the list of favorite pictures
`POST /sapi/media/picture/favorite?action=remove`	USER	Remove a picture or a list of pictures from the list of favorite pictures
`GET /sapi/media/video?action=get`	USER	Retrieve videos for the user

Definition	Security Realm	Description
POST /sapi/media/video?action=delete	USER	Delete videos for the user
POST /sapi/media/video?action=reset	USER	Delete all the videos stored on the media storage provider
GET /sapi/media/video?action=count	USER	Return the number of videos for the user
POST sapi/media/video?action=set-transcoding-status	PUBLIC	Update the transcoding job status
GET /sapi/media/file?action=get	USER	Retrieve files for the user
POST /sapi/media/file?action=delete	USER	Delete files for the user
POST /sapi/media/file?action=reset	USER	Delete all the files stored on the media storage provider
GET /sapi/media/file?action=count	USER	Return the number of files for the user
GET /sapi/media/audio?action=get	USER	Retrieve audio data for the user
POST /sapi/media/audio?action=delete	USER	Delete audio files for the user
POST /sapi/media/audio?action=reset	USER	Delete all the audio items stored on the media storage provider
GET /sapi/media/audio?action=count	USER	Return the number of audio items for the user
POST /sapi/media/set?action=save	USER	Create a media set, which is a bundle of media items having a unique URL
GET /sapi/media/set?action=get	PUBLIC	Return a previously created media set
POST /sapi/media?action=remove-from-sets	USER	Remove media items from all the media sets they are contained in
POST /sapi/media/folder?action=save	USER	Create a folder or update it, in case the folder already exists
GET /sapi/media/folder?action=get	USER	Allow the user to retrieve one or more folders
POST /sapi/media/folder?action=add-item	USER	Add one or more media items to a given folder
POST /sapi/media/folder?action=remove-item	USER	Allow the user to remove media from an existing folder

Definition	Security Realm	Description
POST /sapi/media/folder?action=delete	USER	Let a user delete an existing folder
POST /sapi/media/folder?action=reset	USER	Delete all the folders of the user
POST /sapi/label?action=save	USER	Allow the user to create a label (also empty) with a given name, or to modify an existing one by renaming it or adding a group of items
POST /sapi/label?action=add-item	USER	Allow the user to mark one or more media items with an existing label
POST /sapi/label?action=remove-item	USER	Allow the user to remove a media item from an existing label
POST /sapi/label?action=delete	USER	Allow the user to delete an existing label
POST /sapi/label?action=get	USER	Allow the user to retrieve one or more existing labels
POST /sapi/media?action=change-validation-status	SYSTEM	Allow the administrator to change the validation status of an item
POST /sapi/media?action=emptytrash	USER	Delete all soft deleted items for the user, regardless of their media type
POST /sapi/media/picture?action=emptytrash	USER	Delete all soft deleted pictures for the user
POST /sapi/media/video?action=emptytrash	USER	Delete all soft deleted videos for the user
POST /sapi/media/file?action=emptytrash	USER	Delete all soft deleted files for the user
POST /sapi/media/audio?action=emptytrash	USER	Delete all soft deleted audio items for the user
POST /sapi/media?action=restore	USER	Restore soft deleted items for the user, regardless of their media type
POST /sapi/media/picture?action=restore	USER	Restore soft deleted pictures for the user
POST /sapi/media/video?action=restore	USER	Restore soft deleted videos for the user

Definition	Security Realm	Description
POST /sapi/media/file?action=restore	USER	Restore soft deleted files for the user
POST /sapi/media/audio?action=restore	USER	Restore soft deleted audio items for the user
GET /sapi/media/timeline?action=get	USER	Retrieve a list of JSON objects containing, for each period, the media IDs present on it, ordered by descending date
POST /sapi/media?action=get	USER	Retrieve media or family items information for the user, ordered by descending date
POST /sapi/media?action=import	USER	Allow the user to import into her Digital Life given media content already shared in the family of the user

Definition	Security Realm	Description
Upload binary files		
POST /sapi/upload?action=save	USER	Allow a direct upload in a single request (using multipart) of the binary content and the information about the file, as size, encoding, name, etc
POST /sapi/upload/picture?action=save-metadata	USER	Perform a resumable upload
POST /sapi/upload/file?action=save	USER	Allow a direct upload in a single request using multipart of the binary content and the information of the file, as size, encoding, name, etc
POST /sapi/upload/file?action=save-metadata	USER	Update the binary content or the associated metadata of an exiting media item

Definition	Security Realm	Description
External services		
GET /sapi/externalservice?action=get	USER	Return a list of services to which it is possible to

Definition	Security Realm	Description
		export videos and picture media files
`GET /sapi/externalservice/album?action=get-albums&servicename=<service name>`	USER	Return a list of albums for a specific service
`POST /sapi/externalservice/album?action=create-album`	USER	Create an album in a specific service
`POST /sapi/media/picture?action=export`	USER	Export a media item (picture or video) to a given album
`POST /sapi/externalservice/facebook/friend?action=get`	USER	Update the contacts of the user by adding pictures of Facebook friends
`POST /sapi/externalservice/authorization?action=save`	USER	Request the authorization token for a given service and user ID, and save or update it directly on the server

Definition	Security Realm	Description
Handle authorization		
`GET/POST /sapi/externalservice/<servicename>`	USER	Handle the authorization tokens delivered by external services
`GET/POST /sapi/externalservice/<servicename>?action=revoke`	USER	Handle the authorization tokens delivered by external services when the authorization has to be revoked

Definition	Security Realm	Description
Contacts		
`POST /sapi/contact?action=get`	USER	Retrieve all available contacts (or a single contact) for a user
`POST /sapi/contact?action=contactsave`	USER	Add a new contact or update an existing contact for a user, given the standard information of a vCard
`POST /sapi/contact?action=contactsdelete`	USER	Delete a contact from the user's contact list
`POST /sapi/contact?action=reset`	USER	Reset all contacts for the user

Definition	Security Realm	Description
`POST /sapi/contact?action=photosave`	USER	Add a new picture for the selected contact
`POST /sapi/contact?action=photodelete`	USER	Delete a list of contact pictures
`GET /sapi/contact?action=count`	USER	Return the number of contacts for the user
`GET /sapi/contact?action=photodownload&id=<contactid>`	USER	Retrieve the contact's picture as it is stored on the server
`POST /sapi/contact?action=contactgetdeleted`	USER	Return the list of last dates when at least one contact was deleted
`POST /sapi/contact?action=contactrestore`	USER	Restore contacts that were deleted on the specified date
`GET /sapi/contact?action=export`	USER	Export contacts in comma-separated values (CSV) format compatible with Microsoft Outlook
`POST /sapi/contact/import/service?action=import`	USER	Import contacts from the service account set in the request URI

Definition	Security Realm	Description
Calendar		
`POST /sapi/calendar?action=get`	USER	Retrieve all events for a user in a defined time frame
`POST /sapi/calendar?action=eventcreate`	USER	Add a new event to the user's default calendar
`POST /sapi/calendar?action=eventmodify`	USER	Update an event in the user's default calendar
`POST /sapi/calendar?action=eventmodifyone`	USER	Update an instance of a recurring event in the user's default calendar
`POST /sapi/calendar?action=eventdelete`	USER	Delete an event in the user's default calendar
`POST /sapi/calendar?action=eventdeleteone`	USER	Delete an instance of a recurring event in the user's default calendar
`POST /sapi/calendar?action=busydatesget`	USER	Return all the dates that have at least an event for the user in the indicated date range

Definition	Security Realm	Description
POST /sapi/calendar?action=reset	USER	Reset all events and tasks for the user
GET /sapi/calendar?action=eventgetdeleted	USER	Return the list of last dates when at least one event was deleted
POST /sapi/calendar?action=eventrestore	USER	Restores events which were deleted on the specified date
POST /sapi/calendar/import/service?action=import	USER	Import calendar from the service account set in the request URI

Definition	Security Realm	Description
System		
GET /sapi/system/manufacturer?action=get	PUBLIC	Retrieve all available phone manufacturers
GET /sapi/system/model?action=get	PUBLIC	Retrieve all phone models by the specific manufacturer ID
GET /sapi/system/carrier?action=get	PUBLIC	Retrieve all available carriers worldwide, the carriers for a specific country, or a specific carrier
GET /sapi/system/model?action=get	PUBLIC	Retrieve all phone models by the specific manufacturer ID
GET /sapi/system/timezone?action=get	PUBLIC	Retrieve all available timezones
GET /sapi/system/country?action=get	PUBLIC	Retrieve all available countries
GET /sapi/system/model?action=get-url	PUBLIC	Retrieve the URL of the picture, the URL of the manual configuration instructions, and the URL of the synchronization instructions of a given device
GET /sapi/system/location?action=get	PUBLIC	Translate an IP into a country location
GET /sapi/system/captcha?action=get-url	PUBLIC	Return the URL of an image that represents a token required to validate the session

Definition	Security Realm	Description
`GET /sapi/system/information?action=get`	PUBLIC	Retrieve information about the OneMediaHub server
`GET /sapi/system/media?action=get-used-storage`	SYSTEM	Return the overall storage usage

Definition	Security Realm	Description
SMS		
`POST /sapi/sms?action=send-text`	SYSTEM	Send a generic text SMS to a specific device
`POST /sapi/sms?action=send-activation-link`	SYSTEM	Send the user activation link via SMS to a specific device

Definition	Security Realm	Description
Last activities		
`GET /sapi/activity?action=get`	USER	Return a list containing the last activity per every data source for a given device ID, if specified
`POST /sapi/activity?action=save`	USER	Save last activities for a given device

Definition	Security Realm	Description
EMAIL		
`POST /sapi/email?action=send`	SYSTEM	Send an email message to a specific user or email address of a user
`POST /sapi/email?action=send-activation-link`	SYSTEM	Send an email message with the activation link to a specific user email address

Definition	Security Realm	Description
Family		
`POST /sapi/family?action=save`	SYSTEM	Create a new family of users, or rename an existing one if the ID is provided within the JSON object carried with the request

Definition	Security Realm	Description
`POST /sapi/family?action=delete`	SYSTEM	Delete an existing family
`POST /sapi/family?action=get`	USER	Retrieve a set of families with all the information related to the families inside it
`POST /sapi/family/user?action=add`	SYSTEM	Add a user to an existing family
`POST /sapi/family/user?action=remove`	SYSTEM	Remove a user from an existing family
`POST /sapi/family/media?action=share`	USER	Allows a family member to share some media content on her family hub
`POST /sapi/family/media?action=unshare`	USER	Allows a family member to unshare some media content from her family hub

Appendix B. Deprecated API

This appendix describes the deprecated OneMediaHub Server API.

B.1 Media

B.1.1 Rotate thumbnails of a picture

Rotate the thumbnails of a picture. If the picture has no thumbnails or thumbnails have been disabled, a smaller rotated copy of the picture (called 'view') is provided.

Note

The picture itself is not rotated and the changes will not be propagated when performing a synchronization.

Status: deprecated since v14

Security realm: user

Definition

```
POST /sapi/media/picture?action=rotate-view
```

Request body

- *`data`*: the JSON object containing the ID of the picture to be updated and the rotation amount in degrees. Valid values for the rotation amount are: `90`, `180`, `270`.

Example:

```
{
    "data":{
        "id":"23508",
        "degrees": 90
    }
}
```

Response

A success response is always returned.

Errors

PIC-1004, in addition to generic errors described in Section 4.1.3, “Errors across multiple Server APIs”.

Appendix C. Acronyms

Below is a list of commonly used acronyms found in this guide:

AJAX	Asynchronous JavaScript And XML
API	Application Programming Interface
CSRF	Cross Site Request Forgery
HTTP	HyperText Transfer Protocol
IMAP	Internet Message Access Protocol
IP	Internet Protocol
JSON	JavaScript Object Notation
LUID	Local Unique Identifier: identifier assigned by the client to an item stored on it and unique on the client.
OTA	Over The Air
PDA	Personal Digital Assistant
PIM	Personal Information Management
POP	Post Office Protocol
REST	REpresentational State Transfer
SMS	Short Message Service
SMTP	Simple Mail Transfer Protocol
URI	Uniform Resource Identifier
URL	Uniform Resource Locator
XHR	XMLHttpRequest (W3C)

Appendix D. Media Formats

Supported formats for Pictures, Videos, and Files are:

Media type	Extensions of the supported formats
Pictures	GIF, JFIF, JIF, JPE, JPEG, JPG, PNG
Videos	3G2, 3GP, 3GPP, ASF, AVI, FLV, M4U, M4V, MOV, MOVIE, MP2, MP4, MPA, MPE, MPEG, MPG, WMV
Audio	AAC, M4A, MP3
Files	All what is neither Pictures, nor Videos, nor Audio

Appendix E. Outlook CSV format and Funambol JSON equivalent

Outlook	Funambol JSON equivalent
Account	-
Anniversary	anniversary
Assistant's Name	org[0].assistant
Assistant's Phone	-
Billing Information	-
Birthday	birthday
Business Address	-
Business Address PO Box	-
Business City	address[t=work].city
Business Country	address[t=work].country
Business Fax	phone[t=workfax].v
Business Phone	phone[t=work].v
Business Phone 2	phone[t=work2].v
Business Postal Code	address[t=work].zip
Business State	address[t=work].region
Business Street	address[t=work].street_address
Business Street 2	-
Business Street 3	-
Callback	-
Car Phone	-
Categories	categories
Children	children
Company	org[0].org
Company Main Phone	phone[t=company].v
Department	org[0].department
Directory Server	-
E-mail 2 Address	email[t=home].v
E-mail 3 Address	email[t=work].v
E-mail Address	email[t=main].v
First Name	firstname
Gender	gender
Hobby	hobbies
Home Address	-
Home Address PO Box	-

Outlook	Funambol JSON equivalent
Home City	address[t=home].city
Home Country	address[t=home].country
Home Fax	phone[t=homefax].v
Home Phone	phone[t=home].v
Home Phone 2	-
Home Postal Code	address[t=home].zip
Home State	address[t=home].region
Home Street	address[t=home].street_address
Home Street 2	-
Home Street 3	-
ISDN	-
Initials	initials
Internet Free Busy	-
Job Title	org[0].title
Keywords	-
Language	languages
Last Name	lastname
Location	-
Manager's Name	org[0].manager
Middle Name	middlename
Mileage	mileage
Mobile Phone	phone[t=mobile].v
Notes	notes
Office Location	org[0].officelocation
Organizational ID Number	-
Other Address	-
Other Address PO Box	-
Other City	address[t=other].city
Other Country	address[t=other].country
Other Fax	phone[t=otherfax].v
Other Phone	phone[t=other].v
Other Postal Code	address[t=other].zip
Other State	address[t=other].region
Other Street	address[t=other].street_address
Other Street 2	-
Other Street 3	-
Pager	phone[t=pager].v
Primary Phone	phone[t=other2].v

Outlook	Funambol JSON equivalent
Priority	importance
Private	-
Profession	org[0].role
Radio Phone	-
Referred By	-
Sensitivity	sensitivity
Spouse	spouse
Suffix	suffix
TTY/TDD Phone	-
Telex	-
Title	prefix
User 1	-
User 2	-
User 3	-
User 4	-
Web Page	webpage[t=main].v

References

[1] *Introducing JSON*. http://www.json.org.

[2] *OAuth 2.0*. http://oauth.net/2/.

[3] *RFC 4627 - The application/json Media Type for JavaScript Object Notation (JSON)*. http://www.ietf.org/rfc/rfc4627.txt.

[4] *Representational State Transfer*. http://en.wikipedia.org/wiki/Representational_State_Transfer.

[5] *OMA Provisioning Content*. http://www.openmobilealliance.org/Technical/release_program/docs/ClientProv/V1_1-20090728-A/OMA-WAP-TS-ProvCont-V1_1-20090728-A.pdf.

[6] *Apache HttpClient*. http://hc.apache.org.

[7] *Json-lib*. http://json-lib.sourceforge.net.

[8] *JSON parsing*. http://www.json.org/js.html.

[9] *OneMediaHub Installation and Operation Guide*. OneMediaHub Version 14.5 Installation and Operation Guide.

[10] *RFC 2617 - HTTP Authentication: Basic and Digest Access Authentication*. http://www.ietf.org/rfc/rfc2617.txt.

Colophon

This book is written in DocBook XML, version 5 of the RELAX NG scheme. The XSL-FO and HTML files are generated using xsltproc (compiled against `libxml` version 20632, `libxslt` version 10124, and `libexslt` version 813) and two stylesheet customization layers. The PDF file was generated using Apache FOP version 1.0. The validation of the XML source code (based on XML Inclusions) was accomplished using xmllint (based on `libxml` version 20632) and Jing version 20091111.

In the printed version, the book uses Times as the body font, Helvetica as the title font, and Courier as the monospace font.

The size of the XML source code of the whole book is 894 KB. Pictures are inserted in PNG and JPEG format. The five most frequent elements in this book are: `entry`, `para`, `title`, `row`, and `listitem`.