

OpenDrive REST API Guide

Version 1.1.7

Contents

Contents.....	2
API Partner Roles.....	2
HTTP Verbs.....	2
1. Account Users API:	3
2. Branding:	10
3. Download	15
4. File.....	16
5. Folders API.....	28
6. Notes	43
7. Secure folders.....	55
8. Session API:	57
9. Sharing.....	60
10. Stats	64
11. Tasks:	66
12. Upload API:	92
13. Users	98
14. User Groups API:.....	104
15. OAuth2 authentication protocol.....	106

API Partner Roles

The API has the following roles:

- 1 - basic (partner can use API functions to manage his own account)
- 2 - manager (partner can create users and account users and use all functions to manage their accounts and files)

HTTP Verbs

The OpenDrive API uses appropriate HTTP verbs for each action.

Verb	Descriptions
GET	Used for retrieving resources.
POST	Used for creating resources.

PUT	Used for replacing resources or collections.
DELETE	Used for deleting resources.

API base URL: <https://dev.opendrive.com/api/v1>

Please look at our API Explorer <https://dev.opendrive.com/api/explorer/> to understand and test API calls.

1. Account Users API:

1.1 Folder list (folderslist.json)

Description: List folders or subfolders of an account user

Role: 1 and 2

URL Structure: `/accountusers/folderslist.json/{session_id}/{folder_id}`

Method: GET

Parameters:

session_id: string (required) - Session ID.

folder_id: string (required) - Folder ID (0 to list root folders).

access_email: string - Valid email format required (max 255).

Returns:

Folders:

```
{
  FolderID: string – Folder ID.
  Name: string – Folder name.
  Access: integer- Access rights, 1 – edit, 0 – view.
  ChildFoldersCount: integer - A number of folders in given folder.
  Link: string – the address of given folder.
}
```

Errors:

400: Bad Request: invalid value specified for `access_email`. Expecting email in `name@example.com` format.

404: Not Found: Directory doesn't exist.

1.2 Get an account user info (info.json)

Description: Get an account user info

Role: 1 and 2

URL Structure: `/accountusers/info.json/{session_id}/{access_email}`

Method: GET

Parameters:

session_id: string (required) - Session ID.

access_email: string (required) - Valid email format required (max 255).

Returns:

AccessUserID: string.
FirstName: string - The user's first name.
LastName: string - The user's last name.
Email: string - Account user email address.
AdminMode: integer – Account user admin mode (0 - users, 1 - admin, 2 - files).
AccessAdminLevel: integer – Admin rights level.
AccessActive: integer – Account user status (0 = inactive, 1=active, 2 - blocked).
AccessNotification: boolean – The option is activated by a user with admin rights on the page Users.
AccessPosition: string – Not required field. It is specified by a user with admin rights on the page Users.
AccessPasswordChange: Boolean — The option is activated by a user with admin rights on the page Users.
BwMax: integer – The amount of allowed bandwidth in bytes.
BwUsed: integer – The amount of used bandwidth for the day in bytes.
StorageMax: integer – The amount of allowed storage space in bytes.
StorageUsed: integer – The amount of used storage space in bytes.
Phone: string - Not required field. It is filled by a user on profile settings page
AccessSince: string – Unix timestamp.
AccessFolders: JSON Object – The list of folders a user has access to.
AccessProjects: JSON Object – The list of projects a user has access to.

Errors:

400: Bad Request: invalid value specified for `access_email`. Expecting email in `name@example.com` format.
404: Not Found: Account user not found.

1.3 Search an account users in group (searchuser.json)

Description: Search an account users in group

Role: 1 and 2

URL Structure: /accountusers/searchusers.json/{session_id}

Method: GET

Parameters:

session_id: string (required) - Session ID.
search_query: string - Search query.

Returns:

AccessUserID: string –Account user ID.
FirstName: string – Account user first name.
LastName: string - Account user last name.
Email: string - Account user email address.
AdminMode: integer – Account user admin mode (0 =users, 1=admin, 2=files).
AccessAdminLevel: integer – The level of admin rights.
AccessActive: integer – Account user status (0 = inactive, 1=active, 2 - blocked).
BwMax: integer – The amount of allowed bandwidth in bytes.
BwUsed: integer – The amount of used bandwidth for the day in bytes.
StorageMax: integer – The amount of allowed storage space in bytes.

StorageUsed: integer – The amount of used storage space in bytes.

AccessSince: string – Unix timestamp.

Position: string - not required field. It is set by user with admin rights on the page Users.

Phone: string - Not required field. It is filled by user on profile settings page.

Avatar: string – Path to avatar image.

AvatarColor: string – Background color of default userpic for users who have not set their own

avatars.

Errors:

404: Not Found: invalid permissions.

1.4 List an account users in group (usersingroup.json)

Description: List an account users in group.

Role: 1 and 2

URL Structure: /accountusers/usersingroup.json/{session_id}/{group_id}

Method: GET

Parameters:

session_id: string (required) - Session ID.

group_id: string (required) - Group id.

search_query: string - Search query.

Returns:

AccessUserID: string

FirstName: string - Account user first name.

LastName: string - Account user last name.

Email: string - Account user email address.

AdminMode: integer – Account user admin mode (0 =users, 1=admin, 2=files).

AccessAdminLevel: integer – the level of admin rights.

AccessActive: integer – Account user status (0 = inactive, 1=active, 2 - blocked).

BwMax: integer – The amount of allowed bandwidth in bytes.

BwUsed: integer – The amount of used bandwidth for the day in bytes.

StorageMax: integer – The amount of allowed storage space in bytes.

StorageUsed: integer – The amount of used storage space in bytes.

AccessSince: integer – Unix timestamp.

Position: string - Not required field. It is set by user with admin rights on the page Users.

Phone: string - Not required field. It is filled by user on profile settings page.

Avatar: string – Path to avatar image.

AvatarColor: string – Background color of default userpic for users who have not set their own avatars.

UserFolders: JSON Object – Returns the number of folders a user has an access to and access rights assigned for these folders.

AccessProjects: JSON Object – Returns the number of projects a user has an access to and access rights assigned for these projects.

AccessNotification: boolean – The option is activated by user with admin rights on the page Users.

Errors:

404: Not Found: Invalid permissions.

1.5 Create an account user (accountusers.json)

Description: Create new account user.

Role: 1 and 2

URL Structure: /accountusers.json

Method: POST

Parameters:

session_id: string (required) - Session ID.
access_first_name: string (required) - Account user first name (min 2)(max 50).
access_last_name: string (required) - Account user last name (min 2)(max 50).
access_password: string (required) - Account user password (min 5).
access_email: string (required) - Valid email format required (max 255).
access_admin_mode: integer (required) - Account user admin mode (0 =users, 1=admin, 2=files).
access_notification: integer (required) - Account user notification (0 = off, 1=on).
access_max_storage: integer (required) - Account user max storage size in MB.
access_bw_max: integer (required) - Account user max bandwidth size in MB.
group_id: string (required) - Group id.
access_phone: string – Phone.
access_password_change: integer - Allow user to change password (0 = off, 1=on).
access_position: string - Account user position.
access_send_password: string - Send password by email (0 = off, 1=on).

Returns:

True (200 OK) or error.

Errors:

400: Bad Request: Error parsing JSON, malformed JSON.
400: Bad Request: 'session_id' is required.
403: Forbidden: You have reached maximum amount of account users.
404: Not Found: invalid permissions

1.6 Move an account user to different group (move.json)

Description: Move an account user to another group.

Role: 1 and 2

URL Structure: /accountusers/move.json

Method: POST

Parameters:

session_id: string (required) - Session ID.
access_email: string (required) - Valid email format required (max 255).
group_id: string (required) – New group id.

Returns:

True (200 OK) or error

Errors:

400: Bad Request: `session_id` is required.

400: Bad Request: Error parsing JSON, malformed JSON.

400: Bad Request: Invalid value specified for `access_email`. Expecting email in `name@example.com` format.

404: Not Found: invalid permissions.

1.7 Block, unblock or remove an account user (setaccess.json)

Description: Block, unblock, or remove an account user.

Role: 1 and 2

URL Structure: /accountusers/setaccess.json

Method: POST

Parameters:

session_id: string (required) - Session ID.

access_email: string (required) - Valid email format required (max 255).

access_active: integer (required) - (0 = inactive, 1=active, 2 - blocked).

Returns:

True (200 OK) or error

Errors:

400: Bad Request: Error parsing JSON, malformed JSON.

400: Bad Request: Invalid value specified for `access_email`. Expecting email in `name@example.com` format.

404: Not Found: invalid permissions.

1.8 Set an account user folder access (setfolderaccess.json)

Description: Set the access permission to a folder for a user.

Role: 1 and 2

URL Structure: /accountusers/setfolderaccess.json

Method: POST

Parameters:

session_id: string (required) - Session ID.

access_email: string (required) - Valid email format required (max 255).

foldersObj: Object (required) - Object of root folders and access to them.

required format:

\$foldersObj= array ('folder_id' => access_mode, 'folder_id' => access_mode);

access_mode: integer - (0 = view, 1 = edit, 2 = cancel - disable access)

Returns:

True (200 OK) or error

Errors:

400: Bad Request: invalid value specified for `access_email`. Expecting email in `name@example.com` format.

400: Bad Request: invalid value specified for "foldersArray".

404: Not Found: invalid permissions.

1.9 Update an account user (accountusers.json)

Description: Update and account users profile.

Role: 1 and 2

URL Structure: /accountusers.json

Method: PUT

Parameters:

session_id: string (required) - Session ID.

access_email: string (required) - Valid email format required (max 255).

new_access_email: string (required) - New valid email format required(max 255).

access_first_name: string - Account user first name (min 2)(max 50).

access_last_name: string - Account user last name (min 2)(max 50).

access_password: string - Account user password (min 5).

access_admin_mode: integer (required) - Account user admin mode (0 =users, 1=admin, 2=files).

access_notification: integer - Account user notification (0 = off, 1=on).

access_max_storage: integer - Account user max storage size in MB.

access_bw_max: integer - Account user max bandwidth size in MB.

access_phone: string - Account user phone.

access_position: string - Account user position.

access_password_change : int - Allow user to change password (0 = off, 1=on).

access_admin_level : int - Access admin level (Only for partner accounts).

Returns:

True (200 OK) or error

Errors:

400: Bad Request: `session_id` is required.

400: Bad Request: invalid value specified for `access_email`. Expecting email in `name@example.com` format.

404: Not Found: No user found.

1.10 Delete account (accountusers.json)

Description: Delete account user.

Role: 1 and 2

URL Structure: /accountusers.json/{session_id}/{access_email}

Method: DELETE.

Parameters:

session_id: string (required) - Session ID.

access_email: string (required) - Valid email format required (max 255).

Returns:

True (200 OK) or error.

Errors:

400: Bad Request: `session_id` is required.

400: Bad Request: invalid value specified for `access_email`. Expecting email in `name@example.com` format.

404: Not Found: No user found.

2. Branding:

2.1 Branding user information (branding.json)

Description: Get user branding info.

Role: 2

URL Structure: /branding.json

Method: GET

Parameters:

user_id: string (required) – User ID.

session_id: string (required) – Session ID.

Returns:

UserID: string.

HostMapping: string – Optionally specified by user.

DriveName: string - Optionally specified by user.

MenuColor: string — Specified by user of a default color is used.

FontColor: string - Specified by user or a default color is used.

LoginPageHtml:string – Not used in the new version.

WhiteLabeled: string - Not used in the new version.

FavIcon: image – Uploaded by user or a default image is used.

Subdomain: string - Optionally specified by user.

PartnerDomain: string - Optionally specified by user.

Logo: image - Uploaded by user or a default image is used.

Errors:

400: Bad Request: User not exists.

404: Not Found: Branding is not allowed for current user.

2.2 Check user subdomain exists (checkusersubdomainexists.json)

Description: Check if userdomain already registered by another user.

Role: 2

URL Structure: /branding/checkusersubdomainexists.json/{session_id}/{subdomain}

Method: GET

Parameters:

session_id: string (required) – Session ID.

subdomain: string (required) – Subdomain.

Returns:

True or False.

Errors:

400: Bad Request: User not exists.

404: Not Found: Missing file.

2.3 Branding files (files.json)

Description: Get branding files.

Role: 2

URL Structure: /branding/files.json

Method: GET

Parameters:

type: string (required) - Type('user','account_user','group','logo','favicon').

group_id: integer - Required for type group.

user_id: integer - User ID.

access_user_id: integer - Access User ID.

session_id: string - Session ID.

Returns:

File or Error.

Errors:

400: Bad Request: User required.

404: Not Found: Favicon is missing.

400: Bad Request: Access user required.

400: Bad Request: Invalid group ID.

404: Not Found: Missing file.

2.4 Subdomain branding (subdomainbranding.json)

Description: Get business user subdomain branding.

Role: 2

URL Structure: /branding/subdomainbranding.json/{partner_name}/{subdomain}

Method: GET

Parameters:

partner_name: string (required) – Partner name.

subdomain: string (required) - Subdomain.

Returns:

True (200 OK) or error.

Errors:

400: Bad Request: invalid value specified for `username`. Given string is too short.

404: Not Found: Branding does not exists.

2.5 Create user branding (branding.json)

Description: Create user branding.

Role: 2

URL Structure: /branding.json

Method: POST

Parameters:

session_id: string (required) - Session ID.
host_mapping: string - HostMapping - Host name URL (URL format).
drive_name: string - DriveName - Drive name (min 1)(max 35).
menu_color: string - MenuColor - HEX (max 6).
font_color: string - FonColor - HEX (max 6).
login_page_html: string - LoginPageHTML – text.
white_labeled: integer - WhiteLabeled - (0 = no, 1 = yes).

Returns:

True (200 OK) or error.

Errors:

400: Bad Request: User not exists.
404: Not Found: User not found.

2.6 Upload user branding files (upload.json)

Description: Upload user branding files.

Role: 2

URL Structure: /branding/upload.json

Method: POST

Parameters:

session_id: string (required) - Session ID.
type: string - Type('user','account_user','group','logo','favicon').
group_id: string - Required for type group.

Returns:

True (200 OK) or error.

Errors:

400: Bad Request: User not exists.
404: Not Found: Please select file for upload.

2.7 Update user branding info (branding.json)

Description: Update user branding info.

Role: 2

URL Structure: /branding.json

Method: PUT

Parameters:

session_id: string (required) - Session ID.
host_mapping: string - HostMapping - Host name URL (URL format).
drive_name: string - DriveName - Drive name (min 1)(max 35).
menu_color: string - MenuColor - HEX (max 6).

font_color: string - FonColor - HEX (max 6).
login_page_html: string - LoginPageHTML – text.
white_labeled: integer - WhiteLabeled - (0 = no, 1 = yes).
subdomain: string - Subdomain.

Returns:

True (200 OK) or error.

Errors:

400: Bad Request: User not exists
404: Not Found: Please select file for upload

2.8 Delete user branding info (branding.json)

Description: Delete user branding info.

Role: 2

URL Structure: /branding.json/{session_id}

Method: DELETE

Parameters:

session_id: string (required) - Session ID.

Returns:

True (200 OK) or error.

Errors:

400: Bad Request: User not exists.
404: Not Found: Please select file for upload.

2.9 Delete user image (userimage.json)

Description: Delete user avatar or logo.

Role: 2

URL Structure: /branding/userimage.json

Method: DELETE

Parameters:

session_id: string (required) - Session ID.

image_type: string (required) – Type ('user', 'account_user', 'logo', 'favicon').

Returns:

True (200 OK) or error.

Errors:

400: Bad Request: User not exists
404: Not Found: Please select file for upload.

3. Download

3.1 Download file (file.json)

Description: Download file.

Role: 2

URL Structure: /download/file.json/{file_id}

Method: GET

Parameters:

file_id: string (required) – File ID.
session_id: string – Session ID.
offset: integer - Offset in bytes.
inline: boolean - Disposition, by default send as attachment.
sharing_id: string – Sharing ID.
test: integer - Only test that file can be download.
backup: integer - Download file backup if available.
app: string – Internal.
temp_auth: string – Internal.
preview: string – Internal.
temp_key: string - Temp key for password-protected files.

Returns:

File content (200 OK) or error.

Errors:

400: Bad Request: File not exists.
400: Bad Request: Invalid value specified for 'value'.
404: Not Found: Please select file for upload.

3.2 Download zipped content (all.json)

Description: Download zipped content.

Role: 2

URL Structure: /download/all.json

Method: POST

Parameters:

files: string - Comma-separated files IDs.
folders: string - Comma-separated folder IDs.
session_key: string - Session ID.

Returns:

File content (200 OK) or error

Errors:

400: Bad Request: File not exists.
400: Bad Request: Invalid value specified for 'value'.
404: Not Found: Please select file for upload.

4. File

4.1 Create and get a file expiring link. (expiringlink.json)

Description: Create and get a file expiring link.

Role: 1 and 2

URL Structure: /v1/file/expiringlink.json/{session_id}/{date}/{counter}/{file_id}/{enable}

Method: GET

Parameters:

session_id: string (required) - Session ID.

date: string (required) - Expiration date.

counter: int (required) - Counter.

file_id: string (required) - File ID.

enable: string (required) - Enable.

Returns:

DownloadLink: string - Expiring download link.

StreamingLink: string - Expiring streaming link.

Errors:

400: Bad Request: Invalid file ID.

401: Unauthorized: Session has expired. Please right click on OpenDrive task bar icon, log out and then log back in.

4.2 Get a file expiring link information. (fileexpiringlinks.json)

Description: Get information about an existing file expiring link.

Role: 1 and 2

URL Structure: /v1/file/fileexpiringlinks.json/{session_id}/{file_id}

Method: GET

Parameters:

session_id: string (required) - Session ID.

file_id: string (required) - File ID.

Returns:

DownloadLink: string - Expiring download link.

StreamingLink: string - Expiring streaming link.

CounterMax: int - Maximum uses.

CounterEnable: int - 1 - Maximum uses enabled, 0 - Maximum uses ignored.

ExpiringDate: string - Expiring date.

Counter: int - Number of uses.

Errors:

400: Bad Request: Invalid file ID.

401: Unauthorized: Session has expired. Please right click on OpenDrive task bar icon, log out and then log back in.

4.3 File versions (fileversions.json)

Description: Get file versions.

Role: 1 and 2

URL Structure: /file/fileversions.json/{session_id}/{file_group_id}

Method: GET

Parameters:

session_id: string (required) - Session ID.

file_group_id: integer (required) – File group ID.

Returns:

```
file_id: {
  ID: string – File ID.
  FileId: string – File ID.
  GroupID: integer – Files group ID.
  Version: string – File version ID.
  Name: string – File name.
  Size: integer - File size in bytes.
  DateModified: integer – Last file’s modification date.
  DateUploaded: integer – Last file’s version upload date.
  DirectLink: string – Link for file review.
  Extension: string – File format.
  Owner: string – File owner name.
  ThumbLink: string - Link to share file.
}
```

Errors:

404 Not Found: Folder cannot be copied/moved to itself.

404 Not Found: File or folder with such name already exists.

4.4 File info (info.json)

Description: File details.

Role: 1 and 2

URL Structure: /file/info.json/{file_id}

Method: GET

Parameters:

file_id: string (required) – File ID.

session_id: string - Session ID.

sharing_id: string – Sharing ID.

Returns:

FileId: string - The file’s ID.

Name: string - The file’s name.

GroupID: string – Group ID for files which have several version.

Extension: string – File format.

Size: string - The file size in bytes.

Views: string – Number of file reviews.

Version: string – Actual file`s version.

Downloads: string – Number of file downloads.

DateTrashed: string - The date the file was trashed. 0 if it never was.

DateModified: string - The date last modified.

Access: string – File access type (1 – public, 2 – hidden, 3 - private).

FileHash: string – The hash related to the file content.

Link: string – The link to review of edit the file .

DownloadLink: string - The files download link.

StreamingLink: string - The files streaming link.

OwnerName: string – User name of a user who uploaded the file.

BWExceeded: integer – 1 – download limit has been exceeded for the file. 0 – limit has not been exceeded.

Password: string – File address password (optionally).

EditOnline: integer – An ability to change the file online.

Description: string – File description, added by user optionally.

IsArchive: string – Is file an archive (1 – archive, 0 – not archive).

Date: string – Last file change date.

DateUploaded: integer - The date the file was uploaded.

DateAccessed: string - The date the file was accessed. 0 if it never was accessed.

AccessDisabled: integer – 1 – the file is prohibited for public sharing, only authorized users can work with a file. 0 – no restrictions.

DestURL: string – Not used.

Owner: string – File owner`s ID.

AccessUser: string – The existence of users who have an access to a file.

DirUpdateTime: integer - The directory update time.

Errors:

400: Bad Request: Invalid file ID.

401: Unauthorized: Session has expired. Please right click on OpenDrive task bar icon, log out and then log back in.

404: Not Found: Folder cannot be copied/moved to itself.

409: Conflict: File or folder with such name already exists.

4.5 File path (path.json)

Description: Get a file path.

Role: 1 and 2

URL Structure: /file/path.json/{session_id}/{file_id}

Method: GET

Parameters:

session_id: string (required) - Session ID.

file_id: string (required) - File ID.

Returns:

Path: string - The file pathway.

Errors:

400: Bad Request: Invalid file ID.

401: Unauthorized: Session has expired. Please right click on OpenDrive task bar icon, log out and then log back in.

4.6 File thumbnail (thumb.json)

Description: File thumbnail.

Role: 1 and 2

URL Structure: /file/thumb.json/{file_id}

Method: GET

Parameters:

session_id: string (required) - Session ID.

file_id: string (required) - File ID.

sharing_id: string - Sharing ID.

Returns:

Thumbnail 256px x 256px (200 OK) or error.

Errors:

400: Bad Request: Invalid file ID

401: Unauthorized: Session has expired. Please right click on OpenDrive task bar icon, log out and then log back in.

4.7 Create a new empty file (file.json)

Description: Create a empty new file.

Role: 1 and 2

URL Structure: /file.json

Method: POST

Parameters:

session_id: string (required) - Session ID.

access_folder_id: string (required) - Access folder ID.

folder_id: string (required) - Folder ID.

file_type: string (required) - File type (doc, ppt, xls, txt).

sharing_id: string - Sharing ID.

Returns:

FileId: string – ID of created file.

Name: string – Name of created file.

GroupID: integer – Files group's ID.

Extension: string — File format.
Size: string – File size in bytes.
Views: string — Number of file reviews.
Downloads: string — Number of file downloads.
DateModified: string — Date of last file change.
Access: string — File access type (1 – public, 2 – hidden, 3 - private).
Link: string – Address of file location.
DownloadLink: string – File downloading link.
StreamingLink: string – File stream link.
TempStreamingLink: string — An app to play media files.
Password: string – File access password.
Description: string – File description.
DirUpdateTime: integer – The directory update time.

Errors:

400: Bad Request: Invalid file ID.
400: Bad Request: “Values” is required.
401: Unauthorized: Session has expired. Please right click on OpenDrive task bar icon, log out and then log back in.

4.8 Update file access permission (access.json)

Description: Update file access permission.

Role: 1 and 2

URL Structure: /file/access.json

Method: POST

Parameters:

session_id: string (required) - Session ID.
file_id: string (required) - File ID.
file_ispublic: integer (required) - (0 = private, 1 = public, 2 = hidden).
access_folder_id: integer - Access folder ID.
sharing_id: string - Sharing ID.

Returns:

FileId: string – File ID.
Name: string – File name.
GroupID: string - Files group`s ID.
Extension: string – File extension.
Size: string - File size in bytes.
Views: string - Number of file reviews.
Downloads: string - Number of file downloads.
DateModified: string - Date of last file change.
Access: string - File access type (1 – public, 2 – hidden, 3 - private).

Link: string - Address of file location.
DownloadLink: string - File downloading link.
StreamingLink: string - File stream link.
Password: string - File access password.
Description: string – File description.
DirUpdateTime: integer - The directory update time.

Errors:

400: Bad Request: Invalid file ID.
400: Bad Request: “Values” is required.
401: Unauthorized: Session has expired. Please right click on OpenDrive task bar icon, log out and then log back in.

4.9 Get a file ID (idbypath.json)

Description: Get a file ID by its Path.

Role: 1 and 2

URL Structure: /file/idbypath.json

Method: POST

Parameters:

session_id: string (required) - Session ID.
path: string (required) - File Path.

Returns:

FileId: string – file ID.
DownloadLink: string – file download link.

Errors:

404 Not Found: File not found.
400: Bad Request: `path` is required.

4.10 Copy or move a file (move_copy.json)

Description: Copy or move a file from its original location.

Role: 1 and 2

URL Structure: /file/move_copy.json

Method: POST

Parameters:

session_id: string (required) - Session ID.
src_file_id: string (required) - Source file ID.
dst_folder_id: string (required) - Destination folder ID.
move: string (required) - (true = move, false = copy).
overwrite_if_exists: string (required) - (true, false).

src_access_folder_id: string - Source access folder.
dst_access_folder_id: string - Destination access folder.
src_sharing_id: string - Source Sharing ID.
dst_sharing_id: string - Destination Sharing ID.
new_file_name: string - New name for destination file.

Returns:

FileId: string - File ID.
Name: string - File name.
GroupID: string - Files group's ID.
Extension: string - File extension.
Size: string - File size in bytes.
Views: string - Number of file reviews.
Downloads: string - Number of file downloads.
DateModified: string - Date of last file change.
Access: string - File access type (1 – public, 2 – hidden, 3 - private).
Link: string - Address of file location.
DownloadLink: string - File downloading link.
StreamingLink: string - File stream link.
Password: string – File access password.
Description: string – File description.
DirUpdateTimeSrc: integer - The directory update time.
DirUpdateTime: integer - The directory update time.

Errors:

404 Not Found: Folder cannot be copied/moved to itself.
404 Not Found: File or folder with such name already exists.

4.11 Rename a file (rename.json)

Description: Rename a file.

Role: 1 and 2

URL Structure: /file/rename.json

Method: POST

Parameters:

session_id: string (required) - Session ID.
new_file_name: string (required) - New file name.
file_id: string (required) - File ID.
access_folder_id: string - Access folder ID.
sharing_id: string - Sharing ID.

Returns:

FileId: string – File ID.

Name: string – File name.
GroupID: string - Files group`s ID.
Extension: string - File extension.
Size: string - File size in bytes.
Views: string - Number of file reviews.
Downloads: string - Number of file downloads.
DateModified: string - Date of last file change.
Access: string - File access type (1 – public, 2 – hidden, 3 - private).
Link: string - Address of file location.
DownloadLink: string - File downloading link.
StreamingLink: string - File stream link.
Password: string – File access password.
Description: string – File description.
DirUpdateTime: integer - The directory update time.

Errors:

400: Bad Request: Invalid file ID.
401: Unauthorized: Session has expired. Please right click on OpenDrive task bar icon, log out and then log back in.
404: Not Found: File or folder with such name already exists

4.12 Restore a file (restore.json)

Description: Restore a file from the trash.

Role: 1 and 2

URL Structure: /file/restore.json

Method: POST

session_id: string (required) - Session ID.

file_id: string (required) - File ID.

Returns:

FileId: string – File ID.
Name: string – File name.
GroupID: string - Files group`s ID.
Extension: string - File extension.
Size: string - File size in bytes.
Views: string - Number of file reviews.
Downloads: string - Number of file downloads.
DateModified: string - Date of last file change.
Access: string - File access type (1 – public, 2 – hidden, 3 - private).
Link: string - Address of file location.
DownloadLink: string - File downloading link.
StreamingLink: string - File stream link.
Password: string - File access password.

Description: string – File description.

ParentID: string – Parents ID.

DirUpdateTime: integer - The directory update time.

Errors:

400: Bad Request: Invalid file ID.

401: Unauthorized: Session has expired. Please right click on OpenDrive task bar icon, log out and then log back in.

4.13 Send file link by email (sendbyemail.json)

Description: Send file link by email.

Role: 1 and 2

URL Structure: /file/sendbyemail.json

Method: POST

Parameters:

file_id: string (required) - File ID.

recipient_emails: string (required) - Emails, separated by comma.

captcha_response : string - Captcha response.

message_subject: string - Custom subject. Replaces the default message subject.

message_body: string – User message. Additional text to the message with file link.

session_id: string – Session ID.

send_expiring_enabled : boolean - Send existing expiring link.

Returns:

Recipient_emails: string - Emails, separated by comma.

Errors:

400: Bad Request: Invalid file ID.

401: Unauthorized: Session has expired. Please right click on OpenDrive task bar icon, log out and

then log back in.

4.14 Trash a file (trash.json)

Description: Delete a file to the trash folder.

Role: 1 and 2

URL Structure: /file/trash.json

Method: POST

Parameters:

session_id: string (required) - Session ID.

file_id: string (required) - File ID.

sharing_id: string - Sharing ID.

Returns:

DirUpdateTime: integer - The Directory update time.

Errors:

400: Bad Request: Invalid file ID.

401: Unauthorized: Session has expired. Please right click on OpenDrive task bar icon, log out and then log back in.

4.15 Verify password (verifypassword.json)

Description:

Role: 1 and 2

URL Structure: /file/verifypassword.json

Method: POST

Parameters:

file_id: string (required) - File ID.

password: string(required) – File’s password.

captcha_response: string - Captcha response.

session_id: string - Session ID.

Returns:

Result: true or false.

Errors:

400: Bad Request: Invalid file ID.

401: Unauthorized: Session has expired. Please right click on OpenDrive task bar icon, log out and then log back in.

4.16 Edit file settings (filesettings.json)

Description: Edit file’s settings.

Role: 1 and 2

URL Structure: /file/filesettings.json

Method: PUT

Parameters:

session_id: string (required) - Session ID.

file_id: string (required) - File ID.

file_price: string - File price.

file_name: string - File name.

file_description: string - File description.

file_password: string - File password.

file_dest_url: string - File destination URL.

file_ispublic: integer - File public or no private = 0, public = 1, hidden – 2;

file_edit_online: integer - File edit online.

file_modification_time: integer - File modification time.

sharing_id: string - Sharing ID.

Returns:

True (200 OK) or error

Errors:

400: Bad Request: Invalid file ID.

401: Unauthorized: Session has expired. Please right click on OpenDrive task bar icon, log out and then log back in.

4.17 Delete a file from trash (file.json)

Description: Delete a file from the trash. Note: there is no recovery of this file.

Role: 1 and 2

URL Structure: /file.json/{session_id}/{file_id}

Method: DELETE

Parameters:

session_id: string (required) - Session ID.

file_id: string (required) - File ID.

access_folder_id: string - Access folder ID.

sharing_id: string – Sharing ID.

Returns:

DirUpdateTime: integer - The Directory update time.

Errors:

400: Bad Request: Invalid file ID.

401: Unauthorized: Session has expired. Please right click on OpenDrive task bar icon, log out and then log back in.

4.18 Remove file version (removefileversion.json)

Description: Remove file version.

Role: 1 and 2

URL Structure: /file/removefileversion.json/{session_id}/{file_id}

Method: DELETE

Parameters:

session_id: string (required) - Session ID.

file_id: string (required) - File ID.

Returns:

DirUpdateTime: integer - The Directory update time.

Update_time: integer – a date of last update.

Errors:

400: Bad Request: Invalid file ID.

401: Unauthorized: Session has expired. Please right click on OpenDrive task bar icon, log out and then log back in.

5. Folders API

5.1 Get a folder breadcrumb (breadcrumb.json)

Description: Get a folder breadcrumb.

Role: 1 and 2

URL Structure: /folder/breadcrumb.json/{session_id}/{folder_id}

Method: GET

Parameters:

session_id: string (required) - Session ID.

folder_id: string (required) - Folder ID.

with_subfolders: boolean - Include subfolders in response.

sharing_id: string - Sharing ID.

Returns:

FolderID: string – folder ID.

Name: string – folder name.

SubFolders: array – a list of folders stored in this folder.

Errors:

401: Unauthorized: Session has expired. Please right click on OpenDrive task bar icon, log out and then log back in.

404: Not found: wrong folder ID.

5.2 Create and get a folder expiring link. (expiringlink.json)

Description: Create and get a file expiring link.

Role: 1 and 2

URL Structure: /v1/folder/expiringlink.json/{session_id}/{date}/{counter}/{folder_id}/{enable}

Method: GET

Parameters:

session_id: string (required) - Session ID.

date: string (required) - Expiration date.

counter: int (required) - Maximum uses.

folder_id: string (required) - Folder ID.

enable: int (required) - Enable counter.

Returns:

Link: string - Expiring link.

Errors:

400: Bad Request: Invalid folder ID.

401: Unauthorized: Session has expired. Please right click on OpenDrive task bar icon, log out and then log back in.

5.3 Get a folder expiring link information. (folderexpiringlinks.json)

Description: Get information about an existing file expiring link.

Role: 1 and 2

URL Structure: /v1/folder/folderexpiringlinks.json/{session_id}/{folder_id}

Method: GET

Parameters:

session_id: string (required) - Session ID.

folder_id: string (required) - Folder ID.

Returns:

Link: string - Expiring link.

CounterMax: int - Maximum uses

CounterEnable: int - 1 - Maximum uses enabled, 0 - Maximum uses ignored.

ExpiringDate: string - Expiring date

Counter: int - Number of uses

Errors:

400: Bad Request: Invalid file ID.

401: Unauthorized: Session has expired. Please right click on OpenDrive task bar icon, log out and then log back in.

5.4 Folder info (info.json)

Description: Folder details.

Role: 1 and 2

URL Structure: /folder/info.json/{session_id}/{folder_id}

Method: GET

Parameters:

session_id: string (required) - Session ID.

folder_id: string (required) - Folder ID.

sharing_id: string - Sharing ID.

Returns:

FolderID: string – Folder ID.

Name: string – Folder name.

DateCreated: integer – Folder creating date.

DateTrashed: integer – Folder removing date.

DirUpdateTime: integer - The Directory update time.

Access: integer - File access type (1 – public, 2 – hidden, 3 - private).

PublicUpload: boolean – An ability to upload to a folder for other users.

PublicContent: boolean - An ability to review a folder content for other users.

DateModified: integer – Last folder change date.

Owner: integer – File owner's ID.

Shared: string – The info about is folder shared to other users.

ChildFolders: integer – Number of folders in given folder.

OwnerLevel: integer – Folder owner's plan identifier.

OwnerSuspended: boolean – The marker of suspending user access to Opendrive.

ID: string – Folder identifier in the DB.

Description: string – Folder description. Permission: integer – Folder access rights.

PublicDownload: boolean – An ability to get an access for files for not authorized users.

Link: string – address of folder location.

Lang: string – site version's language.

Encrypted: string – folder encrypting marker.

Errors:

401: Unauthorized: Session has expired. Please right click on OpenDrive task bar icon, log out and then log back in.

404: Not found: wrong folder ID.

5.5 Get item by name (itembyname.json)

Description: Get item by name.

Role: 1 and 2

URL Structure: /folder/itembyname.json/{session_id}/{folder_id}

Method: GET

Parameters:

session_id: string (required) - Session ID.

folder_id: string (required) - Folder ID.

name: string – Name of item.

sharing_id: string – Sharing ID.

encryption_supported: integer – 1 –if the client supports encrypted folder and files.

Returns:

DirUpdateTime: string - The Directory update time.

Errors:

401: Unauthorized: Session has expired. Please right click on OpenDrive task bar icon, log out and then log back in.

404: Not found: wrong folder ID.

5.6 List folder content (list.json)

Description: List folder content.

Role: 1 and 2

URL Structure: /folder/list.json/{session_id}/{folder_id}

Method: GET

Parameters:

session_id: string (required) - Session ID.

folder_id: string (required) - Folder ID (0 for root folder).

search_query: string - search query.

last_request_time: integer - DirUpdateTime value of last getList request.

sharing_id: string - Sharing ID.

encryption_supported: integer - 1 - if the client supports encrypted folders and files.

only_subfolders: boolean - return only child folders.

with_breadcrumbs: boolean - include folder breadcrumbs.

offset: (integer) - offset for items pagination in the folder. The function will return maximum 100 items in each call if offset is present in function input parameters. *Offset* should be used together with parameter *last_request_time*. Set *last_request_time* to 0 in first call to *list.json*, and use *DirUpdateTime* as *last_request_time* from response for next calls to *list.json*. Please compare *DirUpdateTime* from response and *last_request_time*, if values are different, this means that folder content or folder

properties has been changed and client should reload the folder content from 0 offset. If the *offset* parameter is missing, the function will return all items contained in the folder.

order_by: string - order by field (name, extension, modified_date, size, dl_count).

order_type: string - order type (asc (default), desc).

Returns:

DirUpdateTime: string - The Directory update time.

Name: string – folder name.

ParentFolderID: string – parent folder ID.

DirectFolderLink: string – address of a directory where the folder is located.

ResponseType: integer - 1,

Folders:

```
{
  FolderID: string – Folder ID
  Name: string – Folder name
  DateCreated: integer – Folder creating date
  Access: integer - 1
  DateModified: integer – Last folder modification date
  Shared: string – Marker of folder shared to other users
  ChildFolders: integer – Number of folders inside this given folder
  Link: string – Folder location address
  Encrypted: string – Marker of encrypted folder
},
```

Files:

```
{
  FileId: string – File ID
  Name: string – File name
  GroupID: Files group's ID
  Extension: string – File format
  Size: string – File size in bytes
  Views: string – Number of file reviews
  Version: string – Actual file version
  Downloads: string – Number of file downloads
  DateModified: string – Last modification date
  Access: string - File access type (1 – public, 2 – hidden, 3 - private)
  Link: string – File location address
  DownloadLink: string – File downloading link
  StreamingLink: string – File stream link
  TempStreamingLink: string – An application to play media files
  ThumbLink: string – File to file thumbnail
  Password: string – File access password
  EditOnline: integer – An ability to change a file online
```

Errors:

401: Unauthorized: Session has expired. Please right click on OpenDrive task bar icon, log out and then log back in.

5.7 Get a folder path (path.json)

Description: Get a folder path.

Role: 1 and 2

URL Structure: /folder/path.json/{session_id}/{folder_id}

Method: GET

Parameters:

session_id: string (required) - Session ID.

folder_id: string (required) - Folder ID.

Returns:

Folder path: Folder name.

Errors:

401: Unauthorized: Session has expired. Please right click on OpenDrive task bar icon, log out and then log back in.

404: Not found: wrong folder ID

5.8 Trash list (trashlist.json)

Description: List Trash folder content.

Role: 1 and 2

URL Structure: /folder/trashlist.json/{session_id}

Method: GET

Parameters:

session_id: string (required) - Session ID.

count_only: integer - Return only items count in Trash (0 or 1).

last_request_time: int - DirUpdateTime value of last getTrashList request.

offset: int - offset for pagination.

order_by: string - order by field (name, trashed_date, modified_date, size)

order_type: string - order type (asc (default), desc).

Returns:

FilesCount: integer – files number in trash

FoldersCount: integer – folders number in trash

Folders:

[

FolderID: string – Folder ID

Name: string – Folder name

DateCreated: integer – Creating date

DateTrashed: integer – Removing date

DirUpdateTime: integer – Directory update time.

Access: integer - File access type (1 – public, 2 – hidden, 3 - private)

PublicUpload: boolean – An ability to upload files for not authorized users

PublicContent: boolean – An ability to review the folder's content for not authorized users

DateModified: integer – Date of last folder change

Owner: integer – Folder owner's ID

Shared: string – Marker of shared folder
 OwnerLevel: string – File owner’s plan marker
 OwnerSuspended: boolean – Marker of suspended user access to OpenDrive
 Encrypted: string – Encrypted folder’s marker

]

Files: [

FileId: string – File ID
 Name: string – File name
 GroupID: Files group ID
 Extension: string – File format
 Size: string – File size in bytes
 Views: string – Number of file reviews
 Version: string – Actual file version
 Downloads: string – Number of file downloads
 DataTrashed: string- Date of removing
 DateModified: string – Date of last file change
 Access: integer - File access type (1 – public, 2 – hidden, 3 - private)
 Link: string – Address of file location
 DownloadLink: string – File downloading link
 StreamingLink: string – File stream link
 ThumbLink: string – File thumbnail link
 Encrypted: integer – Encrypted folder’s marker
 Password: string – File access password
 TempStreamingLink: string - Application to play media files

]

Errors:

401: Unauthorized: Session has expired. Please right click on OpenDrive task bar icon, log out and then log back in.

404: Not found: Wrong folder ID

5.9 Get user access mode (useraccessmode.json)

Description: Get user access mode for specified folder.

Role: 1 and 2

URL Structure: /folder/useraccessmode.json/{session_id}/{folder_id}

Method: GET

Parameters:

session_id: string (required) - Session ID.

folder_id: string (required) - Folder ID.

sharing_id: string - Sharing ID.

Returns:

UserAccessMode: string – User access mode.

Errors:

401: Unauthorized: Session has expired. Please right click on OpenDrive task bar icon, log out and

then log back in.

404: Not found: wrong folder ID

5.10 Create a folder (folder.json)

Description: Create a new folder.

Role: 1 and 2

URL Structure: /folder.json

Method: POST

Parameters:

session_id: string (required) - Session ID.

folder_name: string (required) - Valid folder name required (max 255).

folder_sub_parent: string - Folder sub parent (folder_id, 0-for root folder).

folder_is_public: integer - (0 = private, 1 = public, 2 = hidden).

folder_public_upl: integer - Public upload (0 = disabled, 1 = enabled).

folder_public_display: integer - Public display (0 = disabled, 1 = enabled).

folder_public_dnl: integer - Public download (0 = disabled, 1 = enabled).

folder_display_subfolders: int - Show subfolders in public folder (0 = disabled, 1 = enabled).

folder_description: string - Folder description.

sharing_id: string - Sharing ID.

Returns:

FolderID: string – Folder ID

Name: string – Folder name

DateCreated: integer – Folder creating date

DirUpdateTime: integer – Directory update time.

Access: integer - Folder access type (1 – public, 2 – hidden, 3 - private)

DateModified: integer - Date of last folder change

Shared: string - Marker of folder shared to other users

Description: string – Folder description

Link: string – File location address

Errors:

400: Bad request: Folder name is invalid or cannot contain any of the following characters \\ / : * ? \\
< > |"

400: Bad request: Invalid value specified for `folder_name`. Maximum 255 characters allowed.

400: Bad request: `folder_name` is required.

401: Unauthorized: Session has expired. Please right click on OpenDrive task bar icon, log out and

then log back in.

5.11 Get a folder ID (idbypath.json)

Description: Get a folder ID by its Path

Role: 1 and 2

URL Structure: /folder/idbypath.json

Method: POST

Parameters:

session_id: string (required) - Session ID.

path: string (required) - File Path.

Returns:

FolderId: string – folder ID

Errors:

401: Unauthorized: Session has expired. Please right click on OpenDrive task bar icon, log out and then log back in.

404: Not found: Folder not found

5.12 Copy or move folder (move_copy.json)

Description: Copy or move a folder to another location.

Role: 1 and 2

URL Structure: /folder/move_copy.json

Method: POST

Parameters:

session_id: string (required) - Session ID.

folder_id: string (required) - Folder ID.

dst_folder_id: string (required) - Destination folder ID.

move: (true - move, false – copy)

src_sharing_id: string - Source Sharing ID.

dst_sharing_id: string - Destination Sharing ID.

new_folder_name: string - New name for destination folder.

copy_recursive: boolean - Do the copy recursively or not. If false, then the folder with files but without subfolders will be copied.

Returns:

FolderID: string – folder ID

Name: string – folder name

DateCreated: integer – folder creating date

DirUpdateTime: integer – Directory update time.

Access: integer - file access type (1 – public, 2 – hidden, 3 - private)

DateModified: integer - date of last folder change

Shared: string - marker of folder shared to other users

Description: string – folder description

Link: string - address of file location

Errors:

401: Unauthorized: Session has expired. Please right click on OpenDrive task bar icon, log out and then log back in.

403: Folder record update failed

404: Not Found: Folder cannot be copied/moved to itself

404: Not found: Folder not found

404: Not Found: File or folder with such name already exists

409: File or folder with such name already exists

5.13 Delete a folder from trash (remove.json)

Description: Delete a folder from trash

Role: 1 and 2

URL Structure: /folder/remove.json

Method: POST

Parameters:

session_id: string (required) - Session ID (20).

folder_id: string (required) - Folder ID.

sharing_id: string - Sharing ID.

Returns:

DirUpdateTime: integer - Directory update time.

Errors:

400: Bad request: Invalid arguments

401: Unauthorized: Session has expired. Please right click on OpenDrive task bar icon, log out and then log back in.

404: Not found: Folder not found

5.14 Rename a folder (rename.json)

Description: Rename a folder.

Role: 1 and 2

URL Structure: /folder/rename.json

Method:

Parameters:

session_id: string (required) - Session ID.

folder_id: string (required) - Folder ID.

folder_name: string (required) - New folder name (max 255).

sharing_id: string - Sharing ID.

Returns:

FolderID: string – folder ID

Name: string – folder name

DateCreated: integer – folder creating date

DirUpdateTime: integer – Directory update time.

Access: integer - file access type (1 – public, 2 – hidden, 3 - private)

DateModified: integer - date of last folder change

Shared: string - marker of folder shared to other users

Description: string – folder description

Link: string - address of file location

Errors:

401: Unauthorized: Session has expired. Please right click on OpenDrive task bar icon, log out and then log back in.

404: Not Found: File or folder with such name already exists

5.15 Restore a folder (restore.json)

Description: Restore a folder from trash

Role: 1 and 2

URL Structure: /folder/restore.json

Method: POST

Parameters:

session_id: string (required) - Session ID.

folder_id: string (required) - Folder ID.

Returns:

FolderID: string – folder ID

Name: string – folder name

DateCreated: integer – folder creating date

DirUpdateTime: integer – Directory update time.

Access: integer - file access type (1 – public, 2 – hidden, 3 - private)

DateModified: integer - date of last folder change

Shared: string - marker of folder shared to other users

Description: string – folder description

Link: string – address of file location

ParentID: string – parent folder ID

Errors:

401: Unauthorized: Session has expired. Please right click on OpenDrive task bar icon, log out and then log back in.

5.16 Send folder link by email (sendbyemail.json)

Description: Send folder link by email.

Role: 1 and 2

URL Structure: /folder/sendbyemail.json

Method: POST

Parameters:

folder_id: mixed (required) - Folder ID.

recipient_emails: string (required) - Emails, separated by comma.

captcha_response : string - Captcha response.

message_subject: string - Custom subject. Replaces the default message subject.

message_body: string – User message. Additional text to the message with folder link.

session_id: string – Session ID.

send_expiring_enabled : boolean - Send existing expiring link.

Returns:

Recipient emails

Errors:

401: Unauthorized: Session has expired. Please right click on OpenDrive task bar icon, log out and then log back in.

404: Not found: Directory not exists

5.17 Update folder access permission (setaccess.json)

Description: Change a folder's access permissions.

Role: 1 and 2

URL Structure: /folder/setaccess.json

Method: POST

Parameters:

session_id: string (required) - Session ID.

folder_id: string (required) - Folder ID.

folder_ispublic: integer (required) - (0 = private, 1 = public, 2 = hidden).

sharing_id: string - Sharing ID.

with_child_files: boolean- true to set access mode for folder and all child files.

Returns:

FolderID: string – folder ID

Name: string – folder name

DateCreated: integer – folder creating date

DirUpdateTime: integer – Directory update time.

Access: integer - file acces type (1 – public, 2 – hidden, 3 - private)

DateModified: integer - last folder change date

Shared: string - marker of shared of folder shared to other users

Description: string – folder description

Link: string - file location address

Errors:

400: Bad request: Not a valid argument.

400: Bad request: Invalid value specified for `with_child_files`. Expected one of (true, false).

400: Bad request: Error parsing JSON, malformed JSON

401: Unauthorized: Session has expired. Please right click on OpenDrive task bar icon, log out and

then log back in

5.18 Trash a folder (trash.json)

Description: Send a folder to the trash bin.

Role: 1 and 2

URL Structure: /folder/trash.json

Method: POST

Parameters:

session_id: string (required) - Session ID.

folder_id: string (required) - Folder ID.

sharing_id: string - Sharing ID.

Returns:

DirUpdateTime: integer - Directory update time.

Errors:

401: Unauthorized: Session has expired. Please right click on OpenDrive task bar icon, log out and then log back in.

404: Not found: Directory doesn't exist

5.19 Edit folder settings (foldersettings.json)

Description: Edit folder settings.

Role: 1 and 2

URL Structure: /folder/foldersettings.json

Method: PUT

Parameters:

session_id: string (required) - Session ID.

folder_id: string (required) - Folder ID.

folder_name: string - Folder name (max 255).

folder_description: string - Folder description (max 255).

folder_access: int - Folder Access.

folder_public_upl: integer - Folder public upload.

folder_public_display: integer - Folder public display.

folder_public_dnl: integer - Folder public download.

folder_display_subfolders: int - Show subfolder in public folder.

sharing_id: string - Sharing ID.

Returns:

FolderID: string – Folder ID

Name: string – Folder name

DateCreated: integer – Folder creating date

DateTrashed: integer – Removing date (for files in trash)

DirUpdateTime: integer – Directory update time.
Access: integer - File access type (1 – public, 2 – hidden, 3 - private)
PublicUpload: boolean - An ability to upload file for not authorized users
PublicContent: boolean - An ability to review folder's content for not authorized users
DateModified: integer - Date of last folder change
Owner: integer – Folder owner's ID
Shared: string – Marker of the folder shared to other users
ChildFolders: integer – Number of child folders in the folder
OwnerLevel: string – ID of file owner's plan
OwnerSuspended: boolean - Marker of suspended user access to OpenDrive
ID: : string – ID in the database
Description: string – Folder description
Permission: integer – File access marker
PublicDownload: string – An ability to download files from the folder for not authorized users
Link: string – Address of folder location
Lang: string – Site version's language
Encrypted: string – Marker of encrypted folder

Errors:

400: Bad request: Invalid folder name
400: Bad request: Folder name is invalid or cannot contain any of the following characters \\ / : * ? \\
< > |"
400: Bad request: Invalid value specified for `folder_name`. Maximum 255 characters allowed.
400: Bad request: `folder_name` is required.
400: Bad request: Invalid value specified for `folder_public_display`. Expected one of (0, 1).
401: Unauthorized: Session has expired. Please right click on OpenDrive task bar icon, log out and

then log back in.

5.20 Empty trash (trash.json)

Description: Empty trash.

Role: 1 and 2

URL Structure: /folder/trash.json/{session_id}

Method: DELETE

Parameters:

session_id: string (required) - Session ID.

Returns:

True (200 OK) or error

Errors:

401: Unauthorized: Session has expired. Please right click on OpenDrive task bar icon, log out and then log back in.

6. Notes

6.1 Get all notes (notes.json)

Description: Get all notes in Notepad or Note list (ListId or NotepadID is required)

Role: 1 and 2

URL Structure: /notes.json/{session_id}/{item_type}/{item_id}

Method: GET

Parameters:

session_id: string (required) - Session ID.

item_type: string - notepad or notelist

item_id: integer (required) - Notepad Id or Note Id.

Returns:

Noteld: string – Note ID

NotelistId: string – Notelist ID the note is belonging to

Created: string - Note creating date

Status: string – Note status

Text: string – Note text

ThumbLink: string – Link attached to the note

FileLink: string – File attached to the note

Errors:

404: Not found: Notepad not found

404: Not found: Note list not found

6.2 Get archived note lists (archived.json)

Description: Get archived note lists and notepads

Role: 1 and 2

URL Structure: /notes/archived.json/{session_id}

Method: GET

Parameters:

session_id: string (required) - Session ID.

Returns:

Notepads: [

NotepadId: string – Notepad ID

Name: string – Notepad name

Created: string - Notepad creating date

Archived: string – Notepad archiving date

Notelists: [

NoteListId: string – Notelist ID

Name: string – Notelist name

Color: string - Notelist color

Size: string - Number of records in the notelist

Order: string – Notelist order value

Trashed: string – Marker of notelist removed to trash

Created: string - Notelist creating date

Errors:

404: Not found: Notepad not found

404: Not found: Note list not found

6.3 Get note list in notepad (notelists.json)

Description: Get note lists in notepad.

Role: 1 and 2

URL Structure: /notes/notelists.json/{session_id}/{notepad_id}

Method: GET

Parameters:

session_id: string (required) - Session ID.

notepad_id: string (required) – Notepad ID.

Returns:

NoteListId: string – Notelist ID

Name: string – Notelist name

Color: string - Notelist color

Size: string - Number of records in notelist

Order: string – Value of order list

Archived: string – Marker of archived notelist

Created: string – Notelist creating date

Errors:

404: Not found: Notepad not found

404: Not found: Note list not found

6.4 Get notepads list (notepads.json)

Description: Get notepads list.

Role: 1 and 2

URL Structure: /notes/notepads.json/{session_id}

Method: GET

Parameters:

session_id: string (required) - Session ID.

Returns:

NotepadId: string - Notepad ID

Name: string - Notepad name

Created: string – Notepad creating date

NotelistCount: string - Number of notelists in notepad

Errors: n/a

6.5 Search in notepads and notes (search.json)

Description: Get notepads list.

Role: 1 and 2

URL Structure: /notes/search.json/{session_id}

Method: GET

Parameters:

session_id: string (required) - Session ID.

search_query: string (required) - Search query.

Returns:

Notes:

NotepadId: string – ID of the notepad containign the record

NoteText: string – Text of the record found by the search query

Notepads:

NotepadId: string – Notepad ID

Name: string – Name of notepad found by the search query.

Errors: n/a

6.6 Get trashed note lists (trashed.json)

Description: Get trashed note lists and notepads.

Role: 1 and 2

URL Structure: /notes/trashed.json/{session_id}

Method: GET

Parameters:

session_id: string (required) - Session ID.

Returns:

Notepads:

NotepadId: string – Notepad ID

Name: string - Notepad name

Created: string - Notepad creating date

Trashed: string - Notepad removing date

Notelists:

NoteListId: string - Notelist ID

Name: string - Notelist name

Color: string – Notelist color

Size: string – Notelist size

Order: string - Number of records in notelist

Trashed: string - Notelist removing date

Created: string - Notelist creating date

Errors: n/a

6.6 Create a new note (notes.json)

Description: Create a new note.

Role: 1 and 2

URL Structure: /notes.json

Method: POST

Parameters:

session_id: string (required) - Session ID.

list_id: string (required) - Note list ID.

text: string (required) - Note text.

status: string - Note status (default, important, done).

Returns:

note_id: string – Record ID

Errors:

400: Bad request: Invalid value specified for `value`

404: Not Found: Note list not found

6.7 Attach the file to a note (attachfile.json)

Description: Attach the file to a note.

Role: 1 and 2

URL Structure: /notes/attachfile.json

Method: POST

Parameters:

session_id: string (required) - Session ID.

note_id: string (required) - Note ID.

file_id: string (required) – File ID.

Returns:

True (200 OK) or error

Errors:

400: Bad request: Invalid value specified for `value`

404: Not Found: File not found

6.8 Create a new notes list in notepad (noteslist.json)

Description: Create a new notes list in notepad.

Role: 1 and 2

URL Structure: /notes/noteslist.json

Method: POST

Parameters:

session_id: string (required) - Session ID.
notepad_id: string (required) - Notepad ID.
name: string (required) - Notepad name.
order: integer (required) - Note list order.
color: string (required) - Note list color in hex RGB.
size: integer (required) - Size of tiles in notepad (1, 2, 4).

Returns:

list_id: string – Created note list ID

Errors:

400: Bad request: Invalid value specified for `value`
404: Not Found: Notepad not found

6.9 Create a new notepad (notepad.json)

Description: Create a new notepad.

Role: 1 and 2

URL Structure: /notes/notepad.json

Method: POST

Parameters:

session_id: string (required) - Session ID.
name: string (required) – Notepad name.

Returns:

notepad_id: string – Created notepad ID

Errors:

400: Bad request: Invalid value specified for `value`
404: Not Found: Notepad not found

6.10 Update a note (notes.json)

Description: Update a note.

Role: 1 and 2

URL Structure: /notes.json

Method: PUT

Parameters:

session_id: string (required) - Session ID.
note_id: string (required) - Note ID.
text: string (required) - Note text.
status: string - Note status (default,important,done).

Returns:

result: boolean – True or False

Errors:

400: Bad request: Invalid value specified for `value`

404: Not Found: Notepad not found

6.11 Archive the notes list (archivenoteslist.json)

Description: Archive the notes list.

Role: 1 and 2

URL Structure: /notes/archivenoteslist.json/{session_id}/{list_id}

Method: PUT

Parameters:

session_id: string (required) - Session ID.

list_id: string (required) - Note list ID.

Returns:

result: boolean – True or False

Errors:

400: Bad request: Invalid value specified for `value`

404: Not Found: Notelist not found

6.12 Archive a notepad (archivenotepad.json)

Description: Archive a notepad.

Role: 1 and 2

URL Structure: /notes/archivenotepad.json/{session_id}/{notepad_id}

Method: PUT

Parameters:

session_id: string (required) - Session ID.

notepad_id: string (required) - Notepad ID.

Returns:

result: boolean – True or False

Errors:

400: Bad request: Invalid value specified for `value`

404: Not Found: Notepad not found

6.13 Update notes list parameters (noteslist.json)

Description: Update notes list parameters.

Role: 1 and 2

URL Structure: /notes/noteslist.json/{session_id}/{list_id}

Method: PUT

Parameters:

session_id: string (required) - Session ID.

list_id: string (required) - Note list ID.

name: string - Notepad name.
order: integer - Note list order.
color: string - Note list color in hex RGB.
size: int - Number of tiles in list (1, 2, 4).

Returns:

result: boolean – True or False

Errors:

400: Bad request: Invalid value specified for `value`
404: Not Found: Notelist not found

6.14 Update notepad name (notepad.json)

Description: Update notepad name.

Role: 1 and 2

URL Structure: /notes/notepad.json/{session_id}/{notepad_id}

Method: PUT

Parameters:

session_id: string (required) - Session ID.
notepad_id: string (required) - Notepad ID.
name: string (required) - Notepad name

Returns:

result: boolean – True or False

Errors:

400: Bad request: Invalid value specified for `value`
404: Not Found: Notelist not found

6.15 Update multiple note lists order (ordernotelists.json)

Description: Update notepad name.

Role: 1 and 2

URL Structure: /notes/ordernotelists.json/{session_id}

Method: PUT

Parameters:

session_id: string (required) - Session ID.
notelists: Object - Paste JSON object data here (required) . Note list array {notelistId1: order1, notelistId2: order2...}.

Returns:

Array

Errors:

400: Bad request: Invalid value specified for `value`
404: Not Found: Notelist not found

6.16 Restore notelist from archive (restorearchivednotelist.json)

Description: Restore notelist from archive

Role: 1 and 2

URL Structure: /notes/restorearchivednotelist.json/{session_id}/{list_id}

Method: PUT

Parameters:

session_id: string (required) - Session ID.

list_id: string (required) - Note list ID.

Returns:

result: boolean – True or False

Errors:

400: Bad request: Invalid value specified for `value`

403: Forbidden: Notes list is not archived

404: Not Found: Notelist not found

6.17 Restore notepad from archive (restorearchivednotepad.json)

Description: Restore notepad from archive.

Role: 1 and 2

URL Structure: /notes/restorearchivednotepad.json/{session_id}/{notepad_id}

Method: PUT

Parameters:

session_id: string (required) - Session ID.

notepad_id: string (required) - Notepad ID.

Returns:

result: boolean – True or False

Errors:

400: Bad request: Invalid value specified for `value`

403: Forbidden: Notepad is not archived

404: Not Found: Notepad not found

6.18 Restore notelist from trash (restoretrashednotelist.json)

Description: Restore notelist from trash.

Role: 1 and 2

URL Structure: /notes/restoretrashednotelist.json/{session_id}/{list_id}

Method: PUT

Parameters:

session_id: string (required) - Session ID.

list_id: string (required) - Note list ID.

Returns:

result: boolean – True or False

Errors:

400: Bad request: Invalid value specified for `value`

403: Forbidden: Notes list is not trashed

404: Not Found: Notelist not found

6.19 Restore notepad from trash (restoretrashednotepad.json)

Description: Restore notepad from trash.

Role: 1 and 2

URL Structure: /notes/restoretrashednotepad.json/{session_id}/{list_id}

Method: PUT

Parameters:

session_id: string (required) - Session ID.

notepad_id: string (required) - Notepad ID.

Returns:

result: boolean – True or False

Errors:

400: Bad request: Invalid value specified for `value`

403: Forbidden: Notepad list is not trashed

404: Not Found: Notepad not found

6.20 Delete a note (notes.json)

Description: Delete a note and all dependent files.

Role: 1 and 2

URL Structure: /notes.json/{session_id}/{note_id}

Method: DELETE

Parameters:

session_id: string (required) - Session ID.

note_id: string (required) - Note ID.

Returns:

result: boolean – True or False

Errors:

400: Bad request: Invalid value specified for `value`

404: Not Found: Note not found

6.21 Delete the notes list (noteslist.json)

Description: Delete the notes list and all dependent items (files, notes).

Role: 1 and 2

URL Structure: /notes/noteslist.json/{session_id}/{list_id}

Method: DELETE

Parameters:

session_id: string (required) - Session ID.

list_id: string (required) - Note list ID.

Returns:

result: boolean – True or False

Errors:

400: Bad request: Invalid value specified for `value`

404: Not Found: Notelist not found

6.22 Trash the notepad (trashnotepad.json)

Description: Put notepad into trash and all dependent items (note lists, files, notes).

Role: 1 and 2

URL Structure: /notes/trashnotepad.json/{session_id}/{notepad_id}

Method: DELETE

Parameters:

session_id: string (required) - Session ID.

notepad_id: string (required) - Notepad ID.

Returns:

result: boolean – True or False

Errors:

404: Not Found: Notepad not found

6.23 Empty trash (trash.json)

Description: Empty trash (Delete all trashed notepads and notelists).

Role: 1 and 2

URL Structure:

URL Structure: /notes/trash.json/{session_id}

Method: DELETE

Parameters:

session_id: string (required) - Session ID.

Returns:

result: boolean – True or False

Errors:

n/a

6.24 Trash the notes list (trashnoteslist.json)

Description: Put note list into trash.

Role: 1 and 2

URL Structure: /notes/trashnoteslist.json/{session_id}/{list_id}

Method: DELETE

Parameters:

session_id: string (required) - Session ID.

list_id: string (required) – Note list ID.

Returns:

result: boolean – True or False.

Errors:

404: Not Found: Note list not found

6.25 Trash the notepad (trashnotepad.json)

Description: Trash the notepad

Role: 1 and 2

URL Structure: /notes/trashnotepad.json/{session_id}/{notepad_id}

Method: DELETE

Parameters:

session_id: string (required) - Session ID.

notepad_id: string (required) – Notepad ID.

Returns:

result: boolean – True or False.

Errors:

404: Not Found: Note list not found

7. Secure folders

7.1 The first stage of Secure Folders authorization (auth.json)

Description: Request the data Secure Folders authorization. The first stage of Secure Folders authorization.

Role: 1 and 2

URL Structure: /securefolders/auth.json/{session_id}/{folder_id}

Method: GET

Parameters:

session_id: string (required) - Session ID of 20 numbers.

folder_id: string (required) - Folder ID.

Returns:

Data: string -

Errors:

403: Forbidden: Open encrypted directory failed

403: Forbidden: This folder is not encrypted

400: Bad request: Invalid argument

7.2 The second stage of Secure Folders authorization (auth.json)

Description: Secure Folders authorization by hash. The second stage of Secure Folders authorization

Role: 1 and 2

URL Structure: /securefolders/auth.json

Method: POST

Parameters:

session_id: string (required) - Session ID of 20 numbers.

folder_id: integer (required) - Folder ID.

hash: string(required) - Hash for password verification.

Returns:

Errors:

403: Forbidden: Access denied. Wrong password.

400: Bad Request: Values is required.

7.3 Set password for secure folder (init.json)

Description: Set password for secure folder

Role: 1 and 2

URL Structure: /securefolders/init.json

Method: POST

Parameters:

session_id: string (required) - Session ID of 20 numbers.

folder_id: integer (required) - Folder ID.

hash: string(required) – Data for password verification.

data: string (required) - Hash for password verification.

Returns:

Errors:

403: Forbidden: Encryption password is already set and can not be changed

8. Session API:

8.1 Session information (info.json)

Description: User account information from session.

Role: 1 and 2

URL Structure: /session/info.json

Method: GET

Parameters:

session_id: string (required) - Session ID.

Returns:

SessionID: string – session ID

UserName: string – username or email

UserFirstName: string – user first name

UserLastName: string – user last name

AccType: string – account type

UserLang: string – user site version's language

UserID: string – user ID

IsAccountUser: string – marker of AccountUser user account's type

DriveName: string – user disk name.

UserLevel: string – user level depending on account type

UserPlan: string – user tariff plan

FVersioning: string – available number of version of one file

UserDomain: string – user domain name

PartnerUsersDomain: string – user parent domain's name

Encoding: string – Unicode transformation format

IsPartner: string – marker of parent user's domain

Errors:

401: Unauthorized: Invalid sessions please relogin.

8.2 Session exists (exists.json)

Description: Used to return the current session status.

Role: 1 and 2

URL Structure: /session/exists.json

Method: POST

Parameters:

session_id: string (required) - Session ID.

Returns:

result: True or Error

Errors:

401: Unauthorized: Invalid sessions please relogin.

8.3 Create a session (login.json)

Description: Creates a login session.

Role: 1 and 2

URL Structure: /session/login.json

Method: POST

Parameters:

username: string (required) - User name (min 4)(max 100).

passwd: string (required) – User password. Minimum 5 characters.

version: string – Application version number (max 10).

partner_id: string - Partner username (Empty for OpenDrive).

captcha_response: string - Captcha response.

Returns:

SessionID: string – The session ID.

UserName: string - The users name or email.

UserFirstName: string - The user’s first name. Minimum 2 characters.

UserLastName: string – The user’s last name. Minimum 2 characters.

AccType: integer – Type of account (1 – personal, 2 – business).

UserLang: string - The user’s language.

English (En)

Spanish (Es)

Portuguese (Pt)

German (De)

French (Fr)

Simplified Chinese (Zhs)

Traditional Chinese (Zht)

Czech (Cz)

Hungarian (Hu)

Dutch (Nl)

Polish (Pl)

Russian (Ru)

Slovak (Sk)

IsAccessUser: integer – (0 – user, 1 – account user)

DriveName: string - The Drive name: OpenDrive

UserLevel: string – user account type

UserPlan: string – user tariff plan

FVersioning: string - available number of version of one file

UserDomain: string – user domain name

PartnerUsersDomain: string – user parent domain name

IsPartner: string – marker of parent user domain

Encoding: string - Unicode transformation format

Errors:

401: Unauthorized: Invalid username or password.

401: Unauthorized: Invalid email or password.

400: Bad Request: Value is required.

8.4 Log out (logout.json)

Description: Log out of the session.

Role: 1 and 2

URL Structure: /session/logout.json

Method: POST

Parameters:

session_id: string (required) - Session ID.

Returns:

True (200 OK) or error

Errors:

400: Bad Requests: Session does not exist

9. Sharing

9.1 List of shared folders (listsharedfolders.json)

Description: List of shared folders

Role: 1 and 2

URL Structure: /sharing/listsharedfolders.json/{session_id}/{sharing_id}

Method: GET

Parameters:

session_id: string (required) - Session ID.

sharing_id: string (required) – shared user ID

Returns:

DirUpdateTime: integer – a date of last update of parent folder

ResponseType: integer – used for caching of folders listing

SharedUserName: string

SharingID: string

SharedFolders: {

FolderID,

Name,

Description,

IsHomeFolder,

IsDeleteable,

IsRenameable,

IsGallery,

IsMP3,

Permission,

Access,

HasSubfolders,

Shared,

Link,

DateCreated,

DateModified,

DateTrashed,

Encrypted,

SharingID,

ShareMode,

DisplaySubfolders,

PublicContent,

PublicDownload,

PublicUpload

}

Errors:

401: Unauthorized: Invalid sessions please relogin.

400: Bad request: Invalid sharing id

9.2 List of users who shared (listsharedusers.json)

Description: List of users who shared

Role: 1 and 2

URL Structure: /sharing/listsharedusers.json/{session_id}

Method: GET

Parameters:

session_id: string (required) - Session ID.

Returns:

DirUpdateTime: integer - date of last update of parent folder.

ResponseType: integer - date of last update of parent folder.

SharedUsers:

SharingID: integer – sharing ID.

Name: string – a user name who shared the folder

Errors:

401: Unauthorized: Invalid sessions please relogin.

9.3 List of share with users (listusers.json)

Description: List of share with users.

Role: 1 and 2

URL Structure: /sharing/listusers.json/{session_id}/{folder_id}

Method: GET

Parameters:

session_id: string (required) - Session ID.

folder_id: string (required) – Folder ID.

Returns:

ShareWithUser:

SharingID: string – sharing ID

Name: string – a user name who has the folder shared

ShareMode: string – Share mode (0 - View only, 1 - Full access mode).

Errors:

400: Bad request: Invalid folder id

401: Unauthorized: Invalid sessions please relogin.

9.4 Share folder with user (sharing.json)

Description: Share folder with user.

Role: 1 and 2

URL Structure: /sharing.json

Method: POST

Parameters:

session_id: string (required) - Session ID.

folder_id: string (required) – Folder ID.

username: string (required)- User name or email.

sharemode: string (required)- Share mode (0 - View only, 1 - Full access mode).

Returns:

DirUpdateTime: integer - date of last update of parent folder.

SharingID: string — sharing ID

Name: string — a user name who has the folder shared

ShareMode: string - Share mode (0 - View only, 1 - Full access mode).

Errors:

400: Bad Request: Invalid value specified for `sharemode`. Expected one of (0, 1).

401: Unauthorized: Invalid sessions please relogin.

404: Not Found: This folder already shared with this user

404: Not found: Directory doesn't exist

404: Not found: User not found

9.5 Set mode (setmode.json)

Description: Set share mode.

Role: 1 and 2

URL Structure: /sharing/setmode.json

Method: PUT

Parameters:

session_id: string (required) - Session ID.

sharing_id: string (required) – Sharing ID.

sharemode: string (required) - Share mode (0 - View only, 1 - Full access mode).

Returns:

True (200 OK) or error

Errors:

400: Bad Request: Invalid shared id.

400: Bad Request: Invalid shared folder id.

401: Unauthorized: Invalid sessions please relogin.

9.6 Delete sharing (sharing.json)

Description: Delete sharing.

Role: 1 and 2

URL Structure: /sharing.json/{session_id}/{sharing_id}

Method: DELETE

Parameters:

session_id: string (required) - Session ID.

sharing_id: string (required) – Sharing ID.

Returns:

DirUpdateTime: integer – date of last update of parent folder.

Errors:

400: Bad Request: Invalid shared id.

401: Unauthorized: Invalid sessions please relogin.

10. Stats

10.1 All stats (allstats.json)

Description: Get all stats.

Role: 1 and 2

URL Structure: /stats/allstats.json/{session_id}/{month}/{year}

Method: GET

Parameters:

session_id: string (required) - Session ID.

month: integer (required) - Start date month (unixtime).

year: integer (required) - Start date year (unixtime).

Returns:

users_dataset: - number of users registered per particular day

up_name_dataset: - daily upload value for a particular day within a selected period

dl_num_dataset: - daily download value for a particular day within a selected period

users_free_dataset: - number of created free accounts for a particular day

users_pay_dataset: - number of created paid accounts for a particular day

monthly_uploaded: integer - monthly upload value

monthly_downloaded: integer - monthly download value

upload_bw_dataset: daily upload value, from first to last day of selected month

download_bw_dataset: daily download value, from first to last day of selected month

total_bw_uploaded: string - total value of upload for entire usage period

total_bw_downloaded: string - total value of download for entire usage period

users_count: string - total number of active accounts

free_users_count: string - total number of active free accounts

month_users_count: string - number of active users per specified month

storage_used: string - total number of used disk space

free_storage_used: string - total number of disk space used by free users

premium_storage_used: string - total number of disk space used by premium users

suspended_storage_used: string - total number of disk space used by suspended users

Errors:

400: Bad Request: Invalid value specified for `month`. Maximum allowed value is 12.

400: Bad Request: Invalid value specified for `year`. Minimum required value is 1970.

401: Unauthorized: Invalid sessions please relogin.

10.2 Get user bandwidth stats (bandwidth.json)

Description: Get user bandwidth stats and dl/up count. Dates returned in PST.

Role: 1 and 2

URL Structure: /stats/bandwidth.json/{session_id}/{month}/{year}

Method: GET

Parameters:

session_id: string (required) - Session ID of 20 numbers.

month: integer (required) - Start date month (unixtime).

year: integer (required) - Start date year (unixtime).

Returns:

up_num_dataset: {} daily upload value for a particular day within a selected period

dl_num_dataset: {} daily download value for a particular day within a selected period

monthly_uploaded: integer - monthly upload value

monthly_downloaded: integer - monthly download value

upload_bw_dataset: {} daily upload value, from first to last day of selected month

download_bw_dataset: {} daily download value, from first to last day of selected month

Errors:

401: Unauthorized: Invalid sessions please relogin.

10.3 Get users stats (bandwidth.json)

Description: Get user bandwidth stats and dl/up count. Dates returned in PST.

Role: 1 and 2

URL Structure: /stats/toplist.json/{session_id}/{year}/{month}/{day}/{stat_type}

Method: GET

Parameters:

session_id: string (required) - Session ID of 20 numbers.

year: integer (required) - Start date year (unixtime).

month: integer (required) - Start date month (unixtime).

day: integer (required) - Start date day (unixtime).

stat_type: string - Stat type (top_daily_dl_bw, top_monthly_dl_bw, top_daily_up_bw, top_monthly_up_bw, top_monthly_file_uploads, top_monthly_file_downloads, top_daily_file_downloads, top_daily_file_uploads).

Returns:

User statistics array

Errors:

401: Unauthorized: Invalid sessions please relogin.

11. Tasks:

11.1 Active tasks (tasks.json)

Description: Get active tasks. Search for tasks (search in task names and comment text).

Role: 1 and 2

URL Structure: /tasks.json

Method: GET

Parameters:

session_id: string - Session ID.

task_id: string - Task ID.

task_list_id: string – Task list ID.

type: string – Task type (active, trashed, archived).

search_query: string – Search query.

with_comments: boolean - Include comments for single task.

Returns:

TaskId: string – Task ID.

Name: string – Task Name

Number: integer – Integer number of the task in the project

Description: string – Task description text

StartDate: string – Task creation date

DueDate: string – Task deadline date

Private: integer – Marker of private task

Priority: integer – Task priority

Progress: integer – Task completion progress

Status: string – Task status

Color: string – Task background color

Trashed: integer – Marker of removed task

Archived: integer – Marker of archived task

OrderNo: integer – Task order number

AssignedFirstName: string – First name of the user the task is assigned to

AssignedLastName: string – Last name of the user the task is assigned to

AssignedToUserId: string – Id of the user the task is assigned to AssignedToAccessUserId: string - права доступа к таску, юзера на которого заасайнен таск

OwnerFirstName: string – Task creator's first name

OwnerLastName: string – Task creator's last name

TagColor: string – Task tag's color

TagText: string - Task tag's text

TagId: string - Task's tag ID

TaskListId: string – Id of task list the task is belonging ot

TaskListName: string - Name of task list the task is belonging to

ProjectId: string - Id of the project the task is belonging to

ProjectName: string - Project name the task is belonging to

ProjectListId: string - Project list ID the task is belonging to

ProjectListName: string - Project list name the task is belonging to

CommentsCount: integer – Number of comments to the task

OwnerUserId: integer – Owner ID

OwnerAccessUserId: integer – Id of account user who created the task

AccessMode: integer – Task access type
Link: string – Link to the task
AccountUsers:
 AccessUserId: string – ID of a user the task is assigned to
 UserId: string – Task creator ID
 FirstName: string – Task creator's first name
 LastName: string - Task creator's last name
Attachments:
 FileName: string – Name of a file attached to the task
 Size: string – File size
 FileId: string – File ID
 ThumbLink: string – Link to attached file
OwnerAvatar: string – Link to avatar image
OwnerAvatarColor: string - Owner's avatar color

Errors:

401: Unauthorized: Session has expired. Please right click on OpenDrive task bar icon, log out and then log back in.
403: Forbidden: Session is required
404: Not Found: Task not found or you do not have permissions to view this project.

11.2 List of archived items (archiveditems.json)

Description: Get list of archived items (projects, task lists or tasks)

Role: 1 and 2

URL Structure: /tasks/archiveditems.json/{session_id}/{item_type}

Method: GET

Parameters:

session_id: (required) string - Session ID.

item_type: string - Archived item type (project, task, task_list, project_list).

Returns:

ProjectListId: string – Project list ID
Name: string – Project list name
Created: integer – Project list creating date
Archived: integer – Project list archiving date
Projects:
 ProjectId: string – Project ID
 ProjectName: string – Project name
 Archived: integer – Project archiving date

TaskListId: string – Task list ID
Name: string – Task list name
DueDate: string – Task list deadline date
Trashed: integer – Marker of removed task list
Status: string – Task list status
Archived: integer – Marker of archived task list

ProjectName: string – Project name
ProjectListName: string - Project list name
AccountUsers:
 AccessUserId: string – Id of the user the task is assigned to
 UserId: string – Task creator's ID
 FirstName: string – First name of the task's creator
 LastName: string - Last name of the task's creator

Errors:
401: Unauthorized: Session has expired. Please right click on OpenDrive task bar icon, log out and then log back in.
403: Forbidden: Session is required

11.3 List items in archive (archivelist.json)

Description: List Items in Archive.

Role: 1 and 2

URL Structure: /tasks/archivelist.json/{session_id}

Method: GET

Parameters:

session_id: (required) string - Session ID.

Returns:

Projects: string – Number of projects on archive
ProjectLists: string – Number of project lists in archive
Tasks: string – Number of tasks in archive
TaskLists: string - Number of task lists in archive

Errors:
401: Unauthorized: Session has expired. Please right click on OpenDrive task bar icon, log out and then log back in.
403: Forbidden: Session is required

11.4 Project list (projectlist.json)

Description: Get project lists.

Role: 1 and 2

URL Structure: /tasks/projectlists.json

Method: GET

Parameters:

session_id: (required) string - Session ID.
 with_projects: boolean - Include projects in response.
 project_list_id: string - Project List ID.
 task_id: string - Task List ID.
 filter: string - Project list filter (archived, trashed, any). By default - return only active.

Returns:

ProjectListId: string – Project list ID
Name: string – Project list name

Created: integer – Project list creating date

Projects:

ProjectId: string – Project ID

ProjectName: string – Project name

Archived: integer – Marker of archived project

Trashed: integer – Marker of removed project

Errors:

400: Bad Request: Task Id is required

401: Unauthorized: Session has expired. Please right click on OpenDrive task bar icon, log out and then log back in.

403: Forbidden: Session is required

11.5 Projects (projects.json)

Description: Get projects.

Role: 1 and 2

URL Structure: /tasks/projects.json

Method: GET

Parameters:

session_id: (required) string - Session ID.

project_id: string - Project ID.

Returns:

ProjectId: string – Project ID

StartDate: string – Project creation date

DueDate: string – Project deadline date

Status: string – Project status

ProjectListId: string – ID of the project list containing this project

ProjectListName: string – Name of the project list containing this project

Name: string – Project name

Errors:

401: Unauthorized: Session has expired. Please right click on OpenDrive task bar icon, log out and then log back in.

403: Forbidden: Session is required

11.6 Search (search.json)

Description: Search for project list, project, task, comment, subcomment.

Role: 1 and 2

URL Structure: /tasks/search.json/{session_id}

Method: GET

Parameters:

session_id: (required) string - Session ID.

search_query: (required) string - Search query.

Returns:

Tasks:

TaskId: string – Task ID

Name: string – Task name

Comments:

TaskId: string - Task ID

Comment: string - Comment text

Subcomment: string – Subcomment text

Projects:

ProjectId: string – Project ID

Name: string – Project name

ProjectLists:

ProjectListId: string - Project list ID

Name: string – Project list name

Errors:

401: Unauthorized: Session has expired. Please right click on OpenDrive task bar icon, log out and then log back in.

403: Forbidden: Session is required

11.7 User tags list (tags.json)

Description: Get user tags list.

Role: 1 and 2

URL Structure: /tasks/tags.json/{session_id}

Method: GET

Parameters:

session_id: (required) string - Session ID.

Returns:

TagId: integer – Tag ID айди тага

Name: string – Tag name

Color: string - Tag background color

Errors:

401: Unauthorized: Session has expired. Please right click on OpenDrive task bar icon, log out and then log back in.

403: Forbidden: Session is required

11.8 Task comments (taskcomments.json)

Description: Get task comments

Role: 1 and 2

URL Structure: tasks/taskcomments.json/{session_id}/{task_id}/{comment_id}

Method: GET

Parameters:

session_id: (required) string - Session ID.

task_id: (required) string - Task ID.

comment_id: – Comment ID

Returns:

CommentId: string – Comment ID

Comment: string – Comment text

Number: string – Comment numeric number
Created: string - Comment creating date
TagId: integer - Tag ID
TagName: string - Tag name
TagColor: string - Tag background color
OwnerFirstName: string – First name of the user who left the comment
OwnerLastName: string – Last name of the user who left the comment
UserId: integer - ID of the user who left the comment
AccessUserId: integer - ID of the user the task is assigned to
OwnerAvatar: string - Link to the avatar of the user who left the comment
OwnerAvatarColor: string - Color of the avatar of the user who left the comment
SubComments:
 SubCommentId: string – Subcomment ID
 SubComment: string – Subcomment text
 Created: string – Subcomment creating date
 AccessUserId: integer - ID of the user the task is assigned to
 UserId: integer - ID of the user who left the subcomment
OwnerFirstName: string - First name of the user who left the subcomment
OwnerLastName: string - Last name of the user who left the subcomment
OwnerAvatar: string - Link to the avatar of the user who left the subcomment
OwnerAvatarColor: string - The color of the avatar of the user who left the subcomment
Attachments:
 FileId: string – ID of the file attached to the file
 FileName: string - Name of the file attached to the file
 Size: string - Size of the file attached to the file
 ThumbLink: string – Link to the file attached to the file

Errors:

400: Bad Request: Invalid value specified for `comment_id`
401: Unauthorized: Session has expired. Please right click on OpenDrive task bar icon, log out and then log back in.
403: Forbidden: Session is required

11.9 Task lists (tags.json)

Description: Get task lists

Role: 1 and 2

URL Structure: /tasks/tasklists.json

Method: GET

Parameters:

session_id: (required) string - Session ID.
project_id: string – Project ID.
task_list_id: string – Task list ID.
with_tasks: string - Include Tasks in list.
filter: string - Items filter (archived, trashed, active).
item_type: string - Items filter.

Returns:

TaskListId: string – Task list ID
Name: string – Task list name
DueDate: string – Task list deadline date
Trashed: integer – Marker of removed task list
Status: string – Task list status
Archived: integer – Marker of archived task list
ProjectName: string – Project name
ProjectListName: string - Project list name
AccountUsers:
 AccessUserId: string – ID of a user the task is assigned to
 UserId: string – Task creator ID
 FirstName: string – Task creator's first name
 LastName: string - Task creator's last name
AccessUserId: string – Id of the user the task is assigned to
UserId: string – Task creator's ID
FirstName: string – Task creator's first name
LastName: string - Task creator's last name

Errors:

401: Unauthorized: Session has expired. Please right click on OpenDrive task bar icon, log out and then log back in.
403: Forbidden: Session is required
404: Not Found: Project not found
404: Not Found: Tasks list not found

11.10 List of trashed items (trasheditems.json)

Description: Get list of trashed items (projects, task lists or tasks)

Role: 1 and 2

URL Structure: /tasks/trasheditems.json/{session_id}/{item_type}

Method: GET

Parameters:

session_id: (required) string - Session ID.

item_type: string - Items filter (project, task, task_list, project_list).

Returns:

ProjectListId: string – Project list ID
Name: string – Project list name
Created: integer – Project list creating date
Archived: integer – Project list archiving date

ProjectId: string – Project ID
StartDate: string – Project creating date
DueDate: string – Project deadline date
Status: string – Project status
ProjectListId: string – Project list ID the project is belonging to

ProjectListName: string – Project list name the project is belonging to
Name: string – Project name

TaskListId: string – Task list ID
Name: string – Task list name
DueDate: string – Task list's deadline date
Trashed: integer – Marker of removed task list
Status: string – Task list status
Archived: integer – Marker of archived task list
ProjectName: string – Project name имя проджекта
ProjectListName: string - Project list name
AccountUsers:
 AccessUserId: string – ID of a user the task is assigned to
 UserId: string – Task creator ID
 FirstName: string – Task creator's first name
 LastName: string - Task creator's last name

TaskId: string – Task ID.
Name: string – Task Name
Number: integer – Integer number of the task in the project
Description: string – Task description text
StartDate: string – Task creation date
DueDate: string – Task deadline date
Private: integer – Marker of private task
Priority: integer – Task priority
Progress: integer – Task completion progress
Status: string – Task status
Color: string – Task background color
Trashed: integer – Marker of removed task
Archived: integer – Marker of archived task
OrderNo: integer – Task order number
AssignedFirstName: string – First name of the user the task is assigned to
AssignedLastName: string – Last name of the user the task is assigned to
AssignedToUserId: string – Id of the user the task is assigned to
AssignedToAccessUserId: string - Task access rights of the user the task is belonging to
OwnerFirstName: string – Task creator's first name
OwnerLastName: string – Task creator's last name
TagColor: string – Task tag's color
TagText: string - Task tag's text
TagId: string - Task's tag ID
TaskListId: string – Id of task list the task is belonging to
TaskListName: string - Name of task list the task is belonging to
ProjectId: string - Id of the project the task is belonging to
ProjectName: string - Project name the task is belonging to
ProjectListId: string - Project list ID the task is belonging to
ProjectListName: string - Project list name the task is belonging to
CommentsCount: integer – Number of comments to the task
OwnerUserId: integer – Owner ID

OwnerAccessUserId: integer – Id of account user who created the task

AccessMode: integer – Task access type

Link: string – Link to the task

AccountUsers:

 AccessUserId: string – ID of a user the task is assigned to

 UserId: string – Task creator ID

 FirstName: string – Task creator's first name

 LastName: string - Task creator's last name

Attachments:

 FileName: string – Name of a file attached to the task

 Size: string – File size

 FileId: string – File ID

 ThumbLink: string – Link to attached file

OwnerAvatar: string – Link to avatar image

OwnerAvatarColor: string - Owner's avatar color

Errors:

401: Unauthorized: Session has expired. Please right click on OpenDrive task bar icon, log out and then log back in.

403: Forbidden: Session is required

11.11 List Items in Trash (trashlist.json)

Description: List Items in Trash

Role: 1 and 2

URL Structure: /tasks/trashlist.json/{session_id}

Method: GET

Parameters:

session_id: (required) string - Session ID.

Returns:

 Projects: string – Number of project in trash

 ProjectLists: string – Number of project lists in trash

 Tasks: string – Number of tasks in trash

 TaskLists: string - Number of task lists in trash

Errors:

401: Unauthorized: Session has expired. Please right click on OpenDrive task bar icon, log out and then log back in.

403: Forbidden: Session is required

11.12 Assigned users (users.json)

Description: Get assigned users.

Role: 1 and 2

URL Structure: /tasks/users.json/{session_id}

Method: GET

Parameters:

session_id: (required) string - Session ID.

project_id: string – Project ID.

Returns:

AccessUserId: string – Id of the user the project is assigned to

UserId: string – Project creator's ID

FirstName: string – Project creator's first name

LastName: string - Project creator's last name

Errors:

401: Unauthorized: Session has expired. Please right click on OpenDrive task bar icon, log out and then log back in.

403: Forbidden: Session is required

11.13 Create a task (tasks.json)

Description: Create a new task.

Role: 1 and 2

URL Structure: /tasks.json/{session_id}/{task_list_id}

Method: POST

Parameters:

session_id: (required) string - Session ID.

task_list_id: string – Task list ID.

task_name: string - Task name.

task_description: string - Task description.

start_date: string - Start date (YYYY-MM-DD).

email_assign_to: string - User email which task will be assigned.

due_date: string - Due date (YYYY-MM-DD).

private: int - Task privacy (0 - public, 1 - private).

priority: int - Task priority (0 - none, 1 - low, 2 - medium, 3 - high).

tag_id: int - Tag ID to (0 - no tags is assigned to task).

Returns:

TaskId: string – Task ID.

Name: string – Task Name

Number: integer – Integer number of the task in the project

Description: string – Task description text

StartDate: string – Task creation date

DueDate: string – Task deadline date

Private: integer – Marker of private task

Priority: integer – Task priority

Progress: integer – Task completion progress

Status: string – Task status

Color: string – Task background color

Trashed: integer – Marker of removed task

Archived: integer – Marker of archived task

OrderNo: integer – Task order number

AssignedFirstName: string – First name of the user the task is assigned to

AssignedLastName: string – Last name of the user the task is assigned to

AssignedToUserId: string – Id of the user the task is assigned to

AssignedToAccessUserId: string - Task access rights of the user the task is belonging to
OwnerFirstName: string – Task creator's first name
OwnerLastName: string – Task creator's last name
TagColor: string – Task tag's color
TagText: string - Task tag's text
TagId: string - Task's tag ID
TaskListId: string – Id of task list the task is belonging ot
TaskListName: string - Name of task list the task is belonging to
ProjectId: string - Id of the project the task is belonging to
ProjectName: string - Project name the task is belonging to
ProjectListId: string - Project list ID the task is belonging to
ProjectListName: string - Project list name the task is belonging to
CommentsCount: integer – Number of comments to the task
OwnerUserId: integer – Owner ID
OwnerAccessUserId: integer – Id of account user who created the task
AccessMode: integer – Task access type
Link: string – Link to the task
AccountUsers:
 AccessUserId: string – ID of a user the task is assigned to
 UserId: string – Task creator ID
 FirstName: string – Task creator's first name
 LastName: string - Task creator's last name
Attachments:
 FileName: string – Name of a file attached to the task
 Size: string – File size
 FileId: string – File ID
 ThumbLink: string – Link to attached file
OwnerAvatar: string – Link to avatar image
OwnerAvatarColor: string - Owner's avatar color

Errors:

401: Unauthorized: Session has expired. Please right click on OpenDrive task bar icon, log out and then log back in.
403: Forbidden: Session is required
403: Forbidden: You've reached maximum number of tasks. Account upgrade is required

11.14 Add a permission (addprojectpermission.json)

Description: Add a permission for project.

Role: 1 and 2

URL Structure: /tasks/addprojectpermission.json/{session_id}/{project_id}

Method: POST

Parameters:

session_id: (required) string - Session ID.
project_id: (required) string – Project ID.
access_user_email: string (required) - Invited access user e-mail.
mode: int (required) - Permission mode (0 - view, 1 - edit, 2 - block).

Returns:

result: True or False

Errors:

400: Bad Request: Project not found

400: Bad Request: User not found

400: Bad Request: Invalid value specified for `mode`. Expected one of (0,1,2).

401: Unauthorized: Session has expired. Please right click on OpenDrive task bar icon, log out and then log back in.

403: Forbidden: Session is required

11.15 New comment (comment.json)

Description: Create a new comment.

Role: 1 and 2

URL Structure: /tasks/comment.json/{session_id}/{task_id}

Method: POST

Parameters:

session_id: (required) string - Session ID.

task_id: string – Task ID.

comment_text: string (required) - Comment text.

Returns:

CommentId: string – ID of created comment

Errors:

400: Bad Request: Task not found

401: Unauthorized: Session has expired. Please right click on OpenDrive task bar icon, log out and then log back in.

403: Forbidden: Session is required

11.16 New project (project.json)

Description: Create a new project.

Role: 1 and 2

URL Structure: /tasks/project.json/{session_id}/{project_list_id}

Method: POST

Parameters:

session_id: (required) string - Session ID.

project_list_id: string – Project list ID.

project_name: string (required) - Project name.

email_assign_to: string - User email which task will be assigned.

start_date: string - Start date (YYYY-MM-DD).

due_date: string - Due date (YYYY-MM-DD).

status: string - Project status (active, completed).

Returns:

ProjectId: string – ID of created project

Name: string – name of created project

Errors:

400: Bad Request: Project list not found

401: Unauthorized: Session has expired. Please right click on OpenDrive task bar icon, log out and

then log back in.

403: Forbidden: Session is required

11.17 New project list (projectlist.json)

Description: Create a new project list.

Role: 1 and 2

URL Structure: /tasks/projectlist.json/{session_id}

Method: POST

Parameters:

session_id: (required) string - Session ID.

list_name: string (required) - List name.

Returns:

ProjectListId: string – ID of created project list

Name: string – Name of created project list

Errors:

401: Unauthorized: Session has expired. Please right click on OpenDrive task bar icon, log out and then log back in.

403: Forbidden: Session is required

11.18 New subcomment (subcomment.json)

Description: Create a new subcomment

Role: 1 and 2

URL Structure: /tasks/subcomment.json/{session_id}/{comment_id}

Method: POST

Parameters:

session_id: (required) string - Session ID.

comment_id: string (required) – Comment ID.

subcomment_text: string (required) - Subcomment text.

Returns:

SubCommentId: string – Subcomment ID

SubComment: string – Subcomment text

Created: string – Subcomment creating date

AccessUserId: integer - An ID of a user a task is assigned to

UserId: integer - An ID of a user who left a subcomment

string - First name of a user, who left a subcomment

OwnerFirstName:

OwnerLastName: string - Last name of a user, who left a subcomment

OwnerAvatar: string — Avatar of a user, who left a subcomment

OwnerAvatarColor: string - A color of user`s avatar, who left a subcomment

Errors:

400: Bad Request: Comment not found

401: Unauthorized: Session has expired. Please right click on OpenDrive task bar icon, log out and then log back in.

403: Forbidden: Session is required

11.19 New tag (tag.json)

Description: Create a new tag

Role: 1 and 2

URL Structure: /tasks/tag.json/{session_id}

Method: POST

Parameters:

session_id: (required) string - Session ID.

name: string (required) - Tag name.

color: string (required) - Tag color in hex RGB.

Returns:

TagId: string – ID of created tag

Errors:

400: Bad Request: Invalid value specified for `color`. Expecting only hexadecimal digits

401: Unauthorized: Session has expired. Please right click on OpenDrive task bar icon, log out and then log back in.

403: Forbidden: Session is required

11.20 New task list (tasklist.json)

Description: Create a new task list

Role: 1 and 2

URL Structure: /tasks/tasklist.json/{session_id}/{project_id}

Method: POST

Parameters:

session_id: (required) string - Session ID.

project_id: string (required) – Project ID.

list_name: string (required) - List name.

due_date: string - Due date (YYYY-MM-DD).

Returns:

TaskListId: string – ID of created task list

Name: string – name of created task list

Errors:

400: Bad Request: Project not found

401: Unauthorized: Session has expired. Please right click on OpenDrive task bar icon, log out and then log back in.

403: Forbidden: Session is required

11.21 Update the task (tasks.json)

Description: Update the task

Role: 1 and 2

URL Structure: /tasks.json/{session_id}/{task_id}

Method: PUT

Parameters:

session_id: (required) string - Session ID.
task_id: string (required) – Task ID.
task_name: string - Task name.
task_description: string - Task description.
order_no: integer - Task order N^o.
task_list_id: string - Task list ID.
access_userid_assign_to: string - Access UserId which task will be assigned (0-for account owner,'none' for nobody).
due_date: string - Due date (YYYY-MM-DD).
private: integer - Task privacy (0 - public, 1 - private).
priority: integer - Task priority (0 - none, 1 - low, 2 - medium, 3 - high).
tag_id: integer - Tag ID to (0 - no tags is assigned to task).
progress: integer - Task progress in %.
status: string - Task status (active, completed).
color: string - Task color in hex RGB.

Returns:

TaskId: string – Task ID.
Name: string – Task name
Number: integer – Numerical order of a task in the project
Description: string – Task test description
StartDate: string – Task creating date
DueDate: string – Task deadline date
Private: integer – Private task marker
Priority: integer – Task priority
Progress: integer – Task`s executing progress
Status: string – Task`s status
Color: string – Task`s background color
Trashed: integer – Removed task marker
Archived: integer – Archived task marker
OrderNo: integer – Task order number
AssignedFirstName: string – First name of a user a task is assigned to
AssignedLastName: string – Last name of a user a task is assigned to
AssignedToUserId: string – ID of a user a task is assigned to
AssignedToAccessUserId: string – Task access rights of a user a task is assigned to
OwnerFirstName: string – First name of task creator
OwnerLastName: string – Last name of task creator
TagColor: string – The task`s tag color
TagText: string – The task`s tag text
TagId: string – The task`s tag ID
TaskListId: string – Task list id where the task is located
TaskListName: string - Task list name where the task is located
ProjectId: string – Project ID where the task is located
ProjectName: string - Project name where the task is located
ProjectListId: string - Project list ID where the task is located
ProjectListName: string - Project list name where the task is located
CommentsCount: integer – Task`s comments number
OwnerUserId: integer – Owner ID

OwnerAccessUserId: integer – ID of account user who created the task

AccessMode: integer – Task access type

Link: string – Task link

AccountUsers:

AccessUserId: string – An ID of a user a task is assigned to

UserId: string – Task's creator ID

FirstName: string – Task's creator first name

LastName: string - Task's creator last name

Attachments:

FileName: string – A name of a file attached to a task

Size: string – File size

FileId: string – File ID

ThumbLink: string – Attachment-file link

OwnerAvatar: string – Avatar image link

OwnerAvatarColor: string - Owner's avatar color

Errors:

400: Bad Request: Invalid value specified for `value`.

401: Unauthorized: Session has expired. Please right click on OpenDrive task bar icon, log out and then log back in.

403: Forbidden: Session is required

404: Bad Request: Task not found

11.22 Archive the task (archive.json)

Description: Archive the task

Role: 1 and 2

URL Structure: /tasks/archive.json/{session_id}/{task_id}

Method: PUT

Parameters:

session_id: (required) string - Session ID.

task_id: string (required) – Task ID.

Returns:

result: True or False

Errors:

401: Unauthorized: Session has expired. Please right click on OpenDrive task bar icon, log out and then log back in.

403: Forbidden: Session is required

403: Forbidden: Task is archived and cannot be updated

404: Not found: Task not found

11.23 Archive the project (archiveproject.json)

Description: Archive the project and all dependent items (task lists, tasks)

Role: 1 and 2

URL Structure: /tasks/archiveproject.json/{session_id}/{project_id}

Method: PUT

Parameters:

session_id: (required) string - Session ID.

project_id: string (required) – Project ID.

Returns:

result: True or False

Errors:

401: Unauthorized: Session has expired. Please right click on OpenDrive task bar icon, log out and then log back in.

403: Forbidden: Session is required

403: Forbidden: Project is archived and cannot be updated

404: Not found: Project not found

11.24 Archive the project list (archiveprojectlist.json)

Description: Archive the project list

Role: 1 and 2

URL Structure: /tasks/archiveprojectlist.json/{session_id}/{list_id}

Method: PUT

Parameters:

session_id: (required) string - Session ID.

list_id: string (required) – Project list ID.

Returns:

result: True or False

Errors:

401: Unauthorized: Session has expired. Please right click on OpenDrive task bar icon, log out and then log back in.

403: Forbidden: Session is required

403: Forbidden: Project list is archived and cannot be updated

404: Not found: Project list not found

11.25 Archive the task list (archivetasklist.json)

Description: Archive the task list

Role: 1 and 2

URL Structure: /tasks/archivetasklist.json/{session_id}/{list_id}

Method: PUT

Parameters:

session_id: (required) string - Session ID.

list_id: string (required) – Task list ID.

Returns:

result: True or False

Errors:

401: Unauthorized: Session has expired. Please right click on OpenDrive task bar icon, log out and

then log back in.

403: Forbidden: Session is required

403: Forbidden: Task list is archived and cannot be updated

404: Not found: Task list not found

11.26 Update the task comment (comment.json)

Description: Update the task comment

Role: 1 and 2

URL Structure: /tasks/comment.json/{session_id}/{comment_id}

Method: PUT

Parameters:

session_id: (required) string - Session ID.

comment_id: string (required) – Comment ID.

comment_text: string - Comment text.

tag_id: integer - Comment Tag ID.

Returns:

CommentId: string – Comment ID

Comment: string – Comment text

Number: string – Order number of comment

Created: string – Comment creating date

TagId: integer – Tag ID

TagName: string – Tag name

TagColor: string – Background color tag

OwnerFirstName: string – First name of a user, who left a comment

OwnerLastName: string – Last name of a user, who left a comment

UserId: integer - An ID of a user who left a comment

AccessUserId: integer - An ID of a user a task is assigned to

OwnerAvatar: string - A link to user's avatar, who left a comment

OwnerAvatarColor: string - A color of user's avatar, who left a comment

SubComments:

SubCommentId: string – Subcomment ID

SubComment: string – Subcomment text

Created: string – Subcomment creating date

AccessUserId: integer - An ID of a user a task is assigned to

UserId: integer - An ID of a user who left a subcomment

OwnerFirstName: string - First name of a user, who left a subcomment

OwnerLastName: string - Last name of a user, who left a subcomment

OwnerAvatar: string - A link to user's avatar, who left a subcomment

OwnerAvatarColor: string - A color of user's avatar, who left a subcomment

Attachments:

FileId: string – ID of a file attached

FileName: string – name of a file attached

Size: string – size of a file attached

ThumbLink: string – link to the file attached

Errors:

401: Unauthorized: Session has expired. Please right click on OpenDrive task bar icon, log out and then log back in.

403: Forbidden: Session is required

404: Not found: Comment not found

11.27 Move the task (move.json)

Description: Move the task to another task list or project

Role: 1 and 2

URL Structure: /tasks/move.json/{session_id}

Method: PUT

Parameters:

session_id: (required) string - Session ID.

task_id: string (required) - Task ID.

insert_after_taskid: string - Insert the task after specified task in task list.

task_list_id: string - Task list ID.

project_id: string - Project ID.

item_type: string - Items filter.

Returns:

result: True or False

Errors:

401: Unauthorized: Session has expired. Please right click on OpenDrive task bar icon, log out and then log back in.

403: Forbidden: Session is required

404: Not found: Task list not found

11.28 Update the project (project.json)

Description: Update the project.

Role: 1 and 2

URL Structure: /tasks/project.json/{session_id}/{project_id}

Method: PUT

Parameters:

session_id: (required) string - Session ID.

project_id: string (required) - Project ID.

project_name: string - Project name.

email_assign_to: string - User email which task will be assigned.

start_date: string - Start date (YYYY-MM-DD).

due_date: string - Due date (YYYY-MM-DD).

status: string - Project status (active, completed).

Returns:

ProjectId: string – PProject ID

StartDate: string – Project creating date

DueDate: string – Project deadline date

Status: string – Project status

ProjectListId: string – Project list ID, where this project is located

ProjectListName: string – A name of a project list, where this project is located

Name: string –Project name

Errors:

401: Unauthorized: Session has expired. Please right click on OpenDrive task bar icon, log out and then log back in.

403: Forbidden: Session is required

404: Not found: Project not found

11.29 Update the project list (projectlist.json)

Description: Update the project list.

Role: 1 and 2

URL Structure: /tasks/projectlist.json/{session_id}/{list_id}

Method: PUT

Parameters:

session_id: (required) string - Session ID.

list_id: string (required) – Project list ID.

list_name: string (required) - List name.

Returns:

result: True or False

Name: string – New name of project list

Errors:

401: Unauthorized: Session has expired. Please right click on OpenDrive task bar icon, log out and then log back in.

403: Forbidden: Session is required

404: Not found: Project list not found

11.30 Restore the task (restore.json)

Description: Restore the task from archive or trash.

Role: 1 and 2

URL Structure: /tasks/restore.json/{session_id}/{task_id}

Method: PUT

Parameters:

session_id: (required) string - Session ID.

task_id: string (required) – Task ID.

Returns:

result: True or False

Errors:

401: Unauthorized: Session has expired. Please right click on OpenDrive task bar icon, log out and then log back in.

403: Forbidden: Session is required

404: Not found: Task not found

11.31 Restore the task list (restoretasklist.json)

Description: Restore the task list from archive or trash.

Role: 1 and 2

URL Structure: /tasks/restoretasklist.json/{session_id}/{list_id}

Method: PUT

Parameters:

session_id: (required) string - Session ID.

list_id: string (required) – Task list ID.

Returns:

result: True or False

Errors:

401: Unauthorized: Session has expired. Please right click on OpenDrive task bar icon, log out and then log back in.

403: Forbidden: Session is required

404: Not found: Task list not found

11.32 Update subcomment (subcomment.json)

Description: Update the task subcomment.

Role: 1 and 2

URL Structure: /tasks/subcomment.json/{session_id}/{subcomment_id}

Method: PUT

Parameters:

session_id: (required) string - Session ID.

subcomment_id: string (required) – Subcomment ID.

subcomment_text: string (required) - Subcomment text.

Returns:

SubCommentId: string – Subcomment ID

SubComment: string – Subcomment text

Created: string – Subcomment creating date

AccessUserId: integer - An ID of a user a task is assigned to

UserId: integer - An ID of a user who left a subcomment

OwnerFirstName: string – First name of a user, who left a subcomment

OwnerLastName: string – Last name of a user, who left a subcomment

OwnerAvatar: string – A link to user's avatar, who left a subcomment

OwnerAvatarColor: string – A color of user's avatar, who left a subcomment

Errors:

401: Unauthorized: Session has expired. Please right click on OpenDrive task bar icon, log out and then log back in.

403: Forbidden: Session is required

404: Not found: Subcomment not found

11.33 Update the tag (tag.json)

Description: Update the tag.

Role: 1 and 2

URL Structure: /tasks/tag.json/{session_id}/{tag_id}

Method: PUT

Parameters:

session_id: (required) string - Session ID.

tag_id: integer (required) – Tag ID.

name: string - Tag name.

color: string - Tag color in hex RGB.

Returns:

result: True or False

Errors:

401: Unauthorized: Session has expired. Please right click on OpenDrive task bar icon, log out and then log back in.

403: Forbidden: Session is required

404: Not found: Tag not found

11.34 Update the task list (tasklist.json)

Description: Update the task list.

Role: 1 and 2

URL Structure: /tasks/tasklist.json/{session_id}/{list_id}

Method: PUT

Parameters:

session_id: (required) string - Session ID.

list_id: string (required) – Task list ID.

list_name: string - Project name.

due_date: string - Due date (YYYY-MM-DD).

complete_all_tasks: boolean - Mark all tasks in the task list completed or uncompleted.

Returns:

Name: string – New name of the task list.

DueDate: string - Due date (YYYY-MM-DD).

Errors:

401: Unauthorized: Session has expired. Please right click on OpenDrive task bar icon, log out and then log back in.

403: Forbidden: Session is required

404: Not found: Task list not found

11.35 Delete the file (attachedfile.json)

Description: Delete the file from comment.

Role: 1 and 2

URL Structure:

/tasks/attachedfile.json/{session_id}/{item_type}/{item_id}/{file_id}

Method: DELETE

Parameters:

session_id: (required) string - Session ID.

item_type: string (required) – Item type (task, comment)

item_id: string (required) – Task or Comment ID.

file_id: string (required) – File ID.

Returns:

result: True or False

Errors:

401: Unauthorized: Session has expired. Please right click on OpenDrive task bar icon, log out and then log back in.

403: Forbidden: Session is required

404: Not found: Task not found

404: Not found: File not found

404: Not found: Comment not found

11.36 Empty trash (emptytrash.json)

Description: Empty trash.

Role: 1 and 2

URL Structure: /tasks/emptytrash.json/{session_id}

Method: DELETE

Parameters:

session_id: (required) string - Session ID.

Returns:

result: True or False

Errors:

401: Unauthorized: Session has expired. Please right click on OpenDrive task bar icon, log out and then log back in.

403: Forbidden: Session is required

11.37 Delete the tag (tag.json)

Description: Delete the tag.

Role: 1 and 2

URL Structure: /tasks/tag.json/{session_id}/{tag_id}

Method: DELETE

Parameters:

session_id: (required) string - Session ID.

tag_id: integer (required) – Tag ID.

Returns:

result: True or False

Errors:

400: Bad Request: Invalid value specified for `tag_id`

401: Unauthorized: Session has expired. Please right click on OpenDrive task bar icon, log out and then log back in.

403: Forbidden: Session is required

404: Not found: Tag not found

11.38 Trash the task (trash.json)

Description: Trash the task.

Role: 1 and 2

URL Structure: /tasks/trash.json/{session_id}/{task_id}

Method: DELETE

Parameters:

session_id: (required) string - Session ID.

task_id: string (required) – Task ID.

Returns:

result: True or False

Errors:

401: Unauthorized: Session has expired. Please right click on OpenDrive task bar icon, log out and then log back in.

403: Forbidden: Session is required

404: Not found: Task not found

11.39 Trash the comment (trashcomment.json)

Description: Trash the comment.

Role: 1 and 2

URL Structure: /tasks/trashcomment.json/{session_id}/{comment_id}

Method: DELETE

Parameters:

session_id: (required) string - Session ID.

comment_id: integer (required) – Comment ID.

Returns:

result: True or False

Errors:

400: Bad Request: invalid value specified for `comment_id`

401: Unauthorized: Session has expired. Please right click on OpenDrive task bar icon, log out and then log back in.

403: Forbidden: Session is required

404: Not found: Comment not found

11.40 Trash the project (trashproject.json)

Description: Trash the project and all dependent items (task lists, tasks).

Role: 1 and 2

URL Structure: /tasks/trashproject.json/{session_id}/{project_id}

Method: DELETE

Parameters:

session_id: (required) string - Session ID.

project_id: string (required) – Project ID.

Returns:

result: True or False

Errors:

401: Unauthorized: Session has expired. Please right click on OpenDrive task bar icon, log out and then log back in.

403: Forbidden: Session is required

404: Not found: Project not found

11.41 Trash the project list (trashprojectlist.json)

Description: Trash the project list and all dependent items (projects, task lists, tasks)

Role: 1 and 2

URL Structure: /tasks/trashprojectlist.json/{session_id}/{list_id}

Method: DELETE

Parameters:

session_id: (required) string - Session ID.

list_id: string (required) – Project list ID.

Returns:

result: True or False

Errors:

401: Unauthorized: Session has expired. Please right click on OpenDrive task bar icon, log out and then log back in.

403: Forbidden: Session is required

404: Not found: Project list not found

11.42 Trash the subcomment (trashsubcomment.json)

Description: Trash the subcomment.

Role: 1 and 2

URL Structure: /tasks/trashsubcomment.json/{session_id}/{subcomment_id}

Method: DELETE

Parameters:

session_id: (required) string - Session ID.

subcomment_id: integer (required) – Subcomment ID.

Returns:

result: True or False

Errors:

400: Bad Request: invalid value specified for `subcomment_id`

401: Unauthorized: Session has expired. Please right click on OpenDrive task bar icon, log out and then log back in.

403: Forbidden: Session is required

404: Not found: Subcomment not found

11.43 Trash the task list (trashtasklist.json)

Description: Trash the task list all dependent tasks.

Role: 1 and 2

URL Structure: /tasks/trashtasklist.json/{session_id}/{list_id}

Method: DELETE

Parameters:

session_id: (required) string - Session ID.

list_id: string (required) – Task list ID.

Returns:

result: True or False

Errors:

401: Unauthorized: Session has expired. Please right click on OpenDrive task bar icon, log out and then log back in.

403: Forbidden: Session is required

404: Not found: Task list not found

12. Upload API:

The Upload API executes the following Posts in the order listed when conducting an upload:

12.1 Check if file exists with such name (checkfileexistsbyname.json)

Description: Check if file exists with such name.

Role: 1 and 2

URL Structure: /upload/checkfileexistsbyname.json/{folder_id}

Method: POST

Parameters:

session_id: string - Session ID.

folder_id: string (required) - Folder ID. If root folder (0), session is required.

name: array (required) - File names.

sharing_id: string - Sharing ID.

Returns:

result: array – File name

Errors:

400: Bad Request: Invalid folder ID.

400: Bad Request: Invalid value specified for `name`.

401: Unauthorized: Session has expired. Please right click on OpenDrive task bar icon, log out and then log back in.

12.2 Close file upload (close_file_upload.json)

Description: Close file upload should be called after file upload.

Role: 1 and 2

URL Structure: /upload/close_file_upload.json

Method: POST

Parameters:

session_id: string (required) - Session ID.

file_id: string (required) - File ID.

file_size: integer (required) - File size in bytes.

temp_location: string - File temp location.

file_time: integer - Time of file creation.

access_folder_id: string - Access folder ID.

file_compressed: integer - File is compressed.

file_hash: string - MD5 File Hash (Optional).

sharing_id: string - Sharing ID.

Returns:

FileId: string - A file's ID.

Name: string - The file's name.

GroupID: string – Group ID for files which have several versions

Extension: string – File format

Size: string - The file size in bytes.

Views: string – Number of file reviews

Version: string – Actual file version

Downloads: string – File downloads number

DateTrashed: string - The date the file was trashed. 0 if it never was.

DateModified: string - The date last modified.

OwnerSuspendet: bollean – True or False

AccType: string – User account type, who uploaded a file

FileHash: string – Hash related to file content

Link: string – A link to review of edit a file

DownloadLink: string - File downloading link.

StreamingLink: string - File streaming link.

OwnerName: string – User who uploaded a file

upload_speed_limit: integer – Files uploading speed limit

download_speed_limit: integer – Files downloading speed limit

BWExceeded: integer – 1 – Download limit for file has been exceeded. 0 – limit has not been exceeded

ThumbLink: string – File share link

Encrypted: string – Folder encrypting marker

Password: string – File access password (optionally)

OwnerLevel: string – Owner's tariff plan

EditOnline: integer – An ability to change file online

ID: string – File ID in DB

FolderID: string – Folder ID

Description: string – File description (optional for users)

IsArchive: string – Is file an archive (1 – archive, 0 – not archive)

Category: string – Category of file

Date: string – last file change date

DateUploaded: integer - The date the file was uploaded.

DateAccessed: string - The date the file was accessed. 0 if it never was accessed.

DirectLinkPublick: string – link for public review of a file

EmbedLink: string – File embed link

AccessDisabled: integer – 1 – a file is prohibited for public charing, only authorized users can work with a file. 0 – no restrictions

Type: string - deprecated

DestURL: string – deprecated

Owner: string – file owner's ID

AccessUser: string – the existence of users who have an access to this file

DirUpdateTime: integer - The directory update time.

FileName: string – File name.

FileDate: string – File creating date

FileDescription: string – File description.
FileDestUrl: string - Deprecated
FileKey: string – File Key
FilePrice: string – File price.
FileVersion: string – File version
FileIp: string – File IP
FileIsPublic: string – Permission for public access to a file
Datetime: string

Errors:

400: Bad Request: Invalid file ID
401: Unauthorized: Session has expired. Please right click on OpenDrive task bar icon, log out and then log back in.
404: Not Found: File was not found

12.3 Create file (create_file.json)

Description: Create file should be called before file upload if file_size and file_hash present, RequireHashOnly can be returned, that means this file with same hash and size already exists in db.

Role: 1 and 2

URL Structure: /upload/create_file.json

Method: POST

Parameters:

session_id: string (required) - Session ID.
folder_id: string (required) - Folder ID.
file_name: string (required) - File Name.
file_description: string - File description (Optional).
access_folder_id: string - Access folder ID.
file_size: integer - File Size in bytes (Optional).
file_hash: string - MD5 File hash (Optional).
sharing_id: string - Sharing ID.
open_if_exists: integer - (1) - file info will be returned if file already exists, (0) - error 409 will be returned if file already exists.

Returns:

FileId: string - File's ID.
Name: string - File's name.
GroupID: string – Group ID for files which has several versions
Extension: string – File format
Size: string - The file size in bytes.

Views: string – File reviews number
Version: string – Actual file version
Downloads: string – File downloads number
Access: string -
Link: string – A link to review or edit a file

DownloadLink: string - The files download link.
StreamingLink: string - The files streaming link.
DirUpdateTime: integer - The directory update time.
TempLocation: string – Location for uploading large files
SpeedLimit: integer – File downloading speed limit
RequireCompression: integer – A file should be compressed by Zlib algorithm (level=6)
RequireHash: integer – MD5 hash should be calculated for file upon calling close_file_upload
RequireHashOnly: integer – If upon calling this function there has been file_hash specified with this file_hash and file_size is existing in the DB, we can call close_file_upload with empty value right away

Errors:

400: Bad Request: Invalid file ID

401: Unauthorized: Session has expired. Please right click on OpenDrive task bar icon, log out and then log back in.

Notes: Create file should be called before upload.

12.4 Open a file for upload (open_file_upload.json)

Description: Open file upload should be called before file upload.

Role: 1 and 2

URL Structure: /upload/open_file_upload.json

Method: POST

Parameters:

session_id: string (required)- Session ID.

file_id: string (required) - File ID.

file_size: integer (required)- File Size.

access_folder_id: string - Access folder ID.

file_hash: string - MD5 File hash.

sharing_id: string - Sharing ID.

Returns:

TempLocation: string – location for uploading large files

RequireCompression: integer – A file should be compressed by Zlib algorithm(level=6)

RequireHash: integer – MD5 hash should be calculated for file upon calling close_file_upload

RequireHashOnly: integer – If upon calling this function there has been file_hash specified with this file_hash and file_size is existing in the DB, we can call close_file_upload with empty value right away

SpeedLimit: integer – File downloading speed limit

Errors:

400: Bad Request: Invalid file ID

401: Unauthorized: Session has expired. Please right click on OpenDrive task bar icon, log out and then log back in.

Notes: Open file upload should be called before file upload.

12.5 Upload file chunk (upload_file_chunk.json)

Description: Attention! It is recommended to use the second version of this function (upload_file_chunk2.json).

Role: 1 and 2

URL Structure: /upload/upload_file_chunk.json

Method: POST

Parameters:

this function expects \$_POST variables:

```
$postData = array(  
    "session_id" => $session_id,  
    "file_id" => $file_id,  
    "temp_location" => $temp_location,  
    "chunk_offset" => $chunk_offset,  
    "chunk_size" => $chunk_size,  
);  
along with file sent like:  
$_FILES['file_data']
```

Returns:

TotalWritten: Integer. File size in bytes.

Errors:

400: Bad Request: Session_id field missing

400: Bad Request: Invalid file ID

401: Unauthorized: Session has expired. Please right click on OpenDrive task bar icon, log out and then log back in.

404: Not Found: Could not open chunk on server.

12.6 Upload file chunk. Second version (upload_file_chunk2.json)

Description: Same as the previous version, but all parameters except 'file_data' in the query line. This function works more stable than the previous version if the user has a upload speed limit.

Role: 1 and 2

URL Structure: /upload/upload_file_chunk2.json/{session_id}/{file_id}

Method: POST

Parameters:

This function expects binary data in \$_FILES['file_data']

Returns:

TotalWritten: Integer. File size in bytes.

Errors:

400: Bad Request: Session_id field missing

400: Bad Request: Invalid file ID

401: Unauthorized: Session has expired. Please right click on OpenDrive task bar icon, log out and then log back in.

404: Not Found: Could not open chunk on server.

13. Users

13.1 Users info (info.json)

Description: Get users info.

Role: 1 and 2

URL Structure: /users/info.json/{session_id}

Method: GET

Parameters:

session_id: string (required) - Session ID

apply_bw: integer – Apply BW.

branding: integer – Include user branding.

Returns:

UserID: integer – User ID.

AccessUserID: integer – Access user ID.

UserName: string – Username.

UserFirstName: string – First name of the user.

UserLastName: string – Last name of the user.

PrivateKey: string – User's private key.

Trial: string – Trial or paid account

UserSince: string – Date and time (YYYY-MM-DD HH:MM:SS).

BwResetLast: string – Last Bandwidth reset date

AccType: string – Account type.

MaxStorage: string – User's disk size

StorageUsed: string – Storage size in bytes.

BwMax: string – Bandwidth size in bytes.

BwUsed: string – The size of daily downloaded data

FVersioning: string – File Versioning

FVersions: string – Number of file's version

DailyStat: integer – Daily user activity stats

UserLang: string – User language

MaxFileSize: string – Max file size available for uploading by users

Level: string – User account type

UserPlan: string – User tariff plan

TimeZone: string – User timezone

MaxAccountUsers: string – Number of account users available for adding by user

IsAccountUser: integer – Marker of account user

CompanyName: string - User company name

Email: string – User's email

Phone: string – User's phone number

Avatar: string – Link to user's avatar

AvatarColor: string – Color of user's avatar

AdminMode: integer – Admin functions access

DueDate: string – Trial period ending date

WebLink: string – User's web link

PublicProfiles: integer – Private marker of a user

RootFolderPermission: integer – Default access rights to all folders created by user

CanChangePwd: integer – An ability to change password
IsPartner: integer – Partner marker of a user
SupportUrl: string – Support page's URL
PartnerUsersDomain: string – User partner domain
Suspended: string – Access to the account has been suspended

Errors:

401: Unauthorized: Session has expired. Please right click on OpenDrive task bar icon, log out and then log back in.

13.2 User logs (userlogs.json)

Description: Get user logs.

Role: 1 and 2

URL Structure: /users/userlogs.json/{session_id}

Method: GET

Parameters:

session_id: string (required) - Session ID
access_user_id: string – Access user id (0 for account owner, " empty for all).
start_date: integer - Start date (Unix timestamp).
end_date: integer - End date (Unix timestamp).
log_type: integer - Log Type ID.
page: integer - Page (from 0).

Returns:

TotalPages: integer – number of pages

CurrentPage: integer – current page

Logs: {

Time: integer – user action's date

User: string – a name of a user who executed the action

LogType: string – a type of an action a user executed

IP: string – user IP

FileName: string – a name of a file an action has been executed on

FileSize: string - a size of a file an action has been executed on

Details: string – action details

App: string – the app used by user

}

NumResults: integer – number of logs

Errors:

401: Unauthorized: Session has expired. Please right click on OpenDrive task bar icon, log out and then log back in.

13.3 Forgot password (forgotpassword.json)

Description: Reset a password

Role: 1 and 2

URL Structure: /users/forgotpassword.json

Method: POST

Parameters:

username : string (required) - User name (min 4)(max 100).

Returns:

result: true or false

Errors:

401: Unauthorized: Session has expired. Please right click on OpenDrive task bar icon, log out and then log back in.

404: Not Found: User not exists.

13.4 Verify an email (verifyemail.json)

Description: Verify an email.

Role: 1 and 2

URL Structure: /users/verifyemail.json

Method: POST

Parameters:

session_id: string (required) - Session ID.

ver_code: string (required)- verification code (max 32).

Returns:

True (200 OK) or error

Errors:

401: Unauthorized: Session has expired. Please right click on OpenDrive task bar icon, log out and then log back in.

13.5 Verify an email after signup (verifyemailsignup.json)

Description: Verify an email after signup.

Role: 1 and 2

URL Structure: /users/verifyemailsignup.json

Method: POST

Parameters:

session_id: string (required) - Session ID.

ver_code: string (required)- verification code (max 32).

Returns:

True (200 OK) or error

Errors:

401: Unauthorized: Session has expired. Please right click on OpenDrive task bar icon, log out and then log back in.

13.6 Update an email (email.json)

Description: Update an email. Email updated with verification code.

Role: 1 and 2

URL Structure: /users/email.json

Method: PUT

Parameters:

session_id: string (required) - Session ID.

email: string (required) - Valid email format required (max 255).

Returns:

result: True or Error

Errors:

401: Unauthorized: Session has expired. Please right click on OpenDrive task bar icon, log out and then log back in.

403: Forbidden: Email already taken

13.7 Update user info (info.json)

Description: Update user info.

Role: 1 and 2

URL Structure: /users/info.json

Method: PUT

Parameters:

session_id: string (required) - Session ID.

first_name: string - First Name (min 2) (max 50).

last_name: string- Last Name (min 2) (max 50).

company_name: string- Company Name (max 255).

phone: string- Phone (max 20).

time_zone: string- Time Zone database version 2013.6.

file_versioning: integer - File Versioning (0 = off, 1=on).

file_versions: integer - File Versions (max 99).

lang: string- (ISO 2 Letter Language Codes).

daily_stat: integer - Daily Stats (0 = off, 1=on).

default_file_permission: stringstring- Default file permission in root folder (0=Private, 1=Public, 2=Hidden).

address_1: string - Address line 1.

address_2: string - Address line 2.

Returns:

result: True or Error

Errors:

400: Bad Request: Invalid value specified for `value`.

401: Unauthorized: Session has expired. Please right click on OpenDrive task bar icon, log out and then log back in.

403: Forbidden: Email already taken

13.8 Update user password (password.json)

Description: Update a password.

Role: 1 and 2

URL Structure: /users/password.json

Method: PUT

Parameters:

session_id: string (required) - Session ID.

newpassword: string (required) - New password (min 5).

oldpassword: string (required) - Old password (min 5).

Returns:

result: True or Error

Errors:

400: Bad Request: Old Password is incorrect.

400: Bad Request: Invalid value specified for `value`.

401: Unauthorized: Session has expired. Please right click on OpenDrive task bar icon, log out and then log back in.

13.9 Update user name (username.json)

Description: Update a username.

Role: 1 and 2

URL Structure: /users/username.json

Method: PUT

Parameters:

session_id: string (required) - Session ID.

username: string (required) - User name (min 4)(max 100).

Returns:

result: True or Error

Errors:

400: Bad Request: Invalid value specified for `username`. Minimum 4 characters required.

401: Unauthorized: Session has expired. Please right click on OpenDrive task bar icon, log out and then log back in.

404: Not Found: There is an existing user associated with this email.

14. User Groups API:

14.1 User groups information (usergroups.json)

Description: Get group info.

Role: 1 and 2

URL Structure: /usergroups.json/{session_id}/{group_id}

Method: GET

Parameters:

session_id: string (required) - Session ID maximum of 20 characters.

group_id: integer (required) – Grou ID.

Returns:

GroupID: string - Group ID.

GroupName: string - Group name (max 100).

GroupActive: string - Group status (1 = Active, 0 = inactive)

GroupMaxStorage: string - Storage size in bytes.

GroupMaxBw: string - Bandwidth size in bytes.

GroupSince: string - Date and time (YYYY-MM-DD HH:MM:SS)

NumUsers: integer – number of users in a group

Errors:

400: Bad Request: invalid value specified for `group_id`

401: Unauthorized: Session has expired. Please right click on OpenDrive task bar icon, log out and then log back in.

404: Not Found: User group not exists

14.2 List users groups (all.json)

Description: See a list of all the user groups for an account.

Role: 1 and 2

URL Structure: /usergroups/all.json

Method: GET

Parameters:

session_id: string (required) - Session ID maximum of 20 characters.

Returns:

GroupID: string - Group ID.

GroupName: string - Group name (max 100).

GroupActive: string - Group status (1 = Active, 0 = inactive)

GroupMaxStorage: string - Storage size in bytes.

GroupMaxBw: string - Bandwidth size in bytes.

GroupSince: string - Date and time (YYYY-MM-DD HH:MM:SS)

NumUsers: integer – number of users in a group

Errors:

401: Unauthorized: Session has expired. Please right click on OpenDrive task bar icon, log out and then log back in.

14.3 Create a group (usergroups.json)

Description: Create a user group for an account.

Role: 1 and 2

URL Structure: /usergroups.json

Method: POST

Parameters:

session_id: string (required) - Session ID.

group_name: string (required) - Group name (max 100 characters).

group_bw_max: integer (required) - Storage size in MB.

group_max_storage: integer (required) - Bandwidth size in MB.

Returns:

Group_id: integer – unique group number.

Errors:

400: Bad Request: invalid value specified for `valid`.

401: Unauthorized: Session has expired. Please right click on OpenDrive task bar icon, log out and then log back in.

14.4 Update a group (usergroups.json)

Description: Update a user group's bandwidth or storage amount.

Role: 1 and 2

URL Structure: /usergroups.json

Method: PUT

Parameters:

session_id: string (required) - Session ID.

group_id: string (required) - Group ID.

group_name: string (required) - Group name (max 100).

group_bw_max: string (required) - Storage size in MB.

group_max_storage: string (required) - Bandwidth size in MB.

group_settings_enable: int - Group settings enable (0 = off, 1=on).

Returns:

True (200 OK) or error

Errors:

400: Bad Request: max_bw and max_storage should be greater than acc owner values

404: Not Found: Empty arguments

14.5 Delete a group (usergroups.json)

Description: Delete a user group.

Role: 1 and 2

URL Structure: /usergroups.json/{session_id}/{group_id}

Method: DELETE

Parameters:

session_id: string (required) - Session ID.

group_id: string (required) - Group ID.

Returns:

True (200 OK) or error

Errors:

401: Unauthorized: Session has expired. Please right click on OpenDrive task bar icon, log out and then log back in.

404: Not Found: update failed or nothing changed

15. OAuth2 authentication protocol

15.1 Grant user access (oauth2/grant.json)

Description: OpenDrive is supporting simplified OAuth 2.0 standard for authorization (Resource Owner Password Credentials Flow).

Applications can exchange user credentials (email and password) with access tokens. When user is authorized through OAuth 2.0 scheme applications must not store email and password on user's side.

Pass to API user's email and password and receive tokens

URL Structure: /oauth2/grant.json

Method: POST

Parameters:

grant_type: string (required) – Grant Type. For this stage “password” value should be used

client_id: string (required) - Partner ID. “OpenDrive” is default partner value.

username: string (required) – Username or E-mail.

password: string (required) – User password.

Returns:

True (200 OK) or error

Successful response:

```
{
  "access_token": "f3749e1a7dce6623f5073f6591ad294e066837b7",
  "expires_in": 86400,
  "token_type": "bearer",
  "scope": null,
  "refresh_token": "7414baaf7a335111ccc1b040e24d34dd7b8e81e2"
}
```

Errors:

401: Invalid username or password

Response has two tokens: **access_token** and **refresh_token**. Applications should save these tokens for future API's calls (instead of classic sessions). For all API functions, that accepts **session_id** parameter, the **session_id** must be set to 'OAUTH' value and **access_token** must presents in GET parameters in all calling methods (GET, PUT, POST, DELETE).

For example:

- a) to retrieve the information about user, the following call (GET request) can be used (please see 13.1 Users):

https://dev.opendrive.com/api/v1/users/info.json/OAUTH?access_token=f3749e1a7dce6623f50

[73f6591ad294e066837b7](#)

- b) update the user's phone (PUT request)

http://dev.opendrive.com/api/v1/users/info.json?access_token=f3749e1a7dce6623f5073f6591ad294e066837b7

The request body will be following JSON object:

```
{
  "session_id": "OAUTH",
  "phone": "1234"
}
```

Errors:

If token has expired or is invalid:

```
{
  "error": {
    "code": 401,
    "message": "",
    "error": "invalid_token",
    "error_description": "The access token provided has expired"
  }
}
```

15.2 Retrieve a new access token (oauth2/grant.json)

The **access_token** lifetime is limited to 86400 seconds, and for **refresh_token** lifetime is 30 days. If the app has received 401 error with expired token message, the app must refresh and obtain new **access_token**.

Upon successful call the **refresh_token** will be updated as well.

URL Structure: /oauth2/grant.json

Method: POST

Parameters:

grant_type: string (required) – Grant Type. For this stage “refresh_token” value should be used

client_id: string (required) - Partner ID. “OpenDrive” is default partner value.

refresh_token: string (required) – The refresh token, that was obtained in **15.1**

Returns:

True (200 OK) or error

Successful response:

```
{
  "access_token": "9fccf1b5f09927727b89ce006490bcf0c4038b17",
  "expires_in": 86400,
  "token_type": "bearer",
  "scope": null,
  "refresh_token": "5a63f924b863870a6e020bed754b0ed817937fa6"
}
```

Errors:

If token is invalid:

```
{  
  "error": "invalid_grant",  
  "error_description": "Invalid refresh token"  
}
```

If token has expired:

```
{  
  "error": "invalid_grant",  
  "error_description": "Refresh token has expired"  
}
```

Copyright ©2008-2017 OpenDrive, Inc. All rights reserved.

For information about OpenDrive trademarks, copyrights and patents refer to the OpenDrive website. All other trademarks and registered trademarks are the property of their respective holders.

No part of this document may be reproduced or transmitted in any form or by any means, electronic or mechanical, for any purpose, without the express written permission of OpenDrive, Inc. Under the law, reproducing includes translating into another language or format. Every effort has been made to ensure that the information in this manual is accurate. OpenDrive, Inc. is not responsible for printing or clerical errors. Information in this document is subject to change without notice.

OpenDrive REST API Guide, version 1.1.6, March, 2017.

If you have any comments or feedback on our documentation, please send them to us at: support@opendrive.com.