Synology

Synology File Station

Official API

Synology Inc.
3F-3, No. 106, Chang-An W. Rd. Taipei 103, Taiwan

Table of Contents

Chapter 1

Introduction

This File Station Official API developer's guide explains how to expand your applications based on the APIs of File Station, allowing your applications to interact with files in DSM via HTTP/HTTPS requests and responses.

This document explains the structure and detailed specifications of various File Station APIs. "Chapter 2: Get Started" describes the basic guidelines on how to use these APIs, which we suggest reading all the way through before you jump into the API specifications. "Chapter 3: Base API" and "Chapter 4: File Station API" list all available APIs and related details.

Chapter 2

Get Started

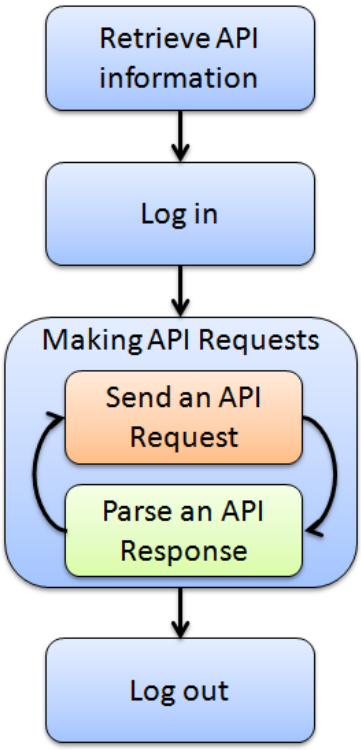
Before making use of File Station APIs to develop your own applications, you need to have basic understanding of API concepts and API procedures.

This chapter explains how to execute and complete API processes in the following five sections:

- **API Workflow**: Briefly introduces how to work with File Station APIs
- **Making Requests**: Elaborates how to construct API requests
- **Parsing Response**: Describes how to parse response data
- **Common Error Code**: Lists all common error codes that might be returned from all File Station APIs
- **Working Example**: Provides an example to request a file operation

API Workflow

The following five-step and easy-to-follow workflow shows how to make your application interact with File Station APIs.



Step 1: Retrieve API Information

First, your application needs to retrieve API information from the target DiskStation to know which APIs are available for use on the target DiskStation. This information can be accessed simply through

a request to /webapi/query.cgi with SYNO.API.Info API parameters. The information provided in the response contains available API name, API method, API path and API version. Once you have all the information at hand, your application can make further requests to all available APIs.

Step 2: Log in

In order to make your application interact with File Station, your application needs to log in with an account and password first. The login process is simply making a request to SYNO.API.Auth API with the `login` method. If successful, the API returns an authorized session ID. You should keep it and pass it in making other API requests.

Step 3: Making API Requests

Once successfully logged in, your application can start to make requests to all available File Station APIs. In the next section “Making Requests,” instructions on how to form a valid API request and how to decode response information will be given.

Step 4: Log out

After finishing with the steps above, your application can end the login session by making another request to SYNO.API.Auth API with the `logout` method.

Making Requests

There are five basic elements that are used to construct a valid request to any API.

- **API name**: Name of the API requested
- **version**: Version of the API requested
- **path**: path of the API. The path information can be retrieved by requesting SYNO.API.Info
- **method**: Method of the API requested
- **_sid**: Authorized session ID. Each API request should pass it, which is retrieved from the response of /webapi/auth.cgi, via either HTTP/HTTPS GET/POST method with “`_sid`” argument. Otherwise, if you pass it within “`id`” value of cookie of HTTP/HTTPS header, this parameter can be ignored.

And the syntax for the request is as follows:

```
GET
/webapi/<CGI_PATH>?api=<API_NAME>&version=<VERSION>&method=<METHOD>[&<PARAMS>][&_si
d=<SID>]
```

Here `<PARAMS>` represents the parameters for the requested method which is optional. Note all parameters need to be escaped. Commas ”,” are replaced by slashes " \", and slashes" \" are replaced by double-slashes "\\", because commas ”,” are used to separate multiple elements in a parameter. Password-relative parameters do not need to be escaped including passwd or password parameter.

Please see the following example. If you want to make a request to the SYNO.API.Info API version 1 with the `query` method on your DiskStation whose address is http://myds.com:port (default port for HTTP is 5000 or 5001 for HTTPS) for the list of all available API methods, the corresponding parameters are:

API name: SYNO.API.Info

version: 1

path: query.cgi

method: query

params: query=all

And the request will look like this:

```
http://myds.com:port/webapi/query.cgi?api=SYNO.API.Info&version=1&method=query&query=all
```

Note that an API's path and supported version information can be acquired by sending a request to SYNO.API.Info. The location of SYNO.API.Info is fixed so that you can always request SYNO.API.Info with /webapi/query.cgi.

Parsing Response

All API responses are encoded in the JSON format, and the JSON response contains elements as follows:

Key	Value	Description
success	true/false	“true”: the request finishes successfully; “false”: the request fails with an error data.
data	<JSON-Style Object>	The data object contains all response information described in each method.
error	<JSON-Style Object>	The data object contains error information when a request fails. The basic elements are described in the next table.

Following describes the format of error information in error element.

Key	Value	Description
code	Error Code	An error code will be returned when a request fails. There are two kinds of error codes: a common error code which is shared between all APIs; the other is a specific API error code (described under the corresponding API spec).
errors	<JSON-Style Array>	The array contains detailed error information of each file. Each element within errors is a JSON-Style Object which contains an error code and other information, such as a file path or name. Note: When there is no detailed information, this error element won’t be responded.

Example 1

Respond an invalid request to get information of File Station without a method parameter

Request:

```
http://myds.com:port/webapi/FilStation/info.cgi?api=SYNO.FileStation.Info&version=1
```

Failed Response:

```
{
	"success":false,
	"error":{
		"code":101
	}
}
```

Example 2

Respond an invalid request with a illegal path to create a folder

Request:

```
http://myds.com:port/webapi/FilStation/info.cgi?
api=SYNO.FileStation.CreateFolder&method=create&version=1&
folder_path=%2Ftest&name=%3A
```

Failed Response:

```
{
	"success":false,
	"error":{
		"code":1100,
		"errors":[{
			"code":418,
			"path":"/test/:"
		}]
	}
}
```

Example 3

Respond a successful request to get information from File Station

Request:

```
http://myds.com:port/webapi/FilStation/info.cgi?api=SYNO.FileStation.Info&version=1
&method=getinfo
```

Success Response:

```
{
	"success":true,
	"data": {
		"is_manager":true,
		"hostname":"DS",
		"support_sharing":true,
		"support_virtual":"cifs,iso"
	}
}
```

Note that to demonstrate examples with clarity, only the data object is included in the response examples given in the following sections.

Common Error Codes

The codes listed below are common error codes of wrong parameters or failed login for all WebAPIs.

Code	Description
100	Unknown error
101	No parameter of API, method or version
102	The requested API does not exist
103	The requested method does not exist
104	The requested version does not support the functionality
105	The logged in session does not have permission
106	Session timeout
107	Session interrupted by duplicate login

The codes listed below are common error codes of file operations for all File Station APIs.

Code	Description
400	Invalid parameter of file operation
401	Unknown error of file operation
402	System is too busy
403	Invalid user does this file operation
404	Invalid group does this file operation
405	Invalid user and group does this file operation
406	Can’t get user/group information from the account server
407	Operation not permitted
408	No such file or directory
409	Non-supported file system
410	Failed to connect internet-based file system (ex: CIFS)
411	Read-only file system
412	Filename too long in the non-encrypted file system
413	Filename too long in the encrypted file system
414	File already exists
415	Disk quota exceeded
416	No space left on device
417	Input/output error
418	Illegal name or path
419	Illegal file name
420	Illegal file name on FAT file system
421	Device or resource busy
599	No such task of the file operation

Working Example

The following demonstrates a working example for requesting a file operation from the DiskStation. To implement this example, simply replace the DiskStation address used in the example (myds.com:port) with your DiskStation address and paste the URL to a browser. Then the JSON response will show up in a response page.

Step 1: Retrieve API Information

In order to make API requests, you should first request to /webapi/query.cgi with SYNO.API.Info to get the SYNO.API.Auth API information for logging in and FileStation API info for file operations.

Request:

```
http://myds.com:port/webapi/query.cgi?api=SYNO.API.Info&version=1&method=query&query=SYNO.API.Auth,SYNO.FileStation.
```

Response:

```
{
	"data":{
		"SYNO.API.Auth": {
			"path": "auth.cgi",
			"minVersion": 1,
			"maxVersion": 3
		},
		"SYNO.FileStation.List": {
			"path": "FileStation/file_share.cgi",
			"minVersion": 1,
			"maxVersion": 1
		},
		…
	},
	"success": true
}
```

Step 2: Login

After the SYNO.API.Auth path and supported version information are returned, you can log in a FileStation session by requesting SYNO.API.Auth API version 3 located at /webapi/auth.cgi.

Request:

```
http://myds.com:port/webapi/auth.cgi?api=SYNO.API.Auth&version=3&method=login&account=admin&passwd=12345&session=FileStation&format=cookie
```

Response:

```
{
       "data":{
              sid: "ohOCjwhHhwghw"
       },
       "success": true
}
```

Step 3: Request a File Station API

After a session is logged in, you can continue to call the method of listing shared folder in SYNO.FileStation.List. The cgi path and version are provided in the response of Step 1, and the list of all tasks can be requested by excluding the `offset` and `limit` parameters.

Request:

```
http://myds.com:port/webapi/FileStation/file_share.cgi?api=SYNO.FileStation.List&ve
rsion=1&method=list_share
```

Response:

```
{
    "data": {
        "offset": 0,
        "shares": [{
            "isdir": true,
            "name": "video",
            "path": "/video"
        },{
            "isdir": true,
            "name": "photo",
            "path": "/photo"
        }],
        "total": 2
    },
    "success": true
}
```

From the response list, it can be observed that there are two shared folders in File Station. Let's say you're interested in the shared folder "photo" and want to know more details about it. In this case, you can make another request to the `getinfo` method. In this request, you will need to add the parameter `additional= real_path,owner,time` for the method to request detailed objects and transfer them in response.

Request:

```
http://myds.com:5000/webapi/FileStation/file_share.cgi?api=SYNO.FileStation.List&ve
rsion=1&method=getinfo&path=%2Fphoto&additional=real_path,owner,time,perm
```

Response:


```
{
    "data": {
        "files": [{
            "additional": {
                "owner": {
                    "gid": 100,
                    "group": "users",
                    "uid": 1024,
                    "user": "admin"
                },
                "real_path": "/volume1/photo",
                "time": {
                    "atime": 1371630215,
                    "crtime": 1352168821,
                    "ctime": 1368769689,
                    "mtime": 1368769689
                }
            },
            "isdir": true,
            "name": "photo",
            "path": "/photo"
        }]
    },
    "success": true
}
```


Step 4: Logout

When finished with the procedure, you should log out of the current session. The session will be ended by calling the `logout` method in SYNO.API.Auth. If you want to log out a specific session, you can pass the `_sid` parameter.

Example:

```
http://myds.com:5000/webapi/auth.cgi?api=SYNO.API.Auth&version=1&method=logout&session=FileStation
```

Chapter 3

Base API

API List

The following table is the overview of two fundamental APIs defined in this chapter:

API Name	Description
SYNO.API.Info	Provide available API info.
SYNO.API.Auth	Perform login and logout.

SYNO.API.Info

Overview

Availability: Since DSM 4.0

Version: 1

Method

Query

Request:

Parameter	Description	Availability
`query`	API names, separated by a comma “,” or use "all" to get all supported APIs.	1 and later

Example:

```
GET /webapi/query.cgi?api=SYNO.API.Info&version=1&method=query&query=all
```

Response:

Contains API description objects.

Parameter	Description	Availability
`key`	API name.	1 and later
`path`	API path.	1 and later
`minVersion`	Minimum supported API version.	1 and later
`maxVersion`	Maximum supported API version.	1 and later

Example:

```
{
	"data":{
		"SYNO.API.Auth": {
			"path": "auth.cgi",
			"minVersion": 1,
```

```
                        "maxVersion": 3
                },
                "SYNO.FileStation.List": {
                        "path": "FileStation/file_share.cgi",
                        "minVersion": 1,
                        "maxVersion": 1
                },
                …
        },
        "success": true
}
```

API Error Code

No specific API error codes.

SYNO.API.Auth

Overview

Availability: Since DSM 4.0

Version: 3 (Since DSM 4.2), 2 (Since DSM 4.1)

Method

Login

Request:

Parameter	Description	Availability
`account`	Login account name.	1 and later
`passwd`	Login account password.	1 and later
`session`	Login session name.	1 and later
`format`	Returned format of session ID. Following are the two possible options and the default value is `cookie`. `cookie`: The login session ID will be set to "id" key in cookie of HTTP/HTTPS header of response. `sid`: The login sid will only be returned as response JSON data and "id" key will not be set in cookie.	2 and later
`otp_code`	Reserved key. DSM 4.2 and later support a 2-step verification option with an OTP code. If it's enabled, the user requires a verification code to log into DSM sessions. However, WebAPI doesn't support it yet.	3 and later

Example:

```
GET
/webapi/auth.cgi?api=SYNO.API.Auth&version=3&method=login&account=admin&passwd=1234
5&session=FileStation&format=cookie
```

Response:

<data> object definitions:

Parameter	Description	Availability
`sid`	Authorized session ID. When the user log in with `format=sid`, cookie will not be set and each API request should provide a request parameter `_sid=<sid>` along with other parameters.	2 and later

Example:

```
{
	sid: "ohOCjwhHhwghw"
}
```

Logout

Request:

Parameter	Description	Availability
session	Session name to be logged out.	1 and later

Example:

```
GET /webapi/auth.cgi?api=SYNO.API.Auth&version=1&method=logout&session=FileStation
```

Response:

No specific response. It returns an empty success response if completed without error.

API Error Code

Code	Description
400	No such account or incorrect password
401	Account disabled
402	Permission denied
403	2-step verification code required
404	Failed to authenticate 2-step verification code

Chapter 4

File Station API

API List

The following table is the overview of all File Station APIs defined in this chapter. All File Station APIs are required to login with SYNO.API.Auth and session=FileStation.

API Name	Description
SYNO.FileStation.Info	Provide File Station info.
SYNO.FileStation.List	List all shared folders, enumerate files in a shared folder, and get detailed file information.
SYNO.FileStation.Search	Search files on given criteria.
SYNO.FileStation.VirtualFolder	List all mount point folders of virtual file system, ex: CIFS or ISO.
SYNO.FileStation.Favorite	Add a folder to user's favorites or do operations on user's favorites.
SYNO.FileStation.Thumb	Get a thumbnail of a file.
SYNO.FileStation.DirSize	Get the total size of files/folders within folder(s).
SYNO.FileStation.MD5	Get MD5 of a file.
SYNO.FileStation.CheckPermission	Check if the file/folder has a permission of a file/folder or not.
SYNO.FileStation.Upload	Upload a file.
SYNO.FileStation.Download	Download files/folders.
SYNO.FileStation.Sharing	Generate a sharing link to share files/folders with other people and perform operations on sharing links.
SYNO.FileStation.CreateFolder	Create folder(s).

API Name	Description
SYNO.FileStation.Rename	Rename a file/folder.
SYNO.FileStation.CopyMove	Copy/Move files/folders.
SYNO.FileStation.Delete	Delete files/folders.
SYNO.FileStation.Extract	Extract an archive and do operations on an archive.
SYNO.FileStation.Compress	Compress files/folders.
SYNO.FileStation.BackgroundTask	Get information regarding tasks of file operations which are run as the background process including copy, move, delete, compress and extract tasks or perform operations on these background tasks.

SYNO.FileStation.Info

Description

Provide File Station information.

Overview

Availability: Since DSM 4.3

Version: 1

Method

getinfo

Description:

Provide File Station information

Request:

No parameters are required

Example:

```
GET /webapi/FileStation/info.cgi?api=SYNO.FileStation.Info&version=1&method=getinfo
```

Response:

<data> object definitions:

Parameter	Type	Description	Availability
`is_manager`	Boolean	If the logged-in user is an administrator.	1 and later
`support_virtual`	String	Types of virtual file system which the logged user is able to mount on. DSM 4.3 supports CIFS and ISO of virtual file system. Different types are separated with a comma, for example: cifs,iso.	1 and later
`support_sharing`	Boolean	If the logged-in user can sharing file(s)/folder(s) or not.	1 and later
`hostname`	String	DSM host name.	1 and later

Example:

```
{
        "hostname": "test",
        "is_manager": true,
        "support_sharing": true,
        "support_virtual": "cifs,iso"
}
```

API Error Code

No specific API error codes

SYNO.FileStation.List

Description

List all shared folders, enumerate files in a shared folder, and get detailed file information.

Overview

Availability: Since DSM 4.3

Version: 1

Method

list_share

Description:

List all shared folders.

Availability:

Since version 1

Parameter	Description	Value	Default Value	Availability
`offset`	Optional. Specify how many shared folders are skipped before beginning to return listed shared folders.	Integer	0	1 and later
`limit`	Optional. Number of shared folders requested. 0 lists all shared folders.	Integer	0	1 and later
`sort_by`	Optional. Specify which file information to sort on. Options include: **name**: file name **user**: file owner **group**: file group **mtime**: last modified time **atime**: last access time **ctime**: last change time **crtime**: create time **posix**: POSIX permission	name, user, group, mtime, atime, ctime, crtime or posix	name	1 and later
`sort_direction`	Optional. Specify to sort ascending or to sort descending. Options include: **asc**: sort ascending **desc**: sort descending	asc or desc	asc	1 and later
`onlywritable`	Optional. “true”: List writable shared folders; “false”: List writable and read-only shared folders.	true or false	false	1 and later

Parameter	Description	Value	Default Value	Availability
additional	Optional. Additional requested file information, separated by commas ",". When an additional option is requested, responded objects will be provided in the specified additional option. Options include: ■ **real_path**: return a real path in volume ■ **size**: return file byte size ■ **owner**: return information about file owner including user name, group name, UID and GID ■ **time**: return information about time including last access time, last modified time, last change time and create time ■ **perm**: return information about file permission ■ **mount_point_type**: return a type of a virtual file system of a mount point ■ **volume_status**: return volume statuses including free space, total space and read-only status	real_path, owner, time, perm, mount_point_type, sync_share, or volume_status	(None)	1 and later

Example:

```
GET
/webapi/FileStation/file_share.cgi?api=SYNO.FileStation.List&version=1&method=list_
share&additional=real_path%2Cowner%2Ctime
```

Response:

<data> object definitions:

Parameter	Type	Description	Availability
total	Integer	Total number of shared folders.	1 and later
offset	Integer	Requested offset.	1 and later
shares	JSON-Style Array	Array of <shared folder> objects.	1 and later

<shared folder> object definition:

Parameter	Type	Description	Availability
path	String	Path of a shared folder.	1 and later
name	String	Name of a shared folder.	1 and later
additional	<shared-folder additional> object	Shared-folder additional object.	1 and later

<shared-folder additional> object definition:

Parameter	Type	Description	Availability
real_path	String	Real path of a shared folder in a volume space.	1 and later
owner	<owner> object	File owner information including user name, group name, UID and GID.	1 and later
time	<time> object	Time information of file including last access time, last modified time, last change time, and creation time.	1 and later
perm	<shared-folder perm> object	File permission information.	1 and later
mount_point_type	String	Type of a virtual file system of a mount point.	1 and later
volume_status	<volume_status> object	Volume status including free space, total space and read-only status.	1 and later

<owner> object definition:

Parameter	Type	Description	Availability
user	String	User name of file owner.	1 and later
group	String	Group name of file group.	1 and later
uid	Integer	File UID.	1 and later
gid	Integer	File GID.	1 and later

<time> object definition:

Parameter	Type	Description	Availability
atime	Linux timestamp in second	Linux timestamp of last access in second.	1 and later
mtime	Linux timestamp in second	Linux timestamp of last modification in second.	1 and later
ctime	Linux timestamp in second	Linux timestamp of last change in second.	1 and later
crtime	Linux timestamp in second	Linux timestamp of create time in second.	1 and later

Note: Linux timestamp in second, defined as the number of seconds that have elapsed since 00:00:00 Coordinated Universal Time (UTC), Thursday, 1 January 1970.

<shared-folder perm> object definition:

Parameter	Type	Description	Availability
share_right	String	“RW: The shared folder is writable; “RO”: the shared folder is read-only.	1 and later
posix	Integer	POSIX file permission, For example, 777 means owner, group or other has all permission; 764 means owner has all permission, group has read/write permission, other has read permission.	1 and later
adv_right	<adv_right> object	Specail privelge of the shared folder.	1 and later
acl_enable	Boolean	If the configure of Windows ACL privilege of the shared folder is enabled or not.	1 and later
is_acl_mode	Boolean	“true”: The privilege of the shared folder is set to be ACL-mode. “false”: The privilege of the shared folder is set to be POSIX-mode.	1 and later
acl	<acl> object	Windows ACL privilege. If a shared folder is set to be POSIX-mode, these values of Windows ACL privileges are derived from the POSIX privilege.	1 and later

<adv_right> object definition:

Parameter	Type	Description	Availability
disable_download	Boolean	If a non-administrator user can download files in this shared folder through SYNO.FileStation.Download API or not.	1 and later
disable_list	Boolean	If a non-administrator user can enumerate files in this shared folder though SYNO.FileStation.List API with `list` method or not.	1 and later
disable_modify	Boolean	If a non-administrator user can modify or overwrite files in this shared folder or not.	1 and later

<acl> object definition:

Parameter	Type	Description	Availability
append	Boolean	If a logged-in user has a privilege to append data or create folders within this folder or not.	1 and later
del	Boolean	If a logged-in user has a privilege to delete a file/a folder within this folder or not.	1 and later
exec	Boolean	If a logged-in user has a privilege to execute files/traverse folders within this folder or not.	1 and later
read	Boolean	If a logged-in user has a privilege to read data or list folder within this folder or not.	1 and later
write	Boolean	If a logged-in user has a privilege to write data or create files within this folder or not.	1 and later

<volume_status> object definition:

Parameter	Type	Description	Availability
freespace	Integer	Byte size of free space of a volume where a shared folder is located.	1 and later
totalspace	Integer	Byte size of total space of a volume where a shared folder is located.	1 and later
readonly	Boolean	“true”: A volume where a shared folder is located is read-only; “false”: It’s writable.	1 and later

Example:

```
{
	"shares": [{
		"isdir": true,
		"name": "video",
		"path": "/video",
		"additional": {
			"owner": {
				"gid": 100,
				"group": "users",
				"uid": 1024,
				"user": "admin"
			},
			"real_path": "/volume1/video",
			"time": {
				"atime": 1374918626,
				"crtime": 1363259974,
				"ctime": 1371713685,
				"mtime": 1371713685
			}
		}
	}, {
		"isdir": true,
		"name": "photo",
		"path": "/photo",
		"additional": {
			"owner": {
				"gid": 100,
```

```
                                "group": "users",
                                "uid": 1024,
                                "user": "admin"
                        },
                        "real_path": "/volume1/photo",
                        "time": {
                                "atime": 1371630215,
                                "crtime": 1352168821,
                                "ctime": 1368769689,
                                "mtime": 1368769689
                        }
                }
        }],
        "offset": 0,
        "total": 2
}
```


list

Description:
Enumerate files in a given folder

Availability:
Since version 1

Request:

Parameter	Description	Value	Default Value	Availability
folder_path	A listed folder path started with a shared folder.	String	(None)	1 and later
offset	Optional. Specify how many files are skipped before beginning to return listed files.	Integer	0	1 and later
limit	Optional. Number of files requested. 0 indicates to list all files with a given folder.	Integer	0	1 and later
sort_by	Optional. Specify which file information to sort on. Options include: **name**: file name **size**: file size **user**: file owner **group**: file group **mtime**: last modified time	name, size, user, group, mtime, atime, ctime, crtime, posix or type	name	1 and later

Parameter	Description	Value	Default Value	Availability
	atime: last access time **ctime**: last change time **crtime**: create time **posix**: POSIX permission **type**: file extension			
`sort_direction`	Optional. Specify to sort ascending or to sort descending Options include: **asc**: sort ascending **desc**: sort descending	asc or desc	asc	1 and later
`pattern`	Optional. Given glob pattern(s) to find files whose names and extensions match a case-insensitive glob pattern. Note: 1. If the pattern doesn't contain any glob syntax (? and *), * of glob syntax will be added at begin and end of the string automatically for partially matching the pattern. 2. You can use "," to separate multiple glob patterns.	Glob patterns	(None)	1 and later
`filetype`	Optional. "file": only enumerate regular files; "dir": only enumerate folders; "all" enumerate regular files and folders	file, dir or all	all	1 and later
`goto_path`	Optional. Folder path started with a shared folder. Return all files and sub-folders within `folder_path` path until `goto_path` path recursively.	String	(None)	1 and later
`additional`	Optional. Additional requested file information, separated by a comma, ",". When an additional option is requested, responded objects will be provided in the specified additional option. Options include: ■ **real_path**: return a real path in volume ■ **size**: return file byte size ■ **owner**: return information about file owner including user name, group name, UID and GID ■ **time**: return information about time including last access time, last modified time, last change time and create time ■ **perm**: return information about file permission	real_path, size, owner, time, perm, type or mount_point_type	(None)	1 and later

Parameter	Description	Value	Default Value	Availability
	■ **mount_point_type**: return a type of a virtual file system of a mount point ■ **type**: return a file extension			

Example:

```
GET
/webapi/FileStation/file_share.cgi?api=SYNO.FileStation.List&version=1&method=list&
additional=real_path%2Csize%2Cowner%2Ctime%2Cperm%2Ctype&folder_path=%2Fvideo
```

Response:

<data> object definitions:

Parameter	Type	Description	Availability
total	Integer	Total number of files.	1 and later
offset	Integer	Requested offset.	1 and later
files	JSON-Style Array	Array of <file> objects.	1 and later

<file> object definition:

Parameter	Type	Description	Availability
path	String	Folder/file path started with a shared folder.	1 and later
name	String	File name.	1 and later
isdir	Boolean	If this file is folder or not.	1 and later
children	<children> object	File list within a folder which is described by a <file> object. The value is returned, only if goto_path parameter is given.	1 and later
additional	<file additional> object	File additional object.	1 and later

<children> object definition:

Parameter	Type	Description	Availability
total	Integer	Total number of files.	1 and later
offset	Integer	Requested offset.	1 and later
files	JSON-Style Array	Array of <file> objects.	1 and later

<file additional> object definition:

Parameter	Type	Description	Availability
real_path	String	Real path started with a volume path.	1 and later
size	Integer	File size.	1 and later
owner	<owner>	File owner information including user name, group	1 and later

Parameter	Type	Description	Availability
	object	name, UID and GID.	
time	<time> object	Time information of file including last access time, last modified time, last change time and create time.	1 and later
perm	<perm> object	File permission information.	1 and later
mount_point_type	String	A type of a virtual file system of a mount point.	1 and later
type	String	File extension.	1 and later

<owner> object definition:

Parameter	Type	Description	Availability
user	String	User name of file owner.	1 and later
group	String	Group name of file group.	1 and later
uid	Integer	File UID.	1 and later
gid	Integer	File GID.	1 and later

<time> object definition:

Parameter	Type	Description	Availability
atime	Linux timestamp in second	Linux timestamp of last access in second.	1 and later
mtime	Linux timestamp in second	Linux timestamp of last modification in second.	1 and later
ctime	Linux timestamp in second	Linux timestamp of last change in second.	1 and later
crtime	Linux timestamp in second	Linux timestamp of create time in second.	1 and later

Note: Linux timestamp, defined as the number of seconds that have elapsed since 00:00:00

Coordinated Universal Time (UTC), Thursday, 1 January 1970.

<perm> object definition:

Parameter	Type	Description	Availability
posix	Integer	POSIX file permission. For example, 777 means owner, group or other has all permission; 764 means owner has all permission, group has read/write permission, other has read permission.	1 and later
is_acl_mode	Boolean	“true”: the privilege of the shared folder is set to be ACL-mode; “false”: the privilege of the shared folder is set to be POSIX-mode.	1 and later

Parameter	Type	Description	Availability
acl	Object	Windows ACL privilege. If a file is set to be POSIX-mode, these values of Windows ACL privilege are derived from the POSIX privilege.	1 and later

<acl> object definition:

Parameter	Type	Description	Availability
append	Boolean	If a logged-in user has a privilege to append data or create folders within this folder or not.	1 and later
del	Boolean	If a logged-in user has a privilege to delete a file/a folder within this folder or not.	1 and later
exec	Boolean	If a logged-in user has a privilege to execute files or traverse folders within this folder or not.	1 and later
read	Boolean	If a logged-in user has a privilege to read data or list folder within this folder or not.	1 and later
write	Boolean	If a logged-in user has a privilege to write data or create files within this folder or not.	1 and later

Example:


```
{
	"files": [{
		"additional": {
			"owner": {
				"gid": 100,
				"group": "users",
				"uid": 1024,
				"user": "admin"
			},
			"perm": {
				"acl": {
					"append": true,
					"del": true,
					"exec": true,
					"read": true,
					"write": true
				},
				"is_acl_mode": false,
				"posix": 777
			},
			"real_path": "/volume1/video/1",
```

```
                        "size": 4096,
                        "time": {
                                "atime": 1370104559,
                                "crtime": 1370104559,
                                "ctime": 1370104559,
                                "mtime": 1369728913
                        },
                        "type": ""
                },
                "isdir": true,
                "name": "1",
                "path": "/video/1"
        }, {
                "additional": {
                        "owner": {
                                "gid": 100,
                                "group": "users",
                                "uid": 1024,
                                "user": "admin"
                        },
                        "perm": {
                                "acl": {
                                        "append": true,
                                        "del": true,
                                        "exec": true,
                                        "read": true,
                                        "write": true
                                },
                                "is_acl_mode": false,
                                "posix": 777
                        },
                        "real_path": "/volume1/video/2.txt",
                        "size": 12800,
                        "time": {
                                "atime": 1369964337,
                                "crtime": 1369964337,
```

```
                                "ctime": 1372410504,
                                "mtime": 1369964408
                            },
                            "type": "TXT"
                        },
                        "isdir": false,
                        "name": "2.txt",
                        "path": "/video/2.txt"
                }],
                "offset": 0,
                "total": 2
        }
```


getinfo

Description:
Get information of file(s)

Availability:
Since version 1

Request:

Parameter	Description	Value	Default Value	Availability
path	One or more folder/file path(s) started with a shared folder, separated by a comma, “,”.	String	(None)	1 and later
additional	Optional. Additional requested file information, separated by a comma, “,”. When an additional option is requested, responded objects will be provided in the specified additional option. Options include: ■ **real_path**: return a real path in volume ■ **size**: return file byte size ■ **owner**: return information about file owner including user name, group name, UID and GID ■ **time**: return information about time including last access time, last modified time, last change time and create time ■ **perm**: return information about file permission ■ **mount_point_type**: return a type of a virtual file system of a	real_path, size, owner, time, perm, type or mount_point_type	(None)	1 and later

Parameter	Description	Value	Default Value	Availability
	mount point ■ **type**: return a file extension			

Example:

```
GET
/webapi/FileStation/file_share.cgi?api=SYNO.FileStation.List&version=1&method=getinfo&additional=real_path%2Csize%2Cowner%2Ctime%2Cperm%2Ctype&path=%2Fvideo%2F1%2C%2Fvideo%2F2.txt
```

Response

<data> object definitions:

Parameter	Type	Description	Availability
`files`	JSON-Style Array	Array of <file> objects.	1 and later

<file> object definition:

Parameter	Type	Description	Availability
`path`	String	Folder/file path started with a shared folder.	1 and later
`name`	String	File name.	1 and later
`isdir`	Boolean	If this file is folder or not.	1 and later
`additional`	<file additional> object	File additional object.	1 and later

<file additional> object definition: Same as definition in the `list` method.

Example:

```
{
	"files": [{
		"additional": {
			"owner": {
				"gid": 100,
				"group": "users",
				"uid": 1024,
				"user": "admin"
			},
			"perm": {
				"acl": {
					"append": true,
					"del": true,
```

```
                                    "exec": true,
                                    "read": true,
                                    "write": true
                             },
                             "is_acl_mode": false,
                             "posix": 777
                      },
                      "real_path": "/volume1/video/1",
                      "size": 4096,
                      "time": {
                             "atime": 1370104559,
                             "crtime": 1370104559,
                             "ctime": 1370104559,
                             "mtime": 1369728913
                      },
                      "type": ""
               },
               "isdir": true,
               "name": "1",
               "path": "/video/1"
        }, {
               "additional": {
                      "owner": {
                             "gid": 100,
                             "group": "users",
                             "uid": 1024,
                             "user": "admin"
                      },
                      "perm": {
                             "acl": {
                                    "append": true,
                                    "del": true,
                                    "exec": true,
                                    "read": true,
                                    "write": true
                             },
```

```
                                "is_acl_mode": false,
                                "posix": 777
                        },
                        "real_path": "/volume1/video/2.txt",
                        "size": 12800,
                        "time": {
                                "atime": 1369964337,
                                "crtime": 1369964337,
                                "ctime": 1372410504,
                                "mtime": 1369964408
                        },
                        "type": "TXT"
                },
                "isdir": false,
                "name": "2.txt",
                "path": "/video/2.txt"
        }]
}
```


API Error Code

No specific API error codes.

SYNO.FileStation.Search

Description

Search files according to given criteria.

This is a non-blocking API. You need to start to search files with the `start` method. Then, you should poll requests with `list` method to get more information, or make a request with the `stop` method to cancel the operation. Otherwise, search results are stored in a search temporary database so you need to call `clean` method to delete it at the end of operation.

Overview

Availability: Since DSM 4.3

Version: 1

Method

start

Description:
Start to search files according to given criteria. If more than one criterion is given in different parameters, searched files match all these criteria.

Availability:
Since version 1

Request:

Parameter	Description	Value	Default Value	Availability
folder_path	A searched folder path starting with a shared folder.	String	(None)	1 and later
recursive	Optional. If searching files within a folder and subfolders recursively or not.	Boolean	true	1 and later
pattern	Optional. Search for files whose names and extensions match a case-insensitive glob pattern. Note: 1. If the pattern doesn't contain any glob syntax (? and *), * of glob syntax will be added at begin and end of the string automatically for partially matching the pattern. 2. You can use "," to separate multiple glob patterns.	Glob patterns	(None)	1 and later
extension	Optional. Search for files whose extensions match a file type pattern in a case-insensitive glob pattern. If you give this criterion, folders aren't matched. Note: You can use commas "," to separate multiple glob patterns.	Glob patterns	(None)	1 and later
filetype	Optional. "file": enumerate regular files; "dir": enumerate folders; "all" enumerate regular files and folders.	file,dir or all	all	1 and later
size_from	Optional. Search for files whose sizes are greater than the given byte size.	Byte size	(None)	1 and later
size_to	Optional. Search for files whose sizes are less than the given byte size.	Byte size	(None)	1 and later
mtime_from	Optional. Search for files whose last modified time after the given Linux timestamp in second.	Linux timestamp in second	(None)	1 and later

Parameter	Description	Value	Default Value	Availability
mtime_to	Optional. Search for files whose last modified time before the given Linux timestamp in second.	Linux timestamp in second	(None)	1 and later
crtime_from	Optional. Search for files whose create time after the given Linux timestamp in second.	Linux timestamp in second	(None)	1 and later
crtime_to	Optional. Search for files whose create time before the given Linux timestamp in second.	Linux timestamp in second	(None)	1 and later
atime_from	Optional. Search for files whose last access time after the given Linux timestamp in second.	Linux timestamp in second	(None)	1 and later
atime_to	Optional. Search for files whose last access time before the given Linux timestamp in second.	Linux timestamp in second	(None)	1 and later
owner	Optional. Search for files whose user name matches this criterion. This criterion is case-insensitive.	String	(None)	1 and later
group	Optional. Search for files whose group name matches this criterion. This criterion is case-insensitive.	String	(None)	1 and later

Note: Linux timestamp in second, defined as the number of seconds that have elapsed since 00:00:00 Coordinated Universal Time (UTC), Thursday, 1 January 1970.

Example:

```
GET
/webapi/FileStation/file_find.cgi?api=SYNO.FileStation.Search&version=1&method=star
t&folder_path=%2Fvideo&pattern=1
```

Response:

<data> object definitions:

Parameter	Type	Description	Availability
taskid	String	A unique ID for the search task	1 and later

Example:

```
{
        "taskid": "51CE617CF57B24E5"
}
```

list

Description:

List matched files in a search temporary database. You can check the finished value in response to know if the search operation is processing or has been finished.

Availability:

Since version 1

Request:

Parameter	Description	Value	Default Value	Availability
taskid	A unique ID for the search task which is gotten from start method.	String	(None)	1 and later
offset	Optional. Specify how many matched files are skipped before beginning to return listed matched files.	Integer	0	1 and later
limit	Optional. Number of matched files requested. -1 indicates to list all matched files. 0 indicates to list nothing.	Integer	0	1 and later
sort_by	Optional. Specify which file information to sort on. Options include: **name**: file name **size**: file size **user**: file owner **group**: file group **mtime**: last modified time **atime**: last access time **ctime**: last change time	name, size, user, group, mtime, atime, ctime, crtime, posix or type	name	1 and later

Parameter	Description	Value	Default Value	Availability
	crtime: create time **posix**: POSIX permission **type**: file extension			
sort_direction	Optional. Specify to sort ascending or to sort descending. Options include: **asc**: sort ascending **desc**: sort descending	asc or desc	asc	1 and later
pattern	Optional. Given glob pattern(s) to find files whose names and extensions match a case-insensitive glob pattern. Note: 1. If the pattern doesn't contain any glob syntax (? and *), * of glob syntax will be added at begin and end of the string automatically for partially matching the pattern. 2. You can use"," to separate multiple glob patterns.	Glob patterns	String	1 and later
filetype	Optional. "file": enumerate regular files; "dir": enumerate folders; "all" enumerate regular files and folders.	file,dir or all	all	1 and later
additional	Optional. Additional requested file information, separated by a comma, ",". When an additional option is requested, responded objects will be provided in the specified additional option. Options include: ■ **real_path**: return a real path in volume ■ **size**: return file byte size ■ **owner**: returns information about file owner including user name, group name, UID and GID ■ **time**: return information about time including last access time, last modified time, last change time and create time ■ **perm**: return information about file permission ■ **type**: return a file extension	real_path, size, owner, time, perm or type	(None)	1 and later

Example:

```
GET
/webapi/FileStation/file_find.cgi?api=SYNO.FileStation.Search&version=1&method=list
&taskid=51CE617CF57B24E5&additional=real_path%2Csize%2Cowner%2Ctime%2Cperm%2Ctype&l
```

```
imit=-1
```

Response:

<data> object definitions:

Parameter	Type	Description	Availability
total	Integer	Total number of matched files.	1 and later
offset	Integer	Requested offset.	1 and later
finished	Boolean	If the searching task is finished or not.	1 and later
files	JSON-Style Array	Array of <file> objects.	1 and later

<file> object definitions:

Same as definition in the `list` method of SYNO.FileStation.List API

Example:

```
{
	"files": [{
		"additional": {
			"owner": {
				"gid": 100,
				"group": "users",
				"uid": 1024,
				"user": "admin"
			},
			"perm": {
				"acl": {
					"append": true,
					"del": true,
					"exec": true,
					"read": true,
					"write": true
				},
				"is_acl_mode": false,
				"posix": 644
			},
			"real_path": "/volume1/video/12",
			"size": 0,
			"time": {
```

```
                        "atime": 1370002059,
                        "crtime": 1370002059,
                        "ctime": 1370002099,
                        "mtime": 1370002059
                    },
                    "type": ""
                },
                "isdir": false,
                "name": "12",
                "path": "/video/12"
            }, {
                "additional": {
                    "owner": {
                        "gid": 100,
                        "group": "users",
                        "uid": 1024,
                        "user": "admin"
                    },
                    "perm": {
                        "acl": {
                            "append": true,
                            "del": true,
                            "exec": true,
                            "read": true,
                            "write": true
                        },
                        "is_acl_mode": true,
                        "posix": 70
                    },
                    "real_path": "/volume1/video/1GFILE.txt",
                    "size": 1073741825,
                    "time": {
                        "atime": 1370522981,
                        "crtime": 1370522690,
                        "ctime": 1370522815,
                        "mtime": 1370522814
```

```
                        },
                        "type": "TXT"
                    },
                    "isdir": false,
                    "name": "1GFILE.txt",
                    "path": "/video/1GFILE.txt"
                }],
                "finished": true,
                "offset": 0,
                "total": 2
        }
```


stop

Description:
Stop the searching task(s). The search temporary database won't be deleted, so it's possible to list the search result using `list` method after stopping it.

Availability:
Since version 1

Request:

Parameter	Description	Value	Default Value	Availability
`taskid`	Unique ID(s) for the search task which are gotten from `start` method. Specify multiple search task IDs by ",".	String	(None)	1 and later

Example:

```
GET
/webapi/FileStation/file_find.cgi?api=SYNO.FileStation.Search&version=1&method=stop
&taskid=51CE617CF57B24E5
```

Response:
No specific response. It returns an empty success response if completed without error.

clean

Description:
Delete search temporary database(s)

Availability:
Since version 1

Request:

Parameter	Description	Value	Default Value	Availability
`taskid`	Unique ID(s) for the search task which are gotten from `start` method. Specify multiple search task IDs by ",".	String	(None)	1 and later

Example:

```
GET
/webapi/FileStation/file_find.cgi?api=SYNO.FileStation.Search&version=1&method=clea
n&taskid=51CE617CF57B24E5
```

Response:

No specific response. It returns an empty success response if completed without error.

SYNO.FileStation.VirtualFolder

Description

List all mount point folders of virtual file system, ex: CIFS or ISO.

Overview

Availability: Since DSM 4.3

Version: 1

Method

list

Description:

List all mount point folders on one given type of virtual file system

Availability:

Since version 1

Request:

Parameter	Description	Value	Default Value	Availability
`type`	A type of virtual file systems, ex: CIFS or ISO.	cifs or iso	(None)	1 and later
`offset`	Optional. Specify how many mount point folders are skipped before beginning to return listed mount point folders in virtual file system.	Integer	0	1 and later
`limit`	Optional. Number of mount point folders requested. 0 indicates to list all mount point folders in virtual file system.	Integer	0	1 and later
`sort_by`	Optional. Specify which file information to sort on. Options include: **name**: file name **user**: file owner **group**: file group **mtime**: last modified time **atime**: last access time **ctime**: last change time **crtime**: create time **posix**: POSIX permission	name, user, group, mtime, atime, ctime, crtime or posix	Name	1 and later

Parameter	Description	Value	Default Value	Availability
sort_direction	Optional. Specify to sort ascending or to sort descending. Options include: **asc**: sort ascending **desc**: sort descending	asc or desc	asc	1 and later
additional	Optional. Additional requested file information, separated by a comma, ",". When an additional option is requested, responded objects will be provided in the specified additional option. Options include: ■ **real_path**: return a real path in volume ■ **size**: return file byte size ■ **owner**: return information about file owner including user name, group name, UID and GID ■ **time**: return information about time including last access time, last modified time, last change time and create time ■ **perm**: return information about file permission ■ **volume_status**: return information about volume status including free space, total space and read-only status	real_path, owner, time, perm, mount_point_type or volume_status	(None)	1 and later

Example:

```
GET
/webapi/FileStation/file_virtual.cgi?api=SYNO.FileStation.VirtualFolder&version=1&m
ethod=list&type=cifs&additional=real_path%2Cowner%2Ctime%2Cperm%2Cmount_point_type%
2Cvolume_status
```

Response:

<data> object definitions:

Parameter	Type	Description	Availability
total	Integer	Total number of mount point folders.	1 and later
offset	Integer	Requested offset.	1 and later
folders	JSON-Style Array	Array of <virtual folder> object.	1 and later

<virtual folder> object definition:

Parameter	Type	Description	Availability
path	String	Path of a mount point folder.	1 and later
name	String	Name of a mount point folder.	1 and later
additional	<virtual-folder additional> object	Virtual folder additional object.	1 and later

<virtual folder> object definition:

Parameter	Type	Description	Availability
real_path	String	Real path started with a volume path.	1 and later
owner	<owner> object	File owner information including user name, group name, UID and GID.	1 and later
time	<time> object	Time information of file including last access time, last modified time, last change time and create time.	1 and later
perm	<perm> object	File permission information.	1 and later
mount_point_type	String	A type of a virtual file system of a mount point.	1 and later
volume_status	<volume_status> object	Volume status including free space, total space and read-only status.	1 and later

<owner>, <time> and <perm> object definition: Same as definition in the `list` method of SYNO.FileStation.List API

<volume_status> object definition: Same as definition in the `list_share` method of SYNO.FileStation.List API

Example:

```
{
        "folders": [{
                "additional": {
                        "mount_point_type": "remote",
                        "owner": {
                                "gid": 100,
                                "group": "users",
                                "uid": 1024,
                                "user": "admin"
                        },
```

```
                        "perm": {
                                "acl": {
                                        "append": true,
                                        "del": true,
                                        "exec": true,
                                        "read": true,
                                        "write": true
                                },
                                "is_acl_mode": false,
                                "posix": 777
                        },
                        "real_path": "/volume1/vidoe/remote",
                        "time": {
                                "atime": 1372313445,
                                "crtime": 1320204670,
                                "ctime": 1371206944,
                                "mtime": 1371206944
                        },
                        "volume_status": {
                                "freespace": 12282422599680,
                                "readonly": false,
                                "totalspace": 801958928384
                        }
                },
                "isdir": true,
                "name": "remote",
                "path": "/video/remote"
        }],
        "offset": 0,
        "total": 1
    }
```


API Error Code

No specific API error codes.

SYNO.FileStation.Favorite

Description

Add a folder to user's favorites or perform operations on user's favorites.

Overview

Availability: Since DSM 4.3

Version: 1

Method

list

Description:

List user's favorites

Availability:

Since version 1

Request:

Parameter	Description	Value	Default Value	Availability
`offset`	Optional. Specify how many favorites are skipped before beginning to return user's favorites.	Integer	0	1 and later
`limit`	Optional. Number of favorites requested. 0 indicates to list all favorites.	Integer	0	1 and later
`status_filter`	Optional. Show favorites with a given favorite status. Options of favorite statuses include: **valid**: A folder which a favorite links to exists **broken**: A folder which a favorite links to doesn't exist or doesn't be permitted to access it **all**: Both valid and broken statuses	valid, broken or all	all	1 and later
`additional`	Optional. Additional requested information of a folder which a favorite links to, separated by a comma, ",". When an additional option is requested, responded objects will be provided in the specified additional option. Options include:	name, size, user, group, mtime, atime, ctime, crtime, posix or type	name	1 and later

Parameter	Description	Value	Default Value	Availability
	■ **real_path**: return a real path in volume ■ **owner**: return information about file owner including user name, group name, UID and GID ■ **time**: return information about time including last access time, last modified time, last change time and create time ■ **perm**: return information about file permission ■ **mount_point_type**: return a type of a virtual file system of a mount point			

Example:

```
GET
/webapi/FileStation/file_favorite.cgi?api=SYNO.FileStation.Favorite&version=1&metho
d=list
```

Response:

<data> object definitions:

Parameter	Type	Description	Availability
`total`	Integer	Total number of favorites.	1 and later
`offset`	Integer	Requested offset.	1 and later
`favorites`	JSON-Style Array	Array of <favorite> objects.	1 and later

<favorite> object definition:

Parameter	Type	Description	Availability
`path`	String	Folder path of a user's favorites, started with a shared folder.	1 and later
`name`	String	Favorite name.	1 and later
`status`	String	Favorite status. Values of favorite status include: **valid**: A folder, which a favorite links to, exists **broken**: A folder, which a favorite links to, doesn't exist or be not permitted to access it.	1 and later
`additional`	<favorite additional> object	Favorite additional object.	1 and later

<favorite additional> object definition:

Parameter	Type	Description	Availability
`real_path`	String	Real path started with a volume path.	1 and later

Parameter	Type	Description	Availability
owner	<owner> object	File owner information including user name, group name, UID and GID.	1 and later
time	<time> object	Time information of file including last access time, last modified time, last change time and create time.	1 and later
perm	<perm> object	File permission information.	1 and later
mount_point_type	String	A type of a virtual file system of a mount point.	1 and later
type	String	File extension.	1 and later

<owner>, <time>, <perm> object definition: Same as definition in the `list` method of SYNO.FileStation.List API

Example:

```
{
        "favorites": [{
                "isdir": true,
                "name": "My Video Shared folder",
                "path": "/video",
                "status": "valid"
        }, {
                "isdir": false,
                "name": "deletedfolder",
                "path": "/share/deletedfolder",
                "status": "broken"
        }],
        "offset": 0,
        "total": 2
}
```

add

Description:
Add a folder to user's favorites

Availability:
Since version 1

Request:

Parameter	Description	Value	Default Value	Availability
path	A folder path starting with a shared folder is added to the user's favorites.	String	(None)	1 and later
name	A favorite name.	String	(None)	1 and later
index	Optional. Index of location of an added favorite. If it's equal to -1, the favorite will be added to the last one in user's favoirete. If it's between 0 ~ total number of favorites-1, the favorite will be inserted into user's favorites by the index.	Integer	-1	1 and later

Example:

```
GET
/webapi/FileStation/file_favorite.cgi?api=SYNO.FileStation.Favorite&version=1&metho
d=add&path=%2Fvideo%2Ffav&name=favorite_video
```

Response:
No specific response. It returns an empty success response if completed without error.

delete

Description:
Delete a favorite in user's favorites.

Availability:
Since version 1

Request:

Parameter	Description	Value	Default Value	Availability
path	A folder path starting with a shared folder is deleted from a user's favorites.	String	(None)	1 and later

Example:

```
GET
/webapi/FileStation/file_favorite.cgi?api=SYNO.FileStation.Favorite&version=1&metho
d=delete&path=%2Fvideo%2Ffav
```

Response:
No specific response. It returns an empty success response if completed without error.

clear_broken

Description:
Delete all broken statuses of favorites.

Availability:
Since version 1.

Request:
No parameters are required.

Example:

```
GET
/webapi/FileStation/file_favorite.cgi?api=SYNO.FileStation.Favorite&version=1&metho
d=clear_broken
```

Response:
No specific response. It returns an empty success response if completed without error.

edit

Description:
Edit a favorite name.

Availability:
Since version 1.

Request:

Parameter	Description	Value	Default Value	Availability
path	A folder path starting with a shared folder is edited from a user's favorites.	String	(None)	1 and later
name	New favorite name.	String	(None)	1 and later

Example:

```
GET
/webapi/FileStation/file_favorite.cgi?api=SYNO.FileStation.Favorite&version=1&metho
d=edit&path=%2Fvideo%2Ffav&name=my_video
```

Response:
No specific response. It returns an empty success response if completed without error.

replace_all

Description:
Replace multiple favorites of folders to the existed user's favorites.

Availability:
Since version 1.

Request

Parameter	Description	Value	Default Value	Availability
path	One or more folder paths starting with a shared folder, separated by a comma "," is added to the user's favorites. The number of paths must be the same as the number of favorite names in the name parameter. The first path parameter corresponds to the first name parameter.	String	(None)	1 and later
name	One or more new favrorite names, separated by a comma ", ". The number of favorite names must be the same as the number of folder paths in the path parameter. The first name parameter corresponding to the first path parameter.	String	(None)	1 and later

Example:

```
GET
/webapi/FileStation/file_favorite.cgi?api=SYNO.FileStation.Favorite&version=1&metho
d=replace_all&path=%2Fvideo%2C%2Fvideo%2Ffav&name=all_video%2Cmy_video
```

Response:

No specific response. It returns an empty success response if completed without error.

API Error Code

Code	Description
800	A folder path of favorite folder is already added to user's favorites.
801	A name of favorite folder conflicts with an existing folder path in the user's favorites.
802	There are too many favorites to be added.

SYNO.FileStation.Thumb

Description

Get a thumbnail of a file.

Note:

1. Supported image formats: jpg, jpeg, jpe, bmp, png, tif, tiff, gif, arw, srf, sr2, dcr, k25, kdc, cr2, crw, nef, mrw, ptx, pef, raf, 3fr, erf, mef, mos, orf, rw2, dng, x3f, raw
2. Supported video formats in an indexed folder: 3gp, 3g2, asf, dat, divx, dvr-ms, m2t, m2ts, m4v, mkv, mp4, mts, mov, qt, tp, trp, ts, vob, wmv, xvid, ac3, amr, rm, rmvb, ifo, mpeg, mpg, mpe, m1v, m2v, mpeg1, mpeg2, mpeg4, ogv, webm, flv, f4v, avi, swf, vdr, iso
   PS: Video thumbnails exist only if video files are placed in the “photo” shared folder or users' home folders.

Overview

Availability: Since DSM 4.3

Version: 1

Method

get

Description:
Get a thumbnail of a file.

Availability:
Since version 1.

Request:

Parameter	Description	Value	Default Value	Availability
`path`	A file path started with a shared folder.	String	(None)	1 and later
`size`	Optional. Return different size thumbnail. Size Options: **small**: small-size thumbnail **medium**: medium-size thumbnail **large**: large-size thumbnail **original**: original-size thumbnail	small, medium, large or original	small	1 and later

Parameter	Description	Value	Default Value	Availability
rotate	Optional. Return rotated thumbnail. Rotate Options: 0: Do not rotate 1: Rotate 90° 2: Rotate 180° 3: Rotate 270° 4: Rotate 360°	0,1,2,3,4	0	1 and later

Example:

```
GET
/webapi/FileStation/file_thumb.cgi?api=SYNO.FileStation.Thumb&version=1&method=get&
%2Fphoto%2Ftest.jpg
```

Response:
Image binary data.

API Error Code

Standard HTTP status codes.

For example, 404 Not Found.

SYNO.FileStation.DirSize

Description

Get the accumulated size of files/folders within folder(s).

This is a non-blocking API. You need to start it with the `start` method. Then, you should poll requests with the `status` method to get progress status or make a request with `stop` method to cancel the operation.

Overview

Availability: Since DSM 4.3

Version: 1

Method

start

Description:

Start to calculate size for one or more file/folder paths.

Availability:

Since version 1.

Request:

Parameter	Description	Value	Default Value	Availability
`path`	One or more file/folder paths starting with a shared folder for calculating cumulative size, separated by a comma, “,”.	String	(None)	1 and later

Example:

```
GET
/webapi/FileStation/file_dirSize.cgi?api=SYNO.FileStation.DirSize&version=1&method=
start&path=%2Fdownload%2F2013Q1Enhencement
```

Response:

<data> object definitions:

Parameter	Type	Description	Availability
`taskid`	String	A unique ID for the task for the calculating size task.	1

Example:

```
{
        "taskid":"51CBD7CD5C76E461"
}
```

status

Description:
Get the status of the size calculating task

Availability:
Since version 1

Request:

Parameter	Description	Value	Default Value	Availability
taskid	A unique ID for the task which is gotten from start method.	String	(None)	1 and later

Example:

```
GET
/webapi/FileStation/file_dirSize.cgi?api=SYNO.FileStation.DirSize&version=1&method=
status&taskid=51CBD7CD5C76E461
```

Response:
<data> object definitions:

Parameter	Type	Description	Availability
finished	Boolean	If the task is finished or not.	1
num_dir	Integer	Number of directories in the queried path(s).	1
num_file	Integer	Number of files in the queried path(s).	1
total_size	Integer	Accumulated byte size of the queried path(s).	1

Example:

```
{
        "finished":true,
        "num_dir":3,
        "num_file":104,
        "total_size":29973265
}
```

stop

Description:
Stop to calculate size

Availability:
Since version 1

Request:

Parameter	Description	Value	Default Value	Availability
`taskid`	A unique ID for the task which is gotten from `start` method.	String	(None)	1 and later

Example:

```
GET
/webapi/FileStation/file_dirSize.cgi?api=SYNO.FileStation.DirSize&version=1&method=
stop&taskid=51CBD7CD5C76E461
```

Response:
No specific response. It returns an empty success response if completed without error.

API Error Code

No specific API error codes.

SYNO.FileStation.MD5

Description

Get MD5 of a file.

This is a non-blocking API. You need to start it with the `start` method. Then, you should poll requests with `status` method to get the progress status, or make a request with the `stop` method to cancel the operation.

Overview

Availability: Since DSM 4.3

Version: 1

Method

start

Description:

Start to get MD5 of a file

Availability:

Since version 1.

Request:

Parameter	Description	Value	Default Value	Availability
`file_path`	A file path starting with a shared folder for calculating MD5 value.	String	(None)	1 and later

Example:

```
GET
/webapi/FileStation/file_md5.cgi?api=SYNO.FileStation.MD5&version=1&method=start&fi
le_path=%2Fdownload%2Fdownload.zip
```

Response:

<data> object definitions:

Parameter	Type	Description	Availability
`taskid`	String	A unique ID for the task for the calculating MD5 task.	1

Example:

```
{
        "taskid":"51CBD95028B22AED"
}
```

status

Description:
Get the status of the calculating MD5 task.

Availability:
Since version 1

Request:

Parameter	Description	Value	Default Value	Availability
`taskid`	A unique ID for the task which is gotten from `start` method.	String	(None)	1 and later

Example:

```
GET
/webapi/FileStation/file_md5.cgi?api=SYNO.FileStation.MD5&version=1&method=status&taskid=51CBD95028B22AED
```

Response:
<data> object definitions:

Parameter	Type	Description	Availability
`finished`	Boolean	Check if the task is finished or not.	1
`md5`	String	MD5 of the requested file.	1

Example:

```
{
        "finished":true,
        "md5":"6336c5a59aa63dd2042783f88e15410a"
}
```

stop

Description:
Stop calculating the MD5 of a file

Availability:
Since version 1

Request:

Parameter	Description	Value	Default Value	Availability
`taskid`	A unique ID for the task which is gotten from `start` method.	String	(None)	1 and later

Example:

```
GET
/webapi/FileStation/file_md5.cgi?api=SYNO.FileStation.MD5&version=1&method=stop&taskid=51CBD95028B22AED
```

Response:
No specific response. It returns an empty success response if completed without error.

API Error Code

No specific API error codes.

SYNO.FileStation.CheckPermission

Description

Check if a logged-in user has a permission to do file operations on a given folder/file.

Overview

Availability: Since DSM 4.3

Version: 1

Method

write

Description:

Check if a logged-in user has write permission to create new files/folders in a given folder

Availability:

Since version 1

Request:

Parameter	Description	Value	Default Value	Availability
`path`	A folder path starting with a shared folder to check write permission.	String	(None)	1 and later
`create_only`	Optional. True by default. If set to true, the permission will be allowed when there is non-existent file/folder.	Boolean	(None)	1 and later

Example:

```
GET
/webapi/FileStation/file_permission.cgi?api=SYNO.FileStation.CheckPermission&versio
n=1&method=write&path=%2Fdownload%2Ftest.zip&create_only=true
```

Response:

The request will get error response if no write permission for the specified path.

API Error Code

No specific API error codes.

SYNO.FileStation.Upload

Description

Upload a file.

Overview

Availability: Since DSM 4.3

Version: 1

Method

upload

Description:

Upload a file by RFC 1867, http://tools.ietf.org/html/rfc1867.

Note that each parameter is passed within each part but binary file data must be the last part.

Availability:

Since version 1

Request:

Parameter	Description	Value	Default Value	Availability
`dest_folder_path`	A destination folder path starting with a shared folder to which files can be uploaded.	String	(None)	1 and later
`create_parents`	Create parent folder(s) if none exist.	Boolean	(None)	1 and later
`overwrite`	Optional. The value could be one of following: (1) true: overwrite the destination file if one exists (2) false: skip the upload if the destination file exists (3) (None): when it's not specified as true or false, the upload will be responded with error when the destination file exists	true/false/(None)	(None)	1 and later
`mtime`	Optional. Set last modify time of the uploaded file, unit: Linux timestamp in millisecond.	Linux timestamp in millisecond	(None)	1 and later
`crtime`	Optional. Set the create time of the uploaded file, unit: Linux timestamp in millisecond.	Linux timestamp in millisecond	(None)	1 and later
`atime`	Optional. Set last access time of the uploaded file, unit: Linux timestamp in millisecond.	Linux timestamp in millisecond	(None)	1 and later

Parameter	Description	Value	Default Value	Availability
`filename` (file part)	File content. Must be the last part.	Binary data	(None)	1 and later

Note: Linux timestamp in millisecond, defined as the number of milliseconds that have elapsed since 00:00:00 Coordinated Universal Time (UTC), Thursday, 1 January 1970.

Example:

```
POST /webapi/FileStation/api_upload.cgi


…
Content-Length:20326728
Content-type: multipart/form-data, boundary=AaB03x


--AaB03x
content-disposition: form-data; name="api"


SYNO.FileStation.Upload
--AaB03x
content-disposition: form-data; name="version"


1
--AaB03x
content-disposition: form-data; name="method"


upload
--AaB03x
content-disposition: form-data; name="dest_folder_path"


/upload/test
--AaB03x
content-disposition: form-data; name="create_parents"


true
--AaB03x
content-disposition: form-data; name="file"; filename="file1.txt"
Content-Type: application/octet-stream
```

```
... contents of file1.txt ...
--AaB03x--
```

Response:

No specific response. It returns an empty success response if completed without error.

API Error Code

Code	Description
1800	There is no Content-Length information in the HTTP header or the received size doesn't match the value of Content-Length information in the HTTP header.
1801	Wait too long, no date can be received from client (Default maximum wait time is 3600 seconds).
1802	No filename information in the last part of file content.
1803	Upload connection is cancelled.
1804	Failed to upload too big file to FAT file system.
1805	Can't overwrite or skip the existed file, if no `overwrite` parameter is given.

SYNO.FileStation.Download

Description

Download file(s)/folder(s).

Overview

Availability: Since DSM 4.3

Version: 1

Method

download

Description:

Download files/folders. If only one file is specified, the file content is responded. If more than one file/folder is given, binary content in ZIP format which they are compressed to is responded.

Availability:

Since version 1

Request:

Parameter	Description	Value	Default Value	Availability
`path`	One or more file/folder paths starting with a shared folder to be downloaded, separated by a commas ",". When more than one file is to be downloaded, files/folders will be compressed as a zip file.	String	(None)	1 and later
`mode`	Mode used to download files/folders, value could be: (1) open: try to trigger the application, such as a web browser, to open it. Content-Type of the HTTP header of the response is set to MIME type according to file extension. (2) download: try to trigger the application, such as a web browser, to download it. Content-Type of the HTTP header of response is set to application/octet-stream and Content-Disposition of the HTTP header of the response is set to attachment.	open or download	open	1 and later

Example:

```
GET
/webapi/FileStation/file_download.cgi?api=SYNO.FileStation.Download&version=1&metho
d=download&path=%2Ftest%2FITEMA_20445972-0.mp3&mode=open
```

Response:
The file content

API Error Code

No specific API error codes.

Note: If `mode` parameter is set to open value, “404 Not Found” of status code of the HTTP header is responded when an error occurs.

SYNO.FileStation.Sharing

Description

Generate a sharing link to share files/folders with other people and perform operations on sharing link(s).

Overview

Availability: Since DSM 4.3

Version: 1

Method

getinfo

Description:

Get information of a sharing link by the sharing link ID

Availability:

Since version 1

Request:

Parameter	Description	Value	Default Value	Availability
id	A unique ID of a sharing link.	String	(None)	1 and later

Example:

```
GET /webapi/FileStation/file_sharing.cgi?api=SYNO.FileStation.Sharing
&version=1&method=getinfo&id=pHTBKQf9
```

Response:

Returned <data> object is a <Sharing_Link> object (defined in the Response Objects section).

Example:

```
        {
                "date_available":"0",
                "date_expired":"0",
                "has_password":false,
                "id":"pHTBKQf9",
                "isFolder":false,
                "link_owner":"admin",
```

```
            "name":"ITEMA_20448251-0.mp3",
            "path":"/test/ITEMA_20448251-0.mp3",
            "status":"valid",
            "url":"http://myds.com:5000/fbsharing/pHTBKQf9"
}
```


list

Description:
List user's file sharing links.

Availability:
Since version 1

Request:

Parameter	Description	Value	Default Value	Availability
offset	Optional. Specify how many sharing links are skipped before beginning to return listed sharing links.	Integer	0	1 and later
limit	Optional. Number of sharing links requested. 0 means to list all sharing links.	Integer	0	1 and later
sort_by	Optional. Specify information of the sharing link to sort on. Options include: **id**: a unique ID of sharing a file/folder **name**: file name **isFolder**: if it's a folder or not **path**: file path **date_expired**: the expiration date for the sharing link **date_available**: the available date for the sharing link start effective **status**: the link accessibility status **has_password**: If the sharing link is protected or not **url**: a URL of a sharing link **link_owner**: the user name of the sharing link owner	name, isFolder, path, date_expired, date_available, status, has_password, id, url or link_owner	(None)	1 and later
sort_direction	Optional. Specify to sort ascending or to sort descending. Options include: **asc**: sort ascending **desc**: sort descending	asc or desc	asc	1 and later
force_clean	Optional. If set to false, the data	Boolean	false	1 and later

Parameter	Description	Value	Default Value	Availability
	will be retrieval from cache database rapidly. If set to true, all sharing information including sharing statuses and user name of sharing owner will be synchronized. It consumes some time.			

Example:

```
GET
/webapi/FileStation/file_sharing.cgi?api=SYNO.FileStation.Sharing&version=1&method=
list&offset=0&limit=10
```

Response:

<data> object definitions:

Parameter	Type	Description	Availability
total	Integer	Total number of sharing links.	1
offset	Integer	Requested offset.	1
links	JSON-Style Array	Array of <Sharing_Link> object.	1

Example:

```
{
	"links":[{
		"date_available":"0",
		"date_expired":"0",
		"has_password":false,
		"id":"pHTBKQf9",
		"isFolder":false,
		"link_owner":"admin",
		"name":"ITEMA_20448251-0.mp3",
		"path":"/test/ITEMA_20448251-0.mp3",
		"status":"valid",
		"url":"http://myds.com:5000/fbsharing/pHTBKQf9"
	}],
	"offset":0,
	"total":1
}
```

create

Description:
Generate one or more sharing link(s) by file/folder path(s)

Availability:
Since version 1

Request:

Parameter	Description	Value	Default Value	Availability
path	One or more file/folder paths with which to generate sharing links, separated by commas “,”.	String	(None)	1 and later
password	Optional The password for the sharing link when accessing it. The max password length are 16 characters.	String	(None)	1 and later
date_expired	Optional. The expiration date for the sharing link, written in the format YYYY-MM-DD. When set to 0 (default), the sharing link is permanent.	YYYY-MM-DD	0	1 and later
date_available	Optional. The available date for the sharing link to become effective, written in the format YYYY-MM-DD. When set to 0 (default), the sharing link is valid immediately after creation.	YYYY-MM-DD	0	1 and later

Note: date of `date_expired` and `date_available` parameter is based on user’s DS date.

Example:

```
GET
/webapi/FileStation/file_sharing.cgi?api=SYNO.FileStation.Sharing&version=1&method=
create&path=%2Ftest%2FITEMA_20445972-0.mp3
```

Response:
<data> object definitions:

Parameter	Type	Description	Availability
links	JSON-Style Array	Array of <Shared_Link> object.	1

<Shared_Link> object definition:

Member	Type	Description	Availability
path	String	A file/folder path of the sharing link.	1
url	String	A created URL of the sharing link.	1
id	String	A created unique ID of the sharing link.	1
qrcode	String	Base64-encoded image of QR code describing the URL of the sharing link.	1

Member	Type	Description	Availability
error	Integer	0 for creating it successfully, otherwise is the error code for failed to create it.	1

Example:

```
{
	"links":[{
		"error":0,
		"id":"y4LmvpaX",
		"path":"/test/ITEMA_20445972-0.mp3",
		"qrcode":"iVBORw0KGgoAAAANSUh...",
		"url":"http://myds.com:5000/fbsharing/y4LmvpaX"
	}]
}
```

delete

Description:
Delete one or more sharing links.

Availability:
Since version 1.

Request:

Parameter	Description	Value	Default Value	Availability
id	Unique IDs of file sharing link(s) to be deleted, separated by commas “,”.	String	(None)	1 and later

Example:

```
GET
/webapi/FileStation/file_sharing.cgi?api=SYNO.FileStation.Sharing&version=1&method=
delete&id=y4LmvpaX
```

Response:
Returns an empty success response if completed without error; otherwise returns error object array contains failed IDs.

clear_invalid

Description:
Remove all expired and broken sharing links

Availability:
Since version 1

Request:
No parameters are required.

Example:

```
GET
/webapi/FileStation/file_sharing.cgi?api=SYNO.FileStation.Sharing&version=1&method=
clear_invalid
```

Response:
No specific response. It returns an empty success response if completed without error.

edit

Description:
Edit sharing link(s)

Availability:
Since version 1

Request:

Parameter	Description	Value	Default Value	Availability
`id`	Unique ID(s) of sharing link(s) to edit, separated by a comma, ",".	Integer	(None)	1 and later
`password`	Optional. If empty string is set, the password is removed. The max length of the password is 16 characters.	String	(None)	1 and later
`date_expired`	Optional. The expiration date for the sharing link, using format YYYY-MM-DD. When set to 0 (default), the sharing link is permanent.	YYYY-MM-DD	(None)	1 and later
`date_available`	Optional. The available date for the sharing link start effective, using format YYYY-MM-DD. When set to 0 (default), the sharing link is valid right after creation.	YYYY-MM-DD	(None)	1 and later

Note: date of `date_expired` and `date_available` parameter is based on user's DiskStation date.

Example:

```
GET /webapi/FileStation/file_sharing.cgi?
api=SYNO.FileStation.Sharing&version=1&method=edit&id=y4LmvpaX&password=123
```

Response:
No specific response. It returns an empty success response if completed without error.

Response Objects

<Sharing_Link> object definition:

Member	Type	Description	Availability
id	String	A unique ID of a sharing link.	1
url	String	A URL of a sharing link.	1
link_owner	String	A user name of a sharing link owner.	1
path	String	A file or folder path of a sharing link.	1
isFolder	String	Whether the sharing link is for a folder.	1
has_password	Boolean	Whether the sharing link has password.	1
date_expired	String	The expiration date of the sharing link in the format YYYY-MM-DD. If the value is set to 0, the link will be permanent.	1
date_available	String	The date when the sharing link becomes active in the format YYYY-MM-DD. If the value is set to 0, the file sharing link will be active immediately after creation.	1
status	String	The accessibility status of the sharing link might be one of the following: (1) **valid**: the sharing link is active. (2) **invalid**: the sharing link is not active because the available date has not arrived yet. (3) **expired**: the sharing link expired. (4) **broken**: the sharing link broke due to a change in the file path or access permission.	1

API Error Code

Code	Description
2000	Sharing link does not exist.
2001	Cannot generate sharing link because too many sharing links exist.
2002	Failed to access sharing links.

SYNO.FileStation.CreateFolder

Description

Create folders.

Overview

Availability: Since DSM 4.3

Version: 1

Method

create

Description:

Create folders.

Availability:

Since version 1

Request:

Parameter	Description	Value	Default Value	Availability
`folder_path`	One or more shared folder paths, separated by commas. If `force_parent` is "true," and `folder_path` does not exist, the `folder_path` will be created. If `force_parent` is "false," `folder_path` must exist or a false value will be returned. The number of paths must be the same as the number of names in the `name` parameter. The first `folder_path` parameter corresponds to the first `name` parameter.	String	(None)	1 and later
`name`	One or more new folder names, separated by commas “,”. The number of names must be the same as the number of folder paths in the `folder_path` parameter. The first `name` parameter corresponding to the first `folder_path` parameter.	String	(None)	1 and later

Parameter	Description	Value	Default Value	Availability
force_parent	Optional. “true”: no error occurs if a folder exists and make parent folders as needed; “false”: parent folders are not created.	Boolean	false	1 and later
additional	Optional. Additional requested file information, separated by commas “,”. When an additional option is requested, responded objects will be provided in the specified additional option. Options include: ■ **real_path**: return a real path in volume ■ **size**: return file byte size ■ **owner**: return information about file owner including user name, group name, UID and GID ■ **time**: return information about time including last access time, last modified time, last change time and create time ■ **perm**: return information about file permission ■ **type**: return a file extension	real_path,size,owner,time,perm or type	(None)	1 and later

Example:

```
GET
/webapi/FileStation/file_crtfdr.cgi?api=SYNO.FileStation.CreateFolder&version=1&method=create&folder_path=%2Fvideo&name=test
```

Response:

<data> object definitions:

Parameter	Type	Description	Availability
folders	JSON-Style Array	Array of <file> objects about file information of a new folder path.	1 and later

<file> object definition:

Same as definition in SYNO.FileStation.List API with `getinfo` method

Example:

```
{
        "folders": [{
                "isdir": true,
                "name": "test",
                "path": "/video/test"
```

```
                }]

        }
```

API Error Code

Code	Description
1100	Failed to create a folder. More information in <errors> object.
1101	The number of folders to the parent folder would exceed the system limitation.

SYNO.FileStation.Rename

Description

Rename a file/folder.

Overview

Availability: Since DSM 4.3

Version: 1

Method

rename

Description:

Rename a file/folder

Availability:

Since version 1

Request:

Parameter	Description	Value	Default Value	Availability
`path`	One or more paths of files/folders to be renamed, separated by commas ",". The number of paths must be the same as the number of names in the `name` parameter. The first `path` parameter corresponds to the first `name` parameter.	String	(None)	1 and later
`name`	One or more new names, separated by commas ",". The number of names must be the same as the number of folder paths in the `path` parameter. The first `name` parameter corresponding to the first `path` parameter.	String	(None)	1 and later
`additional`	Optional. Additional requested file information, separated by commas ",". When an additional option is requested, responded objects will be provided in the specified additional option. Options include: ■ **real_path**: return a real path in volume	real_path,size,owner,time,perm or type	(None)	1 and later

Parameter	Description	Value	Default Value	Availability
	■ **size**: return file byte size ■ **owner**: return information about file owner including user name, group name, UID and GID ■ **time**: return information about time including last access time, last modified time, last change time and create time ■ **perm**: return information about file permission ■ **type**: return a file extension			
`search_taskid`	Optional. A unique ID for the search task which is obtained from `start` method. It is used to update the renamed file in the search result.	String	(None)	1 and later

Example:

```
GET
/webapi/FileStation/file_rename.cgi?api=SYNO.FileStation.Rename&version=1&method=rename&path=%2Fvideo%2Ftest&name=test2
```

Response:

<data> object definitions:

Parameter	Type	Description	Availability
`files`	JSON-Style Array	Array of <file> objects.	1 and later

<file> object definition:

Same as definition in SYNO.FileStation.List API with `getinfo` method

Example:

```
{
	"files": [{
		"isdir": true,
		"name": "test2",
		"path": "/video/test2"
	}]
}
```

API Error Code

Code	Description
1200	Failed to rename it. More information in <errors> object.

SYNO.FileStation.CopyMove

Description

Copy/move file(s)/folder(s).

This is a non-blocking API. You need to start to copy/move files with `start` method. Then, you should poll requests with `status` method to get the progress status, or make a request with `stop` method to cancel the operation.

Overview

Availability: Since DSM 4.3

Version: 1

Method

start

Description:

Start to copy/move files

Availability:

Since version 1

Request:

Parameter	Description	Value	Default Value	Availability
`path`	One or more copied/moved file/folder path(s) starting with a shared folder, separated by commas “,”.	String	(None)	1 and later
`dest_folder_path`	A desitination folder path where files/folders are copied/moved.	String	(None)	1 and later
`overwrite`	Optional. “true”: overwrite all existing files with the same name; “false”: skip all existing files with the same name; (None): do not overwrite or skip existed files. If there is any existing files, an error occurs (error code: 1003).	true, false, (None)	(None)	1 and later
`remove_src`	Optional. “true”: move filess/folders;”false”: copy files/folders	Boolean	false	1 and later

Parameter	Description	Value	Default Value	Availability
accurate_progress	Optional. “true”: calculate the progress by each moved/copied file within sub-folder. “false”: calculate the progress by files which you give in path parameters. This calculates the progress faster, but is less precise.	Boolean	true	1 and later
search_taskid	Optional. A unique ID for the search task which is gotten from SYNO.FileSation.Search API with start method. This is used to update the search result.	String	(None)	1 and later

Example:

```
GET
/webapi/FileStation/file_MVCP.cgi?api=SYNO.FileStation.CopyMove&version=1&method=st
art&path=%2Fvideo%2Ftest.avi&dest_folder_path=%2F%2Fvideo%2Ftest
```

Response:
<data> object definitions:

Parameter	Type	Description	Availability
taskid	String	A unique ID for the copy/move task.	1 and later

Example:

```
        {
                "taskid": "FileStation_51D00B7912CDE0B0"
        }
```

status

Description:
Get the copying/moving status

Availability:
Since version 1

Request

Parameter	Description	Value	Default Value	Availability
taskid	A unique ID for the copy/move task which is obtained from start method.	String	(None)	1 and later

Example:

```
GET
/webapi/FileStation/file_MVCP.cgi?api=SYNO.FileStation.CopyMove&version=1&method=status&taskid=FileStation_51D00B7912CDE0B0
```

Response:

<data> object definitions:

Parameter	Type	Description	Availability
processed_size	Integer	If accurate_progress parameter is “true,” byte sizes of all copied/moved files will be accumulated. If “false,” only byte sizes of the file you give in path parameter is accumulated.	1 and later
total	Integer	If accurate_progress parameter is “true,” the value indicates total byte sizes of files including subfolders will be copied/moved. If “false,” it indicates total byte sizes of files you give in path parameter excluding files within subfolders. Otherwise, when the total number is calculating, the value is -1.	1 and later
path	String	A copying/moving path which you give in path parameter.	1 and later
finished	Boolean	If the copy/move task is finished or not.	1 and later
progress	Double	A progress value is between 0~1. It is equal to processed_size parameter divided by total parameter.	1 and later
dest folder path	String	A desitination folder path where files/folders are copied/moved.	1 and later

Example:

```
{
        "dest_folder_path": "/video/test",
        "finished": false,
        "path": "/video/test.avi",
        "processed_size": 1057,
        "progress": 0.01812258921563625,
        "total": 58325
}
```

stop

Description:

Stop a copy/move task.

Availability:

Since version 1

Request:

Parameter	Description	Value	Default Value	Availability
`taskid`	A unique ID for the copy/move task which is gotten from `start` method.	String	(None)	1 and later

Example:

```
GET
/webapi/FileStation/file_MVCP.cgi?api=SYNO.FileStation.CopyMove&version=1&method=st
op&taskid=FileStation_51D00B7912CDE0B0
```

Response:

No specific response. It returns an empty success response if completed without error.

API Error Code

Code	Description
1000	Failed to copy files/folders. More information in <errors> object.
1001	Failed to move files/folders. More information in <errors> object.
1002	An error occurred at the destination. More information in <errors> object.
1003	Cannot overwrite or skip the existing file because no `overwrite` parameter is given.
1004	File cannot overwrite a folder with the same name, or folder cannot overwrite a file with the same name.
1006	Cannot copy/move file/folder with special characters to a FAT32 file system.
1007	Cannot copy/move a file bigger than 4G to a FAT32 file system.

SYNO.FileStation.Delete

Description

Delete file(s)/folder(s).

There are two kinds of methods; one is a non-blocking method; and the other is a blocking method. With the non-blocking method, you can start the deletion operation using the `start` method. Then, you should poll a request with the `status` method to get more information or make a request with the `stop` method to cancel the operation. With the blocking method, you can directly make requests with `delete` method to delete files/folders, but the response is not returned until the delete operation is completed.

Overview

Availability: Since DSM 4.3

Version: 1

Method

start

Description:

Delete file(s)/folder(s).

This is a non-blocking method. You should poll a request with `status` method to get more information or make a request with `stop` method to cancel the operation.

Availability:

Since version 1

Request:

Parameter	Description	Value	Default Value	Availability
`path`	One or more deleted file/folder paths starting with a shared folder, separated by commas “,”.	String	(None)	1 and later
`accurate_progress`	Optional. “true”: calculates the progress of each deleted file with the sub-folder recursively; “false”: calculates the progress of files which you give in `path` parameters. The latter is faster than recursively, but less precise. Note: Only non-blocking methods suits using the `status` method to get progress.	Boolean	true	1 and later

Parameter	Description	Value	Default Value	Availability
recursive	Optional. “true”: Recursively delete files within a folder. “false”: Only delete first-level file/folder. If a deleted folder contains any file, an error occurs because the folder can’t be directly deleted.	Boolean	true	1 and later
search_taskid	Optional. A unique ID for the search task which is gotten from start method. It’s used to delete the file in the search result.	String	(None)	1 and later

Example:

```
GET
/webapi/FileStation/file_delete.cgi?api=SYNO.FileStation.Delete&version=1&method=st
art&path=%2Fvideo%2Fdel_folder
```

Response:
<data> object definitions:

Parameter	Type	Description	Availability
taskid	String	A unique ID for the delete task.	1 and later

Example:

```
{
        "taskid": "FileStation_51CEC9C979340E5A"
}
```

status

Description:
Get the deleting status

Availability:
Since version 1

Request:

Parameter	Description	Value	Default Value	Availability
taskid	A unique ID for the delete task which is gotten from start method.	String	(None)	1 and later

Example:

```
GET
/webapi/FileStation/file_delete.cgi?api=SYNO.FileStation.Delete&version=1&method=st
atus&taskid=FileStation_51CEC9C979340E5A
```

Response:
<data> object definitions:

Parameter	Type	Description	Availability
processed_num	Integer	If accurate_progress parameter is “true,” the number of all deleted files will be accumulated. If “false,” only the number of file you give in path parameter is accumulated.	1 and later
total	Integer	If accurate_progress parameter is “true,” the value indicates how many files including subfolders will be deleted. If “false,” it indicates how many files you give in path parameter. When the total number is calculating, the value is -1.	1 and later
path	String	A deletion path which you give in path parameter.	1 and later
processing_path	String	A deletion path which could be located at a subfolder.	1 and later
finished	Boolean	Whether or not the deletion task is finished.	1 and later
progress	Double	Progress value whose range between 0~1 is equal to processed_num parameter divided by total parameter.	1 and later

Example:

```
{
        "finished": false,
        "path": "/video/1000",
        "processed_num": 193,
        "processing_path": "/video/1000/509",
        "progress": 0.03199071809649467,
        "total": 6033
}
```

stop

Description:
Stop a delete task

Availability:
Since version 1

Request:

Parameter	Description	Value	Default Value	Availability
taskid	A unique ID for the deletion task which is obtained from start method.	String	(None)	1 and later

Example:

```
GET
/webapi/FileStation/file_delete.cgi?api=SYNO.FileStation.Delete&version=1&method=st
op&taskid=FileStation_51CEC9C979340E5A
```

Response:
No specific response. It returns an empty success response if completed without error.

delete

Description:
Delete files/folders. This is a blocking method. The response is not returned until the deletion operation is completed.

Availability:
Since version 1

Request:

Parameter	Description	Value	Default Value	Availability
path	One or more deleted file/folder path(s) started with a shared folder, separated by a comma, ",".	String	(None)	1 and later
recursive	Optional. "true": Recursively delete files within a folder. "false": Only delete first-level file/folder. If a deleted folder contains any file, an error will occur because the folder can't be directly deleted.	Boolean	true	1 and later
search_taskid	Optional. A unique ID for the search task which is gotten from start method. It's used to delete the file in the search result.	Boolean	(None)	1 and later

Example:

```
GET
/webapi/FileStation/file_delete.cgi?api=SYNO.FileStation.Delete&version=1&method=de
lete&path=%2Fvideo%2Fdel_folder
```

Response:
No specific response. It returns an empty success response if completed without error.

API Error Code

Code	Description
900	Failed to delete file(s)/folder(s). More information in <errors> object.

SYNO.FileStation.Extract

Description

Extract an archive and perform operations on archive files.

Note: Supported extensions of archives: zip, gz, tar, tgz, tbz, bz2, rar, 7z, iso

Overview

Availability: Since DSM 4.3

Version: 1

Method

start

Description:

Start to extract an archive. This is a non-blocking method. You need to start to extract files with `start` method. Then, you should poll requests with `status` method to get the progress status, or make a request with the `stop` method to cancel the operation.

Availability:

Since version 1

Request:

Parameter	Description	Value	Default Value	Availability
`file_path`	A file path of an archive to be extracted, starting with a shared folder	String	(None)	1 and later
`dest_folder_path`	A destination folder path starting with a shared folder to which the archive will be extracted.	String	(None)	1 and later
`overwrite`	Optional. Whether or not to overwrite if the extracted file exists in the destination folder.	Boolean	false	1 and later
`keep_dir`	Optional. Whether to keep the folder structure within an archive.	Boolean	true	1 and later
`create_subfolder`	Optional. Whether to create a subfolder with an archive name which archived files are extracted to.	Boolean	false	1 and later

Parameter	Description	Value	Default Value	Availability
codepage	Optional. The language codepage used for decoding file name with an archive.	DSM supported language, including enu, cht, chs, krn, ger, fre, ita, spn, jpn, dan, nor, sve, nld, rus, plk, ptb, ptg, hun, trk or csy	DSM Codepage Setting	1 and later
password	Optional. The password for extracting the file.	String	(None)	1 and later
item_id	Optional. Item IDs of archived files used for extracting files within an archive, separated by a comma, ",". Item IDs could be listed by requesting `list` method.	Integer	(None)	1 and later

Example:

```
GET
/webapi/FileStation/file_extract.cgi?api=SYNO.FileStation.Extract&version=1&method=
start&file_path=%2Fdownload%2Fdownload.zip&dest_folder_path=%2Fdownload%2Fdownload&
keep_dir=true&create_subfolder=true&overwrite=false
```

Response:

<data> object definitions:

Parameter	Type	Description	Availability
taskid	String	A unique ID for the extract task.	1

Example:

```
{
        "taskid":"FileStation_51CBB59C68EFE6A3"
}
```

status

Description:

Get the extract task status

Availability:

Since version 1

Request:

Parameter	Description	Value	Default Value	Availability
taskid	A unique ID for the extract task.	String	(None)	1 and later

Example:

```
GET
/webapi/FileStation/file_compress.cgi?api=SYNO.FileStation.Compress&version=1&metho
d=status&taskid=FileStation_51CBB59C68EFE6A3
```

Response:
<data> object definitions:

Parameter	Type	Description	Availability
finished	Boolean	If the task is finished or not.	1
progress	Double	The extract progress expressed in range 0 to 1.	1
dest_folder_path	String	The requested destination folder for the task.	1

Example:

```
{
        "dest_folder_path":"/download/download",
        "finished":false,
        "progress":0.1
}
```

stop

Description:
Stop the extract task

Availability:
Since version 1

Request:

Parameter	Description	Value	Default Value	Availability
taskid	A unique ID for the extract task which is gotten from start method.	String	(None)	1 and later

Example:

```
GET
/webapi/FileStation/file_extract.cgi?api=SYNO.FileStation.Extract&version=1&method=
stop&taskid=FileStation_51CBB59C68EFE6A3
```

Response:
No specific response. It returns an empty success response if completed without error.

list

Description:
List archived files contained in an archive

Availability:
Since version 1

Request:

Parameter	Description	Value	Default Value	Availability
file_path	An archive file path starting with a shared folder to list.	String	(None)	1 and later
offset	Optional. Specify how many archived files are skipped before beginning to return listed archived files in an archive.	Integer	0	1 and later
limit	Optional. Number of archived files requested. -1 indicates to list all archived files in an archive.	Integer	-1	1 and later
sort_by	Optional. Specify which archived file information to sort on. Options include: **name**: file name **size**: file size **pack_size**: file owner **mtime**: last modified time	name, size,pack_size or mtime	name	1 and later
sort_direction	Optional. Specify to sort ascending or to sort descending. Options include: **asc**: sort ascending **desc**: sort descending	asc or desc	asc	1 and later
codepage	Optional. The language codepage used for decoding file name with an archive.	DSM supported language, including enu, cht, chs, krn, ger, fre, ita, spn, jpn, dan, nor, sve, nld, rus, plk, ptb, ptg, hun, trk or csy	DSM Codepage Setting	1 and later
password	Optional. The password for extracting the file.	String	(None)	1 and later
item_id	Optional. Item ID of an archived folder to be listed within an archive. (None) or -1 will list archive files in a root folder within an archive.	Integer	(None)	1 and later

Example:

```
GET
/webapi/FileStation/file_extract.cgi?api=SYNO.FileStation.Extract&version=1&method=
```

```
list&file_path=%2Fdownload%2Fdownload.zip&sortby=name&sort_direction=asc&item_id=-1
```

Response:

<data> object definitions:

Parameter	Type	Description	Availability
items	JSON-Style Array	Array of <Archive_Item> objects.	1

<Archive_Item Object> definition:

Member	Type	Description	Availability
itemid	Integer	Item ID of an archived file in an archive.	1 and later
name	String	Filename of an archived file in an archive.	1 and later
size	Integer	Original byte size of an archived file.	1 and later
pack_size	Integer	Archived byte size of an archived file.	1 and later
mtime	String	Last modified time of an archived file.	1 and later
path	String	Relative path of an archived file within in an archive.	1 and later
is_dir	Boolean	Whether an archived file is a folder.	1 and later

Example:

```
{
        "items":[{
                "is_dir":false,
                "item_id":1,
                "mtime":"2013-02-03 00:17:12",
                "name":"ITEMA_20445972-0.mp3",
                "pack_size":51298633,
                "path":"ITEMA_20445972-0.mp3",
                "size":51726464
        },{
                "is_dir":false,
                "item_id":0,
                "mtime":"2013-03-03 00:18:12",
                "name":"ITEMA_20455319-0.mp3",
                "pack_size":51434239,
                "path":"ITEMA_20455319-0.mp3",
                "size":51896448
        }],
        "total":2
```

```
        }
```

API Error Code

Code	Description
1400	Failed to extract files.
1401	Cannot open the file as archive.
1402	Failed to read archive data error
1403	Wrong password.
1404	Failed to get the file and dir list in an archive.
1405	Failed to find the item ID in an archive file.

SYNO.FileStation.Compress

Description

Compress file(s)/folder(s).

This is a non-blocking API. You need to start to compress files with the `start` method. Then, you should poll requests with the `status` method to get compress status, or make a request with the `stop` method to cancel the operation.

Overview

Availability: Since DSM 4.3

Version: 1

Method

start

Description:

Start to compress file(s)/folder(s).

Availability:

Since version 1

Request:

Parameter	Description	Value	Default Value	Availability
`path`	One or more file paths to be compressed, separated by commas ",". The path should start with a shared folder.	String	(None)	1 and later
`dest_file_path`	A destination file path (including file name) of an archive for the compressed archive.	String	(None)	1 and later
`level`	Optional. Compress level used, could be one of following values: (1) **moderate**(default): moderate compression and normal compression speed (2) **store**: pack files with no compress (3) **fastest**: fastest compression speed but less compression (4) **best**: slowest compression speed but optimal compression	moderate, store, fastest or best	moderate	1 and later

Parameter	Description	Value	Default Value	Availability
mode	Optional. Compress mode used, could be one of following values: (1) **add** (default): Update existing items and add new files. If an archive does not exist, a new one is created. (2) **update**: Update existing items if newer on the file system and add new files. If the archive does not exist create a new archive. (3) **refreshen**: Update existing items of an archive if newer on the file system. Does not add new files to the archive. (4) **synchronize**: Update older files in the archive and add files that are not already in the archive.	add, update, refreshen or synchronize	add	1 and later
format	Optional. The compress format, ZIP or 7z format.	zip or 7z	zip	1 and later
password	Optional. The password for the archive.	String	(None)	1 and later

Example:

```
GET
/webapi/FileStation/file_compress.cgi?api=SYNO.FileStation.Compress&version=1&metho
d=start&path=%2Fdownload%2FITEMA_20455319-0.mp3%2C%2Fdownload%2FITEMA_20445972-
0.mp3&dest_file_path=%2Fdownload%2Fdownload.zip&format=zip
```

Response:

<data> object definitions:

Parameter	Type	Description	Availability
taskid	String	A unique ID for the compress task.	1

Example:

```
{
        "taskid":"FileStation_51CBB25CC31961FD"
}
```

status

Description:

Get the compress task status

Availability:

Since version 1

Request:

Parameter	Description	Value	Default Value	Availability
taskid	A unique ID for the compress task.	String	(None)	1 and later

Example:

```
GET
/webapi/FileStation/file_compress.cgi?api=SYNO.FileStation.Compress&version=1&metho
d=status&taskid=FileStation_51CBB25CC31961FD
```

Response:

<data> object definitions:

Parameter	Type	Description	Availability
finished	Boolean	Whether or not the compress task is finished.	1
dest_file_path	String	The requested destination path of an archive.	1

Example:

```
{
        "dest_file_path":"/download/download.zip",
        "finished":true
}
```

stop

Description:

Stop the compress task

Availability:

Since version 1

Request:

Parameter	Description	Value	Default Value	Availability
taskid	A unique ID for the compress task which is obtained from start method.	String	(None)	1 and later

Example:

```
GET
/webapi/FileStation/file_compress.cgi?api=SYNO.FileStation.Compress&version=1&metho
d=stop&taskid=FileStation_51CBB25CC31961FD
```

Response:

No specific response. It returns an empty success response if completed without error.

API Error Code

Code	Description
1300	Failed to compress files/folders.
1301	Cannot create the archive because the given archive name is too long.

SYNO.FileStation.BackgroundTask

Description

Get information regarding tasks of file operations which is run as the background process including copy, move, delete, compress and extract tasks with non-blocking API/methods. You can use the `status` method to get more information, or use the `stop` method to cancel these background tasks in individual API, such as SYNO.FileStation.CopyMove API, SYNO.FileStation.Delete API, SYNO.FileStation.Extract API and SYNO.FileStation.Compress API.

Overview

Availability: Since DSM 4.3

Version: 1

Method

list

Description:

List all background tasks including copy, move, delete, compress and extract tasks

Availability:

Since version 1

Request:

Parameter	Description	Value	Default Value	Availability
`offset`	Optional. Specify how many background tasks are skipped before beginning to return listed background tasks.	Integer	0	1 and later
`limit`	Optional. Number of background tasks requested. 0 indicates to list all background tasks.	Integer	0	1 and later
`sort_by`	Optional. Specify which information of the background task to sort on. Options include: **crtime**: creation time of the background task **finished**: Whether the background task is finished	crtime or finished	crtime	1 and later

Parameter	Description	Value	Default Value	Availability
sort_direction	Optional. Specify to sort ascending or to sort descending. Options include: **asc**: sort ascending **desc**: sort descending	asc or desc	asc	1 and later
api_filter	Optional. List background tasks with one or more given API name(s), separated by commas ",". If not given, all background tasks are listed. Options include: **SYNO.FileStation.CopyMove**: copy/move tasks **SYNO.FileStation.Delete**: delete tasks **SYNO.FileStation.Extract**: extract tasks **SYNO.FileStation.Compress**: compress tasks	SYNO.FileStation.CopyMove, SYNO.FileStation.Delete, SYNO.FileStation.Extract or SYNO.FileStation.Compress	(None)	1 and later

Example:

```
GET
/webapi/FileStation/background_task.cgi?api=SYNO.FileStation.BackgroundTask&version
=1&method=list
```

Response:

<data> object definitions:

Parameter	Type	Description	Availability
total	Integer	Total number of background tasks.	1 and later
offset	Integer	Requested offset.	1 and later
tasks	JSON-Style Array	Array of <background task> objects.	1 and later

<background task> object definition:

Parameter	Type	Description	Availability
api	String	Requested API name.	1 and later
version	String	Requested API version.	1 and later
method	String	Requested API method.	1 and later
taskid	String	A requested unique ID for the background task.	1 and later
finished	Boolean	Whether or not the background task is finished.	1 and later
params	JSON-Style Object	<params> object. Requested parameters in JSON format according to start method of individual API of the background task.	1 and later
path	String	A requested path according to start method of individual API of the background task.	1 and later

Parameter	Type	Description	Availability
processed_num	Interger	A number of processed files according to the response of status method of individual API of the background task.	1 and later
processed_size	Interger	A processed byte size according to the response of status method of individual API of the background task.	1 and later
processing_path	String	A processing file path according to the response of status method of individual API of the background task.	1 and later
total	Interger	A total number/byte size according to the response of status method of individual API of the background task. If API doesn't support it, the value is always -1.	1 and later
progress	Double	A progress value whose range between 0~1 according to the response of status method of individual API of the background task. If API doesn't support it, the value is always 0.	1 and later
taskid	<favorite additional> object	A unique ID according to the response of start method of individual API of the background task.	1 and later

<params> object definition:

Requested parameters in JSON format. Please refer to start method in each API.

Example:

```
{
        "tasks": [{
                "api": "SYNO.FileStation.CopyMove",
                "crtime": 1372926088,
                "finished": true,
                "method": "start",
                "params": {
                        "accurate_progress": true,
                        "dest_folder_path": "/video/test",
                        "overwrite": true,
                        "path": ["/video/test2/test.avi"],
                        "remove_src": false
                },
                "path": "/video/test2/test.avi",
                "processed_size": 12800,
                "processing_path": "/video/test2/test.avi",
```

```
                "progress": 1,
                "taskid": "FileStation_51D53088860DD653",
                "total": 12800,
                "version": 1
        }, {
                "api": "SYNO.FileStation.Compress",
                "crtime": 1372926097,
                "finished": true,
                "method": "start",
                "params": {
                        "dest_file_path": "/video/test/test.zip",
                        "format": "zip",
                        "level": "",
                        "mode": "",
                        "password": "",
                        "path": ["/video/test/test.avi"]
                },
                "progress": 0,
                "taskid": "FileStation_51D53091A82CD948",
                "total": -1,
                "version": 1
        }, {
                "api": "SYNO.FileStation.Extract",
                "crtime": 1372926103,
                "finished": true,
                "method": "start",
                "params": {
                        "create_subfolder": false,
                        "dest_folder_path": "/video/test",
                        "file_path": ["/video/test/test.zip"],
                        "keep_dir": true,
                        "overwrite": false
                },
                "progress": 1,
                "taskid": "FileStation_51D530978633C014",
                "total": -1,
```

```
                "version": 1
            }, {
                "api": "SYNO.FileStation.Delete",
                "crtime": 1372926110,
                "finished": true,
                "method": "start",
                "params": {
                    "accurate_progress": true,
                    "path": ["/video/test/test.avi"]
                },
                "path": "/video/test/test.avi",
                "processed_num": 1,
                "processing_path": "/video/test/test.avi",
                "progress": 1,
                "taskid": "FileStation_51D5309EE1E10BD9",
                "total": 1,
                "version": 1
            }],
            "offset": 0,
            "total": 4
        }
```


API Error Code

No specific API error codes.

clear_finished

Description:

Delete all finished background tasks.

Availability:

Since version 1

Request:

Parameter	Description	Value	Default Value	Availability
taskid	Unique IDs of finished copy, move, delete, compress or extract tasks. Specify multiple task IDs by ",". If it's not given, all finished tasks are deleted.	String	(None)	1 and later

Example:

```
GET
/webapi/FileStation/background_task.cgi?api=SYNO.FileStation.BackgroundTask&version
=1&method=clear_finished
```

Response:

No specific response. It returns an empty success response if completed without error.

API Error Code

No specific API error codes.

Document Revision History

This table describes the changes to the Synology File Station Official API document.

Date	Note
2013-08-27	Initial release