Synology®

DSM Login Web API Guide

Last Apr 19, 2023

Table of Contents

Chapter 1: Introduction

The DSM Login Web API developer's guide explains how to perform DSM logins using Web API and expands your applications based on the APIs of Synology NAS, allowing your applications to interact with DSM or DSM Packages via HTTP/HTTPS requests and responses.

This document explains the basic guidelines on how to use APIs, which we suggest reading all the way through before you jump into the other API specifications.

Chapter 2: Getting Started

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

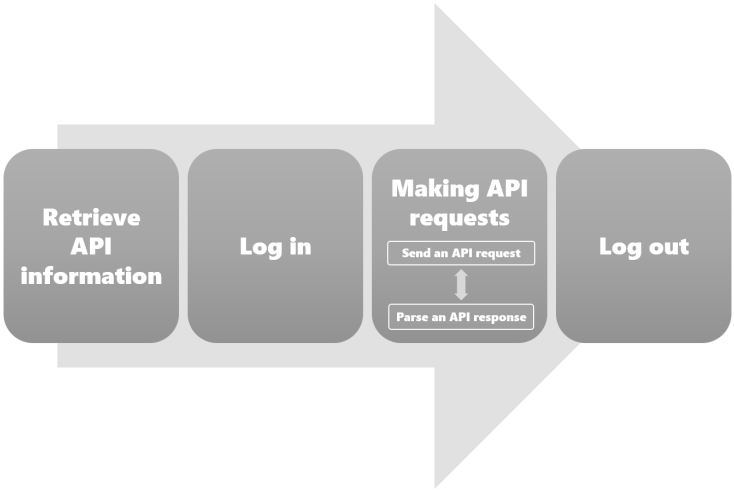
- API Workflow: Briefly introduces how API works.
- Making API Requests: Describes how to construct API requests.
- Parsing API Response: Describes how to parse API responses.
- Common Error Code: Lists of all common error codes that might be returned from all APIs.
- Working Examples: Provides File Station operations as examples.

All query examples are based on DSM 7.0 version.

API Workflow

The following five-step and easy-to-follow workflow show how to make your applications interact with APIs.

Step 1: Retrieve API Information



First, your application needs to retrieve API information from the target Synology NAS to know which APIs are available for use on the target Synology NAS. This information can be accessed simply through a request to /webapi/entry.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: Login

To make your application interact with Synology production, your application needs to log in with an account and password first. The login process is 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 Synology production APIs. In "Making API Requests", instructions on how to form a valid API request and how to decode response information will be given.

Step 4: Logout

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 API Requests

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

- **path**: Path of the API. The path information can be retrieved by requesting SYNO.API.Info.
- **api**: Name of the API requested.
- **version**: Version of the API requested.
- **method**: Method of the API requested.
- **_sid** (optional): If you already store **Cookies** on the login request with SYNO.API.Auth API and will pass this cookie on HTTP/HTTPS header, you may ignore this parameter. Otherwise, pass either HTTP/HTTPS or GET/POST method with _sid argument which value can be retrieved from the login response of SYNO.API.Auth API.
- **SynoToken** (optional): If the server enables **Improve protection against cross-site request forgery attacks**, pass SynoToken into the argument of which the value can be retrieved from the login response of SYNO.API.Auth API.

The syntax for the request is as follows:

```
GET /webapi/<PATH>?api=<API>&version=<VERSION>&method=<METHOD>[&<PARAMS>]
[&_sid=<SID>]
```

<PARAMS> represents the parameters for the requested method which is optional. You should look up requestFormat information that can be retrieved by SYNO.API.Info. If requestFormat indicates "JSON", the params value should be JSON encoded.

To make a request to the SYNO.API.Info API version 1 with the query method on your Synology NAS which address is https://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:

- **path**: entry.cgi
- **api**: SYNO.API.Info
- **version**: 1
- **method**: query

The request will look like:

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

Notes:

- 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/entry.cgi.

Parsing API Response

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

Key	Value	Description
success	true/false	"true": the request completes 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.

The 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 applies to all APIs, and 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 in an error is a JSON-Style Object which contains an error code and other information, such as a file path or name. Note that when there is no detailed information, this error element won't be responded.

Example 1

Respond to an invalid request without a method parameter:

Request:

```
https://myds.com:port/webapi/entry.cgi?api=SYNO.FileStation.Info&version=1
```

Failed Response:

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

Example 2

Respond to an invalid request due to an error code and to show more information about other errors.

Request:

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

Failed Response:

```
{
    "success":false,
    "error":{
        "code":1100,
        "errors":[{
            "code":408,
```

```
                "path":"/test/:"
            }]
        }
}
```

Example 3

Respond to a successful request and retrieve information from File Station.

Request:

```
https://myds.com:port/webapi/entry.cgi?api=SYNO.FileStation.Info&version=&method=get
```

Success Response:

```
{
    "success":true,
    "data": {
    "enable_list_usergrp": false,
    "hostname": "myds",
    "is_manager": true,
    "items": [
        {
            "gid": 100
        },
        {
            "gid": 101
        }
    ],
    "support_file_request": true,
    "support_sharing": true,
    "support_vfs": true,
    "support_virtual": {
        "enable_iso_mount": true,
        "enable_remote_mount": true
```

```
    },
    "support_virtual_protocol": [
        "cifs",
        "nfs",
        "iso"
    ],
    "system_codepage": "cht",
    "uid": 1026
    }
}
```


Notes:

- Only the data object is provided in the given response samples.

Common Error Codes

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

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 duplicated login.
108	Failed to upload the file.
109	The network connection is unstable or the system is busy.

110	The network connection is unstable or the system is busy.
111	The network connection is unstable or the system is busy.
112	Preserve for other purpose.
113	Preserve for other purpose.
114	Lost parameters for this API.
115	Not allowed to upload a file.
116	Not allowed to perform for a demo site.
117	The network connection is unstable or the system is busy.
118	The network connection is unstable or the system is busy.
119	Invalid session.
120-149	Preserve for other purpose.
150	Request source IP does not match the login IP.

Working Example

The following demonstrates a working example for requesting a file operation from the Synology NAS. To implement this example, simply replace the Synology NAS address used in the example (myds.com:port) with your Synology NAS 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/entry.cgi with SYNO.API.Info to get the SYNO.API.Auth API information for logging in.

Request:

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

Response:

```
{
    "data":{
        "SYNO.API.Auth": {
            "path": "entry.cgi",
```

```
            "minVersion": 1,
            "maxVersion": 7
        },
        "SYNO.FileStation.List": {
            "path": "entry.cgi",
            "requestFormat":"JSON"
            "minVersion": 1,
            "maxVersion": 2
        },
        "SYNO.VideoStation.Info": {
            "path": "VideoStation/info.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 to a DSM session by requesting SYNO.API.Auth API version 6 located at /webapi/entry.cgi.

Request:

```
https://myds.com:port/webapi/entry.cgi?
api=SYNO.API.Auth&version=6&method=login&account=<USERNAME>&passwd=
<PASSWORD>&enable_syno_token=yes
```

Response:


```
{
    "data": {
        "did": "8nC0nhJjgiE1XTqM6aKOS6-K1IIs6r-vHNpH72eUe-
XNSWs9OtF5c48EjaqXygEgvnEoARJJDWskZ656CVWl2w",
```

```
            "is_portal_port": false,
            "sid": "K5LlN6r-zkpxg61He2eSS2zIRrPf1aG7L7eGBjAsU8gd7gbtDEuYCtdOH1Y5Kgr-F3_rl86kYyzCzSxzwHGH90",
            "synotoken": "03yhfxW4syRQw"
        },
        "success": true
}
```


Step 3: Request a DSM API

After you have signed in a session, If synotoken is in login response, you should keep it and can continue to call the APIs of DSM listed in SYNO.API.Info. The SynoToken should always be appended on the request parameters. We show a sample of SYNO.FileStation.List API. 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:

```
https://myds.com:port/webapi/entry.cgi?api=SYNO.FileStation.List&version=1&method=list_share&SynoToken=03yhfxW4syRQw
```

Response:


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

```
    "success": true
}
```

It can be observed on the response list that there are two shared folders in File Station. If you're interested in learning more about the share list WebAPI query format, you can acquire the document from File Station.

Step 4: Logout

After you have finished the procedure, you should sign out of the current session. The session will be ended by calling the logout method in SYNO.API.Auth. If you don't use a cookie, log out the session by passing the _sid parameter. It is a good habit to sign out a session when you're done.

Example:

```
https://myds.com:port/webapi/entry.cgi?
api=SYNO.API.Auth&version=6&method=logout[&_sid=K5LlN6r-
zkpxg61He2eSS2zIRrPf1aG7L7eGBjAsU8gd7gbtDEuYCtdOH1Y5Kgr-F3_rl86kYyzCzSxzwHGH90]
```

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 log in and log out.

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/entry.cgi?api=SYNO.API.Info&version=1&method=query
```

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
requestFormat	If this value shows "JSON", use the JSON encoder for all other parameter values.	1 and later

Example:

```
{
    "data":{
        "SYNO.API.Auth": {
            "path": "entry.cgi",
            "minVersion": 1,
            "maxVersion": 7
        },
        "SYNO.FileStation.List": {
            "path": "entry.cgi",
            "requestFormat":"JSON"
            "minVersion": 1,
            "maxVersion": 2
        },
        "SYNO.VideoStation.Info": {
            "path": "VideoStation/info.cgi",
            "minVersion": 1,
            "maxVersion": 1
        },
        ...
    },
    "success": true
}
```

API Error Code

- No specific API error codes.

SYNO.API.Auth

Overview

- Availability: Since DSM 6.0
- Version: 3 - 7; 6 (Recommended)

Method

Login

Request:

Parameter	Description	Availability
account	Login account name.	3 and later
passwd	Login account password.	3 and later
session	(Optional) Login session name for DSM Applications.	3 and later
format	(Optional) 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.	3 and later
otp_code	(Optional) 2-factor authentication option with an OTP code. If it's enabled, the user requires a verification code to log into DSM sessions.	3 and later
enable_syno_token	(Optional) Obtain the CSRF token, also known as SynoToken , for the subsequent request. If set **no**, the server will not produce this token.	6 and later
enable_device_token	(Optional) Omit 2-factor authentication (OTP) with a device id for the next login request.	6 and later

device_name	(Optional) To identify which device can be omitted from 2-factor authentication (OTP), pass this value will skip it.	6 and later
device_id	(Optional) If 2-factor authentication (OTP) has omitted the same enabled device id, pass this value to skip it.	6 and later

Example 1: Login

```
GET /webapi/entry.cgi?api=SYNO.API.Auth&version=6&method=login&account=
<USERNAME>&passwd=<PASSWORD>&session=FileStation&format=cookie
```

Example 2: Login with OTP

```
GET /webapi/entry.cgi?api=SYNO.API.Auth&version=6&method=login&account=
<USERNAME>&passwd=<PASSWORD>&otp_code=<OTP_CODE>
```

Example 3: Login with OTP and to enable to omit 2-factor verification

```
GET /webapi/entry.cgi?api=SYNO.API.Auth&version=6&method=login&account=
<USERNAME>&passwd=<PASSWORD>&otp_code=
<OTP_CODE>&enable_device_token=yes&device_name=<DEVICE_NAME>
```

Example 4: Login with omitted OTP

```
GET /webapi/entry.cgi?api=SYNO.API.Auth&version=6&method=login&account=
<USERNAME>&passwd=<PASSWORD>&device_name=<DEVICE_NAME>&device_id=<DID>
```

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.	3 and later
did	A.k.a. device id, to identify which device.	6 and later
is_portal_port	Irrelevant.	6 and later

synotoken	(Optional) This token avoids Cross-Site Request Forgery. Each subsequent API request should provide a request parameter SynoToken=<synotoken> along with other parameters.	6 and later

Example:

```
{
    "data": {
        "did": "8nC0nhJjgiE1XTqM6aKOS6-K1IIs6r-vHNpH72eUe-XNSWs9OtF5c48EjaqXygEgvnEoARJJDWskZ656CVWl2w",
        "is_portal_port": false,
        "sid": "K5LlN6r-zkpxg61He2eSS2zIRrPf1aG7L7eGBjAsU8gd7gbtDEuYCtdOH1Y5Kgr-F3_rl86kYyzCzSxzwHGH90",
        "synotoken": "03yhfxW4syRQw"
    },
    "success": true
}
```

Logout

Request:

- No specific other parameters.

Example:

```
GET /webapi/entry.cgi?api=SYNO.API.Auth&version=6&method=logout
```

Response:

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

Query Syno Token

This API should be queried via Javascript, and store the SynoToken in Javascript variables. If you reload your web page, SynoToken should be queried again.

Request:

- No specific other parameters.

Example:

```
GET /webapi/entry.cgi?api=SYNO.API.Auth&version=6&method=token
```

Response:

- <data> object definitions:

Parameter	Description	Availability
is_portal_port	Irrelevant.	6 and later
synotoken	(Optional) This token avoids Cross-Site Request Forgery. Each subsequent API request should provide a request parameter SynoToken=<synotoken> along with other parameters.	6 and later

API Error Codes

Code	Description
400	No such account or incorrect password.
401	Disabled account.
402	Denied permission.
403	2-factor authentication code required.
404	Failed to authenticate 2-factor authentication code.
406	Enforce to authenticate with 2-factor authentication code.
407	Blocked IP source.
408	Expired password cannot change.
409	Expired password.
410	Password must be changed.

Copyright and Disclaimer Notices

Synology Inc.
9F, No. 1, Yuandong Rd.
Banqiao Dist., New Taipei City 220632
TAIWAN

THE WARRANTY AND REMEDIES SET FORTH ABOVE ARE EXCLUSIVE AND IN LIEU OF ALL OTHERS, ORAL OR WRITTEN, EXPRESS OR IMPLIED. No Synology dealer, agent, or employee is authorized to make any modification, extension, or addition to this warranty.

Some states do not allow the exclusion or limitation of implied warranties or liability for incidental or consequential damages, so the above limitation or exclusion may not apply to you. This warranty gives you specific legal rights, and you may also have other rights which vary from state to state.