Synology Alarm Clock
Web API

Ver. 1.0

Table of Contents

1. Concept

Synology Alarm Clock provides a programmable interface allowing the 3rd party integrator/installer to develop application that is highly integrated with Synology Alarm Clock. This interface is called “Synology Alarm Clock Web API”, refer to Figure 1-1 for the entire structure:

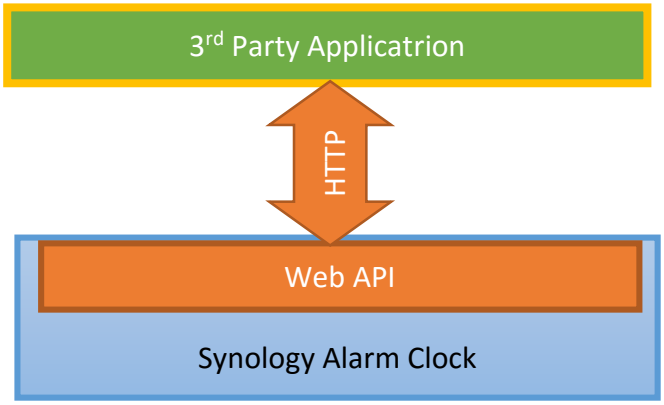

Figure 1-1 Synology Alarm Clock Web API architecture

Synology Alarm Clock Web API provides a set of API interface allowing the 3rd party application to interact with Synology Alarm Clock via HTTP Request/Response call.

Web API is based on HTTP protocol, and Request/Response as the communication structure.

1.1. Request

Use HTTP GET or POST to send the request with API’s URL.

Usage:

```
GET  http://<host:port>/webman/3rdparty/AlarmClock/api.cgi?
     action=<METHOD>&<PARAM LIST>
```

1.2. Response

After receiving the request, API will return the response to the 3rd party application in JSON format.

1.3. Status object

For most operations, the special <status object> is returned.

It is the JSON-encoded object with the following attributes:

Name	Type	Description
status	<string>	Status of operation: • OK • ERROR
data	<string>	Some data as operation result. See specific operation description for details. Error message, if status is ERROR.
stack	<string>	Debug stack trace for error, if status is ERROR

Example:

```
{
  "status": "OK",
  "data": "",
  "stack": ""
 }
```

```
{
  "status": "ERROR",
  "stack": " . . . some error messages . . ."
  "data": "",
 }
```

2. Methods

2.1. Methods list

The following table is the overview of all APIs defined in this section:

Method name	Description
Login	Log in
get_tasks	…
save_tasks	…
get_devices	…
get_playlists	…
get_allinone	…
startplay	…
playstatus	…
stopplay	…
get_log	…
clear_log	…

2.2. Log in

Request:

Parameter	Type	Description
action	<string>	Should be "**login**"
login	<string>	Account name with administrator permissions. See details: http://www.nasalarmclock.com/synologydescription/account_tips/
password	<string>	Password for account

GET or POST request allowed.

Example:

```
GET …/api.cgi?action=login&login=admin&password=xxxxxx
```

Response:

JSON status structure

Name	Type	Description
status	<string>	Status of operation: • OK – if login passed • ERROR – if login failed
stack	<string>	Empty
data	<string>	If login passed, session ID is returned here. If login failed, here is message with error details.

Example:

```
{
  "status": "OK",
  "data": "e1qwdIfy6Z7Cc",
  "stack": ""
 }
```

```
{
  "status": "ERROR",
  "data": "Login or Password is not correct",
  "stack": ""
 }
```

2.3. Get Task List

Request:

Parameter	Type	Description
action	<string>	Should be "**get_tasks**"

GET or POST request allowed.

Example:

```
GET …/api.cgi?action=get_tasks
```

Response:

JSON structure for task list:

Name	Type	Description
status	<string>	Operation status. See <status object> description for details.
stack	<string>	Error stack trace. See <status object> description for details.
data	Collection of *<task object>*	Collection of scheduled tasks.

<task object> - Scheduled alarm clock task description

Name	Type	Description
id	<string>	Task unique id
on	<integer>	0 – task is off 1 – task is on
schedule	<string>	String, containing number of days, when task should trigger. 1 – Monday … 6 – Saturday, 7 - Sunday
hour	<integer>	Scheduled time - hour in 24-hour format: 0 .. 23
minute	<integer>	Scheduled time - minute: 00 .. 59
playlist	<string>	Playlist id – received from AudioStation. See <playlist object> description for details.
device	<string>	Playing device name – received from AudioStation. See <device object> description for details.
volume1	<integer>	The initial volume value in %: 0 .. 100
volume2	<integer>	The final volume value in %: 0 .. 100
volumetime	<float>	Volume increase duration, in minutes. Decimal values are also allowed.
duration	<float>	Playback duration, in minutes. Decimal values are also allowed.

Example:


```
{
  "status": "OK",
  "stack": "",
  "data": [
   {
    "id": "1C8FB3EA-C960-4723-B10A-1D19E853DF16",
    "on": "1",
    "schedule": "1,2,3,4,5,6",
    "hour": "7",
    "minute": "0",
    "playlist": "playlist_shared_normal/174",
    "device": "CEOL piccolo (AirPlay)",
    "duration": "6",
    "volumetime": 3,
    "volume1": 10,
    "volume2": 30
   },
   {
    "id": "DCB2B99F-EA73-4806-8D09-96C1D187699D",
    "on": "0",
    "schedule": "7",
    "hour": "7",
    "minute": "30",
    "playlist": "playlist_shared_normal/137",
    "device": "RX-V671",
    "volumetime": 2,
    "duration": 5,
    "volume1": 5,
    "volume2": 30
   }
  ]
 }
```


2.4. Save Task List

Request:

JSON object:

Parameter	Type	Description
action	<string>	Should be "**save_tasks**"
data	Collection of *<task object>*	Collection of scheduled tasks to save. See *"Get Scheduled Task List"* method for detailed task object description.

Only POST request allowed.

Example:

```
 POST …/api.cgi

{
 "action": "save_tasks",
 "data":[
  {,
   "id": "1C8FB3EA-C960-4723-B10A-1D19E853DF16",
   "on": "1",
   "schedule": "1,2,3,4,5,6",
   "hour": "7",
   "minute": "0",
   "playlist": "playlist_shared_normal/174",
   "device": "CEOL piccolo (AirPlay)",
   "duration": "6",
   "volumetime": 3,
   "volume1": 10,
   "volume2": 30
  },
  {
   "id": "DCB2B99F-EA73-4806-8D09-96C1D187699D",
   "on": "0",
   "schedule": "7",
   "hour": "7",
   "minute": "30",
   "playlist": "playlist_shared_normal/137",
   "device": "RX-V671",
   "volumetime": 2,
   "duration": 5,
   "volume1": 5,
   "volume2": 30
  }
 ]
}
```

Response:

JSON status structure *(see above).*

Example:

```
{
  "status": "OK",
  "data": "",
  "stack": ""
 }
```

2.5. Get devices list

Request:

Parameter	Type	Description
action	<string>	Should be "**get_devices**"

GET or POST request allowed.

Example:

```
GET …/api.cgi?action=get_devices
```

Response:

JSON structure for devices list:

Name	Type	Description
success	<boolean>	Operation status. *true* if operation is ok. *false* if operation is failed.
data	container	

Name	Type	Description
players	Collection of *<player object>*	Collection of playing devices: DLNA, AirPlay, USB, Bluetooth speakers.

<player object> - Device (player) description

Name	Type	Description
id	<string>	Device unique id
name	<string>	Device display name
type	*<string>*	*not used*
is_multiple	*<boolean>*	*not used*
password_protected	*<boolean>*	*not used*
support_seek	*<boolean>*	*not used*

Example:

```
{
   "success":true,
   "data":{
      "players":[
         {
            "id":"uuid:9ab0c000-f668-11de-9976-00a0de865641",
            "is_multiple":false,
```

```
            "name":"RX-V671",
            "password_protected":false,
            "support_seek":false,
            "support_set_volume":true,
            "type":"upnp"
         },
         {
            "id":"uuid:5f9ec1b3-ff59-19bb-8530-0005cd33a83b",
            "is_multiple":false,
            "name":"CEOL piccolo (DLNA)",
            "password_protected":false,
            "support_seek":true,
            "support_set_volume":true,
            "type":"upnp"
         },
         {
            "id":"0005CD33A83B",
            "is_multiple":false,
            "name":"CEOL piccolo (AirPlay)",
            "password_protected":false,
            "support_seek":true,
            "support_set_volume":true,
            "type":"airplay"
         }
      ]
   }
}
```


2.6. Get playlists list

Request:

Parameter	Type	Description
action	<string>	Should be "**get_playlists**"

GET or POST request allowed.

Example:

```
GET …/api.cgi?action=get_playlists
```

Response:

JSON structure for playlist list:

Name	Type	Description
success	<boolean>	Operation status. • *true* if operation is ok. • *false* if operation is failed.
data	container	

Name	Type	Description
offset	<integer>	*not used*
total	<integer>	Total playlists count
playlists	Collection of *<playlist object>*	Collection of playlists, created in AudioStation.

<playlist object> - Playlist description

Name	Type	Description
id	<string>	Playlist unique id
name	<boolean>	Playlist display name
type	*<string>*	*not used*
library	*<string>*	*not used*
sharing_status	*<string>*	*not used*

Example:


```
{
   "success":true,
   "data":{
      "offset": 0,
      "total":2,
      "playlists":[
         {
            "id":"playlist_shared_normal/146",
            "library":"shared",
            "name":"Beatles - A Hard Days Night",
            "sharing_status":"none",
            "type":"normal"
         },
         {
            "id":"playlist_shared_normal/157",
            "library":"shared",
            "name":"Beatles - Abbey Road",
            "sharing_status":"none",
            "type":"normal"
         }
      ]
   }
}
```

2.7. Get playlists list, devices list and task list in one package

Request:

Parameter	Type	Description
action	<string>	Should be "**get_allinone**"

GET or POST request allowed.

Example:

```
GET …/api.cgi?action=get_allinone
```

Response:

JSON structure:

Name	Type	Description
status	<string>	Operation status. See <status object> description for details.
stack	<string>	Error stack trace. See <status object> description for details.
data	container	

Name	Type	Description
tasks	Collection of *<task object>*	*see details in "Get Task List" method description.*
players	Collection of *<player object>*	more compact player description
playlists	Collection of *<playlist object>*	more compact playlist description

<player object> - Device (player) description

Name	Type	Description
id	<string>	Device unique id
name	<string>	Device display name
type	*<string>*	*not used*

<playlist object> - Playlist description

Name	Type	Description
id	<string>	Playlist unique id
name	<boolean>	Playlist display name

Example:


```
{
 "status": "OK",
 "stack": "",
 "data": {

   "players": [
    {
     "name": "CEOL piccolo (DLNA)",
     "type": "upnp",
     "id": "uuid:5f9ec1b3-ff59-19bb-8530-0005cd33a83b"
    }
   ],

   "tasks": [
    {
     "on": "1",
     "playlist": "playlist_shared_normal/174",
     "hour": "7",
     "schedule": "1,2,3,4,5,6",
     "volumetime": 3,
     "duration": "6",
     "volume1": 10,
     "volume2": 30,
     "device": "CEOL piccolo (AirPlay)",
     "id": "1C8FB3EA-C960-4723-B10A-1D19E853DF16",
     "minute": "0"
    }
   ],

   "playlists": [
    {
     "id": "playlist_shared_normal/146",
     "name": "Beatles - A Hard Days Night"
    }
   ]

  }
}
```

2.8. Start demo play

Request:

Parameter	Type	Description
action	<string>	Should be "**startplay**"
guid	<string>	Unique ID for playing task. Generated by 3rd party app and used then for receiving status or controlling playback task.

Only GET request allowed.

Example:

```
GET …/api.cgi?action=startplay&guid=B2AF4EC4-EFEC-40A7-8171
```

Response:

JSON status structure *(see above).*

2.9. Get play status

Request:

Parameter	Type	Description
action	<string>	Should be "**playstatus**"
guid	<string>	Unique ID for playing task, that was used for start playing.

Only GET request allowed.

Example:

```
GET …/api.cgi?action=playstatus&guid=B2AF4EC4-EFEC-40A7-8171
```

Response:

JSON status structure *(see above).*

Name	Type	Description
status		
stack		
data	container	Contains playing status object

Playing status structure:

Name	Type	Description
status	<integer>	Current playing status: • 1 – preparing to play • 2 – playing • 3 – waiting for next song • 4 – playback finished
time1	<string>	Current playback time, in format: *mm:ss*
time2	<string>	Total playback time, in format: *mm:ss* see *duration* attribute in *task object*
volume	<integer>	Current volume, in %: 0 .. 100
song	<string>	Display name of song, that is playing now
player	<string>	Player ID

Example:


```
{
   "status":"OK",
   "data":"{
     \"status\":2,
     \"song\":\"Good morning!\",
     \"time1\":\"01:25\",
     \"time2\":\"05:00\",
     \"volume\":40,
     \"player\":"uuid:9ab0c000-f668-11de-9976-00a0de865641"
     }",
   "stack":""
}
```


2.10. Stop play

Request:

Parameter	Type	Description
action	<string>	Should be "**stopplay**"
playerid	<string>	Unique ID of player. Note! Here used player id, not play task id

Only GET request allowed.

Example:

```
  GET …/api.cgi?action=stopplay&playerid=uuid:9ab0c000-f668-11de-9976-
00a0de865641
```

Response:

JSON status structure *(see above).*

2.11. Get log file content

Request:

Parameter	Type	Description
action	<string>	Should be “get_log”

GET or POST request allowed.

Example:

```
GET …/api.cgi?action=get_log
```

Response:

Raw content of log file

2.12. Clear log file content

Request:

Parameter	Type	Description
action	<string>	Should be “clear_log”

GET or POST request allowed.

Example:

```
GET …/api.cgi?action=clear_log
```

Response:

JSON status structure

Example:

```
{
  "status": "OK",
  "data": "",
  "stack": ""
 }
```