OpenSprinkler Firmware 2.1.7 API Document

(Nov 6*, 2016*)

1. Overview

This document describes the OpenSprinkler Firmware 2.1.7 API, including JSON and HTTP GET commands.

- **Password is required** for all commands (the **pw** parameter). Your device password is referred to as **xxx.**
  - **The password is MD5 hashed by the app / web UI and therefore when using the API commands the password should be MD5 hashed too (all lower-case). You can find online MD5 hash tools to convert plaintext password to MD5.**
- In the following, OpenSprinkler's IP address is referred to as **os-ip**.
- For most commands, **parameters are optional** and the order of parameters does not matter. Parameters that do not appear in the command remain unchanged. Exceptions are the binary-value parameters explained in **Section 5**.
- **Some features (particularly IFTTT notifications)** are only available for OpenSprinkler 2.3 or above, and Linux-based OpenSprinkler (e.g. OpenSprinkler Pi). These features are NOT available for OpenSprinkler 2.0, 2.1, and 2.2.
- **Return Values** are all formatted in JSON. For example **{"fwv": 217}, {"result": 1},** etc. **Return Error Codes:**
  - **{"result":1}** **Success**
  - **{"result":2}** **Unauthorized** (e.g. missing password or password is incorrect)
  - **{"result":3}** **Mismatch** (e.g. new password and confirmation password do not match)
  - **{"result":16}** **Data Missing** (e.g. missing required parameters)
  - **{"result":17}** **Out of Range** (e.g. value exceeds the acceptable range)
  - **{"result":18}** **Data Format Error** (e.g. provided data does not match required format)
  - **{"result":19}** **RF code error** (e.g. RF code does not match required format)
  - **{"result":32}** **Page Not Found** (e.g. page not found or requested file missing)
  - **{"result":48}** **Not Permitted** (e.g. cannot operate on the requested station)

2. Get Controller Variables [Keyword /jc]

/jc?pw=xxx

Return Variables:

- **devt**: Device time (epoch time). This is always the local time.
- **nbrd**: Number of 8-station boards (including main controller).
- **en**: Operation enable bit (same as the 'den' option in Section 4).
- **rd**: Rain delay bit (1: rain delay is currently in effect; 0: no rain delay).
- **rs**: Rain sensor status bit (1: rain is detected from rain sensor; 0: no rain detected).
- **rdst:** Rain delay stop time (0: rain delay not in effect; otherwise: the time when rain delay is over).
- **loc**: Location string (e.g. GPS location, Wunderground PWS station ID).
- **wtkey**: Wunderground API key.
- **sunrise**: Today's sunrise time (number of minutes from midnight).
- **sunset**: Today's sunset time (number of minutes from midnight).
- **eip**: external IP, calculated as (ip[3]<<24)+(ip[2]<<16)+(ip[1]<<8)+ip[0]
- **lwc**: last weather call/query (epoch time)
- **lswc**: last successful weather call/query (epoch time, a value of 0 means it did not receive a successful response)
- **sbits**: Station status bits. Each byte in this array corresponds to an 8-station board and represents the bit field (LSB). For example, 1 means the 1st station on the board is open, 192 means the 7th and 8th stations are open.
- **ps**: Program status data: each element is a 3-field array that stores the **[pid,rem,start]** of a station, where **pid** is the program index (0 means none), **rem** is the remaining water time (in seconds), **start** is the start time. If a station is not running (sbit is 0) but has a non-zero pid, that means the station is in the queue waiting to run.

- **lrun**: Last run record, which stores the [**station index, program index, duration, end time**] of the last run station.
- **curr**: total current draw of all zones (in milliamps). This is only available for controllers with current sensing capability.
- **flwrt**: flow count window in unit of seconds (the firmware defines this as 30 seconds by default).
- **flcrt**: real-time flow count (i.e. number of flow sensor clicks during the last flwrt seconds).
- **wto**: weather options (such as the weights and baseline for the **t**emperature, **h**umidity, **r**ainfall parameters). As this is stored on the uSD card, the return value will be empty for controllers that do not have uSD card.
- **ifkey**: IFTTT Make channel key (*only available for OS 2.3 or above, and Linux-based OpenSprinkler*).

Example Return:

{"devt":1478432421,"nbrd":1,"en":1,"rd":0,"rs":0,"rdst":0,"loc":"pws:KMAAMHER18","wtkey":"xxxxxxxx","sunrise":390,"sunset":996,"eip":xxxxxxxx,"lwc":1478432207,"lswc":1478432207,"lrun":[0,0,0,0],"curr":0,"flcrt":0,"flwrt":30,"sbits":[0,0],"ps":[[0,0,0],[0,0,0],[0,0,0],[0,0,0],[0,0,0],[0,0,0],[0,0,0],[0,0,0]],"wto":{"h":100,"t":100,"r":100,"bh":30,"bt":70,"br":0},"ifkey":"xxxxxxxx"}

3. Change Controller Variables [Keyword /cv]

/cv?pw=xxx&rsn=x&rbt=x&en=x&rd=x&re=x&update=x

Parameters:

- **rsn**: Reset all stations (including those waiting to run). The value doesn't matter: action is triggered if parameter appears.
- **rbt**: Reboot the controller. Binary value (a value of 1 triggers this action).
- **en**: Operation enable. Binary value.
- **rd**: Set rain delay time (in hours). Range is 0 to 32767. A value of 0 turns off rain delay.
- **re**: Set the controller to be in remote extension mode (a value of 1 turns on remote extension mode).

Examples:

- http://os-ip/cv?pw=xxx&rbt=1 (reboot controller)
- http://os-ip/cv?pw=xxx&en=0 (disable operation)
- http://os-ip/cv?pw=xxx&rd=24 (set a 24-hour rain delay)

4. Get Options [Keyword /jo]

/jo?pw=xxx

Return Variables:

- **fwv**: Firmware version (217 means Firmware 2.1.7).
- **fwm**: Firmware minor version (increments with minor revisions to the firmware)
- **tz**: Time zone (floating point time zone value * 4 + 48). For example, GMT+0:00 is 48; GMT-4:00 is 32, GMT+9:30 is 86. Acceptable range is 0 to 108.
- **ntp**: Use NTP sync. **Binary** value.
- **dhcp**: Use DHCP. **Binary** value.
- **ip{1,2,3,4}**: Static IP (ignored if dhcp=1).
- **gw{1,2,3,4}**: Gateway (router) IP (ignored if dhcp=1).
- **dns{1,2,3,4}:** DNS IP (ignored if dhcp=1).
- **ntp{1,2,3,4}**: NTP server IP (ignored if ntp=0).
- **hp{0,1}**: The lower and upper bytes of the HTTP port number. So http_port=(hp1<<8)+hp0.
- **hwv**: Hardware version.
- **hwt**: Hardware type. Values are as follows: 0xAC (172) = AC power type, 0xDC (220) = DC power type.

- **ext**: Number of 8-station expansion boards (not including the main controller).
- **sdt**: Station delay time (in seconds). Acceptable range is -600 to +600 seconds (-10 to +10 minutes), in steps of 5 seconds.
- **mas/mas2**: Master stations 1 and 2 index (a value of 0 means none). This firmware supports up to 2 master stations.
- **mton/mton2**: Master 1 and 2 on adjusted time (in steps of 5 seconds). Acceptable range is 0 to 600 (note: positive).
- **mtof/mtof2**: Master 1 and 2 off adjusted time (in steps of 5 seconds). Acceptable range is -600 to 0 (note: negative).
- **urs**: Attached sensor type. 0: not using sensor; 1: rain sensor; 2: flow sensor; 0xF0 (240): program switch.
- **rso**: Rain sensor type. **Binary** value. 0: normally closed; 1: normally open.
- **wl**: Water level (i.e. % Watering). Acceptable range is 0 to 250.
- **den**: Operation enable bit. **Binary** value. To change this option, you must use the /cv command in Section 3.
- **ipas**: Ignore password. **Binary** value.
- **devid**: Device ID.
- **con/lit/dim**: LCD contrast / backlight / dimming values.
- **bst**: Booster time for DC-powered controller (in milliseconds). Acceptable range is 0 to 1000 (in steps of 4 milliseconds).
- **uwt**: Weather adjustment method. 0: manual adjustment; 1: Zimmerman method. Water restriction setting (for California) is indicated by bit 7.
- **lg**: Enable logging. **Binary** value.
- **fpr{0,1}**: flow pulse rate (scaled by 100) lower/upper byte. The actual flow pulse rate is ((fpr1<<8)+fpr0)/100.0
- **re**: Remote extension mode. **Binary** value.
- **dexp/mexp**: Detected/maximum number of zone expansion boards (-1 means cannot auto-detect).
- **sar**: Special station auto-refresh. **Binary** value. When enabled, special stations (e.g. RF or HTTP stations) will be automatically refreshed to make sure they are in sync with the controller status.
- **ife**: IFTTT events enable. This is a bit field. When a bit is set to 1, an IFTTT notification is sent upon that event.
    - bit 0: Program scheduled
    - bit 1: Rain sensor status update
    - bit 2: Flow sensor status update
    - bit 3: Weather update (e.g. water level change, external IP change)
    - bit 4: Reboot
    - bit 5: Station run

Example Return:

{"fwv":217,"tz":28,"ntp":1,"dhcp":0,"ip1":192,"ip2":168,"ip3":1,"ip4":34,"gw1":192,"gw2":168,"gw3":1,"gw4":1,"hp0":144,"hp1":31,"hwv":23,"ext":0,"sdt":0,"mas":1,"mton":5,"mtof":-5,"urs":2,"rso":1,"wl":100,"den":1,"ipas":0,"dim":20,"bst":320,"uwt":0,"ntp1":50,"ntp2":97,"ntp3":210,"ntp4":169,"lg":1,"mas2":0,"mton2":0,"mtof2":0,"fwm":0,"fpr0":100,"fpr1":0,"re":0,"dns1":8,"dns2":8,"dns3":8,"dns4":8,"sar":1,"ife":33,"reset":0,"dexp":0,"mexp":6,"hwt":220}

5. Change Options [Keyword /co]

/co?pw=xxx&o?=x&loc=x&wtkey=x&wto=x&ifkey=x&ttt=x

IMPORTANT: parameters displayed in **shaded cells** below are binary, and they will be set to 1 as long as the parameter name appears in the command, regardless of the value; conversely, they are set to 0 if the parameter does not appear. For example, **o21=1**, **o21=0**, **o21=on** all set **o21** to 1. Conversely, if you don't provide the parameter **o21**, it will be cleared to 0.

Parameters:

- **loc**: Location string (url encoded). This can be city,state name, zip code, GPS coords, PWS station ID. Every time you use the app to set the location, the app will automatically change it to GPS cords (except for PWS station ID).
- **wtkey**: Wunderground API key.

- **wto**: Weather options in JSON form and stripped of the terminal braces (url encoded). Zimmerman: Has three parameters for adjusting weights and baseline parameters of temp (t), humidity (h) and rain (r).
- **ifkey**: IFTTT Maker channel key (*only available for OS 2.3 or above, and Linux-based OpenSprinkler*).
- **ttt**: Set time manually (epoch time, ignored if ntp=1).
- **o?**: ? is the option index. This corresponds to the same list of options in Section 4 (there the options are listed using JSON names). Please refer to that section for detailed explanations and acceptable range of values.

Idx	JSON	Meaning	Idx	JSON	Meaning	Idx	JSON	Meaning
o1	tz	Time zone	o19	mton	Master on adjusted time	o30	bst	Boost time for DC
o2	ntp	Use NTP	o20	mtof	Master off adjusted time	o31	uwt	Weather method
o3	dhcp	Use DHCP	o21	urs	Attached sensor type	o32-35	ntp1-4	NTP IP
o4-7	ip1-4	Static IP	o22	rso	Rain sensor type	o36	lg	Enable logging
o8-11	gw1-4	Gateway IP	o23	wl	Water level/percentage	o37	mas2	Master2
o12-13	hp0-1	HTTP port	o24	*(not applicable, use /cv)*		o38	mton2	Master2 on delay
o14	*(hardware version, read-only)*		o25	ipas	Ignore password	o39	mtof2	Master 2 off delay
o15	ext	# expansion boards	o26	devid	Device ID	o40	fwm	Fw minor version
o16	*(obsolete)*		o27	con	LCD contrast	o41-42	fpr0-1	Flow pulse rate
o17	sdt	Station delay time	o28	lit	LCD backlight	o43	*(not applicable, use /cv)*	
o18	mas	Master station	o29	dim	LCD dimming	o44-47	dns1-4	DNS IP
						o48	sar	Special auto refresh
						o49	ife	IFTTT events enable

Examples:

- http://os-ip/co?pw=xxx&loc=95050 (change location to zip code 95050)
- http://os-ip/co?pw=xxx&wtkey=abc (change Wunderground API key to abc)
- http://os-ip/co?pw=xxx&o12=144&o13=31 (change HTTP port to 8080, i.e. 31*256+144=8080)
- http://os-ip/co?pw=xxx&o17=30 (set station delay time to 30 seconds)
- http://os-ip/co?pw=xxx&o21=on&o22=on (use rain sensor, and rain sensor type is normally open)

(***In the above example, binary parameters whose names do not appear in the command will all be set to 0***)

6. Set Password [Keyword/sp]

/sp?pw=xxx&npw=xxx&cpw=xxx

Parameters:

- **npw**: New password (should be in MD5 hash).
- **cpw**: Confirmation (should be in MD5 hash).

Examples:

- http://os-ip/sp?pw=xxx&npw=e0ff85143dfa717536cbb668cc8f8e8b&cpw=e0ff85143dfa717536cbb668cc8f8e8b (change password to the MD5 hash of 'sprinkler')

Return Error Code:

- Refer to Section 1 for error codes. In particular, if npw or cpw is missing, the controller will return Data Missing error; if npw and cpw do not match, the controller will return Mismatch error.

7. Get Station Names and Attributes [Keyword /jn]

/jn?pw=xxx

Return Variables:

- **snames**: Array of station names.
- **maxlen**: Maximum number of characters allowed in each name.
- **masop/masop2**: Master/Master2 operation flag (LSB bit field). One byte per board. For example, 254 means the 1st station on this board does not use master, and all other stations on this board use master.
- **ignore_rain**: Ignore rain flag (bit field). Defined similarly to masop. For example, 192 means the 7th and 8th stations on this board ignore rain, and the other stations on this board do not.
- **stn_dis**: Station disable flag (bit field).
- **stn_seq**: Station sequential flag (bit field).
- **stn_spe**: Special station flag (bit field). If marked 1, it means this station is a special station (e.g. RF or remote).

Example Return:

{"snames":["Master","1. Front Yard","2. Back Yard","3. Left Lawn","4. Right Lawn","5. Flower Bed","Landscape Light","Garage Door"],"masop":[254],"ignore_rain":[0],"masop2":[0],"stn_dis":[0], "rfstn":[0], "stn_seq":[0],"maxlen":24}

8. Get Special Station Data [Keyword /je]

/je?pw=xxx

Return Data Format:

{"0":{"st":xxx, "sd":"xxx"}, "1":{"st":xxx, "sd":"xxx"}, ...}

The integer (0, 1, ...) corresponds to the station id (only stations whose stn_spe bits are 1 would appear in the list). "st" indicates the special station type (see list below), and "sd" stores the corresponding special station data:

- st=0: standard station (this should never occur because only stations with st!=0 should appear in the list)
- st=1: RF (radio frequency) station. sd (16 characters) stores the 16-digit hex RF code.
- st=2: remote station. sd stores the 14-digit hex code of the remote station in the following order: ip address (8 bytes), port number (4 bytes), station index (2 bytes).
- st=3: GPIO station. sd stores 3 bytes: the first two define the GPIO index, the third byte indicates active state (1 or 0).
- st=4: HTTP station. sd stores up to 240 bytes, comma separated HTTP GET command with the following data: 1) server name (can be either a domain name or IP address), port number, on command, off command.

Note: GPIO and HTTP stations are *only available for OS 2.3 or above, and Linux-based OpenSprinkler*)

9. Change Station Names and Attributes [Keyword /cs]

/cs?pw=xxx&s?=xxx&m?=xxx&i?=xxx&n?=xxx&d?=xxx&q?=xxx&p?=xxx&sid=xxx&st=xxx&sd=xxx

Parameters:

- **s?**: Station name. ? is the station index (starting from 0). For example, **s0=abc** assigns name 'abc' to the first station.
- **m?:** Master operation flag bit field. ? is the board index (starting from 0). Specifically, m0 is the main controller, m1 is the first expansion board, and so on. Refer to the **masop** variable in Section 7 above. For example, **m0=255** sets all stations on the main controller to use master; **m1=4** sets the 3rd station on the first expansion board to use master.
- **n?:** Master 2 operation flag bit field. Similar to m?, except this is for Master Station 2.
- **i?/d?/q?/p?**: Station ignore_rain/disable/sequential/special flag bit field.
- **sid, st, sd:** Station special data: sid is the station index, st is the special type, sd is the corresponding data. See Section 8.

Examples:

- http://os-ip/cs?pw=xxx&s0=Front%20Lawn&s1=Back%20Lawn (set the name of the first two stations)
- http://os-ip/cs?pw=xxx&m0=127&s7=Garage (set name for station s7 and master operation bits for board m0)
- http://os-ip/cs?pw=xxx&sid=2&st=1&sd=00553300553c001c (set special station data for station s2)

10. Get Station Status [Keyword /js]

/js?pw=xxx

Return Variables:

- **sn**: Array of binary values showing the on/off status of each station. This tells you which stations are open.
- **nstations**: The number of stations (equal to the number of boards times 8).

Example Return:

{"sn":[1,1,0,0,0,0,0,0],"nstations":8}
(the above return shows that the 1st and 2nd stations are currently open).

11. Manual Station Run (previously manual override) [Keyword /cm]

/cm?pw=xxx&sid=xxx&en=xxx&t=xxx

Parameters:

- **sid**: Station index (starting from 0)
- **en:** Enable bit (1: open the selected station; 0: close the selected station).
- **t**: Timer (in seconds). Acceptable range is 0 to 64800 (18 hours). **The timer value must be provided if opening a station**.

Examples:

- http://os-ip/cm?pw=xxx&sid=0&en=1&t=360 (open the 1st station s0 for 6 minutes)
- http://os-ip/cm?pw=xxx&sid=3&en=0 (close the 4th station s3)

An error code will return if you try to open the master station (as the master cannot be operated independently), or open a station that's either already running or in the queue waiting to run. Manually started stations are assigned program ID of 99.

12. Manually Start a Program [Keyword /mp]

/mp?pw=xxx&pid=xxx&uwt=x

Parameters:

- **pid**: program index (starting from 0 as the first program)
- **uwt:** use weather (i.e. applying current water level / percentage). Binary value.

Examples:

- http://os-ip/mp?pw=xxx&pid=0&uwt=0 (start the first program at 100% water percentage, i.e. not using weather)
- http://os-ip/mp?pw=xxx&pid=1&uwt=1 (start the second program using the current water percentage)

Note: all existing queued stations will be reset before launching the manually started program. An error code is returned if the program index is out of bound (either negative or larger than the last program index).

13. Get Program Data [Keyword /jp]

/jp?pw=xxx

Return Variables:

- **nprogs**: Number of existing programs.
- **nboards**: Number of boards (including main controller).
- **mnp**: Maximum number of programs allowed.
- **mnst**: Maximum number of program start times (fixed time type) allowed.
- **pnsize**: Maximum number of characters allowed in program name.
- **pd**: Array of program data. Each element corresponds to a program. See below for data structure.

Program Data Structure:

[flag, days0, days1, [start0, start1, start2, start3], [dur0, dur1, dur2…], name]

- **flag:** a bit field storing program flags
  - bit 0: program enable (1: enabled; 0: disabled)
  - bit 1: use weather adjustment (1: yes; 0: no)
  - bit 2-3: odd/even restriction (0: none; 1: odd-day restriction; 2: even-day restriction; 3: undefined)
  - bit 4-5: program schedule type (0: weekday; 1: undefined; 2: undefined; 3: interval day)
  - bit 6: start time type (0: repeating type; 1: fixed time type)
  - bit 7: undefined
- **days0/days1:**
  - *If (**flag.bits[4..5]==0**), this is a weekday schedule:*
    - **days0.bits[0..6]** store the binary selection bit from Monday to Sunday; days1 is unused.
      For example, **days0=127** means the program runs every day of the week; **days0=21** (0b0010101) means the program runs on Monday, Wednesday, Friday every week.
  - *If (**flag.bits[4..5]==3**), this is an interval day schedule:*
    - **days1** stores the interval day, **days0** stores the remainder (i.e. starting in day).
      For example, days1=3 and days0=0 means the program runs every 3 days, starting from today.
- **start0/start1/start2/start3 (***a value of -1 means the start time is disabled***):**
  - **Start times support using sunrise or sunset with a maximum offset value of +/- 4 hours in minute granularity:**
    - If bits 13 and 14 are both cleared (i.e. 0), this defines the start time in terms of minutes since midnight.
    - If bit 13 is 1, this defines sunset time as start time. Similarly, if bit 14 is 1, this defines sunrise time.
      If either bit 13 or 14 is 1, the remaining 12 bits then define the offset. Specifically, bit 12 is the sign (if true, it is negative); the absolute value of the offset is the remaining 11 bits (i.e. start_time&0x7FF).
  - *If (**flag.bit6==1**), this is a fixed start time type:*
    - **start0, start1, start2, start3** store up to 4 fixed start times (minutes from midnight). Acceptable range is -1 to 1440. If set to -1, the specific start time is disabled.
  - *If (**flag.bit6==0**), this is a repeating start time type:*
    - **start0** stores the first start time (minutes from midnight), **start1** stores the repeat count, **start2** stores the interval time (in minutes); start3 is unused. For example, [480,5,120,0] means: start at 8:00 AM, repeat every 2 hours (120 minutes) for 5 times.
- **dur0, dur1…:** The water time (in seconds) of each station. 0 means the station will not run. The number of elements here must match the number of stations. ***Unlike the previous firmwares, this firmware allows full second-level precision water time from 0 to 64800 seconds (18 hours)***. The two special values are: 1) 65534 represents sunrise to sunset duration; 2) 65535 represents sunset to sunrise duration.
- **name:** Program name

Example Return:

{"nprogs":3, "nboards":1, "mnp":17, "mnst":4, "pnsize":20,"pd":[[3,127,0,[480,2,240,0],[0,2700,0,2700,0,0,0,0],"Summer"],
[2,9,0,[120,0,300,0],[0,3720,0,0,0,0,0,0],"Fall Prog"],[67,16,0,[1150,-1,-1,-1],[0,0,0,0,0,0,0,64800,0],"Pipe"]]}

14. Change Program Data [Keyword /cp]

/cp?pw=xxx&pid=xxx&v=[flag,days0,days1,[start0,start1,start2,start3],[dur0,dur1,dur2...]]&name=xxx

Parameters:

- **pid**: Program index (starting from 0). Acceptable range is -1 to N-1, where N is number of existing programs. If pid=-1, this is adding a new program; otherwise this is modifying an existing program.
- **v:** Program data structure. The format is the same as explained in **Section 13** above, except the name field, given below.
- **name:** Program name (url encoded, without quotes).

Examples:

- http://os-ip/cp?pw=xxx&pid=-1&v=[3,127,0,[480,2,240,0],[1800,1200,0,0,0,0,0,0]]&name=Summer%20Prog
  (add a new program, enabled, use weather adjustment, no restriction, weekday schedule that runs on every day, repeating start time type, start at 8:00 AM, repeat every 4 hours for 2 times, and the running stations are 1st station – 30 minute, 2rd station – 20 minutes, program name is "Summer Prog").

15. Delete Program(s) [Keyword /dp]

/dp?pw=xxx&pid=xxx

Parameters:

- **pid**: Program index (starting from 0). A value of -1 deletes all existing programs. Acceptable range is -1 to N-1, where N is number of existing programs.

Examples:

- http://os-ip/dp?pw=xxx&pid=1 (delete the second program)
- http://os-ip/dp?pw=xxx&pid=-1 (delete all programs)

16. Move Up (Re-order) a Program [Keyword /up]

/up?pw=xxx&pid=xxx

Parameters:

- **pid**: Program index (starting from 0). Acceptable range is 0 to N-1, where N is number of existing programs.

Examples:

- http://os-ip/up?pw=xxxr&pid=2 (move the third program up before the second, i.e. switch the order of them)
- http://os-ip/up?pw=xxx&pid=0 (will do nothing because the first program cannot be moved up)

17. Start Run-Once Program [Keyword /cr]

/cr?pw=xxx&t=[x,x,x,....x,x]

Parameters:

- **t**: Timer value for each station. A value of 0 means the station will not run.

Examples:

- http://os-ip/cr?pw=xxx&t=[60,0,60,0,60,0,600,0]
  (start a run-once program that turns on the 1st, 3rd, 5th, and 7th stations for 1 minute each)
- Run-Once started stations are assigned program ID of 254.

18. Get Log Data [Keyword /jl]

/jl?pw=xxx&start=xxx&end=xxx&type=xxx or **/jl?pw=xxx&hist=n&type=xxx**

Return Value:

An array of log records for the time between **start** and **end** (both are epoch times), or the past n days (using **hist**). The maximum time span is 365 days. Each record is a 4-element array in the format of **[pid, sid, dur, end]**, where:

- **pid** is the program index (starting from 1 as the first program. 0 is a special value, see below)
- **sid** is the station index (starting from 0)
- **dur** is the duration (in seconds)
- **end** is the end time (epoch time).
- if flow sensor is enabled, there will be an additional field recording the flow rate during the station run.

When **pid=0**, it indicates a special event record, and the second field (sid) is a string indicating the event **type** (see list below). Parameter **hist** specifies the number of days to go back in history, starting from today. So hist=0 returns the log data of the current day; hist=1 returns the log data of the current day and previous day, and so on.

Parameter **type** specifies which type of special log event you want to query. The supported types are:

- rs: rain sensed event
- rd: rain delay events
- fl: flow sensor readings
- wl: watering level/percentage logs

Example Return: (e.g. by requesting http://os-ip/jl?pw=xxx&start=1413567367&end=1413657367)
[[3,17,616,1413511817], [0,"rd",86400,1413511845], [254,1,5,1413512107], [1,3,2700,1413552661], [5,3,1200,1413559201]]

19. Delete Log Data [Keyword /dl]

/dl?pw=xxx&day=n

Parameters:

- **day**: The day for which log data should be deleted. The parameter value is the epoch time of the day divided by 86400. For example, 16361 is Oct 18, 2014. If day=all, all log files will be deleted.

Examples:

- http://os-ip/dl?pw=xxx&day=16361 (delete the log file for Oct 18, 2014)
- http://os-ip/dl?pw=xxx&day=all (delete all log files)

20. Change Javascript URL [Keyword /cu]

/cu?pw=xxx&jsp=xxx

Parameters:

- **jsp**: Javascript path to be changed to (url encoded).

Examples:

- http://os-ip/cu?pw=xxx&jsp=http%3A%2F%2Fui.opensprinkler.com%2Fjs
  (change Javascript path to http://ui.opensprinkler.com/js)

21. Get all [Keyword /ja]

/ja?pw=xxx

This command returns the combined result of /jc, /jo, /jn, /js, /jp in one command, and put them under the "settings", "options", "stations", "status", "programs" subsections respectively.

Example Return:

{"settings":{.....},"options":{.....},"stations":{.....},"status":{.....},"programs":{.....}}

where ..... is identical to the data obtained from querying each command individually.