SMART WATER MANAGEMENT

API Documentation

smart water technology

updated September 24, 2019

plageAdd

speed 1 (if the equipement get speed, else default 0)

SMART WATER MANAGEMENT

name	**API V2.2**
route	**server side**
availability	**www**
endpoint	**https://api.swimo.io**
folder	**/cgi-bin/**
method	**HEADER, GET, POST**
update	**2019/07/25**
design	**Frederick Lemaitre**
author	**Frederick Lemaitre**

name	**local API V2.2**
route	**controller side**
availability	**local network**
endpoint	**http://<local_IP>**
folder	**/cgi-bin/**
method	**GET**
update	**2019/07/25**
design	**Frederick Lemaitre**
authors	**Christophe Blaess & Frederick Lemaitre**

As an example, when we talk to the server, we treat the information this way

https://api.swimo.io/cgi-bin/plageAdd?number=1&start=00:30&end=03:30&days=1234567
with a **Header**
< appid : r9isU2RbI1ajFO172Z0nj0LB8XbpDXGQlBOxCgbGJg >
the appid as token is obtained by getToken server side with user credentials

on local network, when we talk to the controller, we treat this way

http://192.168.0.12/cgi-bin/plageAdd?number=1&start=00:30&end=03:30&days=1234567&api=FR576-690
the api as apikey is recovered by getAll server side and printed on the controller itself, it acts as controller credentials
the IP **192.168.0.12** is obtained by getAll server side and can be found via user internet box also,.
(default factory IP on Swimo network is 192.168.240.1)

SMART WATER MANAGEMENT

	Server Auth	**Local** Auth	**Method**	**Answer**	**Synchro**	**Target**	**Few words**
createprofile	HEADER	✘	POST	JSON	✘	DISTRI	create a user profile
deleteprofile	HEADER	✘	DELETE	JSON	✘	DISTRI	delete a user profile (not available is system live)
apikeys	HEADER	✘	GET	JSON	✘	DISTRI	get available apikey/system on stock
customers	HEADER	✘	GET	JSON	✘	DISTRI	get customers list
makecontroller	HEADER	✘	POST	JSON	✘	DISTRI	hook a system on existing user without system
deletesystem	HEADER	✘	DELETE	JSON	✘	DISTRI	delete a controller/system from a user profile
addcontroller	HEADER	✘	POST	JSON	✘	USER	add a new controller to a user with system(s)
register	HEADER	✘	GET	JSON	✘	USER	user registration service
password-reset	HEADER	✘	GET	JSON	✘	BOTH	reset password and get temporarily token
password-renew	HEADER	✘	GET	JSON	✘	BOTH	renew password with temporalily token
getToken	HEADER	✘	GET	JSON	✘	BOTH	get a valid token to talk with the API
getAll	HEADER	GET	GET	JSON	✘	BOTH	get user profile, sensors, equipements, alarms
updateSyst	HEADER	GET	GET	JSON	✔	USER	update user and controller datas
updateAlarm	HEADER	GET	GET	JSON	✘	USER	update alerts
getAnalyse	HEADER	GET	GET	JSON	✘	USER	get sensors datas
updateSensor	HEADER	GET	GET	JSON	✔	USER	update sensors
sensorAdd	HEADER	GET	GET	JSON	✔	USER	add a new sensor
sensorDel	HEADER	GET	GET	JSON	✔	USER	del an existing sensor
getDevices	HEADER	GET	GET	JSON	✘	USER	get devices datas
updateDevices	HEADER	GET	GET	JSON	✔	USER	update devices
relayAdd	HEADER	GET	GET	JSON	✔	USER	add a new equipement
relayDel	HEADER	GET	GET	JSON	✔	USER	del an existing equipement
plageAdd	HEADER	GET	GET	JSON	✔	USER	add a new slot time
plageDel	HEADER	GET	GET	JSON	✔	USER	del an existing slot time
calibrationStart	✘	GET	GET	true/false	✘	SENSOR	start a calibration
calibrationCancel	✘	GET	GET	true/false	✘	SENSOR	cancel a calibration in progress
calibrationResult	✘	GET	GET	number	✘	SENSOR	get result of calibration point
machineServer	SERVER	SERVER	GET	JSON	✔	SYNCHRO	M2M equipments synchronization
analyseServer	SERVER	SERVER	GET	JSON	✔	SYNCHRO	M2M sensors synchronization
updateDaily	SERVER	SERVER	GET	JSON	✔	SYNCHRO	M2M contract and timezone synchronization
getSSID	✘	GET	GET	number	✘	WIFI	get a list of available network around
setSSID	✘	GET	GET	number	✘	WIFI	set a network with password reach internet
resetWifi	✘	GET	GET	number	✘	WIFI	reset wifi network as factory state

Parse the API

POST services are strictly reserved to distributors and cannot be called with a valid user token. SYNCHRO services are M2M services between controllers and servers. Services can be called using cURL, jQuery, Ruby, Python requests, node , php, go.. with HEADERS user, code, appid as authentificators and GET/POST as parameters. Some services exit only in Local mode (auth) and others only server side (auth) - see above.

service example / register

CURL

```
curl --location --request GET «https://api.swimo.io/cgi-bin/register?pass&serial» \
  --header «user: contact@iot.flowers» \
  --header «code: ar98oice»
```

JQUERY

```
var form = new FormData();
var settings = {
  «url»: «https://api.swimo.io/cgi-bin/register?pass&serial»,
  «method»: «GET»,
  «timeout»: 0,
  «headers»: {
    «user»: «contact@iot.flowers»,
    «code»: «ar98oice»
  },
  «processData»: false,
  «mimeType»: «multipart/form-data»,
  «contentType»: false,
  «data»: form
};

$.ajax(settings).done(function (response) {
  console.log(response);
});

require «uri»
require «net/http»

RUBY
url = URI(«https://api.swimo.io/cgi-bin/register?pass&serial»)

https = Net::HTTP.new(url.host, url.port)
https.use_ssl = true

request = Net::HTTP::Get.new(url)
request[«user»] = «contact@iot.flowers»
request[«code»] = «ar98oice»

response = https.request(request)
puts response.read_body
```

PYTHON

```
import requests
url = 'https://api.swimo.io/cgi-bin/register?pass&serial'
headers = {
  'user': 'contact@iot.flowers',
  'code': 'ar98oice'
}
response = requests.request('GET', url, headers = headers, data = payload, files = files, allow_redirects=False, timeout=undefined, allow_redirects=false)
print(response.text)
```

NODE

```
var https = require('https');

var options = {
  'method': 'GET',
  'hostname': 'api.swimo.io',
  'path': '/cgi-bin/register?pass=&serial=',
  'headers': {
    'user': 'contact@iot.flowers',
    'code': 'ar98oice'
  }
};

var req = https.request(options, function (res) {
  var chunks = [];

  res.on(«data», function (chunk) {
    chunks.push(chunk);
  });

  res.on(«end», function (chunk) {
    var body = Buffer.concat(chunks);
    console.log(body.toString());
  });

  res.on(«error», function (error) {
    console.error(error);
  });
});

var postData = «------WebKitFormBoundary7MA4YWxkTrZu0gW\r\nContent-Disposition: form-data; ------WebKitFormBoundary7MA4YWxkTrZu0gW--»;

req.setHeader('content-type', 'multipart/form-data; boundary=----WebKitFormBoundary7MA4YWxkTrZu0gW');

req.write(postData);

req.end();
```

PHP

```
$curl = curl_init();

curl_setopt_array($curl, array(
  CURLOPT_URL => «https://api.swimo.io/cgi-bin/register?pass&serial»,
  CURLOPT_RETURNTRANSFER => true,
  CURLOPT_ENCODING => «»,
  CURLOPT_MAXREDIRS => 10,
  CURLOPT_TIMEOUT => 0,
  CURLOPT_FOLLOWLOCATION => false,
  CURLOPT_HTTP_VERSION => CURL_HTTP_VERSION_1_1,
  CURLOPT_CUSTOMREQUEST => «GET»,
  CURLOPT_HTTPHEADER => array(
    «user: contact@iot.flowers»,
    «code: ar98oice»
  ),
));

$response = curl_exec($curl);
$err = curl_error($curl);

curl_close($curl);

if ($err) {
  echo «cURL Error #:» . $err;
} else {
  echo $response;
} ?>
```

GO

```
package main

import (
  «fmt»
  «bytes»
  «mime/multipart»
  «os»
  «path/filepath»
  «net/http»
  «io/ioutil»
)

func main() {
  url := «https://api.swimo.io/cgi-bin/register?pass&serial»
  method := «GET»

  payload := &bytes.Buffer{}
  writer := multipart.NewWriter(payload)
  err := writer.Close()
  if err != nil {  fmt.Println(err)}

  client := &http.Client {
    CheckRedirect: func(req *http.Request, via []*http.Request) error {
      return http.ErrUseLastResponse
    },
  }
  req, err := http.NewRequest(method, url, payload)

  if err != nil {
    fmt.Println(err)
  }
  req.Header.Add(«user», «contact@iot.flowers»)
  req.Header.Add(«code», «ar98oice»)

  req.Header.Set(«Content-Type», writer.FormDataContentType())
  res, err := client.Do(req)
  defer res.Body.Close()
  body, err := ioutil.ReadAll(res.Body)

  fmt.Println(string(body))
}
```

Services dedicated to distributors

create profile, hook system, add controller, delete system, delete user

General rules

For Distributors services, the «appid» must be a valid distributor token to operate DISTRI services.

Any other user tokens are not valid.

Open a distributor account using automation.ac or contact contact@iot.flowers

POST https://api.swimo.io/cgi-bin/createprofile

the best way for an automation user profile creation

parameters :

https://api.swimo.io/cgi-bin/createprofile?serial=0010255555

Method : Header Param : appid,user,code

POST : serial, send

Headers

appid	YmtiTGQ4TTIZSFBWYmorUTUyM3lwZz09 a valid distributor token received by getToken
user	test@fred.com a new user email address
code	automation a temporarily user password

optional Params (to hook a system on user profile)

serial	0010155555 an existing serial system owned by distributor
send	1 1 to send a notification by email to the user, 0 or nothing to do not send

Answer :

True: (received user account)

'master'=>new id

'idu'=>new id

'userEmail'=>test@fred.com

'token'=>token user

'token_distri'=>YmtiTGQ4TTIZSFBWYmorUTUyM3lwZz09

'emailSent'=>1

false : _MISSING_EMAIL, _MISSING_PASSWORD, _INVALID_PERMISSION, _MISSING_TOKEN, _INVALID_SERIAL, _MISSING_SERIAL, _EMAIL_ALREADY_REGISTERED, _SERIAL_ALREADY_REGISTERED

a cURL example :

```
curl -X POST -H «appid: NkZUQkZmbHFubFk3Y2Nqb1BHeHJBQT09» \
-H «user: email@emailuser.com» \
-H «code: anypassword» \
-F «serial=001012758» \
-F «send=1» \
https://api.swimo.io/cgi-bin/createprofile
```

DELETE https://api.swimo.io/cgi-bin/deleteprofile

the best way for an automation user profile deletion

parameters :

https://api.swimo.io/cgi-bin/deleteprofile

Method : Header Param : appid,user

DELETE :

Headers

appid	YmtiTGQ4TTlZSFBWYmorUTUyM3lwZz09 a valid distributor token received by getToken
user	207 idu of user

a cURL example :

curl -X DELETE -H «appid: NkZUQkZmbHFubFk3Y2Nqb1BHeHJBQT09»
**
*-H «user: 207» *
https://api.swimo.io/cgi-bin/createprofile

Answer :

true : _USER_TOKEN_DELETED

false : _SYSTEM_LIVE_THIS_USER_CAN_NOT_BE_DELETED
_USER_UNREGISTERED
_MISSING_TOKEN
_INVALID_PERMISSION

POST https://api.swimo.io/cgi-bin/apikeys

retrieve all systems available in distributor stock

parameters :

https://api.swimo.io/cgi-bin/apikeys

Method : Header Param : appid

Headers

appid YmtiTGQ4TTlZSFBWYmorUTUyM3lwZz09
a valid distributor token received by getToken

Answer :
True: (received list of serial/ apikey)
‘serial’=>serial number
‘apikey’=>apikey
‘date’=>date of purchase
‘token_distri’=>YmtiTGQ4TTlZSFBWYmorUTUyM3lwZz09

false : _NO_STOCK, _INVALID_PERMISSION, _MISSING_TOKEN

GET https://api.swimo.io/cgi-bin/customers

performed a list of sensors with it user hooked on distributor account with a sort by keywords based on sensor type

parameters :

https://api.swimo.io/cgi-bin/sensors?key=treatment

Method : Header Param : appid

GET: key, send

Headers

appid YmtiTGQ4TTlZSFBWYmorUTUyM3lwZz09
a valid distributor token received by getToken

Optional Params

key a valid idu <207>
based on user unique id

Answer :
True: (received all users sorted idu) or a specific user
'user'=> an array with all account and controller details

POST https://api.swimo.io/cgi-bin/makecontroller

hook a system/controller on an existing user list in customers

parameters :

https://api.swimo.io/cgi-bin/makecontroller
Method : Header Param : appid

POST : serial, idu, send

false : _INVALID_PERMISSION, _MISSING_TOKEN, _INVALID_SERIAL

Headers

appid YmtiTGQ4TTlZSFBWYmorUTUyM3lwZz09
a valid distributor token received by getToken

Params

serial 00101010101
a valid serial number owned by distributor

idu 207
a valid user id owned by distributor

Answer :
True: (received list of serial/ apikey)
‘idu’=>
‘master’=>
‘new-token’=>
‘token_distri’=>YmtiTGQ4TTlZSFBWYmorUTUyM3lwZz09

DELETE https://api.swimo.io/cgi-bin/deletesystem

delete a controller, all sensors and relays setup from a user

parameters :

https://api.swimo.io/cgi-bin/deletesystem

Method : Header Param : appid

POST : serial

Headers

appid YmtiTGQ4TTlZSFBWYmorUTUyM3lwZz09
a valid distributor token received by getToken

param

serial 00120009090
a valid serial number owned by distributor

Answer :
True: _SYSTEM_DEVICES_SENSORS_TOKEN_DELETED

Warning : *when a system is deleted, user token is also deleted, so the only way to retrieve the user is to call* **customers** *service*

Services dedicated to users

manage profile, add new system, get datas, drive controllers, manage controllers...

General rules

For User services, the «appid» must be a valid user token to operate USER/BOTH services.

Open a user account using this API or any other website consumming this API without owning a controller is not allowed at this time.

POST https://api.swimo.io/cgi-bin/addcontroller

to add a new controller on an existing user profile with a valid controller living

parameters :

https://api.swimo.io/cgi-bin/addcontroller

Method : Header Param : appid

POST : serial

Headers

user YmtiTGQ4TTlZSFBWYmorUTUyM3lwZz09
a valid user token received by getToken

parameter :

serial 00102020202
a valid serial number

Answer :

True:
‘master’=>
‘idu’=>
‘userEmail’=>
‘new_token’=>
‘token_user’=>
‘emailSent’=>

GET https://api.swimo.io/cgi-bin/register

register a user account with a valid idSystem (user interface)

parameters :

https://api.swimo.io/cgi-bin/register?pass= AUto56709&serial=00101567890

Method : Header Param : user,code

METHOD : GET PARAMS pass :confirmation serial :serial number

Headers

user	contact@iot.flowers
code	AUto56709

Params

pass	AUto56709 same password as code
serial	00101567890 a valid serial system

Answer : True: (received token) «oR5zdhuVrje4yVqCkTR860i659fqVi04uQECHBVBIY»
false : _SERIAL_ALREADY_REGISTERED, _SERIAL_UNAVAILABLE , _EMAIL_ALREADY_REGISTER , _TWO_PASSWORDS_INCORRECT

GET https://api.swimo.io/cgi-bin/password-reset

to launch a procedure to renew password creating a temporarily token sent automatically by email

parameters :

https://api.swimo.io/cgi-bin/password-reset?serial=0010255555

Method : Header Param : user

Headers

user	test@fred.com user email address

Answer :

True: _EMAIL_SENT

False: _EMAIL_UNKNOWN

An email is sent to the user email with a temporarily token as a key to change his password

GET https://api.swimo.io/cgi-bin/password-renew

To renew password with a temporarily token generated by password-reset

parameters :

https://api.swimo.io/cgi-bin/password-renew?pass=AUto56709

Method : Header Param : appid,user,code

METHOD : GET PARAMS pass :

Headers

appid	<temporarily token received by email>
user	contact@iot.flowers
code	AUto56709

Params

pass	AUto56709 same password as code

Answer :

True: _GO_TO_LOGIN

false : _TWO_PASSWORDS_INCORRECT, _WRONG_TOKEN, _EMAIL_DIFFERENT, _MISSING_TOKEN

GET https://api.swimo.io/cgi-bin/getToken

Get a valid token as appid parameter in header to talk with the entire API

parameters :

https://api.swimo.io/cgi-bin/getToken

Method : Header Param : user, code

Headers

user	test@fred.com user email address
code	AUto56709 user password

Answer : True: (received token) «oR5zdhuVrje4yVqCkTR860i659fqVi04uQECHBVBIY»
false : _MISSING_HEADER, _INVALID_ACCOUNT

GET https://api.swimo.io/cgi-bin/getAll

retrieve a list of your users if there is more than one account, or all datas of this account.

parameters :

https://api.swimo.io/cgi-bin/getAll

Method : Header Param : appid

Headers

appid	oR5zdhuVrje4yVqCkTR860i659fqVi04uQECHBVBlY

Optional Param

ct	1 (for distributor, 1 remains appid token in the answer , else token distri must be stored on the application)

Answer :

false : _INVALID_ACCOUNT

True : regarding type of account (distributor, user, multi user) answer is different as below ...

GET https://api.swimo.io/cgi-bin/getAll

a distributor or multi pool user list

For distributors a list of users is the answer with get a token to reach this particular account and a token pro to retrieve this list sort by last name Desc


```
{
   «systems»: [
      {
         «master»: «101»,
         «idu»: «101»,
         «lastName»: «CRONIMUS»,
         «firstName»: «THOMAS»,
         «address»: «5 rue des Pommiers «,
         «city»: «HOERDT»,
         «zip»: «67720»,
         «phone»: «0610611614»,
         «userEmail»: «thomas.cronimus@sfr.fr»,
         «token»: «SEjrEGZA1q0v3tCheqhdxk3FhwyAEu2xLzEom03sCs»,
         «currentDate»: «2019-07-27 18:32:29»,
         «token_distri»: «MFo1RTZtVW91RXJ0TlFtVWlEZjVsUT09»,
         «idDistri»: «1159»
      },
{
         «master»: «178»,
         «idu»: «180»,
         «lastName»: «Sarl le Weekend»,
         «firstName»: «H495/2»,
         «address»: «2000 route des sanguinaires»,
         «city»: «ajaccio»,
         «zip»: «2000»,
         «phone»: «»,
         «userEmail»: «001021856@iot.flowers»,
         «token»: «MUsvQzJ5KzAvUkI0YXh4aTJySDV4dz09»,
         «currentDate»: «2019-07-27 18:36:13»,
         «token_distri»: «cHRoNDgyUDU4eDhZaGN2UUt3dTlUQT09»,
         «idDistri»: «1161»
      }
   ],

«waiting»:
```


a list of user without systems hooked on them


```
   «multicompte»: [
      {
         «companyname»: «CLAIR' AZUR»,
         «companyadresse»: «100 rue des alisiers»,
         «companycp»: «06600»,
         «companyville»: «Antibes»,
         «companyphone»: «0820333233»,
         «companyemail»: «services@clairazur.com»,
         «companyweb»: «clairazur.automation.ac»,
         «supportnom»: «BARRALIS»,
         «supportprenom»: «Alexandre»,
         «supportphone»: «0820 333 233»,
         «supportemail»: «contact@clairazur.com»,
         «avatar»: «a13c7e080fb8f64b4005643ba22729a7.png»,
         «lat»: «43.6103»,
         «long»: «7.0742»,
         «idmaint»: «1161»,
         «idDistri»: «1161»,
         «mot»: «Le  leader français du spa & spa de nage»
      }
   ]
}
...
```

SMART WATER MANAGEMENT

GET https://api.swimo.io/cgi-bin/getAll

a single account answer


```
{
  «version»: [
    {
      «newVersion»: «v.0.1.1»
    }
  ],
  «user»: [
    {
      «master»: «166»,
      «idu»: «166»,
      «idSystem»: «001021776»,
      «apikey»: «FR576-690»,
      «swimo»: «»,
      «version»: «v.0.1.0»,
      «ipMachine»: «192.168.240.1»,
      «email»: «aquaphilou81@gmail.com»,
      «nom»: «»,
      «prenom»: «aquaphilou81»,
      «adresse»: «»,
      «cp»: «»,
      «city»: «»,
      «phone»: «»,
      «avatar»: «»,
      «timezone»: «Europe/Paris»,
      «lang»: «FR»,
      «lat»: «0»,
      «long»: «0»,
      «idCity»: «»,
      «volume»: «10»,
      «typeBassin»: «1»,
      «typeAbris»: «_INT»,
      «unit»: «metric»,
      «typeFiltration»: «»,
      «datefiltre»: «0000-00-00»,
      «typeTraitement»: «»,
      «typeRevetement»: «»,
      «typeHeater»: «»,
      «typeSupTraitement»: «»,
      «pompephmoins»: «0»,
      «pompephplus»: «0»,
      «notifCLient»: «0»,
      «notifDistri»: «0»,
      «contrat»: «0»,
      «idDistri»: «1133»
    }
  ],
  «alarme»: [
    {
      «idAlarm»: «781»,
      «titreAlarm»: «ALERTE»,
      «dateAlarm»: «2019-02-14 12:51:22»,
      «messageAlarm»: «Controleur N° - 001021776  - déconnecté»
    },
    {
      «idAlarm»: «953»,
      «titreAlarm»: «ALERTE»,
      «dateAlarm»: «2019-03-14 10:37:46»,
      «messageAlarm»: «Controleur N° - 001021776  - déconnecté»
    },
    {
      «idAlarm»: «973»,
      «titreAlarm»: «ALERTE»,
      «dateAlarm»: «2019-03-14 16:00:23»,
      «messageAlarm»: «Controleur N° - 001021776  - déconnecté»
    },
    {
      «idAlarm»: «995»,
      «titreAlarm»: «ALERTE»,
      «dateAlarm»: «2019-03-15 00:00:22»,
      «messageAlarm»: «Controleur N° - 001021776  - déconnecté»
    },
    {
      «idAlarm»: «1432»,
      «titreAlarm»: «ALERTE»,
      «dateAlarm»: «2019-04-15 00:00:40»,
      «messageAlarm»: «Controleur N° - 001021776  - déconnecté»
    },
    {
      «idAlarm»: «1956»,
      «titreAlarm»: «ALERTE»,
      «dateAlarm»: «2019-05-18 00:00:01»,
      «messageAlarm»: «Controleur N° - 001021776  - déconnecté»
    },
    {
      «idAlarm»: «2318»,
      «titreAlarm»: «ALERTE»,
      «dateAlarm»: «2019-06-18 00:00:01»,
      «messageAlarm»: «Controleur N° - 001021776  - déconnecté»
    }
  ],
  «meteo»: null,
  «accueil_analyse»: [
    {
      «sensorType»: «4»,
      «nmSensor»: «1»,
      «currentDate»: «2019-03-13 19:02:46»,
      «etatSensor»: «1»,
      «nameSensor»: «Electrode pH»,
      «minSensor»: «-1»,
      «maxSensor»: «7.87»,
      «alarmMin»: «6.5»,
      «alarmMax»: «8»,
      «liveSensor»: [
        «-1»,
        «7.87»,
        «7.86»,
        «7.85»,
        «7.85»,
        «7.84»,
        «7.84»,
        «7.83»
      ],
      «live»: -1,
      «unitConsigne»: «pt»,
      «offsetSensor»: «0»,
      «typeCalibration»: «2»,
      «dateCalib»: «0000-00-00 00:00:00»,
      «idSystem»: «001021776»
    },
    {
      «sensorType»: «5»,
      «nmSensor»: «2»,
      «currentDate»: «2019-07-16 18:06:58»,
      «etatSensor»: «1»,
      «nameSensor»: «taux Redox»,
      «minSensor»: «-182»,
      «maxSensor»: «1372»,
      «alarmMin»: «200»,
      «alarmMax»: «900»,
      «liveSensor»: [
        «-1»,
        «528»,
        «506»,
        «466»,
        «466»,
        «415»,
        «437»,
        «492»
      ],
      «live»: -1,
      «unitConsigne»: «mV»,
      «offsetSensor»: «0»,
      «typeCalibration»: «2»,
      «dateCalib»: «0000-00-00 00:00:00»,
      «idSystem»: «001021776»
    },
    {
      «sensorType»: «1»,
      «nmSensor»: «4»,
      «currentDate»: «2019-03-08 16:51:50»,
      «etatSensor»: «1»,
      «nameSensor»: «temperature»,
      «minSensor»: «-1»,
      «maxSensor»: «38.31»,
      «alarmMin»: «14»,
      «alarmMax»: «37»,
      «liveSensor»: [
        «-1»,
        «16.21»,
        «16.34»,
        «16.52»,
        «16.52»,
        «16.78»,
        «17»,
        «17.26»
      ],
      «live»: -1,
      «unitConsigne»: «°C»,
      «offsetSensor»: «0»,
      «typeCalibration»: «0»,
      «dateCalib»: «0000-00-00 00:00:00»,
      «idSystem»: «001021776»
    }
  ],
  «accueil_appareil»: [],
  «plages»: [
    {
      «idSystem»: «001021776»,
      «nmAction»: «1»,
      «start»: «20:53»,
      «end»: «21:00»,
      «days»: «1234567»
    },
    {
      «idSystem»: «001021776»,
      «nmAction»: «15»,
      «start»: «19:41»,
      «end»: «20:09»,
      «days»: «1234567»
    }
  ],
  «infos»: [
    {
      «companyname»: «IOT FLOWERS»,
      «companyadresse»: «78 avenue Raymond Poincaré»,
      «companycp»: «75116»,
      «companyville»: «PARIS»,
      «companyphone»: «0987655433»,
      «companyemail»: «fred.lemaitre@iotflowers.com»,
      «companyweb»: «automation.ac»,
      «supportnom»: «LEMAITRE»,
      «supportprenom»: «FRED»,
      «supportphone»: «06 80 24 60 92»,
      «supportemail»: «contact@iot.flowers»,
      «avatar»: «1ecbe-396018665628b104a3678bbdc5a.png»,
      «mot»: «la domotique piscine 2.0»,
      «idmaint»: «1133»,
      «idDistri»: «1133»
    }
  ],
  «LienGet»: [
    {
      «token_pro»: «MFo1RTZtVW91RXJ0T-lFtVWlEZjVsUT09»
    }
  ]
}
```

GET https://api.swimo.io/cgi-bin/updateSyst

update user account

parameters :

https://api.swimo.io/cgi-bin/updateSyst?nuser=1&npro=1&vol=1.6&abris=_EXT&nom=POOL TEST&prenom=LEM

Method :Header Param :appid

Headers

appid	r9isU2Rbl1ajFO172Z0nj0LB8XbpDXGQlBO

optional Params

unit	metric	metric, imperial (°C, °F)
email		new email address
lang	FR	FR, EN, ES
nuser	1	notif user 0 (non) , 1 (oui)
npro	1	notif pro 0 , 1
vol	1.6	volume in m3 (step=0.1)
abris	_EXT	_INT /_EXT
nom	POOL TEST	last name
prenom	LEM	first name
adresse		address
cp		ZIP
ville		city
phone		phone
contrat		0 ou 1 (1=Genius mode beta)
ip		public ip only
type	1	type pool=1, spa=0
timezone	Europe/paris	list of available zone name*

Answer : True:
false : _MISSING_HEADER, _INVALID_ACCOUNT

GET https://api.swimo.io/cgi-bin/updateAlarm

set notification as blind

parameters :

https://api.swimo.io/cgi-bin/updateAlarm?idal=1876&stat=0

Method :Header Param :appid

Headers

appid	*r9isU2Rbl1ajFO172Z0nj0LB8Xb*

Params

idal	*1876*	
		idAlarm found in getAll
stat	*1*	
		0 = visible, 1 = unvisible

update of the notifications in getAll;
With the status = 1, the notification is as deleted and will not be visible by the user, but remains existing in database.

Answer : True:
false : invalid_token

GET https://api.swimo.io/cgi-bin/getAnalyse?number=1

recovers all sensors, values, variations and status. By adding number = (sensor n °), we get the detail and history of a specific sensor

parameters :

https://api.swimo.io/cgi-bin/getAnalyse?number=1&histo=30

Headers

appid r9isU2Rbl1ajFO172Z0nj0

Optional Params

number 1
optional number of sensor
histo 30
7, 30,90, 180, number of days

Answer :


```
{
  «analyse»: [
    {
      «sensorType»: «4»,
      «number»: «1»,
      «nmSensor»: «1»,
      «currentDate»: «2019-07-27 18:48:13»,
      «etatSensor»: «1»,
      «nameSensor»: «Electrode pH»,
      «alarmMin»: «6»,
      «alarmMax»: «9»,
      «liveSensor»: «6.29»,
      «unitConsigne»: «pt»,
      «offsetSensor»: «0»,
      «typeCalibration»: «2»,
      «dateCalib»: «0000-00-00 00:00:00»
    }
  ],
  «historique»: [
    {
      «releve»: «6.31»,
      «insertion»: «2019-07-27 16:00:05»
    },
    {
      «releve»: «5.48»,
      «insertion»: «2019-07-26 16:00:04»
    },
    {
      «releve»: «6.62»,
      «insertion»: «2019-07-20 16:00:04»
    },
    {
      «releve»: «6.64»,
      «insertion»: «2019-07-20 08:00:05»
    },
    {
      «releve»: «8.17»,
      «insertion»: «2019-07-16 16:00:03»
    },
    {
      «releve»: «8.33»,
      «insertion»: «2019-07-16 08:00:04»
    },
    {
      «releve»: «8.34»,
      «insertion»: «2019-07-15 16:00:04»
    }
  ]
}
```

GET https://api.swimo.io/cgi-bin/getAnalyse

recovers all sensors, values, variations and status. By adding number = (sensor n °), we get the detail and history of a specific sensor

parameters :

https://api.swimo.io/cgi-bin/getAnalyse

Headers

appid r9isU2Rbl1ajFO172Z0nj0

answer :


```
{
  «analyse»: [
    {
      «idTheme»: «_PH»,
      «sensorType»: «4»,
      «number»: «1»,
      «etatSensor»: «1»,
      «nameAnalyse»: «Valeur Ph «,
      «variation»: 0,
      «liveSensor»: «6.29»,
      «unitSensor»: «pt»,
      «name2»: «item»,
      «variation2»: 2,
      «live2»: «6.51»,
      «unit2»: «»,
      «name3»: «»,
      «variation3»: 2,
      «live3»: «6.33»,
      «unit3»: «»,
      «currentDate»: «2019-07-27 18:48:13»
    },
    {
      «idTheme»: «_ORP»,
      «sensorType»: «5»,
      «number»: «2»,
      «etatSensor»: «1»,
      «nameAnalyse»: «orp/redox»,
      «variation»: 2,
      «liveSensor»: «323.8»,
      «unitSensor»: «mV»,
      «name2»: «item»,
      «variation2»: 2,
      «live2»: «333.3»,
      «unit2»: «»,
      «name3»: «»,
      «variation3»: 2,
      «live3»: «377.9»,
      «unit3»: «»,
      «currentDate»: «2019-07-27 18:48:13»
    },
    {
      «idTheme»: «_EC»,
      «sensorType»: «6»,
      «number»: «3»,
      «etatSensor»: «1»,
      «nameAnalyse»: «conductivité»,
      «variation»: 2,
      «liveSensor»: «2.29»,
      «unitSensor»: «mS/cm»,
      «name2»: «item»,
      «variation2»: 2,
      «live2»: «2.26»,
      «unit2»: «»,
      «name3»: «»,
      «variation3»: 2,
      «live3»: «2.24»,
      «unit3»: «»,
      «currentDate»: «2019-07-27 18:48:13»
    },
    {
      «idTheme»: «_TEMP»,
      «sensorType»: «1»,
      «number»: «4»,
      «etatSensor»: «1»,
      «nameAnalyse»: «temperature»,
      «variation»: 0,
      «liveSensor»: «35.94»,
      «unitSensor»: «°C»,
      «name2»: «item»,
      «variation2»: 0,
      «live2»: «36.23»,
      «unit2»: «»,
      «name3»: «»,
      «variation3»: 0,
      «live3»: «36.58»,
      «unit3»: «»,
      «currentDate»: «2019-07-27 18:48:13»
    },
    {
      «idTheme»: «_FLOW_SWITCH»,
      «sensorType»: «9»,
      «number»: «7»,
      «etatSensor»: «1»,
      «nameAnalyse»: «debit/flux»,
      «variation»: 2,
      «liveSensor»: «1»,
      «unitSensor»: «L/h»,
      «name2»: «item»,
```

GET https://api.swimo.io/cgi-bin/updateSensor?number=1

update a sensor

parameters :

https://api.swimo.io/cgi-bin/updateSensor?nmSensor=4&nameSens=tempo

Headers

appid	r9isU2Rbl1ajFO172Z0nj0LB8XbpDXGQlBOxCgbGJg	

Params

number	4	
setsens	0.1	offset sensor
min	6	value min ALARM to receive notification
max	7.2	value max ALARM to receive notification
mSensor	4	SAME AS NUMBER for V1
nameSens	tempo	rename the sensor

MIN and MAX alarms are not only here to send notifications. In some kind of conditions, those alarms may interrupted a part of treatment or allowing something that should not run.
As an example, PH sensor alarms is generally maintained between 6.8 to 7.8.
In case of a ph over 7.8, the chlorine pump will stopped until the PH level is restored between those 2 alarms.

Answer :	**True**	**: true**
	False	**: invalid_system**

GET https://api.swimo.io/cgi-bin/sensorAdd

to add a new sensor connected on the controller

parameters :

https://api.swimo.io/cgi-bin/sensorAdd?sensor=2&nmSensor=8

Headers

appid r9isU2Rbl1ajFO172Z0nj0LB8XbpDXGQlBOxCgbGJg

Params

sensor 2
type sensor

number 8
n° of signal

nmSensor 8
SAME AS NUMBER for V1

Answer
True : true
false : invalid_token

Sensors **numbers are 1 to 23 regarding the controller signals**
Modbus sensors numbers are 13, 17, 18 and 19 (can be increased)

SensorType are choosen regarding the type of signal needed even if some signal are hard connect to sensor number like PH sensor is always connected to 1

CHECK SENSORS LIST and SENSORS SIGNAL BOARD

For scalable API calls, number is the primary variable to retrieve Sensors

GET https://api.swimo.io/cgi-bin/sensorDel

to delete a new sensor connected on the controller

parameters :

https://api.swimo.io/cgi-bin/sensorDel?number=8

Headers

appid r9isU2Rbl1ajFO172Z0nj0LB8XbpDXGQlBOxCgbGJg

Params

number 8
n° of signal

nmSensor 8
SAME AS NUMBER for V1

Answer
True : true
false : invalid_token

GET https://api.swimo.io/cgi-bin/getDevices?number=4

recovers all devices, values, variations and status. By adding number = (relay n °), we get the detail and history of a specific device

parameters :

https://api.swimo.io/cgi-bin/getDevices?nmAction=4

Headers

appid r9isU2Rbl1ajFO172Z0nj0

optional Params

number 2

OPTION : to get details and time slots

nmAction 2

same as number for V1

histo 2

1=yesterday , 2= 2 days ago...

Answer :{

«devices»: [
{
«nameAction»: «pompe Chlore»,
«idActionType»: «4»,
«number»: «4»,
«nmAction»: «4»,
«textDevice»: «ECO»,
«isOff»: «1»,
«securite»: «1»,
«selectedIndex»: «2»,
«stateAct»: [
«1/2/3»
],
«codeMode»: [
«ON/OFF/AUTO»
],
«power»: «60»,
«unitPower»: «ml/mn»,
«consigne»: «1»,
«unitConsigne»: «mg/l»,
«sequence»: «0»,
«niveau»: «92.94»,
«vitesse»: 0,
«typeVitesse»: «ON»,
«interval»: «30»,
«etalon»: «0.1»,
«nmSensor»: «13»,
«selectedSeq»: «1»,
«statProg»: [
«1/2/3»
],
«codeProg»: [
«Plage/Eco/Max»
],
«typeAlarm»: «2074» last type from smart device
«alarm»: «128» last alarm
«currentDate»: «2019-07-27 20:03:46»
}
],
«plages»: [],
«consos»: [
{
«temps»: «637»,
«currentDate»: «2019-07-27 19:55:39»
},
{
«temps»: «545»,
«currentDate»: «2019-07-27 19:43:38»
},
{
«temps»: «117»,
«currentDate»: «2019-07-27 19:19:37»
},
{
«temps»: «126»,
«currentDate»: «2019-07-27 19:07:36»
},
{
«temps»: «189»,
«currentDate»: «2019-07-27 18:43:34»
},
{
«temps»: «24»,
«currentDate»: «2019-07-27 18:19:32»
},
{
«temps»: «185»,
«currentDate»: «2019-07-27 17:55:30»
},
{
«temps»: «556»,
«currentDate»: «2019-07-27 17:43:30»
},
{
«temps»: «106»,
«currentDate»: «2019-07-27 17:19:28»
},
{
«temps»: «227»,
«currentDate»: «2019-07-27 17:07:27»
},
{
«temps»: «436»,
«currentDate»: «2019-07-27 16:43:25»
},
{
«temps»: «296»,
«currentDate»: «2019-07-27 16:31:24»
},
{
«temps»: «78»,
«currentDate»: «2019-07-27 16:07:23»
},
{
«temps»: «229»,
«currentDate»: «2019-07-27 15:43:21»
},
{
«temps»: «293»,
«currentDate»: «2019-07-27 15:31:20»
},

«name»:»sensorType»,:[

{

DEFAULT SETUP WHEN ADD A NEW SENSOR

«id»:»1»,

«sensorType» :» 1», Sensor Type to use to add a new sensor in SensorAdd service

«nmSensorDefault»:»4»,

«idTitre» :» _PT100», Type of signal regarding the Board

«theme»:»temperature_adc»,

«nameSensor» :» temperature»,database name never changes, never shows to user

«nameAnalyse» :» temperature», name updatable by user using updateSensor?name=

«defaultSet»:»20»,

«typeCalibration» :» 0», number of point for a new calibration (0> no calibration)

«defaultCalibrationBas»:»0»,

«defaultCalibrationHaut»:»0»,

«unitSensor» :» °C»,unit sensor could be change °F or C° in updateSyst?unit=

«minSensor»:»14»,

«maxSensor»:»30»,

«alarmMin» :» 5», min Alarm updatable by user using updateSensor?minAlarm= for notification and smart automatism

«alarmMax» :» 37» max Alarm updatable by user using updateSensor?maxAlarm= for notification and smart automatism

},

{

«id»:»2»,
«sensorType»:»2»,
«nmSensorDefault»:»5»,
«idTitre»:»_EC»,
«theme»:»sonde_sel»,
«nameSensor»:»Sonde sel»,
«nameAnalyse»:»Salinité»,
«defaultSet»:»4»,
«typeCalibration»:»0»,
«defaultCalibrationBas»:»0»,
«defaultCalibrationHaut»:»0»,
«unitSensor»:»g\/l»,
«minSensor»:»3»,
«maxSensor»:»5»,
«alarmMin»:»3»,
«alarmMax»:»8»

},

{

«id»:»3»,
«sensorType»:»3»,
«nmSensorDefault»:»6»,
«idTitre»:»_4_20MA1»,
«theme»:»pressure_adc»,
«nameSensor»:»sonde Pression»,
«nameAnalyse»:»Pression»,
«defaultSet»:»1»,
«typeCalibration»:»1»,
«defaultCalibrationBas»:»4»,
«defaultCalibrationHaut»:»20»,
«unitSensor»:»Bar»,
«minSensor»:»0.8»,
«maxSensor»:»0.95»,
«alarmMin»:»0.5»,
«alarmMax»:»2»

},

{

«id»:»4»,
«sensorType»:»4»,
«nmSensorDefault»:»1»,
«idTitre»:»_PH»,
«theme»:»ph_adc»,
«nameSensor»:»Electrode pH»,
«nameAnalyse»:»Valeur Ph «,
«defaultSet»:»7.2»,
«typeCalibration»:»2»,
«defaultCalibrationBas»:»0»,
«defaultCalibrationHaut»:»14»,
«unitSensor»:»pt»,
«minSensor»:»6.2»,
«maxSensor»:»7.5»,
«alarmMin»:»6»,
«alarmMax»:»9»

},

{

«id»:»5»,
«sensorType»:»5»,
«nmSensorDefault»:»2»,
«idTitre»:»_ORP»,
«theme»:»orp_adc»,
«nameSensor»:»Electrode ORP»,
«nameAnalyse»:»Traitement»,
«defaultSet»:»650»,
«typeCalibration»:»2»,
«defaultCalibrationBas»:»0»,
«defaultCalibrationHaut»:»1200»,
«unitSensor»:»mV»,
«minSensor»:»450»,
«maxSensor»:»750»,
«alarmMin»:»300»,
«alarmMax»:»1000»

},

{

«id»:»6»,
«sensorType»:»6»,
«nmSensorDefault»:»3»,
«idTitre»:»_EC»,
«theme»:»ec_adc»,
«nameSensor»:»Electrode EC»,
«nameAnalyse»:»conductivité»,
«defaultSet»:»1»,
«typeCalibration»:»2»,
«defaultCalibrationBas»:»0»,
«defaultCalibrationHaut»:»10»,
«unitSensor»:»mS\/cm»,
«minSensor»:»1»,
«maxSensor»:»3»,
«alarmMin»:»0»,
«alarmMax»:»7»

},

{

«id»:»7»,
«sensorType»:»7»,
«nmSensorDefault»:»13»,
«idTitre»:»_RS485_CL»,
«theme»:»cl_adc»,
«nameSensor»:»Chlore ampero»,
«nameAnalyse»:»Chlore actif»,
«defaultSet»:»0.4»,
«typeCalibration»:»2»,
«defaultCalibrationBas»:»1»,
«defaultCalibrationHaut»:»2»,
«unitSensor»:»mg\/l»,
«minSensor»:»0.4»,
«maxSensor»:»1.4»,
«alarmMin»:»0»,
«alarmMax»:»1.6»

},

{

«id»:»9»,
«sensorType»:»9»,
«nmSensorDefault»:»7»,
«idTitre»:»_FLOW_SWITCH»,
«theme»:»flow_rate_gpio»,
«nameSensor»:»detecteur de debit»,
«nameAnalyse»:»Débit»,
«defaultSet»:»0»,
«typeCalibration»:»0»,
«defaultCalibrationBas»:»0»,
«defaultCalibrationHaut»:»0»,
«unitSensor»:»L\/h»,
«minSensor»:»0»,
«maxSensor»:»0»,
«alarmMin»:»0»,
«alarmMax»:»0»

},

{

«id»:»10»,
«sensorType»:»10»,
«nmSensorDefault»:»8»,
«idTitre»:»_CAN_LEVEL»,
«theme»:»water_level_gpio_ph_minus»,
«nameSensor»:»detecteur bidon»,
«nameAnalyse»:»niveau ph moins»,
«defaultSet»:»0»,
«typeCalibration»:»0»,
«defaultCalibrationBas»:»0»,
«defaultCalibrationHaut»:»0»,
«unitSensor»:»»,
«minSensor»:»0»,
«maxSensor»:»0»,
«alarmMin»:»0»,
«alarmMax»:»0»

},

{

«id»:»11»,
«sensorType»:»11»,
«nmSensorDefault»:»9»,
«idTitre»:»_FLOW_METER»,
«theme»:»water_level_gpio_flocculant»,
«nameSensor»:»Débimetre»,
«nameAnalyse»:»niveau floculant»,
«defaultSet»:»0»,
«typeCalibration»:»0»,
«defaultCalibrationBas»:»0»,
«defaultCalibrationHaut»:»0»,
«unitSensor»:»»,
«minSensor»:»0»,
«maxSensor»:»0»,
«alarmMin»:»0»,
«alarmMax»:»0»

},

{

«id»:»12»,
«sensorType»:»12»,
«nmSensorDefault»:»10»,
«idTitre»:»_4_20MA2»,
«theme»:»water_level_gpio_cl»,
«nameSensor»:»detecteur bidon traitement»,
«nameAnalyse»:»niveau traitement»,
«defaultSet»:»0»,
«typeCalibration»:»0»,
«defaultCalibrationBas»:»0»,
«defaultCalibrationHaut»:»0»,
«unitSensor»:»»,
«minSensor»:»0»,
«maxSensor»:»0»,
«alarmMin»:»0»,
«alarmMax»:»0»

},

{

«id»:»13»,
«sensorType»:»13»,
«nmSensorDefault»:»11»,
«idTitre»:»_ROLLER_SENSOR»,

```
    «theme»:»roller_level_gpio»,
    «nameSensor»:»detecteur fermeture volet»,
    «nameAnalyse»:»Volet fermé»,
    «defaultSet»:»0»,
    «typeCalibration»:»0»,
    «defaultCalibrationBas»:»0»,
    «defaultCalibrationHaut»:»0»,
    «unitSensor»:»»,
    «minSensor»:»0»,
    «maxSensor»:»0»,
    «alarmMin»:»0»,
    «alarmMax»:»0»
  },
  {
    «id»:»14»,
    «sensorType»:»14»,
    «nmSensorDefault»:»12»,
    «idTitre»:»_WATER_LEVEL»,
    «theme»:»niveau»,
    «nameSensor»:»niveau»,
    «nameAnalyse»:»NIveau d'eau»,
    «defaultSet»:»0»,
    «typeCalibration»:»0»,
    «defaultCalibrationBas»:»2»,
    «defaultCalibrationHaut»:»0»,
    «unitSensor»:»»,
    «minSensor»:»4.5»,
    «maxSensor»:»7»,
    «alarmMin»:»3.5»,
    «alarmMax»:»8»
  },
  {
    «id»:»15»,
    «sensorType»:»15»,
    «nmSensorDefault»:»5»,
    «idTitre»:»_SWITCH»,
    «theme»:»reed»,
    «nameSensor»:»Detecteur reed»,
    «nameAnalyse»:»reed sensor»,
    «defaultSet»:»0»,
    «typeCalibration»:»0»,
    «defaultCalibrationBas»:»2»,
    «defaultCalibrationHaut»:»0»,
    «unitSensor»:»»,
    «minSensor»:»0»,
    «maxSensor»:»1»,
    «alarmMin»:»0»,
    «alarmMax»:»1»
  },
  {

  {
    «id»:»18»,
    «sensorType»:»17»,
    «nmSensorDefault»:»6»,
    «idTitre»:»_TEMP_4201»,
    «theme»:»TEMP»,
    «nameSensor»:»Temperature Sauna»,
    «nameAnalyse»:»Temperature Sauna»,
    «defaultSet»:»0»,
    «typeCalibration»:»2»,
    «defaultCalibrationBas»:»2»,
    «defaultCalibrationHaut»:»0»,
    «unitSensor»:»°C»,
    «minSensor»:»0»,
    «maxSensor»:»0»,
    «alarmMin»:»5»,
    «alarmMax»:»80»
  },
  {
    «id»:»19»,
    «sensorType»:»18»,
    «nmSensorDefault»:»15»,
    «idTitre»:»_RS485_NORSUP»,
    «theme»:»temp_rs_norsup»,
    «nameSensor»:»Temperature»,
    «nameAnalyse»:»Temperature Norsup»,
    «defaultSet»:»90»,
    «typeCalibration»:»0»,
    «defaultCalibrationBas»:»0»,
    «defaultCalibrationHaut»:»0»,
    «unitSensor»:»°C»,
    «minSensor»:»0»,
    «maxSensor»:»0»,
    «alarmMin»:»50»,
    «alarmMax»:»100»
  },
  {
    «id»:»21»,
    «sensorType»:»19»,
    «nmSensorDefault»:»16»,
    «idTitre»:»_RS485_TEMP»,
    «theme»:»temp_adc»,
    «nameSensor»:»Temperature hammam»,
    «nameAnalyse»:»Temperature Hammam»,
    «defaultSet»:»45»,
    «typeCalibration»:»0»,
    «defaultCalibrationBas»:»0»,
    «defaultCalibrationHaut»:»0»,
    «unitSensor»:»°C»,
    «minSensor»:»0»,
    «maxSensor»:»0»,
    «alarmMin»:»30»,
    «alarmMax»:»65»
  },
  {
    «id»:»22»,
    «sensorType»:»20»,
    «nmSensorDefault»:»10»,
    «idTitre»:»_TYPE_TOP»,
    «theme»:»flotteur haut»,
    «nameSensor»:»flotteur haut»,
    «nameAnalyse»:»flotteur haut»,
    «defaultSet»:»0»,
    «typeCalibration»:»0»,
    «defaultCalibrationBas»:»0»,
    «defaultCalibrationHaut»:»0»,
    «unitSensor»:»»,
    «minSensor»:»0»,
    «maxSensor»:»0»,
    «alarmMin»:»0»,
    «alarmMax»:»1»
  },
  {
    «id»:»23»,
    «sensorType»:»21»,
    «nmSensorDefault»:»11»,
    «idTitre»:»_TYPE_MEDIUM»,
    «theme»:»flotteur milieu»,
    «nameSensor»:»flotteur milieu»,
    «nameAnalyse»:»flotteur milieu»,
    «defaultSet»:»0»,
    «typeCalibration»:»0»,
    «defaultCalibrationBas»:»0»,
    «defaultCalibrationHaut»:»0»,
    «unitSensor»:»»,
    «minSensor»:»0»,
    «maxSensor»:»0»,
    «alarmMin»:»0»,
    «alarmMax»:»1»
  },
  {
    «id»:»24»,
    «sensorType»:»22»,
    «nmSensorDefault»:»12»,
    «idTitre»:»_TYPE_BOTTOM»,
    «theme»:»flotteur bas»,
    «nameSensor»:»flotteur bas»,
    «nameAnalyse»:»flotteur bas»,
    «defaultSet»:»0»,
    «typeCalibration»:»0»,
    «defaultCalibrationBas»:»0»,
    «defaultCalibrationHaut»:»0»,
    «unitSensor»:»»,
    «minSensor»:»0»,
    «maxSensor»:»0»,
    «alarmMin»:»0»,
    «alarmMax»:»1»
  },
  {
    «id»:»25»,
    «sensorType»:»23»,
    «nmSensorDefault»:»2»,
    «idTitre»:»_ORP»,
    «theme»:»ORP_CL»,
    «nameSensor»:»Orp Chlore»,
    «nameAnalyse»:»ORP Chlore actif»,
    «defaultSet»:»0.4»,
    «typeCalibration»:»2»,
    «defaultCalibrationBas»:»1»,
    «defaultCalibrationHaut»:»2»,
    «unitSensor»:»mg\/l»,
    «minSensor»:»0.4»,
    «maxSensor»:»1.4»,
    «alarmMin»:»0»,
    «alarmMax»:»1.6»
  }
 ]
}
]
```

SENSORS Board signals

The swimo V3 board signals VS number in API call

Analogic Hard signals:	Sensor number in API call	Type of Sensor	SensorType inclued a list more important that the numbers availables and we can not connect all sensor Type on the controller at the same times (see Sensor-Type)
Ph signal	1	Analogic Ph ONLY	
Orp signal	2	Orp/redox ONLY	
conductivity signal	3	Conductivity 1K ONLY	
Temperature PT100 signal	4	PT100, 2 or 3 wires	
Temperature NTC10 signal	15	NTC10K 2 wires	
Humidity signal	16	Humlidity 3 wires IOT	
~~Free chlorine potentiostatic~~	~~14~~		
Analogic variable signals :			
4- 20 mA signal 1	5	Any 4-20mA sensors in 12V like Temperature, Pressure, water level, flowmeter	
4-20 mA signal 2	6		
RS485 modBus signal :	13,17,18,19	Any Modbus signal	
Free Chlorine Sensor		0x01	
Norsup Heater temperature		0x32	
Carel Hammam temperature		0x02	
Digital signals :		Any reed/dry contact flow detector, switch detector, level detector hight and low, door detector...	
Tor 1	7		
Tor 2	8		
Tor 3	9		
Tor 4	10		
Tor 5	11		
Tor 6	12		

GET https://api.swimo.io/cgi-bin/getDevices

recovers all devices, values, variations and status. By adding number = (relay n °), we get the detail and history of a specific device

parameters :

https://api.swimo.io/cgi-bin/getDevices

Headers

appid r9isU2Rbl1ajFO172Z0nj0

answer


```
{
  «devices»: [
    {
      «nameAction»: «pompe Chlore»,
      «idActionType»: «4»,
      «number»: «4»,
      «nmAction»: «4»,
      «textDevice»: «_ECO_MODE»,
      «soldeNiveau»: «92.94»,
      «consoJour»: «03:13:29»,
      «codeConsigne»: «1»,
      «unitConsigne»: «mg/l»,
      «isOff»: «0»,
      «securite»: «1»,
      «selectedIndex»: «2»,
      «stateAct»: [
        «1/2/3»
      ],
      «codeMode»: [
        «ON/OFF/AUTO»
      ],
      «currentDate»: «2019-07-27 20:07:40»
    },
    {
      «nameAction»: «pompe ph-»,
      «idActionType»: «2»,
      «number»: «3»,
      «nmAction»: «3»,
      «textDevice»: «_ECO_MODE»,
      «soldeNiveau»: «41.07»,
      «consoJour»: «00:00:00»,
      «codeConsigne»: «7»,
      «unitConsigne»: «pt»,
      «isOff»: «0»,
      «securite»: «1»,
      «selectedIndex»: «1»,
      «stateAct»: [
        «1/2/3»
      ],
      «codeMode»: [
        «ON/OFF/AUTO»
      ],
      «currentDate»: «2019-07-27 20:07:40»
    },
    {
      «nameAction»: «Pompe secu»,
      «idActionType»: «9»,
      «number»: «1»,
      «nmAction»: «1»,
      «textDevice»: «_ECO_MODE»,
      «soldeNiveau»: «-1»,
      «consoJour»: «00:00:00»,
      «codeConsigne»: «»,
      «unitConsigne»: «»,
      «isOff»: «0»,
      «securite»: «1»,
      «selectedIndex»: «1»,
      «stateAct»: [
        «1/2/3»
      ],
      «codeMode»: [
        «ON/OFF/AUTO»
      ],
      «currentDate»: «2019-07-27 20:07:40»
    },
    {
      «nameAction»: «Pompe floculant»,
      «idActionType»: «14»,
      «number»: «2»,
      «nmAction»: «2»,
      «textDevice»: «_ECO_MODE»,
      «soldeNiveau»: «-1»,
      «consoJour»: «00:00:00»,
      «codeConsigne»: «»,
      «unitConsigne»: «»,
      «isOff»: «0»,
      «securite»: «0»,
      «selectedIndex»: «1»,
      «stateAct»: [
        «1/2/3»
      ],
      «codeMode»: [
        «ON/OFF/AUTO»
      ],
      «currentDate»: «2019-07-27 20:07:40»
    }
  ]
}
```

GET https://api.swimo.io/cgi-bin/updateDevice?number=1

update a device

parameters :

https://api.swimo.io/cgi-bin/updateDevice?number=2&con=760

Headers

appid	r9isU2Rbl1ajFO172Z0nj0LB8XbpDXGQlBOxCgbGJg

Params

number	1	MUST EXIST
val	40	value power/flow of the device
con	760	setpoint when exists
seq	Eco	name of the program (only in V1)
bid	20	level of tank if exits to update volume
nmAction	2	same as number for V1
unitPower	ml/mn	unity
vit	2	only used for led light sequence (1 to 18)
interval	20	not yet used
type	ON	not yet used
sensor	5	not yet used (to hook a sensor to this device)
etalon	0	standard to manage setpoint range
codeSeq	0	0,1 or 2 TIME SLOT/ECO/MAX
index	0	0,1 or 2 ON/OFF/AUTO
name	Heater	rename the device as you want

GET https://api.swimo.io/cgi-bin/relayAdd

to add a new device connected on the controller

parameters :

https://api.swimo.io/cgi-bin/relayAdd?action=3&nmAction=14

Headers

appid r9isU2Rbl1ajFO172Z0nj0LB8XbpDXGQlBOxCgbGJg

Params

action 3
an action Type regarding the list
number 14
relay number
nmAction 8
SAME AS NUMBER V1
speed 1
(if the equipement get speed, else default 0)

Answer
True : true
false : invalid_token

Relay numbers are 1 to 15 regarding the controller relays

Modbus numbers are 16 to 19 (can be inceased)

ActionType are choosen regarding the type of automatism and sensor(s) required to get smart automation.

For scalable API calls, number is the primary variable to retrieve devices

GET https://api.swimo.io/cgi-bin/relayDel

to delete a new device connected on the controller

parameters :

https://api.swimo.io/cgi-bin/relayDel?number=14

Headers

appid r9isU2Rbl1ajFO172Z0nj0LB8XbpDXGQlBOxCgbGJg

Params

number 14
n° of signal
nmSensor 14
SAME AS NUMBER for V1

Answer
True : true
false : invalid_token

GET https://api.swimo.io/cgi-bin/plageAdd

to add a new time slot for a specific device

parameters :

https://api.swimo.io/cgi-bin/plageAdd?number=1&start=00:30&end=03:30&-days=1234567

Headers

appid r9isU2Rbl1ajFO172Z0nj0LB8XbpDXGQlBOxCgbGJg

Params

number 1
relay 1 to 15 and 16 to 30 as modbus line
start 00:30
start time slot
end 03:30
end time slot
days 1234567
as option , if not indicated, add all days
nmAction
same as number

Answer
True : true
false : invalid_token

GET https://api.swimo.io/cgi-bin/plageDel

to delete a new an existing time slot on a specific device

parameters :

https://api.swimo.io/cgi-bin/plageDel?number=1&start=00:30

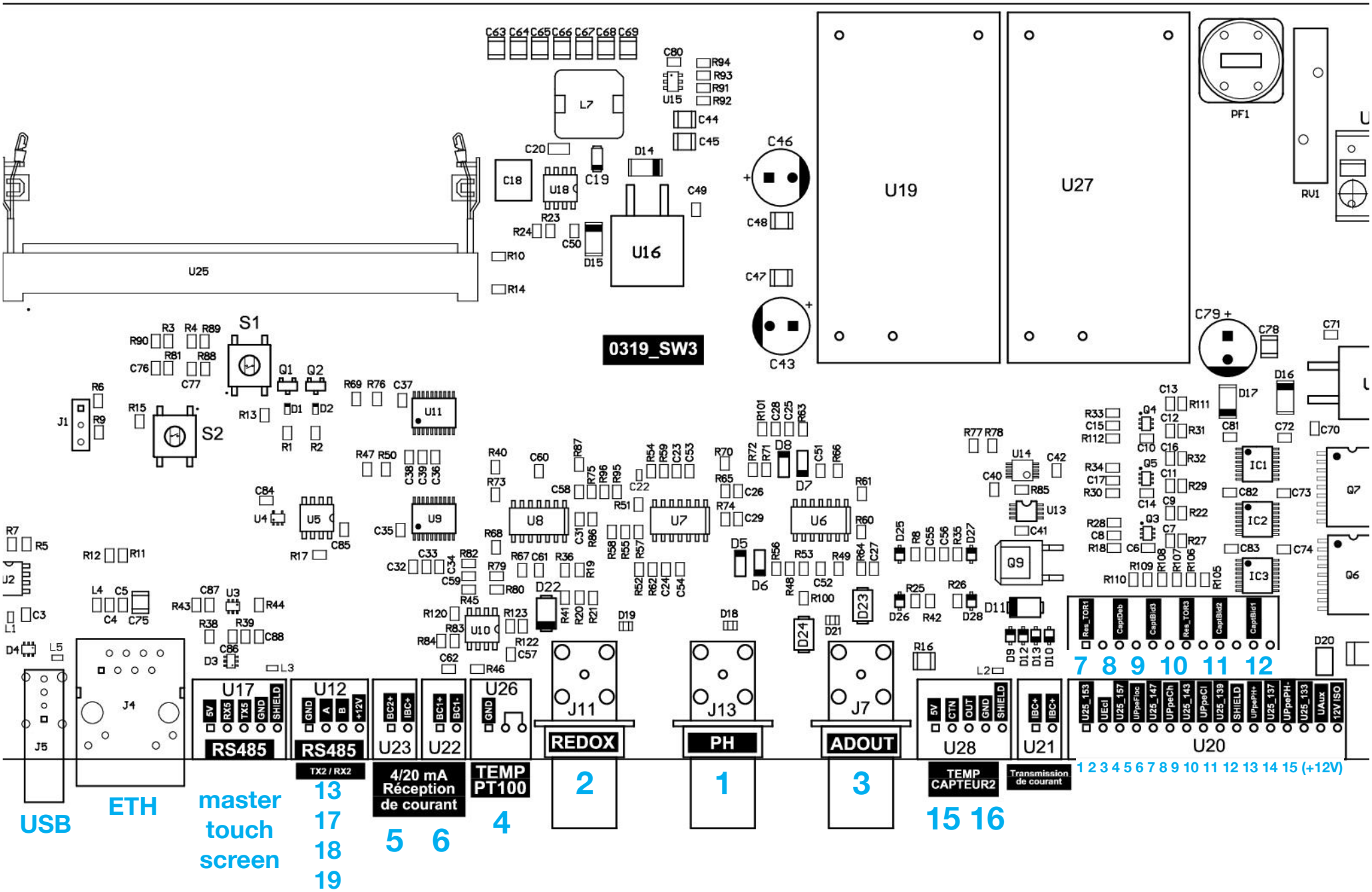
Headers

appid r9isU2Rbl1ajFO172Z0nj0LB8XbpDXGQlBOxCgbGJg

Params

number 1
relay to uninstall
nmAction
same as number
start 00:30
time slot to delete

Answer
True : true
false : invalid_token


0319_SW3
U19
U27
U16
U25
PF1
RV1
S1
S2
J1
J2
J4
J5
J7
J11
J13
U17
RS485
U12
RS485
TX2 / RX2
U23
4/20 mA
Réception
de courant
U22
U26
TEMP
PT100
REDOX
PH
ADOUT
U28
TEMP
CAPTEUR2
U21
Transmission
de courant
U20
USB
ETH
master
touch
screen
13
17
18
19
5
6
4
2
1
3
15 16
7 8 9 10 11 12
1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 (+12V)

```
actionType»,
  :[
    {
      «id»:»1»,
      «idActionType»:»1»,
      «idTitre»:»_FILTRATION_PUMP»,
      «theme»:»_ECO_MODE»,
      «unitPower»:»M3\/h»,
      «nameAction»:»pompe Filtration»,
      «SensorType»:»3»,
      «SensorType2»:»9»,
      «nmActionDefault»:»1»,
      «stateAct»:»1\/2\/3»,
      «stateNameAct»:»ON\/OFF\/AUTO»,
      «stateProg»:»1\/2\/3\/4»,
      «statProgName»:»Plage\/Night\/Day\/Winter»,
      «defaultPower»:»12»,
      «etalon»:»»,
      «unitType»:»»,
      «interval»:»»,
      «defaultConsigne»:»»,
      «defaultBidon»:»»,
      «actionPilote»:»1»,
      «needPilote»:»0»,
      «typeCode»:»0»,
      «securite»:»1»
    },
    {
      «id»:»2»,
      «idActionType»:»2»,
      «idTitre»:»_PHPMINUS_PUMP»,
      «theme»:»_ECO_MODE»,
      «unitPower»:»ml\/mn»,
      «nameAction»:»pompe ph-»,
      «SensorType»:»1»,
      «SensorType2»:»4»,
      «nmActionDefault»:»2»,
      «stateAct»:»1\/2\/3»,
      «stateNameAct»:»ON\/OFF\/AUTO»,
      «stateProg»:»1\/2\/3»,
      «statProgName»:»Plage\/Eco\/Max»,
      «defaultPower»:»35»,
      «etalon»:»0.3»,
      «unitType»:»pt»,
      «interval»:»30»,
      «defaultConsigne»:»7.2»,
      «defaultBidon»:»20»,
      «actionPilote»:»0»,
      «needPilote»:»1»,
      «typeCode»:»0»,
      «securite»:»1»
    },
    {
      «id»:»3»,
      «idActionType»:»3»,
      «idTitre»:»_PHPLUS_PUMP»,
      «theme»:»_ECO_MODE»,
      «unitPower»:»ml\/mn»,
      «nameAction»:»pompe ph+»,
      «SensorType»:»1»,
      «SensorType2»:»4»,
      «nmActionDefault»:»3»,
      «stateAct»:»1\/2\/3»,
      «stateNameAct»:»ON\/OFF\/AUTO»,
      «stateProg»:»1\/2\/3»,
      «statProgName»:»Plage\/Eco\/Max»,
      «defaultPower»:»35»,
      «etalon»:»0.3»,
      «unitType»:»pt»,
      «interval»:»30»,
      «defaultConsigne»:»7.1»,
      «defaultBidon»:»20»,
      «actionPilote»:»0»,
      «needPilote»:»1»,
      «typeCode»:»0»,
      «securite»:»1»
    },
    {
      «id»:»4»,
      «idActionType»:»4»,
      «idTitre»:»_CHLORINE_PUMP»,
      «theme»:»_ECO_MODE»,
      «unitPower»:»ml\/mn»,
      «nameAction»:»pompe Chlore»,
      «SensorType»:»13»,
      «SensorType2»:»5»,
      «nmActionDefault»:»4»,
      «stateAct»:»1\/2\/3»,
      «stateNameAct»:»ON\/OFF\/AUTO»,
      «stateProg»:»1\/2\/3»,
      «statProgName»:»Plage\/Eco\/Max»,
      «defaultPower»:»60»,
      «etalon»:»0.1»,
      «unitType»:»mg\/l»,
      «interval»:»30»,
      «defaultConsigne»:»1.2»,
      «defaultBidon»:»20»,
      «actionPilote»:»0»,
      «needPilote»:»1»,
      «typeCode»:»0»,
      «securite»:»1»
    },
    {
      «id»:»5»,
      «idActionType»:»5»,
      «idTitre»:»_DOSING_PUMP»,
      «theme»:»_ECO_MODE»,
      «unitPower»:»ml\/mn»,
      «nameAction»:»pompe de dosage»,
      «SensorType»:»0»,
      «SensorType2»:»8»,
      «nmActionDefault»:»5»,
      «stateAct»:»1\/2\/3»,
      «stateNameAct»:»ON\/OFF\/AUTO»,
      «stateProg»:»1\/2\/3»,
      «statProgName»:»Plage\/Eco\/Max»,
      «defaultPower»:»1»,
      «etalon»:»»,
      «unitType»:»»,
      «interval»:»30»,
      «defaultConsigne»:»»,
      «defaultBidon»:»20»,
      «actionPilote»:»0»,
      «needPilote»:»1»,
      «typeCode»:»0»,
      «securite»:»1»
    },
    {
      «id»:»6»,
      «idActionType»:»6»,
      «idTitre»:»_CHLORINATOR»,
      «theme»:»_ECO_MODE»,
      «unitPower»:»KW\/h»,
      «nameAction»:»Electrolyse»,
      «SensorType»:»2»,
      «SensorType2»:»7»,
      «nmActionDefault»:»6»,
      «stateAct»:»1\/2\/3»,
      «stateNameAct»:»ON\/OFF\/AUTO»,
      «stateProg»:»1\/2\/3»,
      «statProgName»:»Plage\/Eco\/Max»,
      «defaultPower»:»1»,
      «etalon»:»50»,
      «unitType»:»mV»,
      «interval»:»»,
      «defaultConsigne»:»760»,
      «defaultBidon»:»»,
      «actionPilote»:»0»,
      «needPilote»:»1»,
      «typeCode»:»0»,
      «securite»:»0»
    },
    {
      «id»:»7»,
      «idActionType»:»7»,
      «idTitre»:»_HEATER»,
      «theme»:»_ECO_MODE»,
      «unitPower»:»KW\/h»,
      «nameAction»:»Chauffage»,
      «SensorType»:»4»,
      «SensorType2»:»2»,
      «nmActionDefault»:»7»,
      «stateAct»:»1\/2\/3»,
      «stateNameAct»:»ON\/OFF\/AUTO»,
      «stateProg»:»1\/2\/3»,
      «statProgName»:»Plage\/Eco\/Max»,
      «defaultPower»:»2»,
      «etalon»:»3»,
      «unitType»:»°C»,
      «interval»:»»,
      «defaultConsigne»:»29»,
      «defaultBidon»:»»,
      «actionPilote»:»0»,
      «needPilote»:»1»,
      «typeCode»:»0»,
      «securite»:»0»
    },
    {
      «id»:»8»,
      «idActionType»:»8»,
      «idTitre»:»OPENING_ROLLER»,
      «theme»:»_ECO_MODE»,
      «unitPower»:»KW\/h»,
      «nameAction»:»Ouverture volet»,
      «SensorType»:»0»,
      «SensorType2»:»0»,
      «nmActionDefault»:»8»,
      «stateAct»:»1\/2»,
      «stateNameAct»:»OUVRIR\/ARRET»,
      «stateProg»:»»,
      «statProgName»:»»,
      «defaultPower»:»2»,
      «etalon»:»»,
      «unitType»:»»,
      «interval»:»»,
      «defaultConsigne»:»»,
      «defaultBidon»:»»,
      «actionPilote»:»0»,
      «needPilote»:»0»,
      «typeCode»:»0»,
      «securite»:»2»
    },
    {
      «id»:»9»,
      «idActionType»:»9»,
      «idTitre»:»_LIGHT»,
      «theme»:»_ECO_MODE»,
      «unitPower»:»KW\/h»,
      «nameAction»:»Eclairage»,
      «SensorType»:»0»,
      «SensorType2»:»0»,
      «nmActionDefault»:»9»,
      «stateAct»:»1\/2\/3»,
      «stateNameAct»:»ON\/OFF\/AUTO»,
      «stateProg»:»1\/2\/3»,
      «statProgName»:»Plage\/Eco\/Max»,
```

«defaultPower»:»3»,
«etalon»:»»,
«unitType»:»»,
«interval»:»»,
«defaultConsigne»:»»,
«defaultBidon»:»»,
«actionPilote»:»0»,
«needPilote»:»0»,
«typeCode»:»0»,
«securite»:»1»
},
{
«id»:»10»,
«idActionType»:»10»,
«idTitre»:»_VARIABLE_PUMP»,
«theme»:»_ECO_MODE»,
«unitPower»:»M3\/h»,
«nameAction»:»pompe Filtration variable»,
«SensorType»:»3»,
«SensorType2»:»9»,
«nmActionDefault»:»13»,
«stateAct»:»1\/2\/3»,
«stateNameAct»:»ON\/OFF\/AUTO»,
«stateProg»:»1\/2\/3\/4»,
«statProgName»:»Plage\/Night\/Day\/Winter»,
«defaultPower»:»12»,
«etalon»:»»,
«unitType»:»»,
«interval»:»»,
«defaultConsigne»:»»,
«defaultBidon»:»»,
«actionPilote»:»1»,
«needPilote»:»0»,
«typeCode»:»1»,
«securite»:»1»
},
{
«id»:»11»,
«idActionType»:»11»,
«idTitre»:»_ELECTRO_VALVE»,
«theme»:»_ECO_MODE»,
«unitPower»:»KW\/h»,
«nameAction»:»Vannes 6 voies aurto»,
«SensorType»:»3»,
«SensorType2»:»9»,
«nmActionDefault»:»11»,
«stateAct»:»1\/2\/3»,
«stateNameAct»:»ON\/OFF\/AUTO»,
«stateProg»:»»,
«statProgName»:»»,
«defaultPower»:»1»,
«etalon»:»»,
«unitType»:»»,
«interval»:»»,
«defaultConsigne»:»»,
«defaultBidon»:»»,
«actionPilote»:»1»,
«needPilote»:»1»,
«typeCode»:»1»,
«securite»:»0»
},
{
«id»:»12»,
«idActionType»:»12»,
«idTitre»:»_LED_MOB»,
«theme»:»_ECO_MODE»,
«unitPower»:»KW\/h»,
«nameAction»:»Eclairage modbus»,
«SensorType»:»0»,
«SensorType2»:»0»,
«nmActionDefault»:»12»,
«stateAct»:»1\/2\/3»,
«stateNameAct»:»ON\/OFF\/AUTO»,
«stateProg»:»1\/2\/3»,
«statProgName»:»Plage\/Eco\/Max»,
«defaultPower»:»1»,
«etalon»:»»,
«unitType»:»»,
«interval»:»»,
«defaultConsigne»:»»,
«defaultBidon»:»»,
«actionPilote»:»0»,
«needPilote»:»0»,
«typeCode»:»1»,
«securite»:»0»
},
{
«id»:»13»,
«idActionType»:»13»,
«idTitre»:»_REDOX_PUMP»,
«theme»:»_ECO_MODE»,
«unitPower»:»ml\/mn»,
«nameAction»:»pompe Redox»,
«SensorType»:»2»,
«SensorType2»:»5»,
«nmActionDefault»:»4»,
«stateAct»:»1\/2\/3»,
«stateNameAct»:»ON\/OFF\/AUTO»,
«stateProg»:»1\/2\/3»,
«statProgName»:»Plage\/Eco\/Max»,
«defaultPower»:»35»,
«etalon»:»40»,
«unitType»:»mV»,
«interval»:»30»,
«defaultConsigne»:»650»,
«defaultBidon»:»20»,
«actionPilote»:»0»,
«needPilote»:»1»,
«typeCode»:»0»,
«securite»:»1»
},
{
«id»:»14»,
«idActionType»:»14»,
«idTitre»:»_AUX»,
«theme»:»_ECO_MODE»,
«unitPower»:»KW\/h»,
«nameAction»:»Auxiliaire»,
«SensorType»:»0»,
«SensorType2»:»7»,
«nmActionDefault»:»6»,
«stateAct»:»1\/2\/3»,
«stateNameAct»:»ON\/OFF\/AUTO»,
«stateProg»:»1\/2\/3»,
«statProgName»:»Plage\/Eco\/Max»,
«defaultPower»:»1»,
«etalon»:»»,
«unitType»:»»,
«interval»:»»,
«defaultConsigne»:»»,
«defaultBidon»:»»,
«actionPilote»:»0»,
«needPilote»:»1»,
«typeCode»:»0»,
«securite»:»0»
},
{
«id»:»15»,
«idActionType»:»15»,
«idTitre»:»_CLOSING_ROLLER»,
«theme»:»_ECO_MODE»,
«unitPower»:»KW\/h»,
«nameAction»:»Fermeture volet»,
«SensorType»:»0»,
«SensorType2»:»0»,
«nmActionDefault»:»10»,
«stateAct»:»1\/2»,
«stateNameAct»:»FERMER\/ARRET»,
«stateProg»:»»,
«statProgName»:»»,
«defaultPower»:»2»,
«etalon»:»»,
«unitType»:»»,
«interval»:»»,
«defaultConsigne»:»»,
«defaultBidon»:»»,
«actionPilote»:»0»,
«needPilote»:»0»,
«typeCode»:»1»,
«securite»:»2»
},
{
«id»:»16»,
«idActionType»:»16»,
«idTitre»:»_LED»,
«theme»:»_ECO_MODE»,
«unitPower»:»KW\/h»,
«nameAction»:»Eclairage led»,
«SensorType»:»0»,
«SensorType2»:»0»,
«nmActionDefault»:»9»,
«stateAct»:»1\/2\/3»,
«stateNameAct»:»ON\/OFF\/AUTO»,
«stateProg»:»1\/2\/3»,
«statProgName»:»Plage\/Eco\/Max»,
«defaultPower»:»3»,
«etalon»:»»,
«unitType»:»»,
«interval»:»»,
«defaultConsigne»:»»,
«defaultBidon»:»»,
«actionPilote»:»0»,
«needPilote»:»0»,
«typeCode»:»0»,
«securite»:»0»
},
{
«id»:»17»,
«idActionType»:»17»,
«idTitre»:»_ROBOT»,
«theme»:»_ECO_MODE»,
«unitPower»:»KW\/h»,
«nameAction»:»Robot nettoyage»,
«SensorType»:»0»,
«SensorType2»:»0»,
«nmActionDefault»:»10»,
«stateAct»:»1\/2\/3»,
«stateNameAct»:»ON\/OFF\/AUTO»,
«stateProg»:»1\/2\/3»,
«statProgName»:»Plage\/Eco\/Max»,
«defaultPower»:»2»,
«etalon»:»»,
«unitType»:»»,
«interval»:»»,
«defaultConsigne»:»»,
«defaultBidon»:»»,
«actionPilote»:»0»,
«needPilote»:»0»,
«typeCode»:»1»,
«securite»:»3»
},
{
«id»:»18»,
«idActionType»:»18»,

actionType»,
:[
{
«id»:»18»,
«idActionType»:»18», Type of action used to add a new device with relayAdd service
«idTitre»:»_UV_LAMP», code we use to manage langage
«theme»:»_ECO_MODE»,
«unitPower»:»KW\/h»,
«nameAction»:»UV + oxy», name can be updated by the user using updateDevicer?name=
«SensorType»:»2», sensor in correlation for smart water management
«SensorType2»:»0»,
«nmActionDefault»:»10»,
«stateAct»:»1\/2\/3»,
«stateNameAct»:»ON\/OFF\/AUTO»,
«stateProg»:»1\/2\/3»,
«statProgName»:»Plage\/Eco\/Max»,
«defaultPower»:»100», Power in kw/h to calcule consumption or in ml/mm to calculate consumption of tank of product (ph minus, ph maximus, Chlorine etc)
«etalon»:»50», standard can be updated by the user using updateDevice?etalon= to set a new standard for smart setpoint range. In ECO mode means that for a setpoint of 650 mv (ORP sensor) the treatement restart at 650 -etalon(50)= 600 mv
«unitType»:»mV»,
«interval»:»»,
«defaultConsigne»:»150», Setpoint can be updated by the user using updateDevice?con= to set a new setpoint .
«defaultBidon»:»», Level of product in the tank/can, can be updated by the user using updateDevice?bid= to set a new level when change the 20 L can. if empty or -1 not in use for this device.

«actionPilote»:»0»,
«needPilote»:»0»,
«typeCode»:»1»,
«securite»:»1»
},
{
«id»:»19»,
«idActionType»:»19»,
«idTitre»:»_SAUNA»,
«theme»:»_ECO_MODE»,
«unitPower»:»KW\/h»,
«nameAction»:»Sauna»,
«SensorType»:»6»,
«SensorType2»:»0»,
«nmActionDefault»:»10»,
«stateAct»:»1\/2\/3»,
«stateNameAct»:»ON\/OFF\/AUTO»,
«stateProg»:»1\/2\/3»,
«statProgName»:»Plage\/Eco\/Max»,
«defaultPower»:»3»,
«etalon»:»3»,
«unitType»:»°C»,
«interval»:»»,
«defaultConsigne»:»45»,
«defaultBidon»:»»,
«actionPilote»:»0»,
«needPilote»:»0»,
«typeCode»:»1»,
«securite»:»1»
},
{
«id»:»20»,
«idActionType»:»20»,
«idTitre»:»_HAMMAM»,
«theme»:»_ECO_MODE»,
«unitPower»:»KW\/h»,
«nameAction»:»Hammam»,
«SensorType»:»19»,
«SensorType2»:»0»,
«nmActionDefault»:»10»,
«stateAct»:»1\/2\/3»,
«stateNameAct»:»ON\/OFF\/AUTO»,
«stateProg»:»1\/2\/3»,
«statProgName»:»Plage\/Eco\/Max»,
«defaultPower»:»3»,
«etalon»:»3»,
«unitType»:»°C»,
«interval»:»»,
«defaultConsigne»:»45»,
«defaultBidon»:»»,
«actionPilote»:»0»,
«needPilote»:»0»,
«typeCode»:»1»,
«securite»:»1»
},
{
«id»:»21»,
«idActionType»:»21»,
«idTitre»:»_VALVE»,
«theme»:»_ECO_MODE»,
«unitPower»:»KW\/h»,
«nameAction»:»Electrovanne»,
«SensorType»:»11»,
«SensorType2»:»0»,
«nmActionDefault»:»10»,
«stateAct»:»1\/2\/3»,
«stateNameAct»:»ON\/OFF\/AUTO»,
«stateProg»:»1\/2\/3»,
«statProgName»:»Plage\/Eco\/Max»,
«defaultPower»:»3»,
«etalon»:»3»,
«unitType»:»»,
«interval»:»»,
«defaultConsigne»:»0»,
«defaultBidon»:»»,
«actionPilote»:»0»,
«needPilote»:»0»,
«typeCode»:»1»,
«securite»:»1»
},

{
«id»:»22»,
«idActionType»:»22»,
«idTitre»:»_CAREL_LIGHT»,
«theme»:»_ECO_MODE»,
«unitPower»:»KW\/h»,
«nameAction»:»Eclairage Carel»,
«SensorType»:»0»,
«SensorType2»:»0»,
«nmActionDefault»:»17»,
«stateAct»:»1\/2»,
«stateNameAct»:»ON\/OFF»,
«stateProg»:»»,
«statProgName»:»»,
«defaultPower»:»1»,
«etalon»:»»,
«unitType»:»»,
«interval»:»»,
«defaultConsigne»:»»,
«defaultBidon»:»»,
«actionPilote»:»0»,
«needPilote»:»0»,
«typeCode»:»1»,
«securite»:»1»
},
{
«id»:»23»,
«idActionType»:»23»,
«idTitre»:»_CAREL_EUCA»,
«theme»:»_ECO_MODE»,
«unitPower»:»KW\/h»,
«nameAction»:»Pompe Eucalyptus Carel»,
«SensorType»:»0»,
«SensorType2»:»0»,
«nmActionDefault»:»18»,
«stateAct»:»1\/2»,
«stateNameAct»:»ON\/OFF»,
«stateProg»:»»,
«statProgName»:»»,
«defaultPower»:»1»,
«etalon»:»»,
«unitType»:»»,
«interval»:»»,
«defaultConsigne»:»»,
«defaultBidon»:»»,
«actionPilote»:»0»,
«needPilote»:»0»,
«typeCode»:»1»,
«securite»:»1»
},
{
«id»:»24»,
«idActionType»:»24»,
«idTitre»:»_NORSUP»,
«theme»:»_ECO_MODE»,
«unitPower»:»KW\/h»,
«nameAction»:»Chauffage»,
«SensorType»:»4»,
«SensorType2»:»2»,
«nmActionDefault»:»7»,
«stateAct»:»1\/2\/3»,
«stateNameAct»:»ON\/OFF\/AUTO»,
«stateProg»:»1\/2\/3»,
«statProgName»:»Plage\/Eco\/Max»,
«defaultPower»:»2»,
«etalon»:»3»,
«unitType»:»°C»,
«interval»:»»,
«defaultConsigne»:»29»,
«defaultBidon»:»»,
«actionPilote»:»0»,
«needPilote»:»1»,
«typeCode»:»0»,
«securite»:»0»
},

Next services are not available remotely

so you need to be connected on the controller itself, on its local IP or throught a redirection to this local IP. For the purpose of this documentation, we will used the factory IP on Swimo network before it connexion to user box.

GET http://192.168.240.1/calibrateStart

to calibrate a sensor

http://192.168.240.1/cgi-bin/calibrateStart?number=1&point=1&value=4&api=FR45556

Params

number 1
number of the sensor
point 1
number of point (from getAnalyse)
«typeCalibration»: «2», means that 2 baths to calibrate this sensor
value 4
value of the bath
api apikey

Answer
True : true
false : false

When true a timer is launched for 60 seconds

when 60 seconds is reached, you MUST send

http://192.168.240.1/cgi-bin/calibrateResult?number=1&api=FR45556

Answer
True : 0 to 4096 a value from the DAC component on the PCB
false : false

then for this case of typeCalibration = 2, we launched the point 2

GET http://192.168.240.1/calibrateResult

to get the result for each bath

http://192.168.240.1/cgi-bin/calibrateStart?number=1&point=2&value=7&api=FR45556

Params

number 1
number of the sensor
point 2
number of point (from getAnalyse)
«typeCalibration»: «2», means that 2 baths to calibrate this sensor
value 7
value of the bath
api apikey

Answer
True : true
false : false

a new timer of 60 seconds is launched.

then you must send

http://192.168.240.1/cgi-bin/calibrateResult?number=1&api=FR45556

Answer
True : 0 to 4096 a value from the DAC component on the PCB
false : false

the calibration is done

GET http://192.168.240.1/calibrateCancel

to cancel a calibration in progress

http://192.168.240.1/cgi-bin/calibrateCancel?number=1&api=FR45556

Params

number	1 number of the sensor
api	apikey

Answer

True	: true

GET http://192.168.240.1/getSSID

to get a list of network reached by the controller

http://192.168.240.1/cgi-bin/getSSID?api=FR45556

Params

api	apikey

Answer

True	: a list of network separate by /n
false	: error

GET http://192.168.240.1/resetWifi

to set the controller as factory setup and relaunch Swimo-xxxx network

http://192.168.240.1/cgi-bin/resetWifi?api=FR45556

Params

api	apikey

Answer

True	: true
false	: error

You must physically restart the controller, switch off and on after 5 seconds

GET http://192.168.240.1/setSSID

to get a list of network reached by the controller

http://192.168.240.1/cgi-bin/setSSID?PASS=freebox&PASS=tousdansladrome`&api=FR45556

Params

SSID	**network** (case sensitive)
PASS	password (case sensitive)
api	apikey

Answer

True	: true
false	: error

you must physically restart the controller, switch off and on after 5 seconds

The swimo network disappears, and the controller is now part of the network where he is connected, so that's means that the user box DHCP will attribute a new IP to this device. Then the controller send this new IP to the server to update getAll.

GET https://api.swimo.io/cgi-bin/getTimezone

retrive list of time zone availbale in the world

parameters :

https://api.swimo.io/cgi-bin/getTimezone

Headers

appid	r9isU2Rbl1ajFO172Z0nj0LB8XbpDXGQlBOxCgbGJg valid appid

a list of all time zone and country code in the world


```
{
   «zone»: [
      {
         «timezone»: «Pacific/Wallis»,
         «country»: «WF»
      },
      {
         «timezone»: «Pacific/Wake»,
         «country»: «UM»
      },
      {
         «timezone»: «Pacific/Tongatapu»,
         «country»: «TO»
      },
      {o
         «timezone»: «Pacific/Tarawa»,
         «country»: «KI»
      },
      {
         «timezone»: «Pacific/Tahiti»,
         «country»: «PF»
      },
      {
         «timezone»: «Pacific/Saipan»,
         «country»: «MP»
      },
      {
         «timezone»: «Pacific/Rarotonga»,
         «country»: «CK»
      },
      {
         «timezone»: «Pacific/Port_Moresby»,
         «country»: «PG»
      },
      {
         «timezone»: «Pacific/Pohnpei»,
         «country»: «FM»
      },
      {
         «timezone»: «Pacific/Pitcairn»,
         «country»: «PN»
      },
      {
         «timezone»: «Pacific/Palau»,
         «country»: «PW»
      },
      {
         «timezone»: «Pacific/Pago_Pago»,
         «country»: «AS»
      },
      {
         «timezone»: «Pacific/Noumea»,
         «country»: «NC»
```

M2M services are not available for consumption

thoses services update server every 10 minutes on basic contract and can be changed server side for each client or on all the distributor park.
Bandwidth changes are billed to actual data consumption

- 3 services are running to synchronize server orders and datas storage.

- All services on the local network are doubled to the servers

-A «remotely firmware program» is running in the background to ensure the automatic update of the controllers

4 firmwares are available

- release 0.40.1 - a production firmware for pool & spa industry
- dev version 0.36.7 - a private production firmware for biomass reactor, with a specific IHM to manage 11 processus and 17 equipements
- test version 0.40.6 - a beta test firmware for new improvment (stability unknown)
- futur 0.50.0 - a beta firmware for next generation of board, a wired screen, and so on.