Thinking Bits

V1.8 01-06-2015

Thinking Cleaner API introduction

Discovery
When the ThinkingCleaner module is not configured yet it generates a wifi network. The SSID of this network is formatted as follows: ThinkingCleaner- [first 4 byte of the MAC address]. You can connect to the SoftAP network without a password. If your browser does not support Bonjour you can also use 192.168.1.3 for Thinking Cleaner acces while in wifi direct mode.
This mode is meant only for configuring the Wifi network settings.

When the setup has been done the Thinking Cleaner can be found in several ways.

1. Use a http call : https://thinkingsync.com/mydevices , gives a list of all Thinking Cleaners that are active on the same location (outgoing IP-address)
2. https://thinkingsync.com/mydevices?uuid=deviceUuid , where “deviceUuid” contains the UUID of the Thinking Cleaner. The UUID can be found in full_status.json. This call will automatically redirect to the local IP of the Thinking Cleaner
3. https://thinkingsync.com/api/v1/discover/devices , gives back a json response e.g.: [{“local_ip”:”192.168.1.6”,”uuid”:"d9258affde7d34e","name":"TC-1","device_type":"tc500"}, {"local_ip":"192.168.1.5","uuid":"c1a2a445e0c44321","name":"TC-2","device_type":"tc500"}]
4. Bonjour (ZeroConf mulitcast DNS):
   ThinkingCleaner uses Bonjour to make itself discoverable on the network. When it is not configured yet it registers a service with the following service identifier: xxxx. It also registers itself as a http service with the following service identifier: xxxx. After the Thinking Cleaner module has been configured, it will only register with the http service identifier. For example, if you named your Thinking Cleaner "James", then you can go to http://James.local.

   Apple OSX supports mDNS (Bonjour) automatically, Windows and Linux users will have to install a mDNS package. If you have iTunes on your machine, bonjour is probably installed already and you only have to activate it.
   To install mDNS on windows: http://support.apple.com/kb/DL999
   To install mDNS on linux: https://help.ubuntu.com/community/HowToZeroconf

API change log:

V1.0.84:

* New FULL_STATUS.JSON variables added:
  - “modelnr”: 600 : Roomba model, can be set in web-interface. Adds changed behavior for R800 series.
  - “restart_AC" : 0 : New feature for MAX mode, Roomba will restart once after fully charged. This way you Roomba clean rooms of up to 150m2. We advise to set “dock_at” to 25% so Roomba has enough time to find a dock. **Multiple homebases (>2) are required in different locations of the large room!**

V1.8: Firmware 1.0.80 and higher

* Most values in FULL_STATUS.JSON are now values, no strings anymore. This change will be included in firmware 1.0.80 and higher. To be compatible with older firmware versions you will have to be sure you can accept values as strings and as value.
* New FULL_STATUS.JSON variables added:
  - “dock_at": 10 : Roomba will start docking at 10% charge
  - “stop_at": 6 : Roomba will stop at 6% charge
  - “modelnr”: 600 : Roomba model, can be set in web-interface. Adds changed behavior for R800 series.

  Currents are now formatted as signed values.
* Added Webhook settings in firmware 1.0.80 and higher, see page 14.

First setup
When you are configuring the module, you need to supply the SSID of a network you want the Thinking Cleaner module to connect to. You can use the module to discover networks.

wifi
Use the "/wifi.json" command to get a list of networks (only in SoftAP mode)

Example:
`http://thinking.local/wifi.json`
returns:


```
{
       "action" : "wifi",
       "result" : "success",
       "ssid" : "Thinking08bd",
       "scancount" : "5",
       "scanlist" : [
           {"ssid":"Landgoed Soelen WiFi",
            "bssid":"",
            "privacy":"13",
            "wlan":"1",
            "channel":"1",
            "strength":"1"},
          {"ssid":"Landgoed Soelen WiFi",
           "bssid":"",
           "privacy":"13",
           "wlan":"1",
           "channel":"6",
           "strength":"2"},
          {"ssid":"bits",
           "bssid":"",
           "privacy":"9",
           "wlan":"1",
           "channel":"11",
           "strength":"3"} ]
}
```


Configure
Configure the Thinking Cleaner module with a new name, network and password
Example:
http://thinking.local/configure.json
The following data is set in the request body:

`name=My_Thinking_Cleaner&password=MySecret&ssid=MyNetword&wlan=3&sec=wpa&key=MyPassword`
returns:


```
{
  "action" : "configure",
  "result" : "success"
}
```


The „success” is for the command, not as indication of a successful new connection.

Polling for status information
The ThinkingClean module can return a complete status information of the Roomba.

STATUS.JSON
Example:
http://thinking.local/status.json returns:

```
{
  "action" : "status",
  "result" : "success",
  "status" : {
    “name” : “Thinking”,
    "battery_charge" : 100,
    "capacity" : "2696",
    "cleaner_state" : "st_base_full",
    "cleaning" : "0",
    "schedule_serial_number" : “12”,
    "near_homebase" : "0"
  }
}
```

—Explanation———————

- **name** shows the name you entered during setup
- **battery_charge** gives back 0 to 100 in percent.
  Value is calculated by Roomba and can be wrong when the battery has been removed recently or when the Roomba was reset. When Roomba is in sleep (CLEAN button off) the percentage is calculated by Thinking Cleaner.
- **capacity** is the battery capacity calculated bij Roomba in mA.

- **cleaner_state** gives back left column. Right is the text message associated:

```
st_base           On homebase: Not Charging
st_base_recon     On homebase: Reconditioning Charging
st_base_full      On homebase: Full Charging
st_base_trickle   On homebase: Trickle Charging
st_base_wait      On homebase: Waiting
st_plug           Plugged in: Not Charging
st_plug_recon     Plugged in: Reconditioning Charging
st_plug_full      Plugged in: Full Charging
st_plug_trickle   Plugged in: Trickle Charging
st_plug_wait      Plugged in: Waiting
st_stopped        Stopped
st_clean          Cleaning
st_cleanstop      Stopped with cleaning
st_clean_spot     Spot cleaning
st_clean_max      Max cleaning
st_delayed        Delayed cleaning will start soon ..
st_dock           Searching Homebase
st_pickup         Roomba picked up
st_remote         Remote control driving
st_wait           Waiting for command
st_off            Off
st_error          Error
st_locate         Find me!
st_unknown        Unknown state
```

- **cleaning:** contains "0" when Roomba is not cleaning and "1" when busy with cleaning.
- **schedule_serial_number:** This number is incremented every time the schedule is changed. So this number can be used to check if a local copy of the Thinking Cleaner schedule is still valid.
- **near_homebase:** “1” means that the Roomba has visual contact with a homebase

FULL_STATUS.JSON
Example:
http://thinking.local/full_status.json
returns:


```
{
      "action":"full_status",
      "result":"success",
      "firmware" : {
        "version" : "1.0.84-310C-EU",
        "wifi_version" : "310C",
        "uuid" : "8aa855d4axxxxxx",
        “mac_address":"00:1E:C0:15:XX:XX",
        "DHCP" : 1,
        "has_been_backed_up" : 1,
        "has_auth_token" : 1,
        "boot_status" : "run success",
        "boot_version" : "3",
        "auto_update" : 1,
        "auto_dock" : 1,
        "restart_AC" : 0,
        "dock_at" : 10,
        "stop_at" : 6,
        "time_h_m" : "16:10"
    },
      "tc_status" : {
        "name" : "kompas",
        "modelnr" : "500",
      "cleaning_time" : "0",
      "cleaning_time_total" : "260",
      "cleaning_distance" : "4680",
          "dirt_detected" : 7,
          "bin_status" : 0,
        "server_connection" : 1,
        "vacuum_drive" : 0,
        "clean_delay" : 60,
        "cleaning" : 0,
        "schedule_serial_number" : 28
    },
      "power_status" : {
            "cleaner_state" : "st_base_trickle",
            "current" : 23,
            "charge" : 1618,
        "battery_charge" : "100",
        "capacity" : 1618,
            "voltage" : 17116,
            "temperature" : 41,
        "battery_condition" : "60",
        "low_power" : 0
      },
      "buttons" : {
            "clean_button" : 0,
            "spot_button" : 0,
            "dock_button" : 0
      },
      "sensors" : {
            "bumper_state" : 0,
            "bumper_left_state" : 0,
            "bumper_right_state" : 0,
            "wheel_drop_left" : 0,
            "wheel_drop_right" : 0,
            "wall" : 0,
            "cliff_left" : 0,
            "cliff_front_left" : 0,
            "cliff_right" : 0,
            "cliff_front_right" : 0,
```

```
            "virtual_wall" : 0,
         "dirt_detect" : 0,
         "light_bump" : 0,
         "mainbrush_current" : 0,
         "sidebrush_current" : 0,
         "homebase_detected" : 0,
         "near_homebase" : 0,
         "IR_Omni" : 0,
         "IR_Left" : 0,
         "IR_Right" : 0
      },
      "webview" : {
         "advanced" : "1"
  }
}
```


Explanation: 1 means ON or TRUE and 0 is Off or FALSE. Current, charge and capacity are in mA. Battery_condition is in percent. This value is calculated by Roomba internally and is reset every time the battery is removed or Roomba is reset. A correct value will be shown after a full charge and a cleaning cycle until empty and full charge again.
The battery charge is calculated by Roomba and does not take the external current consumption from Thinking Cleaner into account. This can result in faulty readings, we are aware of this and are busy making a new battery charge function inside the Thinking Cleaner.

cleaning: contains "0" when Roomba is not cleaning and "1" when busy with cleaning.

schedule_serial_number: This number is incremented every time the schedule is changed. So this number can be used to check if a local copy of the Thinking Cleaner schedule is still valid.

near_homebase: "1" means that the Roomba has visual contact with a homebase

All "current" values are in mA and

Sending Commands
Actuator commands are send with the /command.json query and the command as parameter. The following commands are available:

Clean
This command starts a cleaning cycle like pressing the 'Clean' button on the Roomba. When Roomba is sleeping this command will wake Roomba first and execute the Dock command. When Roomba is cleaning, this command will stop the Roomba.

Example:
http://thinking.local/command.json?command=clean
returns:


```
{
  "action" : "clean",
  "result" : "success
}
```


Spot
This command gives the Roomba the command to perform a spot cleaning.

Example:
http://thinking.local/command.json?command=spot
returns:


```
{
   "action" : "spot",
   "result" : "success"
}
```

Delayed Clean command
This command starts a cleaning cycle after xx minutes.

Example:
http://thinking.local/command.json?command=delayedclean
returns:

```
{
  "action" : "delayed clean",
  "result" : "success"
}
```


Delayed Clean time setting
This command starts a cleaning cycle after xx minutes. Minutes can be between 30 and max. 240 (=4 hours)

Example:
http://thinking.local/command.json?command=CleanDelay&minutes=60
returns:

```
{
  "action" : "CleanDelay",
  "result" : "success"
}
```


Maxclean
This command starts a max cleaning cycle on the Roomba. Roomba will start cleaning and it will stop and go back to the dock when the battery charge is below 20%. When Roomba is sleeping this command will wake Roomba first and execute the Dock command. When Roomba is cleaning, this command will stop the Roomba.

In large rooms Roomba will often stop cleaning in the middel of the room. Every 8 meters Roomba will turn and after three times with no virtual wall or a real life obstacle, Roomba will go into an error state and stop cleaning. In the Thinking Cleaner max mode this error is overruled and Roomba will start again automatically.

Be careful, iRobot dit have a good reason to put this error in the Roomba. So if you let your Roomba go outside, it will not stop! Please only use this max mode when Roomba cannot escape from the building it is in.

Example:
http://thinking.local/command.json?command=max
returns:

```
{
  "action" : "max",
  "result" : "success
}
```


Dock
This command gives the Roomba the command to go back to the docking station. When Roomba is sleeping this command will wake Roomba and execute the Dock command. When Roomba is cleaning it will stop the cleaning cycle and start docking.

Example:
http://thinking.local/command.json?command=dock
returns:

```
{
    "action" : "dock",
    "result" : "success"
}
```


Driving from the homebase will cause Roomba to go backwards and turn 180 degrees. This is to be able to drive off the homebase manually. When an obstacle is detected Roomba will slow down.

Drive only:
This command gives the Roomba the command to do drive in a certain direction with a certain speed. Backwards is also supported. Brush and vacuum motors are off
degrees: 180 = forward
degrees: 0 = spinright
degrees: 360 = spinleft
speed: -500 up to +500
degrees: 1..179 = right turn, the lower is more turn
degrees: 181..359 = left turn, the higher is more turn
Example drive forward with speed 200:
http://thinking.local/command.json?command=drive_only&degrees=180&speed=200
returns:

```
{
    "action" : "drive_only"
    "result" : "success"
}
```

Find Me
This command gives the Thinking Cleaner the command to play the Find Me sound.

Example:
http://thinking.local/command.json?command=find_me
returns:

```
{
    "action":"find_me",
    "result":"success"
}
```

Stop driving Roomba
This command will stop driving Thinking Cleaner asap.
Example:
http://thinking.local/command.json?command=drivestop
returns:

```
{
    "action":"drivestop",
    "result":"success"
}
```

Forward driving Roomba
This command will start Roomba driving forward for one second. You will have to repeat this command every 0.5 seconds to get an uninterrupted drive.
Example:
http://thinking.local/command.json?command= forward
returns:

```
{
    "action":"command",
    “result": ""
}
```

Instead of “forward” there are a few other commands you can use to drive Rooma:
`driveleft, driveright, spinleft, spinright,`

Driving Roomba in exception situations
With this command you can set Roomba to drive even when there is an error. **Be careful, you can for example drive it down the stairs when you use this!**
This setting is reset automatically when Roomba is docked or charging.

Example:
http://thinking.local/command.json?command= DriveAlways
returns:

```
{
    "action":"command",
    “result": ""
}
```

`To force reset: DriveNormal`

Driving Roomba with or without vacuum and brushes on
Default Thinking Cleaner will drive your Roomba without the vacuum and brushes on. To turn them on use the following command.

Example:
http://thinking.local/command.json?command= VacuumDriveON
returns:

```
{
    "action":"command",
    “result": ""
}
```

To reset: VacuumDriveOFF

Exit dock command
This command will start Roomba driving backwards and turn 180. Is meant to be used for leaving the dock.
Example:
http://thinking.local/command.json?command= leavehomebase
returns:

```
{
    "action":"command",
    “result": ""
}
```

Scheduling
You can set and retrieve and change the schedule of the Roomba with the following commands:

Get schedule
Retrieve the Roomba's current schedule.

schedule: 0 to 6 where 0 equals Monday, 1 Tuesday, etc.
index: 0 to 3, each schedule can contain max. 4 timers.
time: in seconds 0 to 86399 for 0 to 23:59:99.
command:
- clean = 0
- max = 1
- dock = 2
- stop = 3

Example:
http://thinking.local/schedule.json

Response:

```
{
    "action":"schedule",
    "result":"success",
    "schedule": {
      "0":[],
      "1":[
         {"index":"0","time":"28800","command":"0"},
{"index":"1","time":"75600","command":"0"}
          ],
      "2":[],
      "3":[],
      "4":[],
      "5":[],
      "6":[]
  }
}
```

Two schedules on Tuesday, one at 08:00 and one at 21:00.

Add Schedule
You can add a scheduled command using the add_schedule query. Days are numbered, starting with 0 on Monday. Time is given as the amount of seconds since 00:00.

Example:
http://thinking.local/add_schedule.json?day=0&time=72000&command=0 [GET]

In this example we add a scheduled 'clean' command on Monday at 20:00.

clean = 0
max = 1
dock = 2
stop = 3

Response:

```
{
    "action": "add_schedule",
    "result" : "success"
}
```
After this command the ScheduleContainerSN in status.json will be incremented.

Remove Schedule
You can remove a scheduled command. For this you have to use the weekday and index of the schedule. To get the index of the schedule, you first need to call schedule.json. Beware, when you remove schedules, the index of all the other schedules of a day may changes. So after every time you removed a schedule, you have to call schedule.json again to get the new indexes.

Example:
http://thinking.local/remove_schedule.json?day=0&index=0 [GET]

In this example we remove the scheduled command on monday with index 0.

Response:
```
{
    "action": "remove_schedule",
    "result" : "success"
}
```
After this command the ScheduleContainerSN in status.json will be incremented.

Change Schedule
You can change a schedule. For this you have to use the weekday and index of the schedule. To get the index of the schedule, you first need to call /schedule.json.

Example:
http://thinking.local/change_schedule.json?day=0&index=0&time=0&command=0 [GET]

In this example we change the scheduled command on monday with index 0.

Response:
```
{
    "action": "remove_schedule",
    "result" : "success"
}
```
After this command the ScheduleContainerSN in status.json will be incremented.

Power Off command
This command will switch Roomba off.

Example:
http://thinking.local/command.json?command= poweroff
returns:

```
{
    "action":"command",
    “result": ""
}
```

XML API:
Because some home automation systems (like Gira Homeserver for KNX smart home) do not connect with json files we have included a status.xml and full_status.xml with the same data as in the .json files:

status.xml

```
<status>
   <name>Thinking-Cleaner</name>
   <modelnr>86</modelnr>
   <battery_charge>86</battery_charge>
   <cleaner_state>st_wait</cleaner_state>
   <cleaning>0</cleaning>
   <near_homebase>1</near_homebase>
   <bin_status>0</bin_status>
   <cleaning_time>54</cleaning_time>
   <light_bump>0</light_bump>
   <virtual_wall>0</virtual_wall>
   <wall>0</wall>
   <bumper_state>0</bumper_state>
</status>
```

full_status.xml:

```
<full_status>
<firmware>
<version>1.0.84-310C-EU</version>
<wifi_version>310C</wifi_version>
<uuid>8aa855d4a7xxxxxx</uuid>
<mac_address>00:1E:C0:15:XX:XX</mac_address>
<dhcp>1</dhcp>
<has_been_backed_up>1</has_been_backed_up>
<has_auth_token>1</has_auth_token>
<boot_status>run success</boot_status>
<boot_version>3</boot_version>
<auto_update>1</auto_update>
<auto_dock>1</auto_dock>
<restart_AC>0</restart_AC>
<dock_at>10</dock_at>
<stop_at>6</stop_at>
<time_h_m>16:14</time_h_m>
</firmware>
<tc_status>
<name>kompas</name>
<modelnr>600</modelnr>
<cleaning_time>0</cleaning_time>
<cleaning_time_total>260</cleaning_time_total>
<cleaning_distance>4680</cleaning_distance>
<dirt_detected>7</dirt_detected>
<bin_status>0</bin_status>
<serverconnection>1</serverconnection>
<vacuum_drive>0</vacuum_drive>
<clean_delay>60</clean_delay>
<cleaning>0</cleaning>
<schedule_serial_number>28</schedule_serial_number>
</tc_status>
<power_status>
<cleaner_state>st_base_trickle</cleaner_state>
<current>-31</current>
<charge>1618</charge>
<battery_charge>100</battery_charge>
<capacity>1618</capacity>
<voltage>17005</voltage>
<temperature>41</temperature>
<battery_condition>60</battery_condition>
<low_power>0</low_power>
</power_status>
```

```
<buttons>
<clean_button>0</clean_button>
<spot_button>0</spot_button>
<dock_button>0</dock_button>
</buttons>
<sensors>
<bumper_state>0</bumper_state>
<bumper_left_state>0</bumper_left_state>
<bumper_right_state>0</bumper_right_state>
<wheel_drop_left>0</wheel_drop_left>
<wheel_drop_right>0</wheel_drop_right>
<wall>0</wall>
<cliff_left>0</cliff_left>
<cliff_front_left>0</cliff_front_left>
<cliff_right>0</cliff_right>
<cliff_front_right>0</cliff_front_right>
<virtual_wall>0</virtual_wall>
<dirt_detect>0</dirt_detect>
<light_bump>0</light_bump>
<mainbrush_current>0</mainbrush_current>
<sidebrush_current>0</sidebrush_current>
<homebase_detected>0</homebase_detected>
<near_homebase>0</near_homebase>
<IR_Omni>0</IR_Omni>
<IR_Left>0</IR_Left>
<IR_Right>0</IR_Right>
</sensors>
</full_status>
```

schedule.xml:

```
<root>
<action>schedule</action>
<serial_number>8</serial_number>
<schedule>
{ "0":[ {"index":"0","time":"44100", "command":"0"} ], "1":[ ], "2":
[ {"index":"0","time":"47700", "command":"1"} ], "3":[ ], "4":[ ], "5":[ ], "6":
[ ]}
</schedule>
</root>
```

WEBHOOK (web callback or HTTP push API, available in firmware 1.0.80 and newer) :
The web hook setting will eliminate the need for constant polling the Thinking Cleaner status.
In the web-app on the “Options” page click on advanced settings and the web hook settings will show. You can set a webhook address, path and portnumber. Thinking Cleaner will make a HTTP GET request to your service on every status change. No information is sent with the hook so you use it to trigger a json request to get the info you want.

1) Only HTTP requests, we do not support HTTPS yet.
2) To switch off, leave the webhook field empty
3) The maximum length of both the webhook url and path is 64 characters

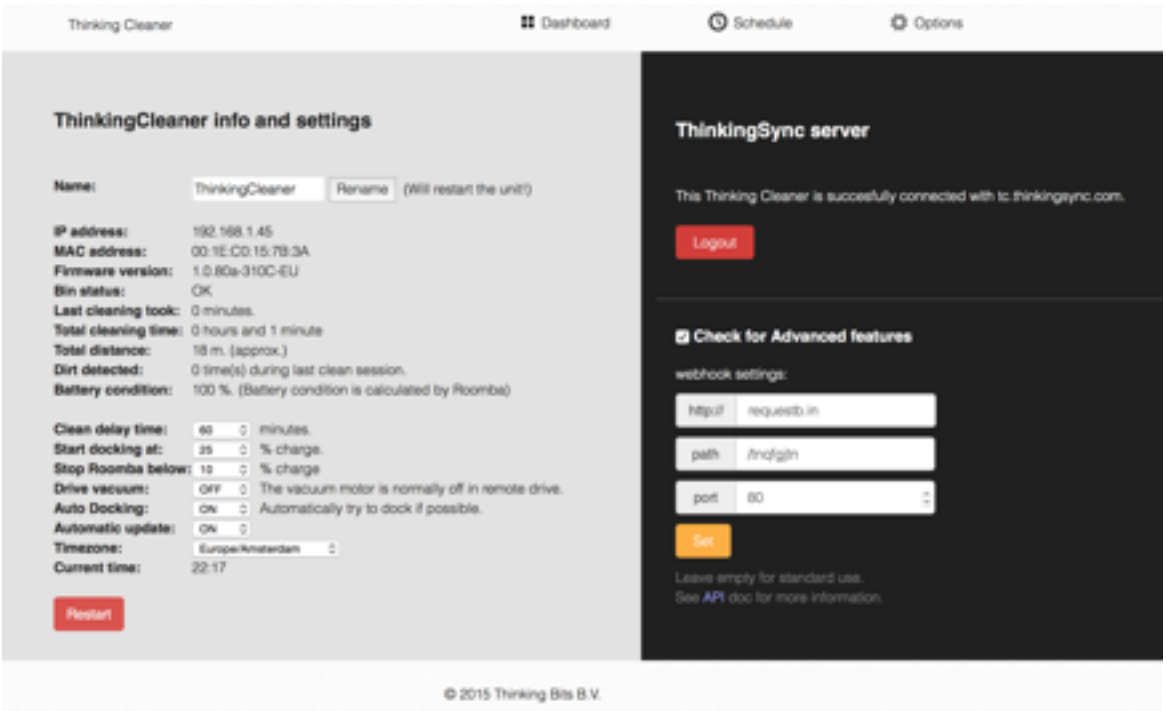
The webhook can also be set with a json request but only within the same wifi network (local).
register_webhook.json

```
h_url  : webhook url, your server for the web hook. (may not end with a '/')
h_path : contains the path of the web hook on your server, must start with a '/'
h_port : webhook port, if you need a port number, put it here

Example:
POST: register_webhook.json
Request body: h_url=myserver.com&h_path=/webhook.html&h_port=nr
```

Thinking Cleaner will not send parameters or a message body in the request, but there is some information in the headers.
The following headers are included in the web hook GET request: Local-Ip, Device-Name and Uuid

Web-app Options page with webhook settings:



- OSX, iPhone and iOS are trademarks of Apple Inc., registered in the U.S. and other countries
- Android is a trademark of Google Inc.
- iRobot, Roomba and Virtual Wall are trademarks of iRobot Corporation
- Thinking Bits and the Thinking Bits logo are registered trademarks of Thinking Bits BV.