



**Version: 1.0**

---

## **Developer Guide v1.0**

Document Release Date: July 20, 2017



Copyright (c) 2017, OCEAN  
All rights reserved.

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

1. Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.
2. Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.
3. The name of the author may not be used to endorse or promote products derived from this software without specific prior written permission.

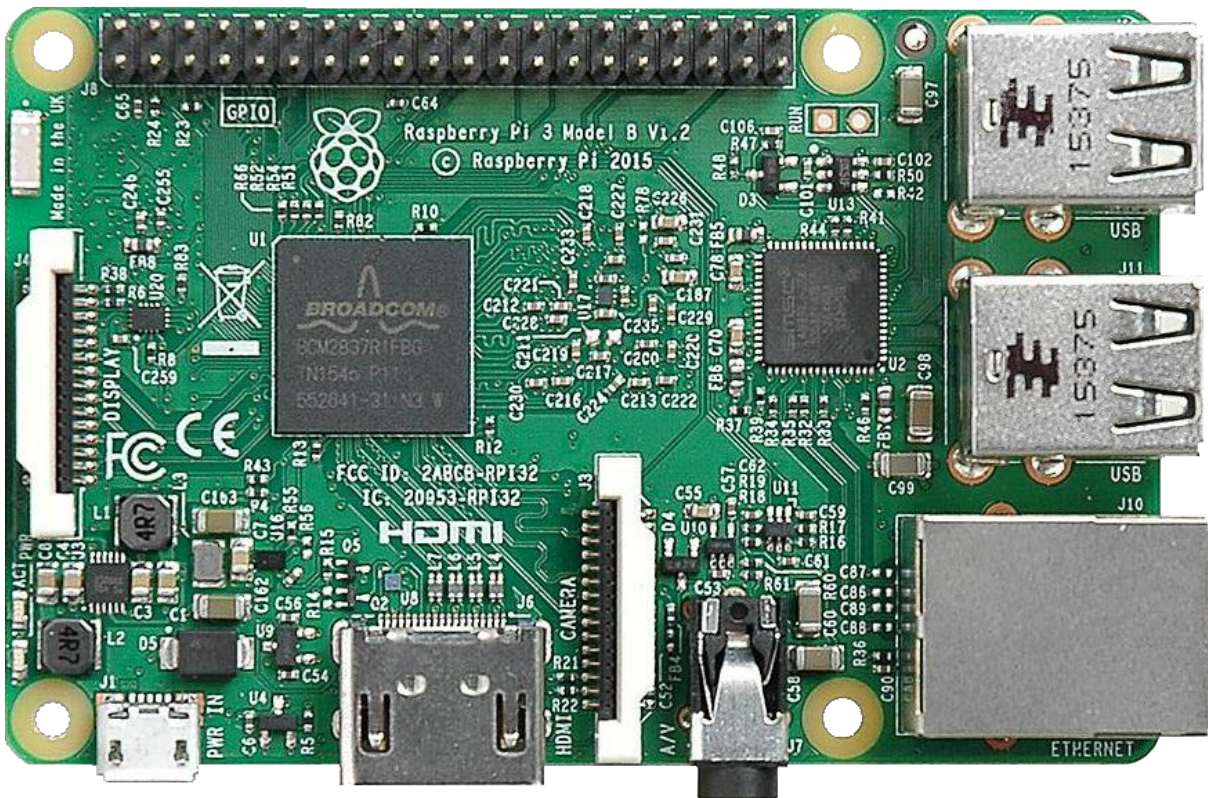
THIS SOFTWARE IS PROVIDED BY THE AUTHOR "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE AUTHOR BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

## nCube: Thyme for Node.js

nCube: Thyme for Node.js is developed based on oneM2M AE Node.js. It supports MQTT, HTTP, CoAP communication protocols.

nCube: Thyme for Node.js can be developed with the Internet connection and Node.js development environment. In this document, Raspberry pi 3 Model B is used.

This document aims to explain the process of developing nCube: Thyme for Node.js that uploads various information in everyday life and provides the control of user actions. The example contains the measurement and upload of Co2 concentration level and the control of LED.



Based on this document, it is possible to create various low-powered IoT devices to detect temperature, humidity, Co2 concentration and the infrared sensor that captures the presence or the movement direction of the people.

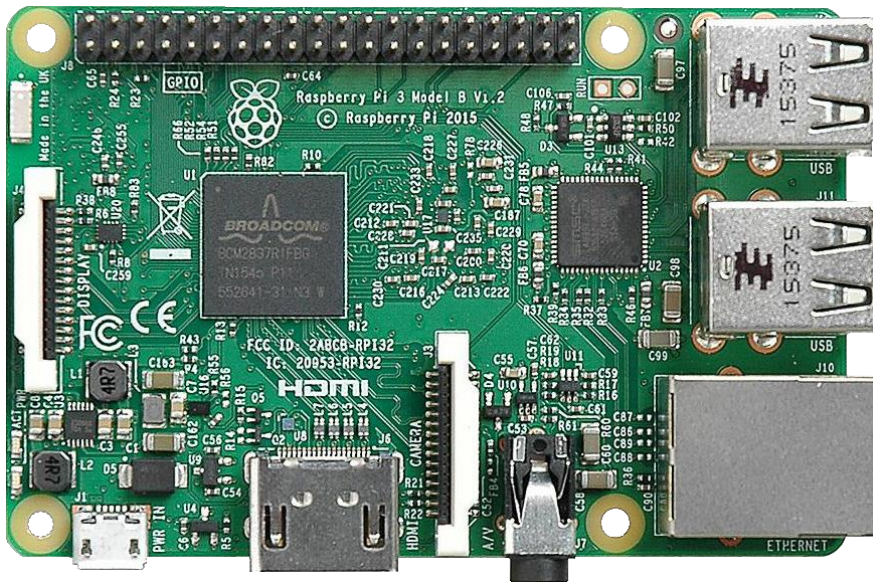
## Contents

1. Hardware .....	5
1.1. Raspberry pi 3 Model B .....	5
1.2. Co2 Sensor .....	5
1.3. PL2303 USB UART Board .....	6
1.4. RGB-LED .....	6
2. Environment .....	6
2.1. Raspbian OS Installation .....	7
2.2. Node.js Installation .....	10
2.3. Samba FTP Server Installation .....	12
2.4. nCube: Thyme for Node.js download .....	15
3. nCube: Thyme for Node.js Execution .....	16
3.1. nCube: Thyme for Node.js + Co2 Sensor .....	16
3.2. nCube: Thyme for Node.js + RGB-LED .....	16
3.3. nCube: Thyme for Node.js Configuration .....	18
3.4. nCube: Thyme for Node.js Package Installation	19 오류! 책갈피가 정의되어 있지 않습니다.
4. nCube: Thyme for Node.js Execution Exercise .....	20
Appendix A .....	23

# 1. Hardware

In chapter 1, hardware specifications for the development of nCube-Thyme for Node.js will be introduced in order of Raspberry pi 3 Model B, Co2 sensor, RGB-LED sensor and Raspberry pi 3 Model B.

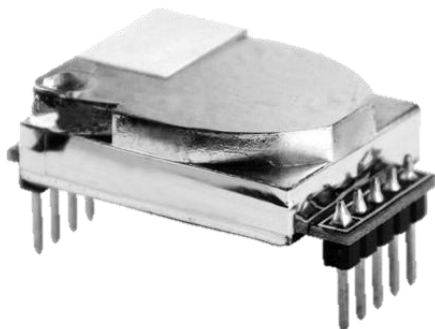
## 1.1. Raspberry pi 3 Model B



Raspberry pi 3 Model B contains BCM2837 64Bit Quad Core Processor (1.2GHz) produced by Broadcom. Consist of 1 GB SDRAM, four USB ports, GPIO (General Purpose Input/Output) 40 pin, Ethernet port WiFi, Bluetooth LE and Power supply using 5 pin micro USB (5V, 2.5A).

It supports OS including NOOBS, Ubuntu Mate, Windows 10 IoT Core, RICS. In this document, Raspbian will be used for the installation.

## 1.2. Co2 Sensor

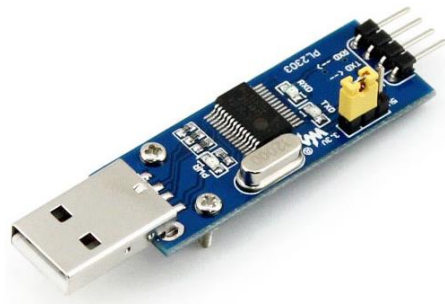


The M1106 Co2 sensor measures the Co2 values from the air and uploads the value to Arduino using the ART port. Output values of 8 bytes (0x16, 0x05, 0x01, 0x02, 0x72, 0x01, 0xD6, 0x9) can be obtained from the input values of 4 bytes (0x11, 0x01, 0x01, 0xED). The output



value of 5, 6 bytes (0x02, 0x72) indicates the Co2 concentration. For instance, 0x0272 = 626 is identical with 626ppm value of the Co2 concentration.

### 1.3. PL2303 USB UART Board



PL2303 USB UART board is a converter that translates UART serial communication to USB communication.

### 1.4. RGB-LED

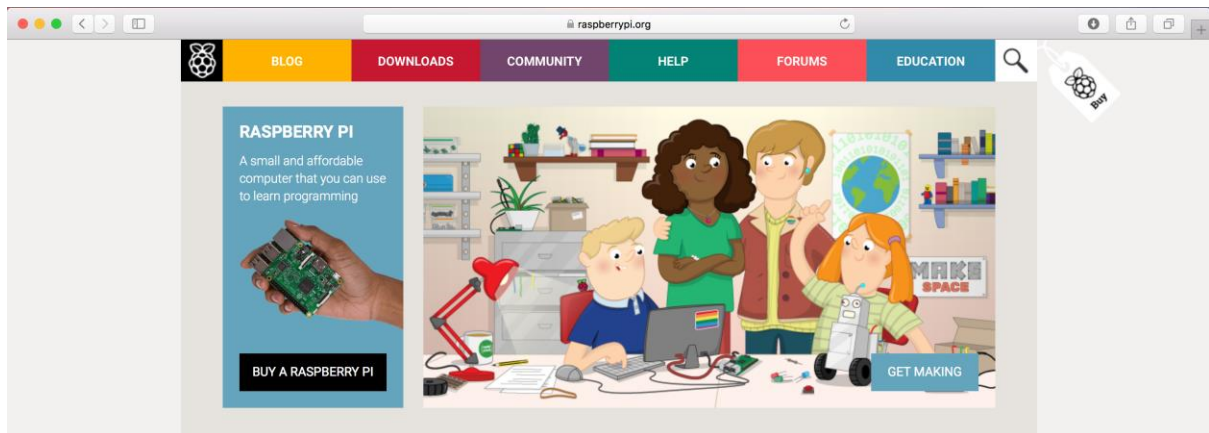


FE-RGB 3-color LED board operates red, green and blue LEDs depending on the input of R, G, B pins. The combination of input values of lights (2(LED On, Off)<sup>3</sup>(LED three types)-1(RGB off case) = 7 types) including three basic red, green and blue can be created.

## 2. Environment

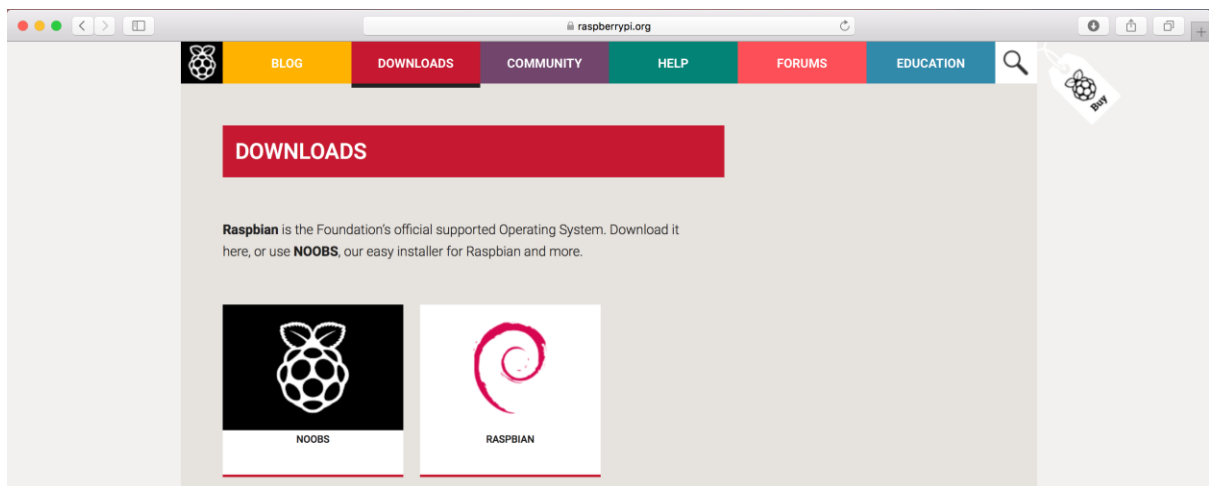
In chapter 2, environmental setting for nCube: Thyme for Node.js development is explained including the installation of Raspbian OS, node.js, samba FTP server and nCube-Thyme.

## 2.1. Raspbian OS Installation



<https://www.raspberrypi.org>

From the URL above, go to Raspberry pi homepage and click *DOWNLOADS* tab.



Click RASPBIAN icon and download Raspbian OS.

RASPBIAN (E) 형식 ×

용량(P):  
7.39GB

파일 시스템(F):  
FAT32(기본값)

할당 단위 크기(A):  
32KB

장치 기본값 복원(D)

블록 레이블(L):  
RASPBIAN

포맷 옵션(O)  
☒ 빠른 포맷(Q)

시작(S) 닫기(C)

Prepare micro SD that is larger than 8GB and format it into FAT32.

**sourceforge**  [Browse](#) [Enterprise](#) [Blog](#) [Deals](#) [Help](#) [Create](#) [Log In or Join](#)

SOLUTION CENTERS Resources Newsletters Cloud Storage Providers Business VoIP Providers Internet Speed Test Call Center Providers

**ONLY \$5.00** PCB PROTOTYPE Lead time 2-3 days  
Sponsor hobbyist- first pcb order free(up to \$50) [Order Now](#)


**Win32 Disk Imager**  
A Windows tool for writing images to USB sticks or SD/CF cards  
Brought to you by: [gruemaster](#), [tuxinator2009](#)

Summary Files Reviews Support Wiki Feature Requests Bugs Code Mailing Lists Blog

★ 4.0 Stars (88)  
↓ 77,266 Downloads (This Week)  
📅 Last Update: 2017-06-15

[Download](#)  
Win32DiskImager-1.0.0-binary.zip

[Browse All Files](#)

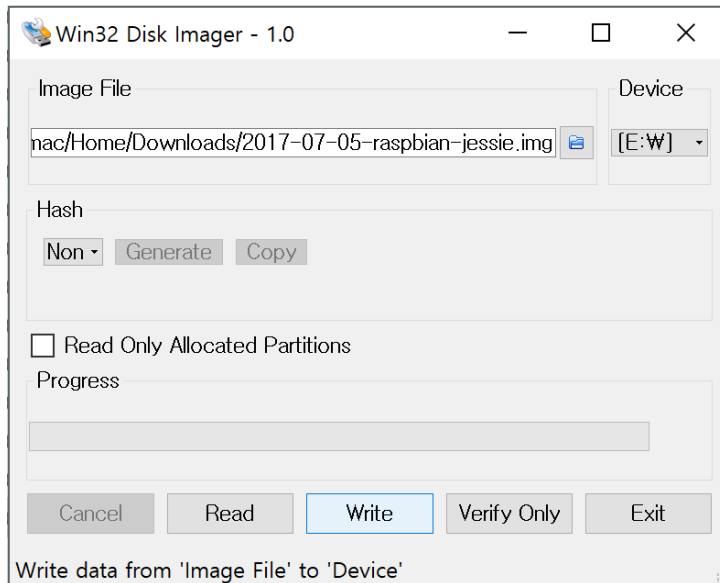


**DEDICATED SERVERS**  
with 10Gbps ports from \$350  
[ORDER SERVER](#)  
LOS ANGELES ATLANTA AMSTERDAM  
10Gbps.io  
FRANKFURT

<https://sourceforge.net/projects/win32diskimager/>

From the URL link above, download and execute win32 diskimager.





Select Raspbian OS image from the download Image File section, select formatted microSD from the Device section. Click Write button to write Raspbian OS into microSD.

After the steps above, put microSD into Raspberry pi 3, and connect the power.

```
deory — pi@raspberrypi: ~ — ssh pi@192.168.0.98 — 80×24
[Deoryui-MacBook-Pro:~ deory$ ssh pi@192.168.0.98
[pi@192.168.0.98's password:

The programs included with the Debian GNU/Linux system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.
Last login: Wed Jul  5 21:06:55 2017

SSH is enabled and the default password for the 'pi' user has not been changed.
This is a security risk - please login as the 'pi' user and type 'passwd' to set
a new password.

pi@raspberrypi:~ $
```

For the remote control of RaspberryPi 3, use `ssh` command from the terminal or use `ssh` by installing putty. The host address is identical with the ip address of Raspberry pi 3. The account is pi and the password is raspberry.

```
pi@raspberrypi ~ $ sudo apt-get update
.....
Reading package lists... Done
pi@raspberrypi ~ $ sudo apt-get upgrade
.....
```

Before installing node.js and samba, repository update and package upgrade should be conducted. Commands are presented as above.

## 2.2. Node.js Installation

Install Node.js package to execute nCube: Thyme for Node.js.

```
pi@raspberrypi ~ $ mkdir node
pi@raspberrypi ~ $ cd node
pi@raspberrypi ~/node $ sudo apt-get remove nodejs
pi@raspberrypi ~/node $ sudo wget https://node-arm.herokuapp.com/node_latest_armhf.deb
pi@raspberrypi ~/node $ sudo dpkg -i node_latest_armhf.deb (패키지 설치 명령어)
pi@raspberrypi ~/node $ node -v (버전 확인 명령어)
pi@raspberrypi ~/node $ npm -v (추가 라이브러리 설치도구 버전 확인 명령어)
```

Delete existing Node.js package and download new Node.js package. Input commands as above in exact order.

```

deory — pi@raspberrypi: ~/node — ssh pi@192.168.0.98 — 80x24
[pi@raspberrypi:~ $ mkdir node
[pi@raspberrypi:~ $ cd node
[pi@raspberrypi:~/node $ sudo apt-get remove nodejs
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following packages were automatically installed and are no longer required:
  libc-ares2 libv8-3.14.5
Use 'apt-get autoremove' to remove them.
The following packages will be REMOVED:
  nodejs
0 upgraded, 0 newly installed, 1 to remove and 0 not upgraded.
After this operation, 2,772 kB disk space will be freed.
[Do you want to continue? [Y/n] y
(Reading database ... 112775 files and directories currently installed.)
Removing nodejs (0.10.29~dfsg-2) ...
Processing triggers for man-db (2.7.5-1~bpo8+1) ...
pi@raspberrypi:~/node $ █

```

Create node directory and deletion of existing node.js package is conducted.

```

deory — pi@raspberrypi: ~/node — ssh pi@192.168.0.98 — 90x24
pi@raspberrypi:~/node $ sudo wget https://node-arm.herokuapp.com/node_latest_armhf.deb
--2017-07-16 16:33:59-- https://node-arm.herokuapp.com/node_latest_armhf.deb
Resolving node-arm.herokuapp.com (node-arm.herokuapp.com)... 54.197.246.226, 54.204.11.224,
, 23.21.114.115, ...
Connecting to node-arm.herokuapp.com (node-arm.herokuapp.com)|54.197.246.226|:443... conne
cted.
HTTP request sent, awaiting response... 200 OK
Length: 5617566 (5.4M) [application/x-debian-package]
Saving to: 'node_latest_armhf.deb'

node_latest_armhf.deb 100%[=====>] 5.36M 823KB/s in 13s

2017-07-16 16:34:15 (418 KB/s) - 'node_latest_armhf.deb' saved [5617566/5617566]

pi@raspberrypi:~/node $ █

```

*wget* command is used to download new Node.js package.

```

deory — pi@raspberrypi: ~/node — ssh pi@192.168.0.98 — 90x24
[pi@raspberrypi:~/node $ sudo dpkg -i node_latest_armhf.deb
Selecting previously unselected package node.
(Reading database ... 110052 files and directories currently installed.)
Preparing to unpack node_latest_armhf.deb ...
Unpacking node (4.2.1-1) ...
Setting up node (4.2.1-1) ...
Processing triggers for man-db (2.7.5-1~bpo8+1) ...
pi@raspberrypi:~/node $ █

```

*dpkg* command is used to install the downloaded Node.js package.

```

deory — pi@raspberrypi: ~/node — ssh pi@192.168.0.98 — 90x24
[pi@raspberrypi:~/node $ node -v
v4.2.1
[pi@raspberrypi:~/node $ npm -v
2.14.7
pi@raspberrypi:~/node $ █

```

After the installation, version information is shown as above.

## 2.3. Samba FTP Server Installation

Samba FTP server installation is for the file sharing with Raspberry pi 3 in the development environment.

```
pi@raspberrypi ~ $ sudo apt-get install samba samba-common-bin
.....
Do you want to continue [Y/n]? Y
```

```
deory — pi@raspberrypi: ~ — ssh pi@192.168.0.98 — 90x24
[pi@raspberrypi:~ $ sudo apt-get install samba samba-common-bin
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following packages were automatically installed and are no longer required:
  libc-ares2 libv8-3.14.5
Use 'apt-get autoremove' to remove them.
The following extra packages will be installed:
  attr libaio1 libasn1-8-heimdal libfile-copy-recursive-perl libhcrypto4-heimdal
  libhdb9-heimdal libheimbase1-heimdal libhx509-5-heimdal libkrb5-26-heimdal
  libroken18-heimdal libwind0-heimdal python-crypto python-dnspython python-ldb
  python-ntdb python-samba python-tdb samba-dsdb-modules samba-vfs-modules tdb-tools
  update-inetd
Suggested packages:
  python-crypto-dbg python-crypto-doc bind9 bind9utils ctdb ldb-tools smbldap-tools
  winbind heimdal-clients
```

Samba installation is completed using *apt-get* commands.

```
pi@raspberrypi ~ $ sudo smbpasswd -a pi
New SMB password: (원하는 패스워드 입력)
Retype new SMB password: (원하는 패스워드 입력)
Added user pi.
```

```
deory — pi@raspberrypi: ~ — ssh pi@192.168.0.98 — 90x24
[pi@raspberrypi:~ $ sudo smbpasswd -a pi
New SMB password:
Retype new SMB password:
Added user pi.
pi@raspberrypi:~ $ █
```

New user registration can be done using *smbpasswd* command.

```
pi@raspberrypi ~ $ sudo nano /etc/samba/smb.conf
```

..... (가장 마지막 줄 밑에)

```
[pi]
```

```
comment = raspberry pi folder
```

```
path = /home/pi
```

```
valid users = pi
```

```
writable = yes
```

```
browseable = yes
```

```
<Ctrl>+<X> → Y → <Enter>
```

```
deory — pi@raspberrypi: ~ — ssh pi@192.168.0.98 — 90x24
```

```
GNU nano 2.2.6
```

```
File: /etc/samba/smb.conf
```

```
# Please note that you also need to set appropriate Unix permissions
# to the drivers directory for these users to have write rights in it
; write list = root, @lpadmin
```

```
[pi]
```

```
comment = raspberry pi folder
```

```
path = /home/pi
```

```
valid user = pi
```

```
writable = yes
```

```
browseable = yes
```

User setting is successfully done by modifying */etc/samba/smb.conf* file.

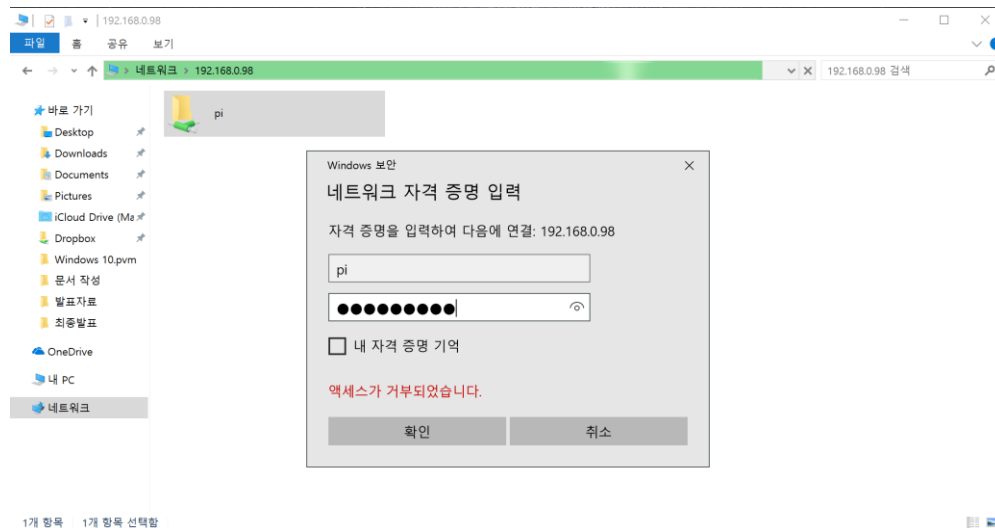
```
pi@raspberrypi ~ $ sudo service smbd restart
```

```
deory — pi@raspberrypi: ~ — ssh pi@192.168.0.98 — 90x24
```

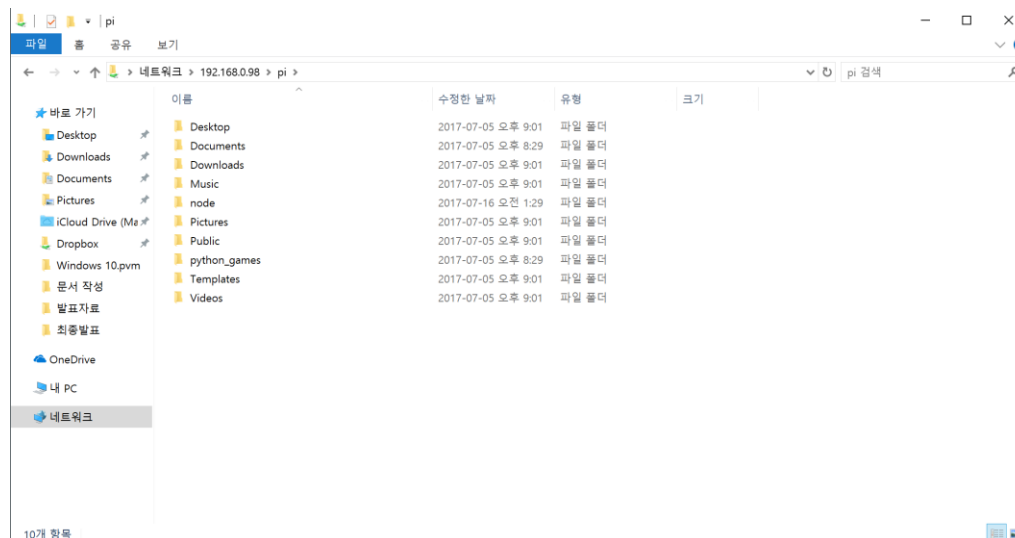
```
[pi@raspberrypi:~ $ sudo service smbd restart
```

```
pi@raspberrypi:~ $
```

Restart samba to set modified user settings by using *sudo service smbd restart* command.



From the explorer window, put Raspberry pi ip address (e.g. 192.168.0.98) in address text box. Account and password are identical to pi and samba user account and password.

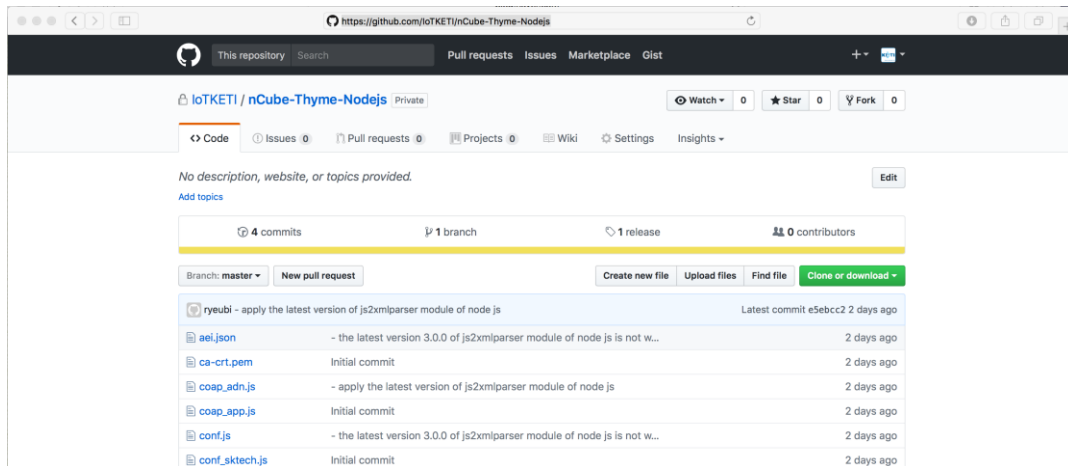


Access to /home/pi directory in Raspberry pi is successfully done as above.

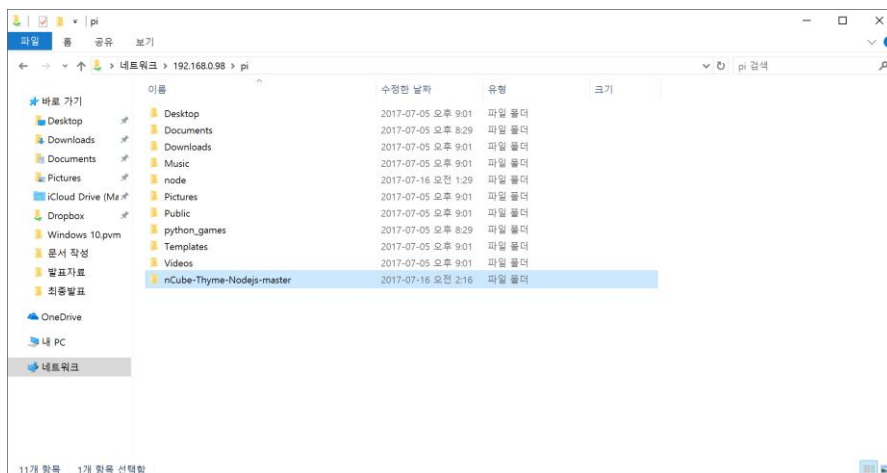


## 2.4. nCube: Thyme for Node.js Download

<https://github.com/loTKETI/nCube-Thyme-Node.js>



From the URL above, access to nCube-Thyme-Node.js repository in loTKETI github. Download nCube:Thyme for Node.js.



Download nCube:Thyme for Node.js and save at Raspberry pi using file explorer.

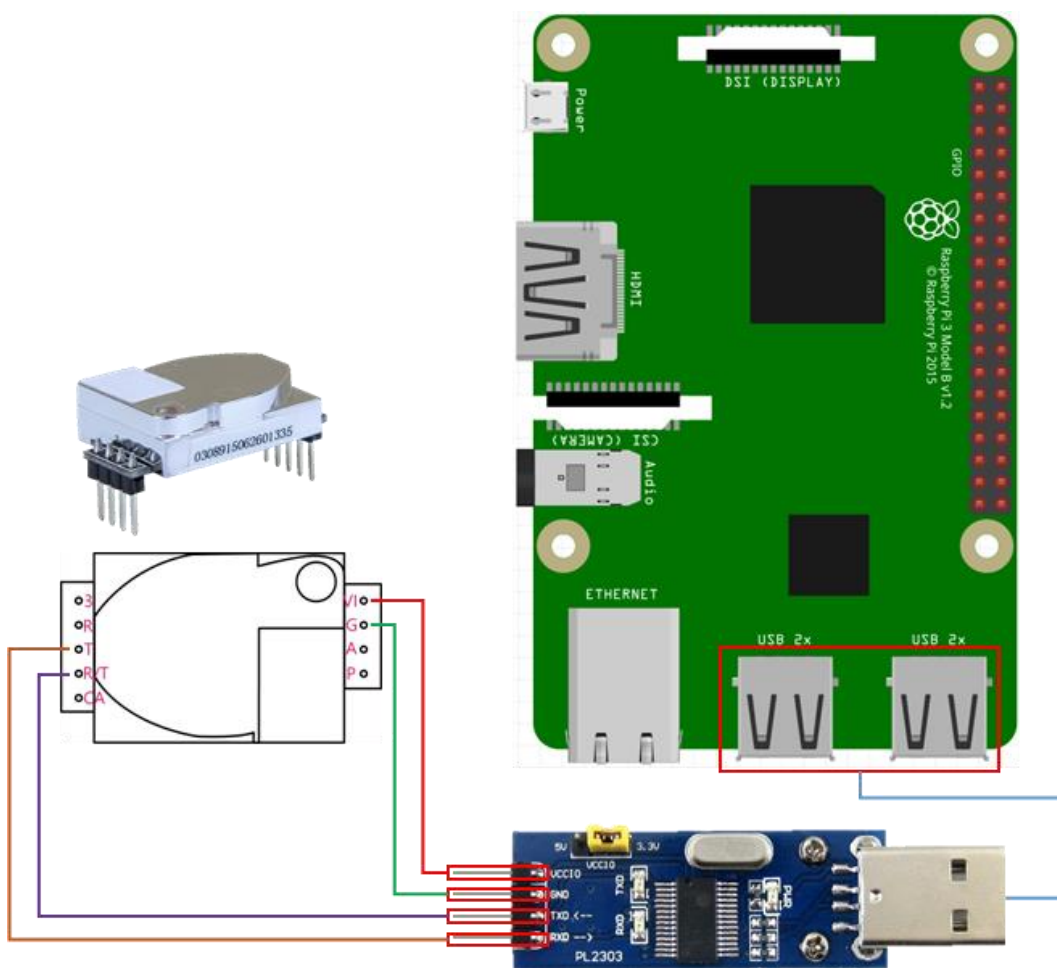


Download is successfully done in Raspberry pi as above.

## 3.nCube: Thyme for Node.js Execution

In chapter 3, nCube: Thyme for Node.js board installation of Co2 sensor, RGB-LED and upload of Co2 concentration to IoT Platform will be done. By using IoT Platform, RGB-LED in nCube: Thyme for Node.js control will also be handled.

### 3.1. nCube: Thyme for Node.js + Co2 Sensor



Raspberry Pi 3 and Co2 sensor connection is as above.

### 3.2. nCube: Thyme for Node.js + RGB-LED

Connect RGB-LED according to Raspberry Pi 3 GPIO pin setting.

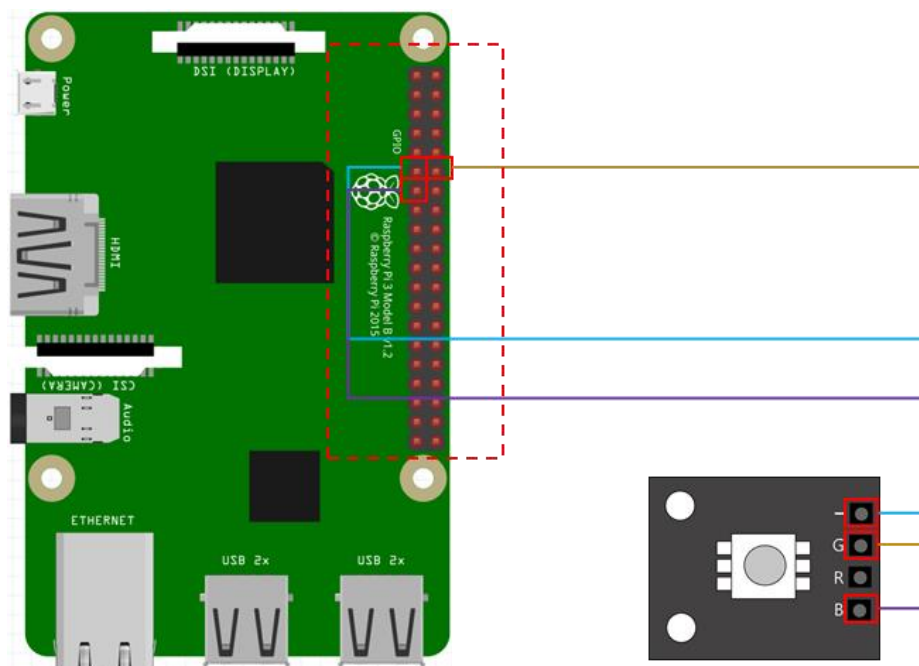
deory — pi@raspberrypi: ~ — ssh pi@192.168.0.98 — 90x27

pi@raspberrypi:~\$ gpio readall

Pi 3											
BCM	wPi	Name	Mode	V	Physical	V	Mode	Name	wPi	BCM	
		3.3v			1	2		5v			
2	8	SDA.1	IN	1	3	4		5v			
3	9	SCL.1	IN	1	5	6		0v			
4	7	GPIO. 7	IN	1	7	8	0	IN	TxD	15	
		0v			9	10	1	IN	RxD	16	
17	0	GPIO. 0	IN	0	11	12	0	IN	GPIO. 1	1	
27	2	GPIO. 2	IN	0	13	14		0v			
22	3	GPIO. 3	IN	0	15	16	0	IN	GPIO. 4	4	
		3.3v			17	18	0	IN	GPIO. 5	5	
10	12	MOSI	IN	0	19	20	0	IN	0v		
9	13	MISO	IN	0	21	22	0	IN	GPIO. 6	6	
11	14	SCLK	IN	0	23	24	1	IN	CE0	10	
		0v			25	26	1	IN	CE1	11	
0	30	SDA.0	IN	1	27	28	1	IN	SCL.0	31	
5	21	GPIO.21	IN	1	29	30		0v			
6	22	GPIO.22	IN	1	31	32	0	IN	GPIO.26	26	
13	23	GPIO.23	IN	0	33	34		0v			
19	24	GPIO.24	IN	0	35	36	0	IN	GPIO.27	27	
26	25	GPIO.25	IN	0	37	38	0	IN	GPIO.28	28	
		0v			39	40	0	IN	GPIO.29	29	

BCM	wPi	Name	Mode	V	Physical	V	Mode	Name	wPi	BCM	
Pi 3											

*gpio readall* command presents the arrangement and use of the GPIO pin.



Connect Raspberry Pi 3 and RGB-LED as above.

### 3.3. nCube: Thyme for Node.js Configuration

```

conf.useprotocol = 'http';

// build cse
cse.host      = '203.253.128.161';
cse.port      = '7579';
cse.name      = 'Mobius';
cse.id        = '/Mobius';
cse.mqttport  = '1883';

// build ae
if (aei != 'S') {
  ae.id       = aei;
} else {
  ae.id       = 'S';
}
ae.id        = 'ae-edu2';
ae.parent    = '/' + cse.name;
ae.name      = 'ae-edu2';
ae.appid     = 'measure_co2';
ae.port      = '9727';
ae.bodytype  = 'json';
ae.tasport   = '3105';

// build cnt
var count = 0;
cnt_arr[count] = {};
cnt_arr[count].parent = '/' + cse.name + '/' + ae.name;
cnt_arr[count++].name = 'cnt-co2';
cnt_arr[count] = {};
cnt_arr[count].parent = '/' + cse.name + '/' + ae.name;
cnt_arr[count++].name = 'cnt-led';
//cnt_arr[count] = {};
//cnt_arr[count].parent = '/' + cse.name + '/' + ae.name;
//cnt_arr[count++].name = 'cnt-cam';

// build sub
count = 0;
//sub_arr[count] = {};
//sub_arr[count].parent = '/' + cse.name + '/' + ae.name + '/' + cnt_arr[1].name;
//sub_arr[count].name = 'sub-ctrl';
//sub_arr[count++].nu = 'mqtt://' + cse.host + '/' + ae.id;

sub_arr[count] = {};
sub_arr[count].parent = '/' + cse.name + '/' + ae.name + '/' + cnt_arr[1].name;
sub_arr[count].name = 'sub-ctrl2';

//var ip = require("ip");
//sub_arr[count++].nu = conf.userprotocol + '://' + ip.address() + ':' + ae.port + '/noti';

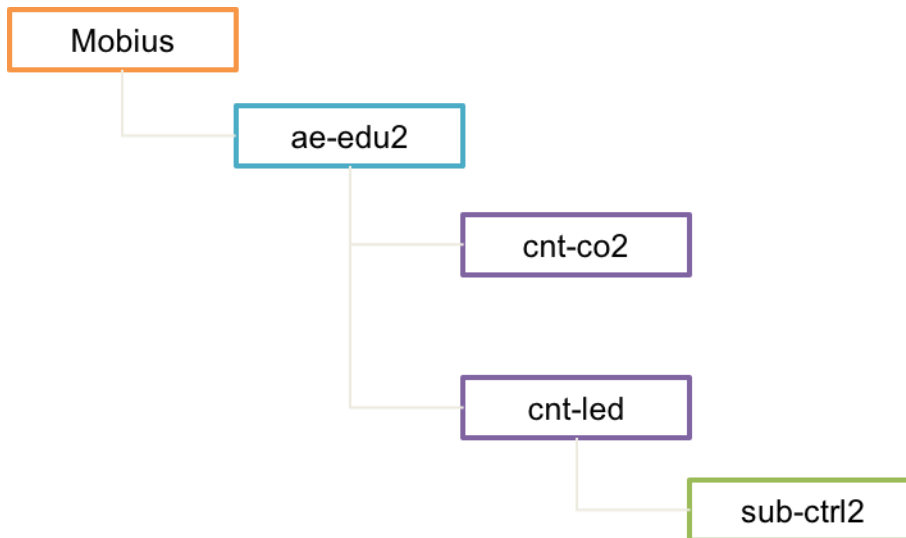
//sub_arr[count++].nu = 'mqtt://' + cse.host + '/' + ae.id + '?rcn=9';
sub_arr[count++].nu = 'mqtt://' + cse.host + '/' + ae.id + '?ct=' + ae.bodytype;
//var ip = require("ip");
//sub_arr[count++].nu = 'http://' + ip.address() + ':' + ae.port + '/noti';
//sub_arr[count++].nu = 'coap://203.254.173.104:' + ae.port + '/noti';

// build acp: not complete
acp.parent = '/' + cse.name + '/' + ae.name;
acp.name = 'acp-' + ae.name;
acp.id = ae.id;

conf.usesecondary = 'disable';

```

*conf.js* file in nCube: Thyme for Node.js is as above.



The resource structure to be created at IoT Platform by nCube: Thyme for Node.js is as above.

### 3.4. nCube: Thyme for Node.js Package Installation

```

deory — pi@raspberrypi: ~/nCube-Thyme-Nodejs-master — ssh pi@192.168.0.98 — 90x24
[pi@raspberrypi:~/nCube-Thyme-Nodejs-master $ npm install
npm WARN package.json thyme@1.7.2 No repository field.
npm WARN package.json thyme@1.7.2 No README data

> websocket@1.0.24 install /home/pi/nCube-Thyme-Nodejs-master/node_modules/websocket
> (node-gyp rebuild 2> builderror.log) || (exit 0)

make: Entering directory '/home/pi/nCube-Thyme-Nodejs-master/node_modules/websocket/build'
CXX(target) Release/obj.target/bufferutil/src/bufferutil.o
SOLINK_MODULE(target) Release/obj.target/bufferutil.node
COPY Release/bufferutil.node
CXX(target) Release/obj.target/validation/src/validation.o
SOLINK_MODULE(target) Release/obj.target/validation.node
COPY Release/validation.node
make: Leaving directory '/home/pi/nCube-Thyme-Nodejs-master/node_modules/websocket/build'
http@0.0.0 node_modules/http

twitter@1.7.1 node_modules/twitter
├─┬ deep-extend@0.5.0
│   └─ request@2.81.0 (aws-sign2@0.6.0, tunnel-agent@0.6.0, forever-agent@0.6.1, oauth-sign@0.8.2, is-typedarray@1.0.0, caseless@0.12.0, safe-buffer@5.1.1, stringstream@0.0.5, aws4@1.6.0, isstream@0.1.2, json-stringify-safe@5.0.1, extend@3.0.1, performance-now@0.2.0, uuid@3.1.0, qs@6.4.0, combined-stream@1.0.5, mime-types@2.1.15, tough-cookie@2.3.2, form-data@2.1.4, hawk@3.1.3, http-signature@1.1.1, har-validator@4.2.1)
└─ websocket@1.0.24 node_modules/websocket
    └─ yaeti@0.0.6
        └─ typedarray-to-buffer@3.1.2 (is-typedarray@1.0.0)
            └─ debug@2.6.8 (ms@2.0.0)
                └─ nan@2.6.2

xml2js@0.4.17 node_modules/xml2js
├─ sax@1.2.4
└─ xmlbuilder@4.2.1 (lodash@4.17.4)
pi@raspberrypi:~/nCube-Thyme-Nodejs-master $ ]
  
```

Inside nCube: Thyme for Node.js directory, input command

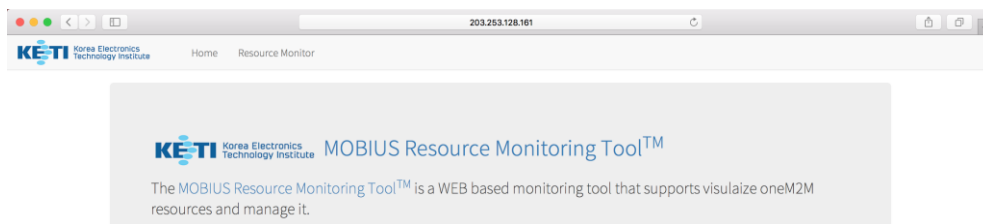
```
sudo npm install
```

and install Node.js package for nCube: Thyme for Node.js execution.

## 4. nCube: Thyme for Node.js Execution Exercise

In this chapter, execution of nCube: Thyme for Node.js and the monitoring of uploading Co2 data and RGB-LED control of Raspberry Pi using Mobius Resource Monitor will be handled.

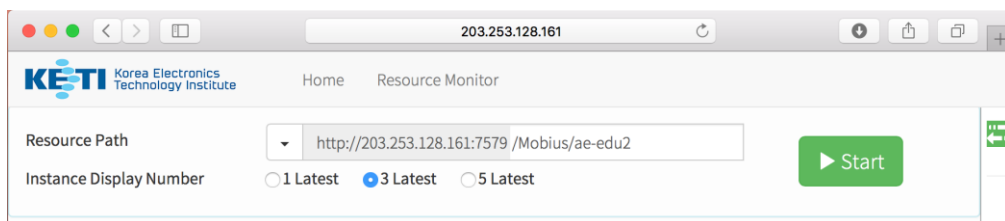
<http://203.253.128.161:7575/>



From the URL above, go to Mobius Resource Monitor web version.

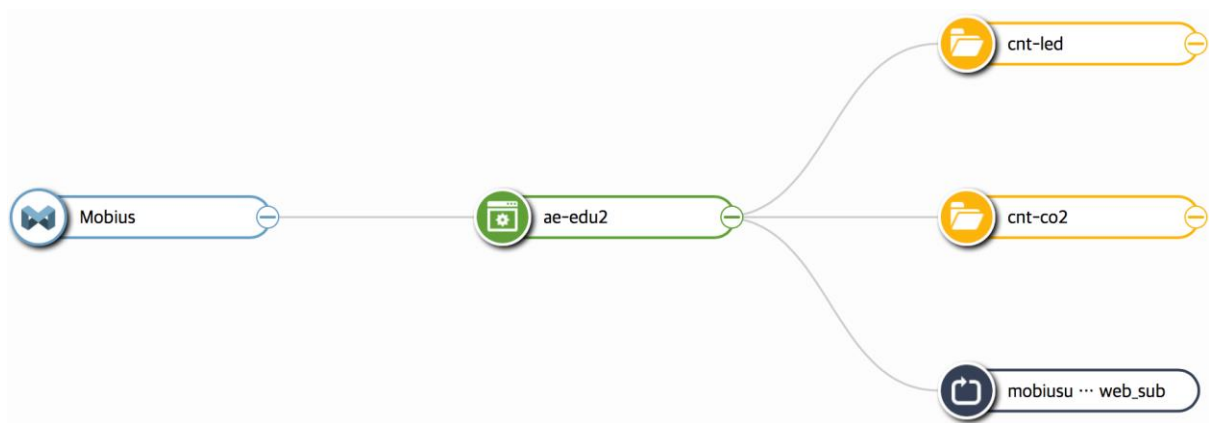
Select the *Resource Monitor* tab to use *Resource Monitor function* and go to Resource Monitor page.

<http://203.253.128.161:7579/Mobius/ae-edu2>

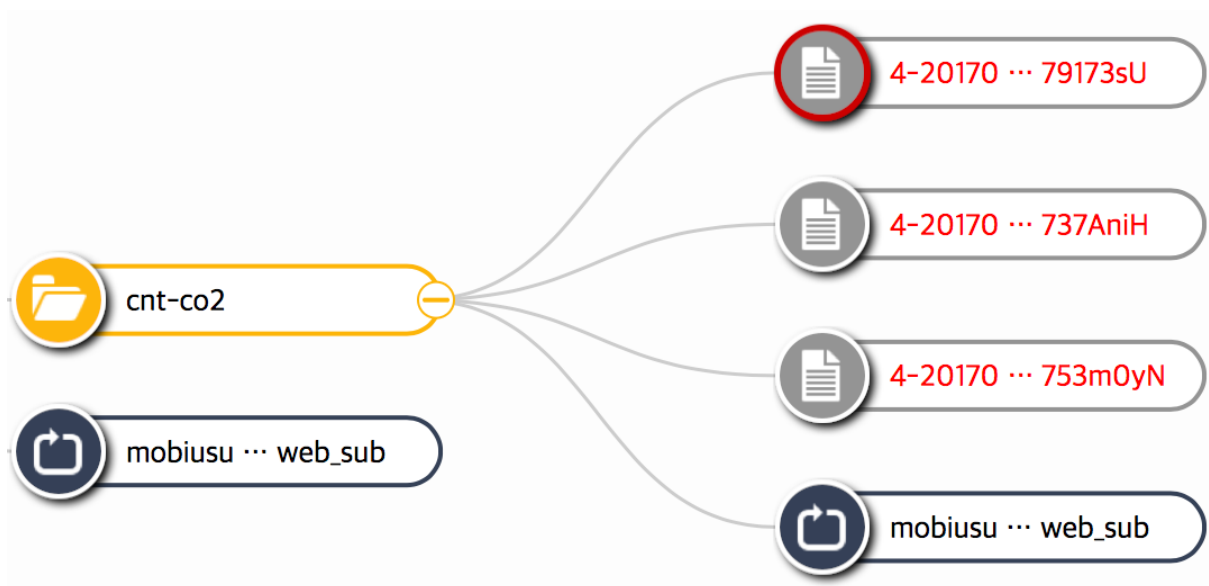


Put the address above into Resource Path and click *Start* button. The resource tree created by nCube-Mint board will be presented.

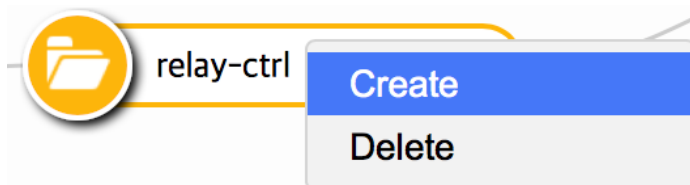




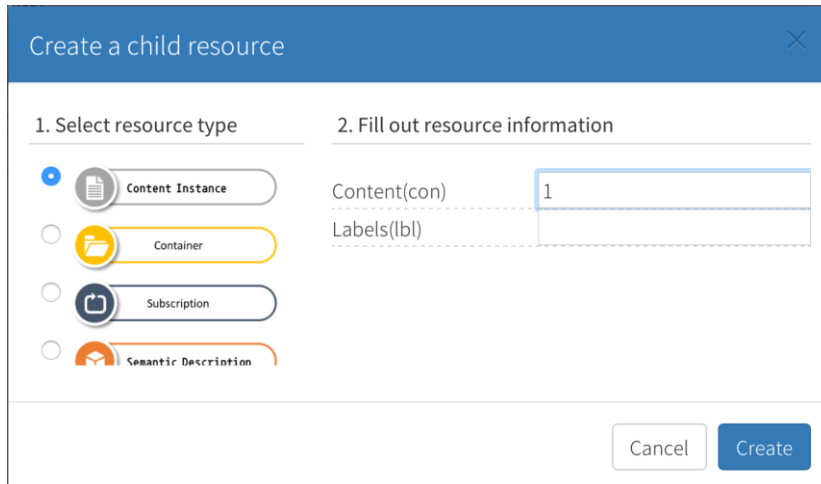
Queried resource structure is as above.



Co2 concentration data is uploaded under cnt-co2 as above.



Right click over *relay-ctrl cnt* and click *Create* button.



From *the Create a child resource* window, confirm that Content Instance is checked in Select resource type section on the left. Put 1 in the Content(con) section in Fill out resource information on the right. Then, click *Create* button.

```
deory — pi@raspberrypi: ~/nCube-Thyme-Nodejs-master/tas_sample/tas_led...
[pi@raspberrypi:~/nCube-Thyme-Nodejs-master/tas_sample/tas_led $ node app.js ]
tas init ok
Argument 0 is ./led
Argument 1 is 0
Init Light!

upload Connected
download Connected - cnt-led hello
Received: {"ctname":"cnt-led","con":"hello"}
{"id":"led#1","con":"1"} <----
Argument 0 is ./led
Argument 1 is 1
Green Light ON!
```

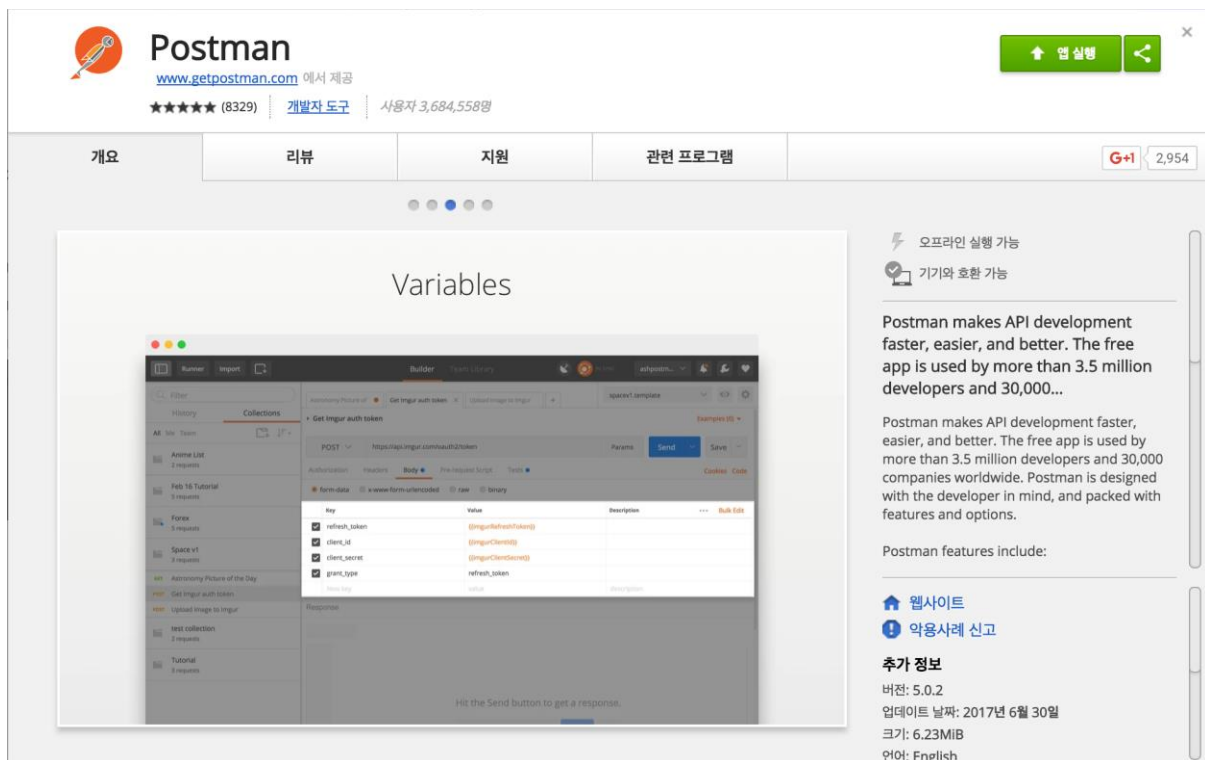


## Appendix A

# nCube: Thyme for Node.js Resource Query and Control Using Postman

Browsing Co2 data updated with nCube: Thyme for Node.js and controlling LED are available with Postman Application in Google Chrome. Moreover, AE, CNT, CIN creation and querying with API is also available.

[https://chrome.google.com/webstore/detail/postman/fhbjgfbiflinjbdggehcdcbncdddomo?utm\\_source=chrome-ntp-icon](https://chrome.google.com/webstore/detail/postman/fhbjgfbiflinjbdggehcdcbncdddomo?utm_source=chrome-ntp-icon)



Download Postman Application in Google Chrome from the URL link above, and execute.

The screenshot shows the OCEAN website's 'Download' page for 'Yellow Turtle'. The page is titled 'Download - Yellow Turtle' and features a 'Latest Version' section with 'Mobius' and 'Yellow Turtle' options. Below this, there is a 'Files' section with a table of download links.

Name	Download Link
Mobius Installation Guide Korea	<a href="#">Installation_Guide_Mobius_Yellow_Turtle_v2.3.4_KR.pdf</a>
Mobius Installation Guide English	<a href="#">Installation_Guide_Mobius_Yellow_Turtle_v2.0_ENG.pdf</a>
REST API for Mobius Yellow Turtle	<a href="#">REST_API_for_Mobius_Yellow_Turtle_v2.0_ENG.pdf</a>
Mobius Yellow Turtle Source	<a href="#">mobius-yt-v2.3.8.zip</a>
Mysql DB Script	<a href="#">Mobiusdb_v2.2.20.sql(TableSpace Regeneration Required !)</a>
POSTMAN Environment Script	<a href="#">mobius-yt-7579.postman_environment.json</a>
POSTMAN Script	<a href="#">mobius-yt-release1.postman_collection.json</a> <a href="#">test-as-virtual-device.postman_collection.json</a> <a href="#">test-for-group-resource-fanoutpoint.postman_collection.json</a>

Go to the ocean official website (<http://www.iotocean.org>), Download>Mobius Yellow Turtle>POSTMAN Environment Scrip and click mobius-yt-7579.postman\_environment.json and download Postman API collection.

The screenshot shows the Postman application interface. On the left, the 'Collections' list is visible, showing 'mobius-yt-release1' with 41 requests. The main area is the 'Builder' tab, which is currently empty, showing a 'GET' request with a placeholder for the URL and a 'Send' button.

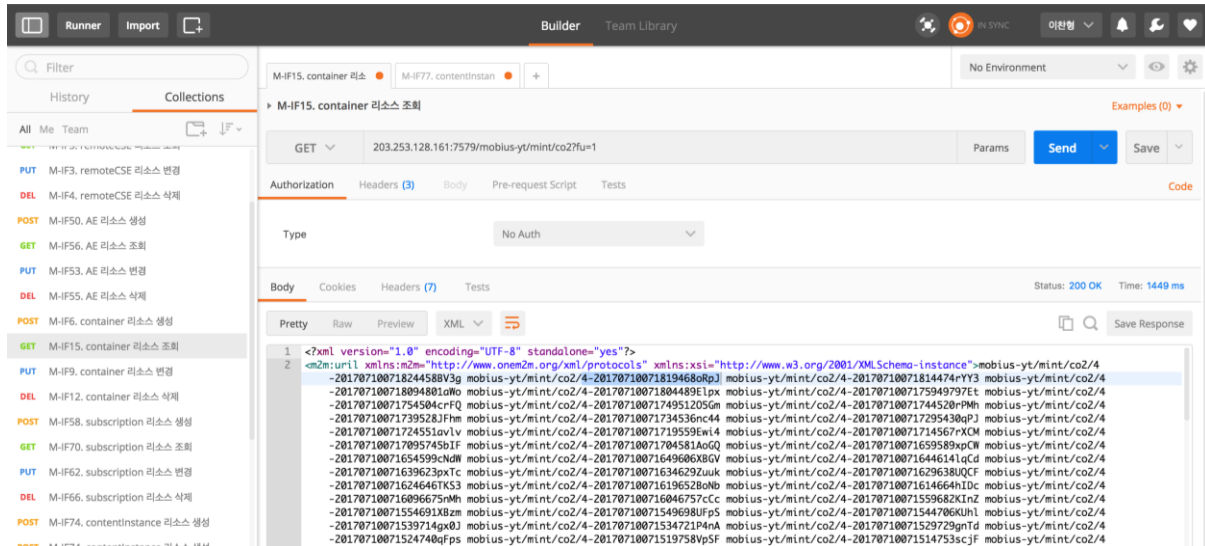
Open API collection in Postman by *import* → *import file* → *choose files*, then APIs are added under *Collections* list on the left.

Query *cin* that contains Co2 data, create *cin* for RGB-LED control is available using added

APIs.

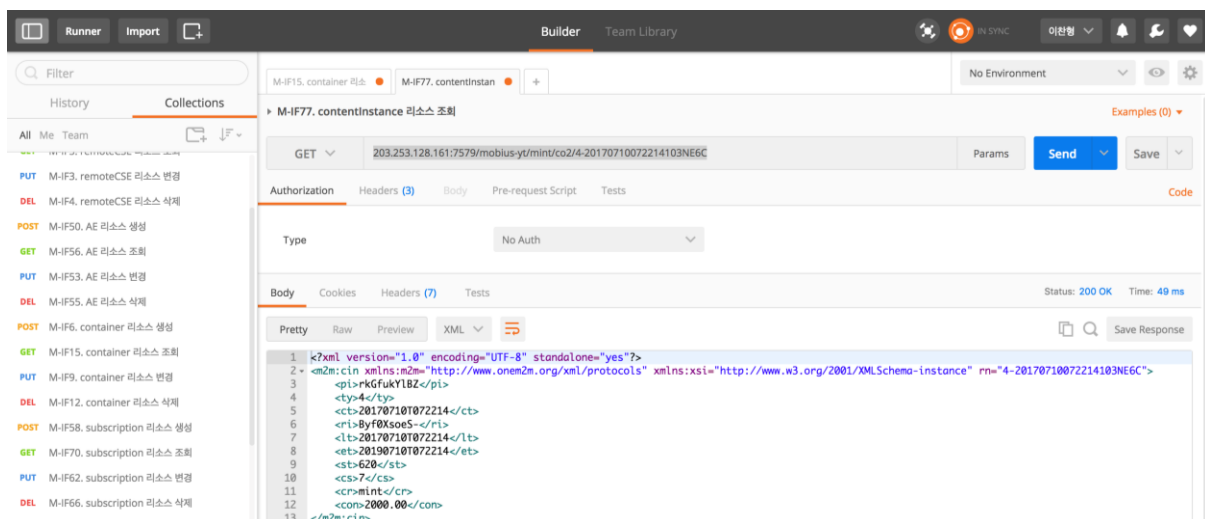
From the list on the left, select container resource inquiry. Input URL as follows and click blue *Send* button. The list of *cin* created in co2 are shown as follows.

203.253.128.161:7579/Mobius/ae-edu2/cnt-co2?fu=1



Copy one of *cin* names, then select *contentinstance* resource query from the list on the left. Input URL as follows then click *Send* button. Co2 concentration values those are uploaded under *con* field inside *cin* are shown as follows.

203.253.128.161:7579/Mobius/ae-edu2/cnt-co2/[contentinstance resource name]



Select *contentinstance* Resource Creation from the left list for RGB-LED control. Input URL as follows and put the values under *con* and *Body* section. Click *Send* button and the RGB-LED control will successfully work.

203.253.128.161:7579/Mobius/ae-edu2/cnt-led

The screenshot shows a REST client interface with a sidebar on the left listing various API endpoints. The main panel displays a POST request to the URL `203.253.128.161:7579/mobius-yt/mint/led-ctrl`. The request body is an XML document. The response status is `201 Created` with a time of `96 ms`.

**Request:**

```
POST 203.253.128.161:7579/mobius-yt/mint/led-ctrl
```

**Body:**

```
<?xml version="1.0" encoding="UTF-8"?>
<m2m:cin xmlns:m2m="http://www.onem2m.org/xml/protocols" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
  <con>5</con>
</m2m:cin>
```

**Response:**

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
2 - <m2m:cin xmlns:m2m="http://www.onem2m.org/xml/protocols" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" rn="4-20170710120106215885n">
3   <ty>4</ty>
4   <pi>SjzNy8YlRw</pi>
5   <ri>B1f5K21-rW</ri>
6   <ct>20170710120106</ct>
7   <et>20170710120106</et>
```