



OCEAN

nCube:Thyme for Node.js

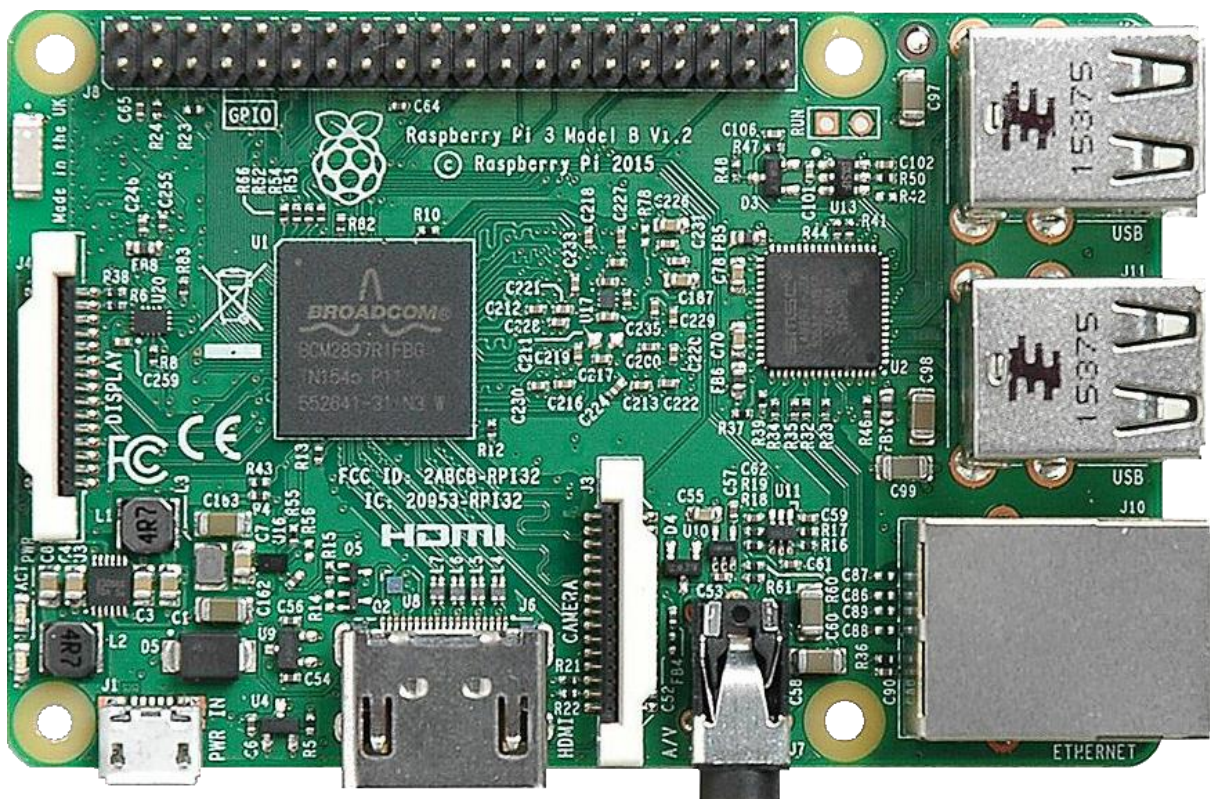
Ver.1.0

nCube:Thyme for Node.js

nCube:Thyme for Node.js 는 oneM2M AE Node.js 를 이용하여 구현한 것이다. 지원하는 통신 프로토콜은 MQTT, HTTP, CoAP 이다.

nCube:Thyme for Node.js 는 인터넷 연결과, Node.js 환경이 갖추어 진 어떠한 환경에서도 동작할 수 있으며, 본 문서에서는 Raspberry pi 3 Model B 를 기준으로 설명한다.

Mobius IoT Platform 에 생활속에서 수집될 수 있는 다양한 정보를 업로드하고, 사용자의 동작 제어를 제공하는 nCube:Thyme for Node.js 보드를 제작하는 과정의 설명을 목적으로 하며, nCube:Thyme for Node.js 를 활용한 Co2 농도의 측정 및 업로드, LED 제어를 제공하는 예제를 통해 설명한다.



본 문서의 내용을 바탕으로 nCube:Thyme for Node.js 에 다양한 센서를 연결하여 생활 속 곳곳에서 온도, 습도, Co2 농도, 적외선을 이용한 특정 공간에서의 사람의 존재 유무나 이동방향 등을 탐지할 수 있는 저전력 IoT Device 제작이 가능하다.

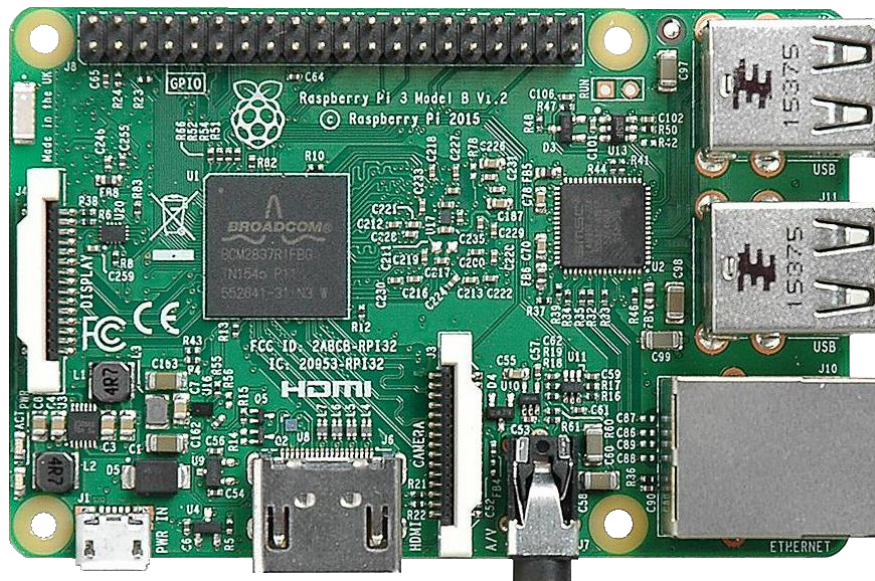
목차

1 하드웨어	4
1 - 1 Raspberry pi 3 Model B	4
1 - 2 Co2 센서	5
1 - 3 PL2303 USB UART Board	5
1 - 4 RGB-LED	5
2 환경설정	6
2 - 1 Raspbian OS 설치	6
2 - 2 Node.js 설치	10
2 - 3 Samba FTP 서버 설치	12
2 - 4 nCube:Thyme for Node.js 다운로드	15
3 nCube:Thyme for Node.js 구동 준비	17
3 - 1 nCube:Thyme for Node.js + Co2 Sensor	17
3 - 2 nCube:Thyme for Node.js + RGB-LED	18
3 - 3 nCube:Thyme for Node.js configure	19
3 - 4 nCube:Thyme for Node.js Packages Install	21
4 nCube:Thyme for Node.js 구동 실습	22
Appendix A	25

1 하드웨어

1장에서는 nCube:Thyme for Node.js 를 구성하기 위한 하드웨어에 대해 간단히 설명한다. 기본이 되는 Adafruit Feather M0 에 대한 소개를 시작으로, Co2 센서, RGB-LED 센서, Adafruit Power Relay FeatherWing 에 대해 소개한다.

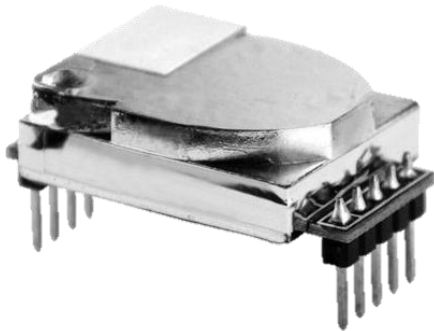
1 - 1 Raspberry pi 3 Model B



Raspberry pi 3 Model B 는 1.2GHz 로 동작하는 Broadcom 의 BCM2837 64Bit Quad Core Processor 를 탑재하고 있으며, 1GB SDRAM 과, 4 개의 USB 포트, GPIO(General Purpose Input/Output)40 pin, 이더넷 포트와 WiFi, Bluetooth LE 를 지원한다. 전원은 5pin micro USB 를 통해 5V, 2.5A 를 입력받는다.

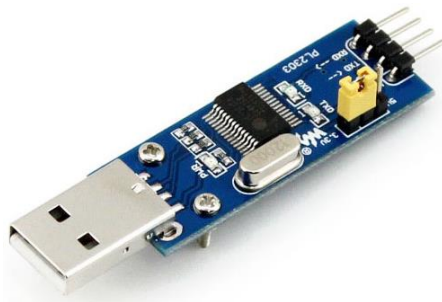
NOOBS, Ubuntu Mate, Windows 10 IoT Core, RISC 등의 OS 를 지원하며, 본 문서에서는 Raspbian 을 설치하여 사용한다.

1 - 2 Co2 센서



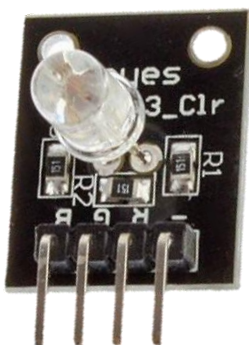
CM1106 Co2 센서는 대기중의 Co2 농도를 측정하여 UART 포트를 이용해 Arduino 로 업로한다. 4 바이트(0x11, 0x01, 0x01, 0xED) 입력값을 통해 8 바이트의 출력값(0x16, 0x05, 0x01, 0x02, 0x72, 0x01, 0xD6, 0x9)을 얻을 수 있으며, 5, 6 바이트의 출력값 0x02, 0x72 가 Co2 농도를 나타내는 수치로써, 위의 예에서 0x0272 = 626, 즉 Co2 농도가 626ppm 임을 나타낸다.

1 - 3 PL2303 USB UART Board



PL2303 USB UART 보드는 UART 방식의 시리얼 통신을 USB 통신으로 변환해 주는 컨버터이다.

1 - 4 RGB-LED



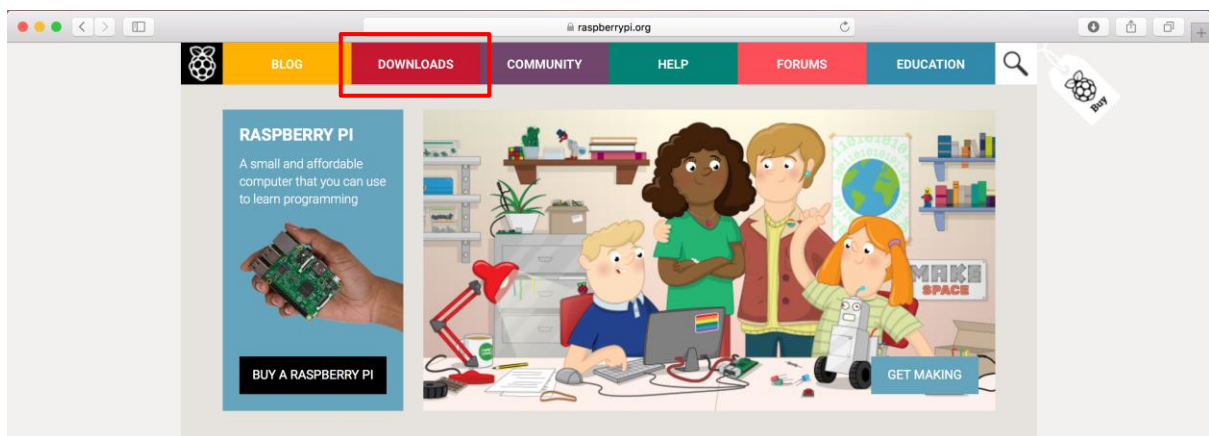
FE-RGB 3 색 LED 보드는 각 R, G, B 핀의 입력에 따라 Red, Green, Blue LED 를 동작시킨다.

기본적인 RGB 3 가지 불빛 이외에도, 입력값의 조합에 따라 하나 이상의 빛($2(\text{LED On, OFF})^3(\text{LED 3 개}) - 1(\text{RGB off case}) = 7$ 가지)을 낼 수 있다.

2 환경설정

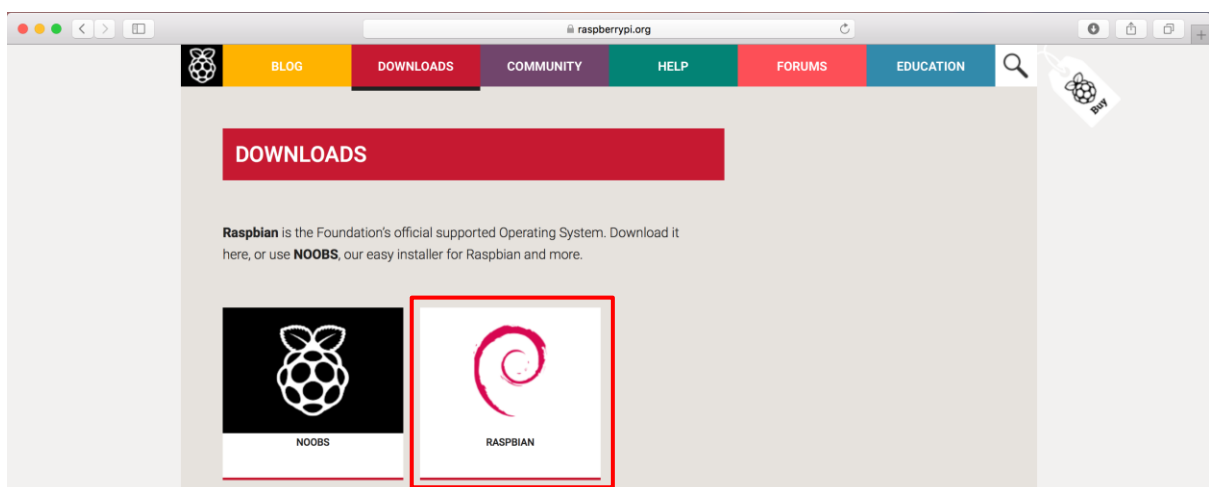
2 장에서는 nCube:Thyme for Node.js 보드를 제작하기 위한 환경설정에 대해 설명한다.
Raspbian OS 의 설치, node.js 설치, samba FTP 서버 설치, nCube-Thyme 설치의 내용으로 구성하였다.

2 - 1 Raspbian OS 설치



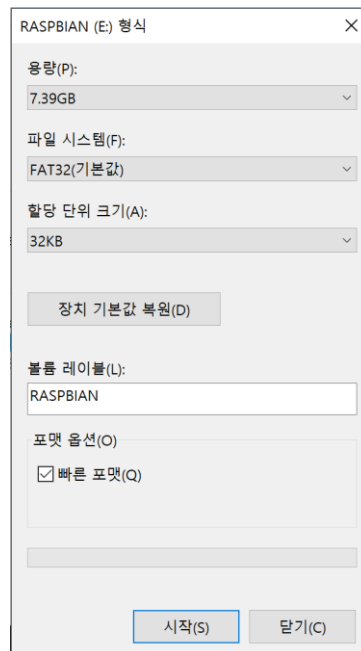
<https://www.raspberrypi.org>

위의 URL 을 통해 Raspberry pi 홈페이지에 접속하고, DOWNLOADS 탭을 클릭한다.



RASPBIAN 아이콘을 클릭하여 Raspbian OS 를 다운로드 받는다.

nCube:Thyme for Node.js 개발 가이드



RASPBIAN (E) 형식

용량(P):
7.39GB

파일 시스템(F):
FAT32(기본값)

할당 단위 크기(A):
32KB

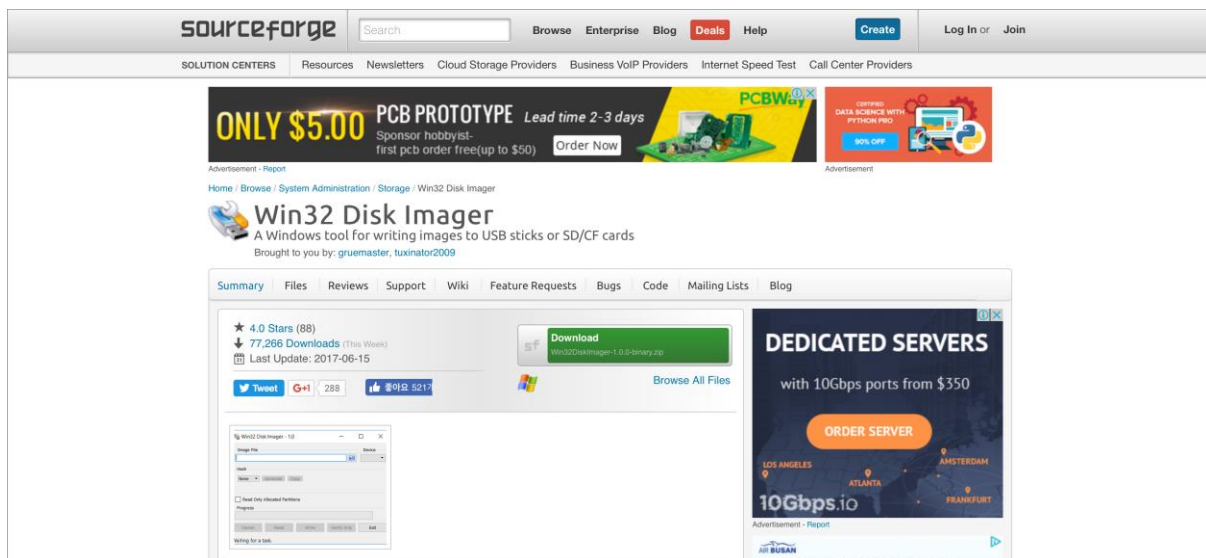
장치 기본값 복원(D)

볼륨 레이블(L):
RASPBIAN

포맷 옵션(O)
☒ 빠른 포맷(Q)

시작(S) 닫기(C)

8GB 이상의 microSD 를 준비하고, FAT32 형식으로 포맷한다.



sourceforge

ONLY \$5.00 PCB PROTOTYPE Lead time 2-3 days

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

DEDICATED SERVERS
with 10Gbps ports from \$350

ORDER SERVER

10Gbps.io

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

위의 URL 을 통해 win32 disk imager 를 다운로드 받고, 실행시킨다.

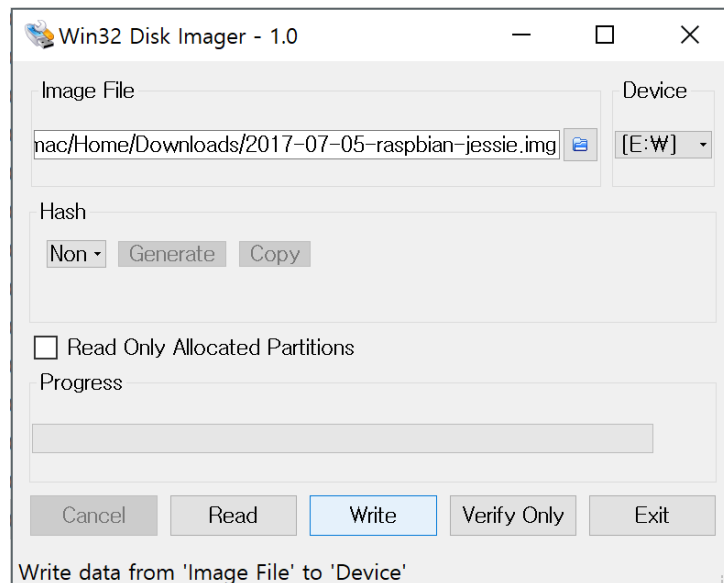


Image File 항목에 다운로드받은 Raspbian OS 이미지를 선택하고, Device 항목에 포맷한 microSD 를 선택하고, Write 버튼을 클릭하여 Raspbian OS 를 microSD 에 덮어씌운다.

작업이 완료된 후 microSD 를 Raspberry pi 3 에 삽입하고, 전원을 연결한다.

```
deory — pi@raspberrypi: ~ — ssh pi@192.168.0.98 — 80x24
[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:~ $
```

원격으로 Raspberry Pi 3 에 접근하여 작업하기 위해 터미널의 ssh 명령어 사용 또는, putty 를 설치를 통한 ssh 를 사용한다. Host 주소는 Raspberry pi 3 의 ip 주소이며, 계정은 pi, 비밀번호는 raspberry 이다.


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

node.js 와 samba 설치 이전에 repository update 와 패키지 upgrade 를 하도록 한다. 각 명령어는 위의 그림을 참조한다.

2 - 2 Node.js 설치

nCube:Thyme for Node.js 구동을 위해 Node.js 패키지를 설치한다.

```
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)
```

기존에 설치된 Node.js 패키지를 제거하고 , 새로 Node.js 패키지를 다운로드 받고, 설치하기 위해 위의 명령어를 차례대로 입력한다.

```
deory — pi@raspberrypi: ~/node — ssh pi@192.168.0.98 — 80×24
[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 $ █
```

node 디렉토리를 생성하고, 기존의 node.js 패키지를 제거하였다.

```
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 명령어를 이용하여 새로운 Node.js 패키지를 다운로드 받았다.

```
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 명령어를 이용하여 다운로드 받은 Node.js 패키지를 설치하였다.

```
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 $
```

Node.js 설치가 완료되어, 버전 확인을 통해 Node.js 설치가 완료된 것을 확인하였다.

2 - 3 Samba FTP 서버 설치

개발환경에서 Raspberry pi 3 와 파일공유를 위해 Samba FTP 서버를 설치한다.

```
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
```

apt-get 명령어를 이용하여 Samba 설치를 완료하였다.

```
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:~ $
```

smbpasswd 명령어를 이용하여 Samba 의 새 사용자를 등록하였다.

```
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
```

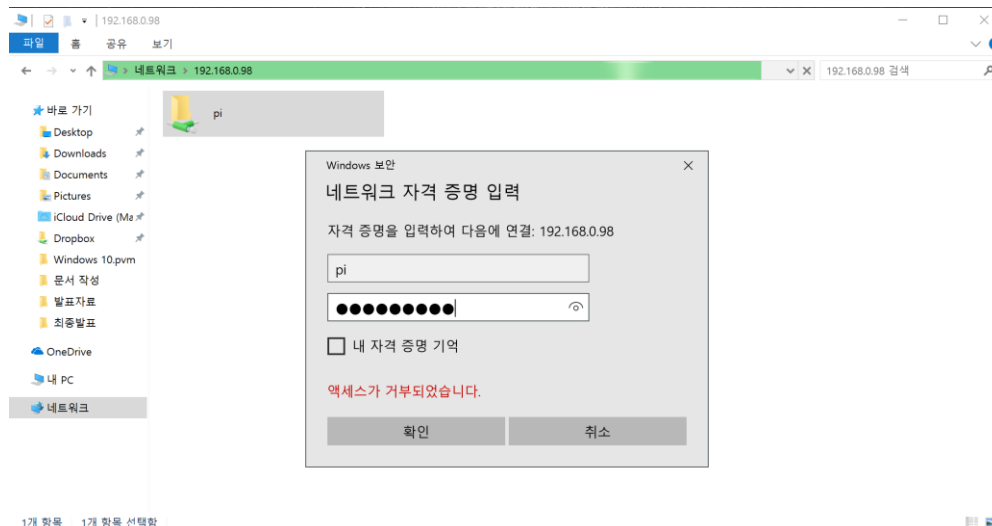
/etc/samba/smb.conf 파일을 수정하여 사용자 설정을 완료하였다.

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

```
deory — pi@raspberrypi: ~ — ssh pi@192.168.0.98 — 90x24
pi@raspberrypi:~ $ sudo service smbd restart
pi@raspberrypi:~ $
```

sudo service smbd restart 명령어를 통해 변경된 사용자 설정이 반영될 수 있도록 samba 를 재시작 하였다.

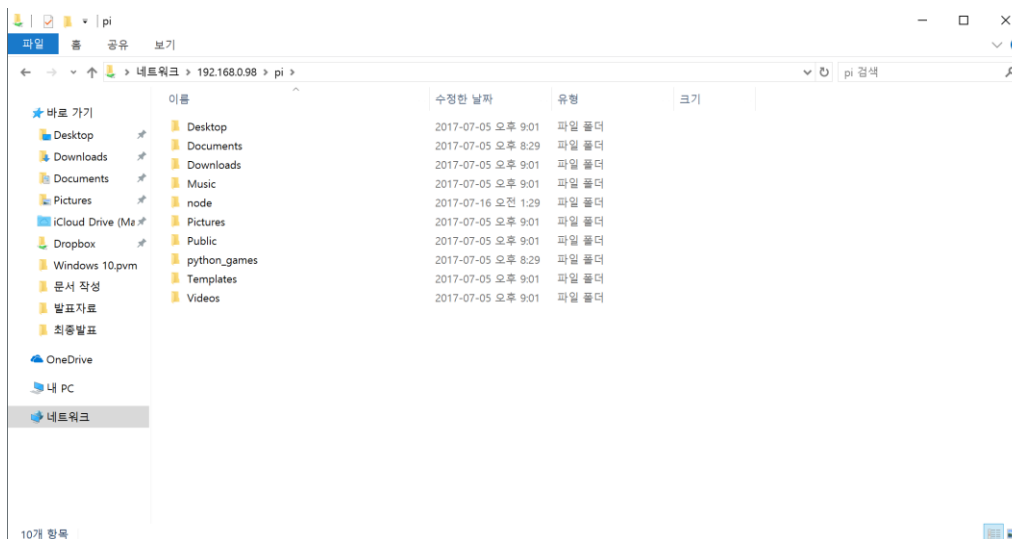
nCube:Thyme for Node.js 개발 가이드



Windows 탐색기의 주소입력 창에

\\Raspberry pi ip 주소 (ex. \\192.168.0.98)

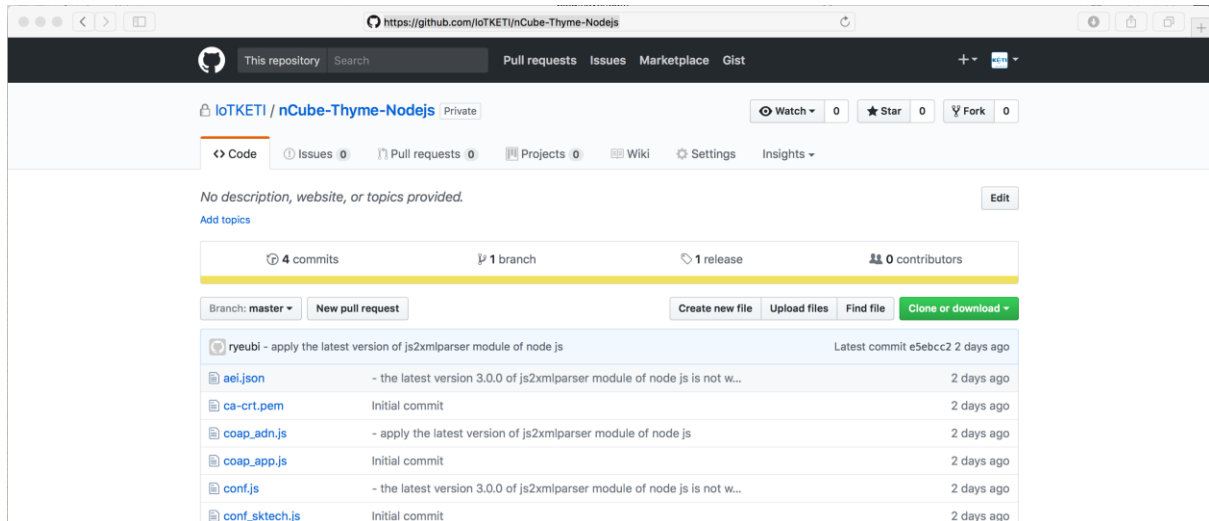
를 입력하고, 계정과 비밀번호를 각각 pi 와 samba 사용자 추가 시 입력하였던 비밀번호를 입력한다.



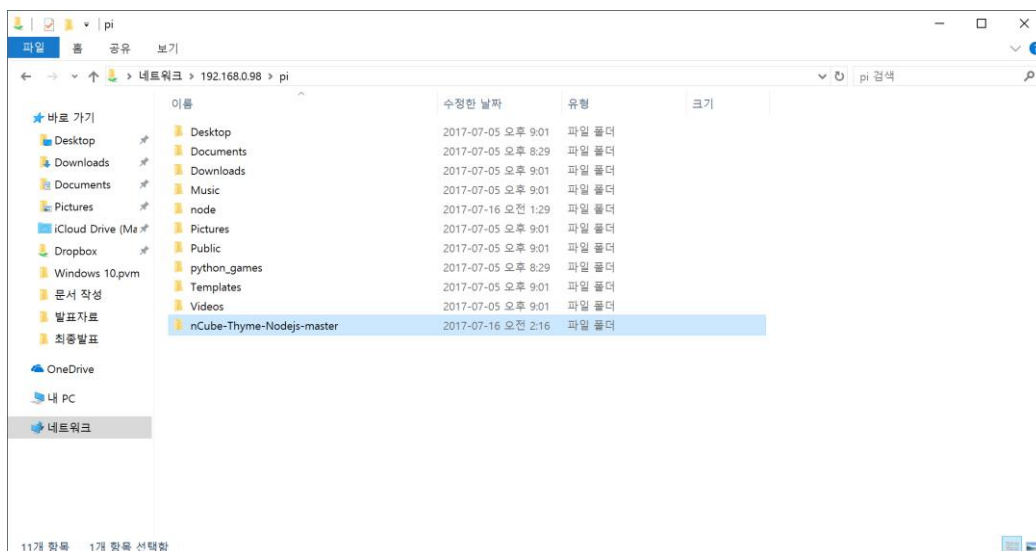
Raspberry pi 의 /home/pi 디렉토리에 접근이 가능한것을 확인할 수 있다.

2 - 4 nCube:Thyme for Node.js 다운로드

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



위의 URL 을 통해 loTKETI github 의 nCube-Thyme-Nodejs 레파지토리에 접속하고, nCube:Thyme for Node.js 를 다운로드 받는다. .



nCube:Thyme for Node.js 를 다운로드 받고, 파일탐색기를 통해 Raspberry pi 에 저장한다.
nCube:Thyme for Node.js

nCube:Thyme for Node.js 개발 가이드

```
deory — pi@raspberrypi: ~ — ssh pi@192.168.0.98 — 90x24
[pi@raspberrypi:~ $ ls
Desktop    Downloads  nCube-Thyme-Nodejs-master  Pictures  python_games  Videos
Documents  Music      node                        Public    Templates
pi@raspberrypi:~ $
```

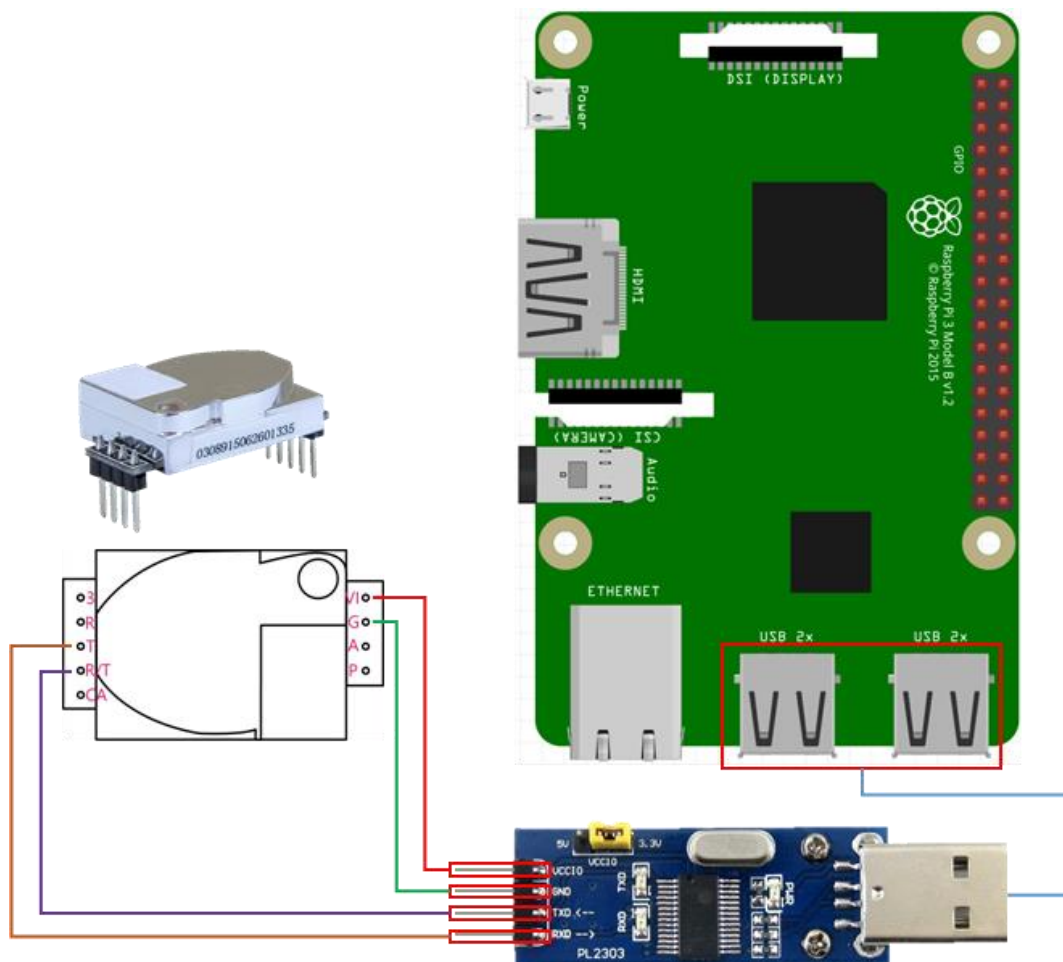
Raspberry pi 에 성공적으로 다운로드 된 것을 확인할 수 있다.

3 nCube:Thyme for Node.js 구동 준비

이번 장에서는 nCube:Thyme for Node.js 보드에 Co2 센서, RGB-LED 를 부착하여, Co2 농도를 IoT Platform 에 업로드 하고, IoT Platform 을 이용하여 nCube:Thyme for Node.js 보드에 부착된 RGB-LED 를 제어한다.

3 - 1 nCube:Thyme for Node.js + Co2 Sensor

Raspberry Pi 3 에 USB UART 컨버터를 이용하여 Co2 센서를 연결시킨다.



Raspberry Pi 3 와 Co2 센서의 연결은 위의 그림을 참조한다.

3 - 2 nCube:Thyme for Node.js + RGB-LED

Raspberry Pi 3 의 GPIO 핀 설정을 확인하여 RGB-LED 를 연결한다.

```
deory — pi@raspberrypi: ~ — ssh pi@192.168.0.98 — 90x27
pi@raspberrypi:~$ gpio readall
```

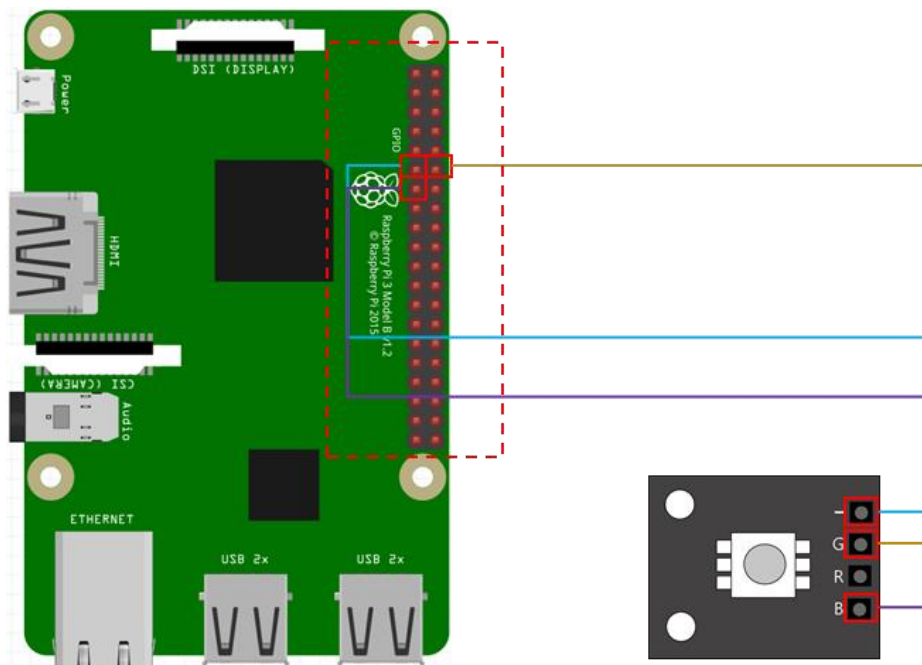
Pi 3											
BCM	wPi	Name	Mode	V	Physical	V	Mode	Name	wPi	BCM	
		3.3v			1	2		5v			
2	8	SDA.1	IN	1	3	4		5v			
3	9	SCL.1	IN	1	5	6		0v			
4	7	GPIO. 7	IN	1	7	8	0	IN	15	14	
		0v			9	10	1	IN	16	15	
17	0	GPIO. 0	IN	0	11	12	0	IN	1	18	
27	2	GPIO. 2	IN	0	13	14		0v			
22	3	GPIO. 3	IN	0	15	16	0	IN	4	23	
		3.3v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

```

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

```

gpio readall 명령어를 통해 GPIO 핀의 배열과 용도에 대해 알 수 있다.



Raspberry Pi 3 와 RGB-LED 를 위의 그림과 같이 연결한다.

3 - 3 nCube:Thyme for Node.js configure

```
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-ctrl';

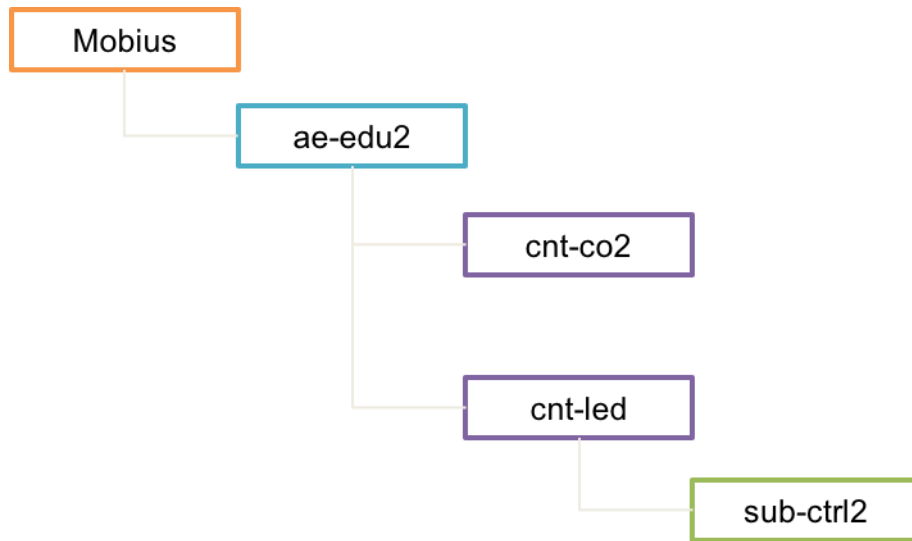
//var ip = require("ip");
//sub_arr[count++].nu = conf.userprotocol + '://' + ip.address() + ':' + ae.port + '/noti';

//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';
```

nCube:Thyme for Node.js 의 conf.js 파일의 내용은 위와 같다.



nCube:Thyme for Node.js 가 IoT Platform 에 생성할 리소스 구조는 위의 그림과 같이 나타낼 수 있다.

3 - 4 nCube:Thyme for Node.js Packages Install

```
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-sign@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 $
```

다운로드 받은 nCube:Thyme for Node.js 디렉토리 내에서 npm install 명령어를 입력하여 nCube:Thyme for Node.js 구동에 필요한 Node.js 패키지를 설치한다.

또한, tas_sample 디렉토리 아래의 tas_co2, tas_led 디렉토리 내에서도 npm install 명령을 실행하여준다.

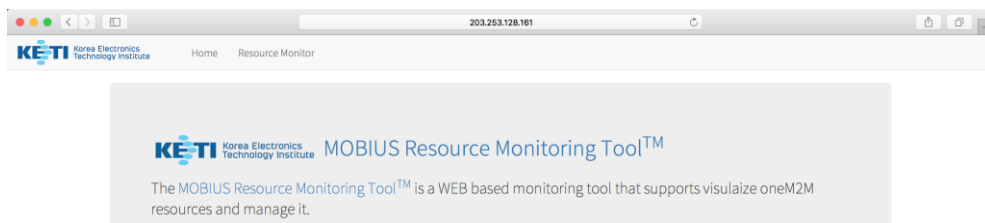
4 nCbue:Thyme for Node.js 구동 실습

이번 장에서는 nCube:Thyme for Node.js 를 직접 구동시키고, Mobius Resource Monitor 를 이용하여 업로드되는 Co2 데이터를 조회하고, Raspberry Pi 에 연결된 RGB-LED 를 제어해본다.

다운로드 받은 nCube:Thyme for Node.js 디렉토리 내에서 node thyme.js 명령어를 입력하여 nCube:Thyme for Node.js 를 구동시킨다.

새로운 터미널을 열고, nCube:Thyme for Node.js 디렉토리 아래의 tas_sample 디렉토리 아래의 tas_co2, tas_led 디렉토리 내에서 각각 node app.js 명령어를 입력하여 TAS 를 구동시킨다.

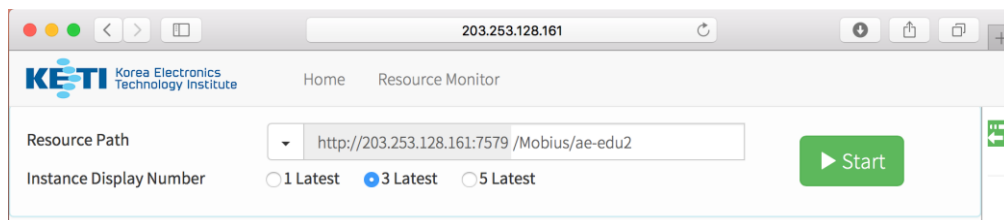
<http://203.253.128.161:7575/>



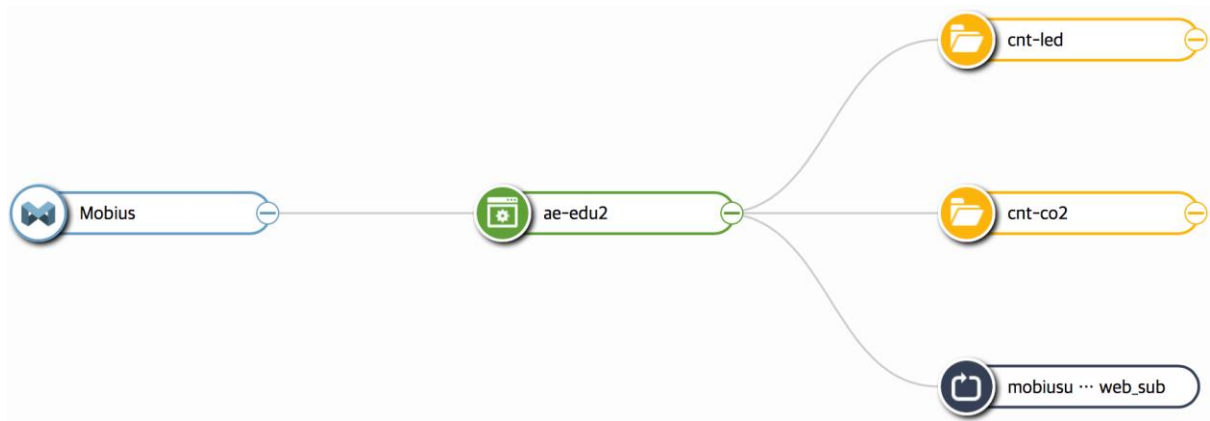
위의 URL 을 통해 웹버전의 Mobius Resource Monitor 에 접속한다.

Resource Monitor 기능을 이용하기 위해 상단의 메뉴에서 Resource Monitor 탭을 클릭하여 Resource Monitor 페이지에 접속한다.

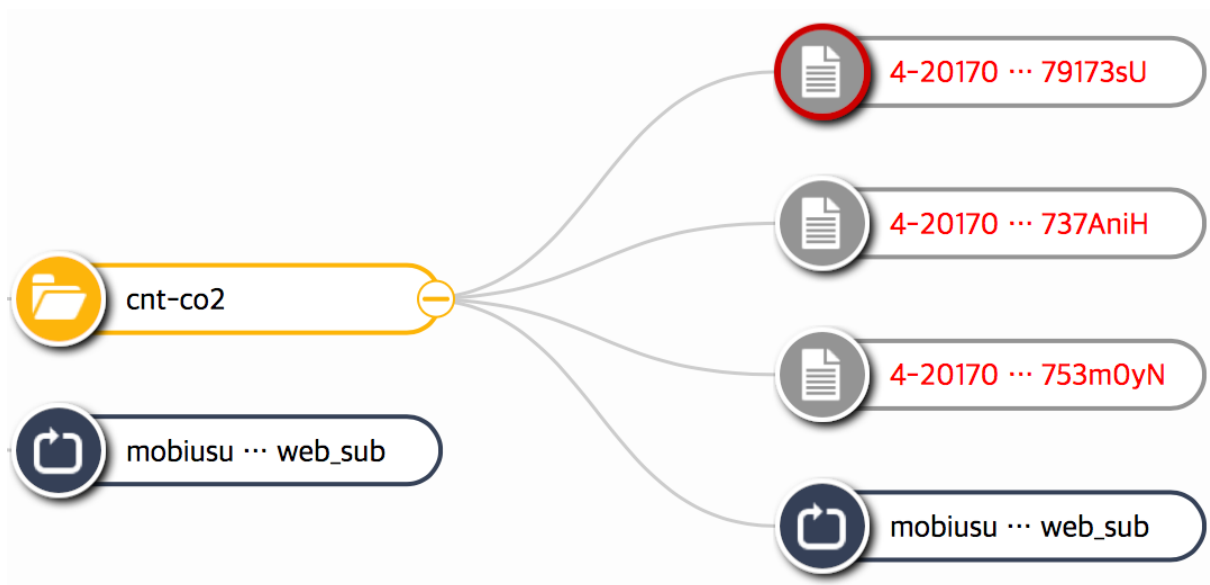
<http://203.253.128.161:7579/Mobius/ae-edu2>



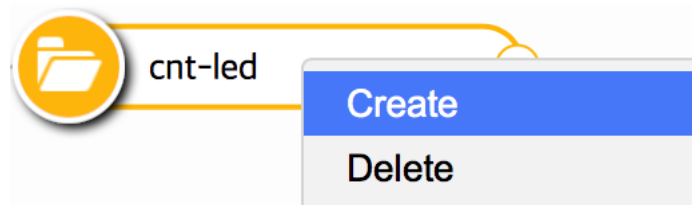
Resource Path 에 위의 주소를 넣고 Start 버튼을 클릭하면, nCube-Mint 보드가 생성한 Resource Tree 를 볼 수 있다.



조회된 리소스 구조는 위의 그림과 같다.



cnt-co2 에 측정된 co2 농도 데이터가 업로드되고 있는것을 확인할 수 있다.



cnt-led 위에서 마우스 오른쪽을 클릭하여 메뉴를 열고, Create 버튼을 클릭한다.

Create a child resource 창에서 좌측의 Select resource type 에 Content Instance 가 체크되어 있는것을 확인하고, 우측의 Fill out resource information 에 Content 에 1 을 넣고 Create 버튼을 클릭한다.

```
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!
```



tas-led 가 notification 을 수신하고, 초록색 LED 를 켜는것을 확인할 수 있다.

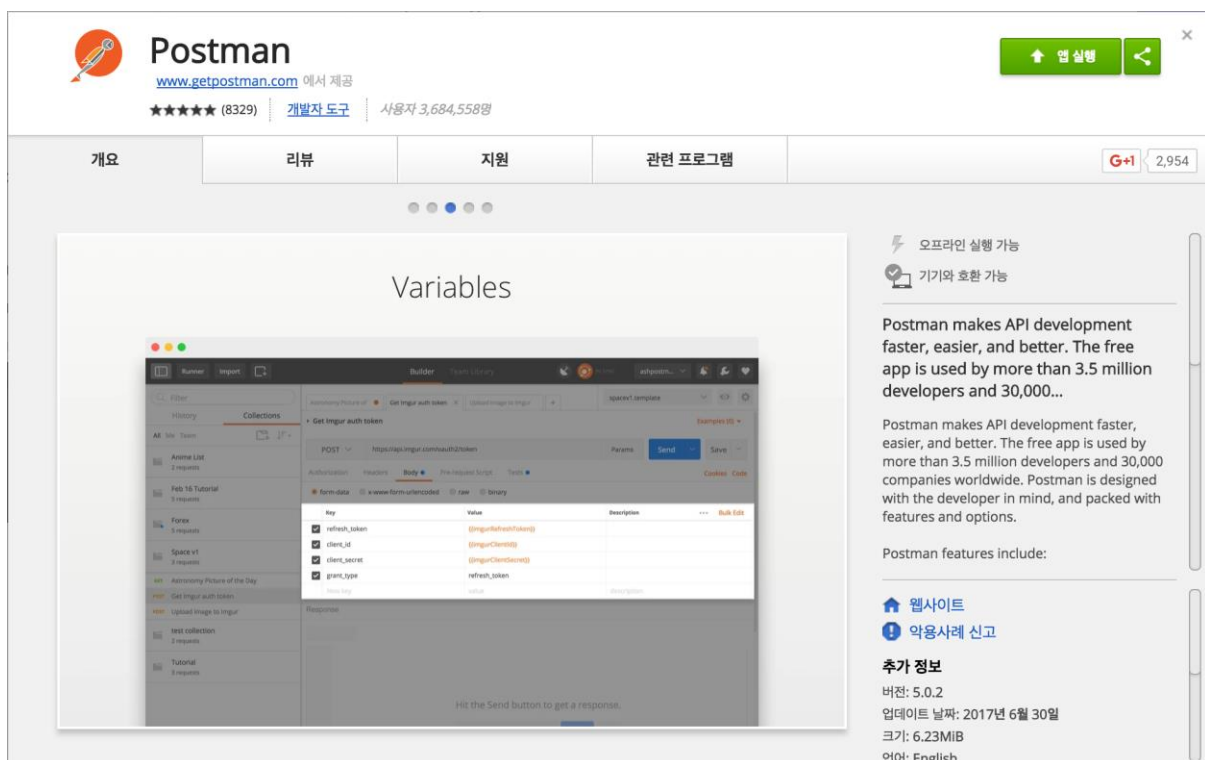
Appendix A.

Postman 을 활용한 nCube:Thyme for Node.js

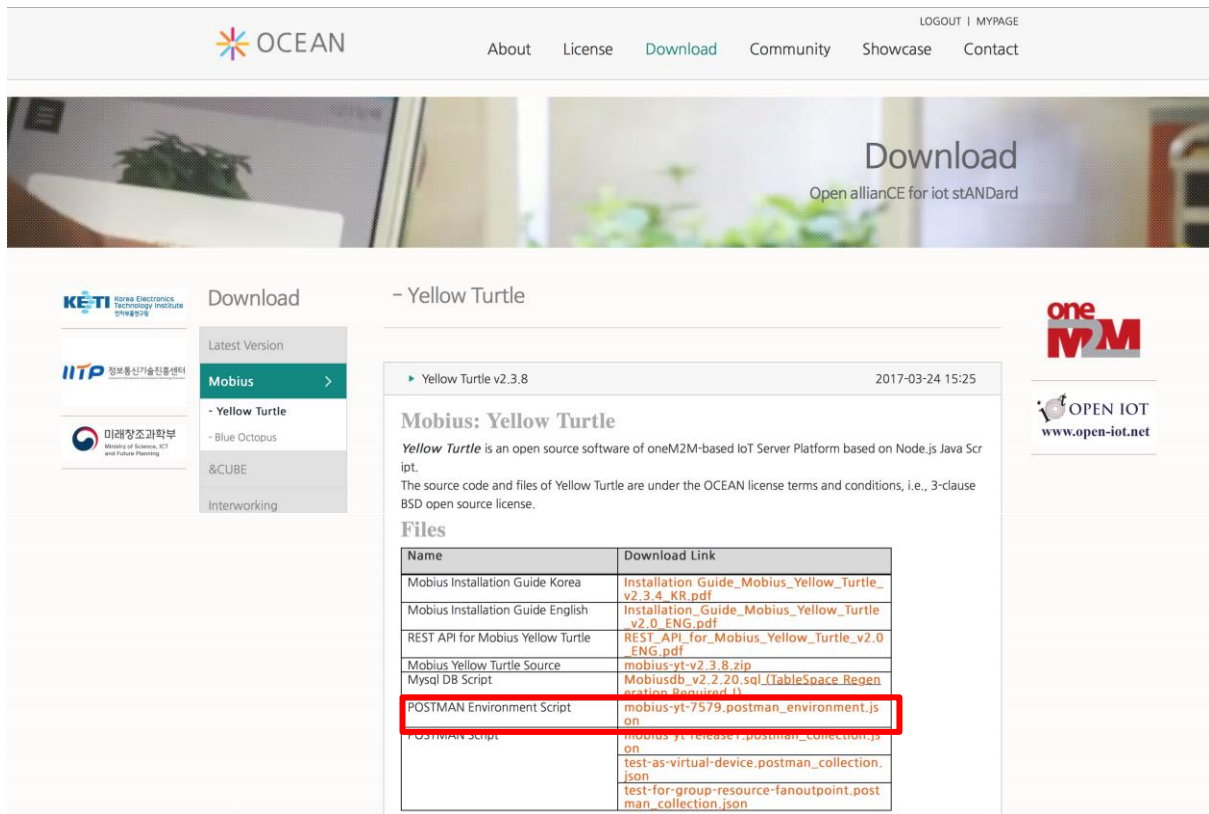
Resource 조회와 제어

Google Chrome 의 Postman 앱을 이용해서 nCube:Thyme for Node.js 가 업로드 한 Co2 데이터를 확인하고, LED 를 제어해본다. 이외에도 API 를 활용한 AE, CNT, CIN 생성 및 조회가 가능하다.

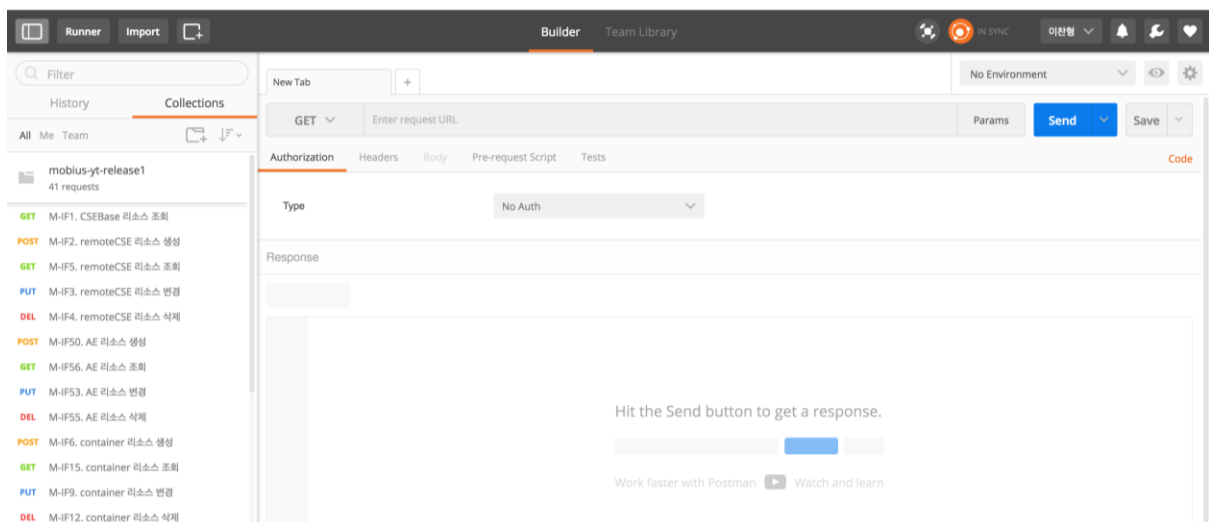
https://chrome.google.com/webstore/detail/postman/fhbjgbiflinjbdgghehcddcbnccdddomop?utm_source=chrome-ntp-icon



Google Chrome 을 이용해서 위의 URL 에 접속해 Postman 앱을 다운로드 및 설치하고, 실행시킨다.



http://www.iotocean.org 의 Download, Mobius Yellow Turtle, POSTMAN Environment Script 아래의 mobius-yt-7579.postman_environment.json 을 클릭하여 Postman API collection 을 다운로드 받는다.



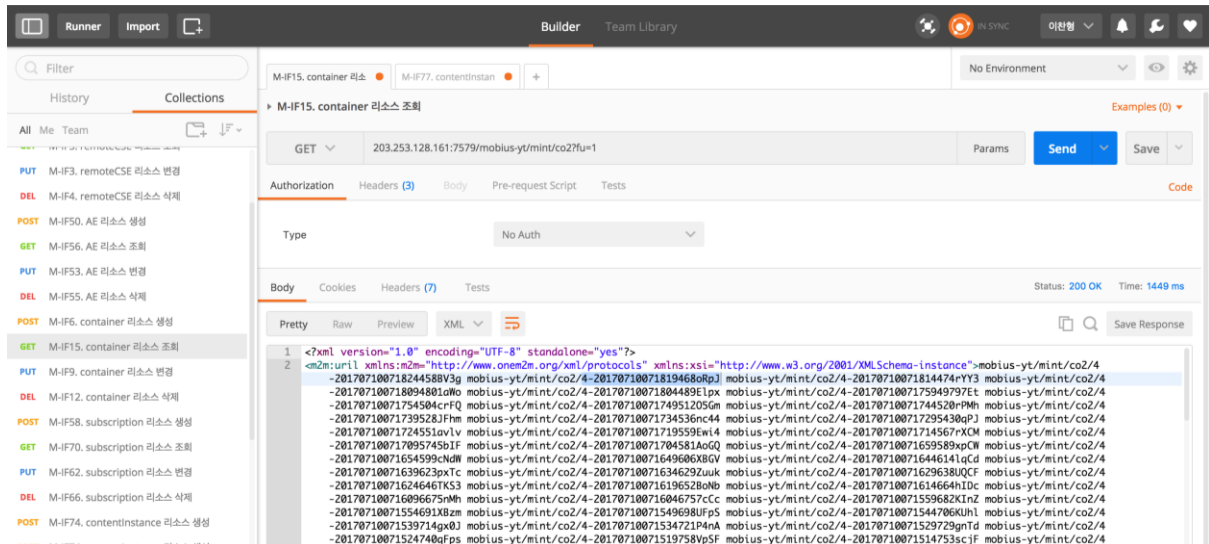
Postman 에서 import → import file → choose files 를 클릭하여 위에서 다운로드 받은 API collection 을 열면 Postman 좌측의 Collections 목록 아래에 API 들이 추가된 것을 볼 수 있다.

nCube:Thyme for Node.js 개발 가이드

추가된 API 를 이용하여 Co2 데이터를 담고 있는 cin 조회, RGB-LED 를 제어하기 위한 cin 생성을 할 수 있다.

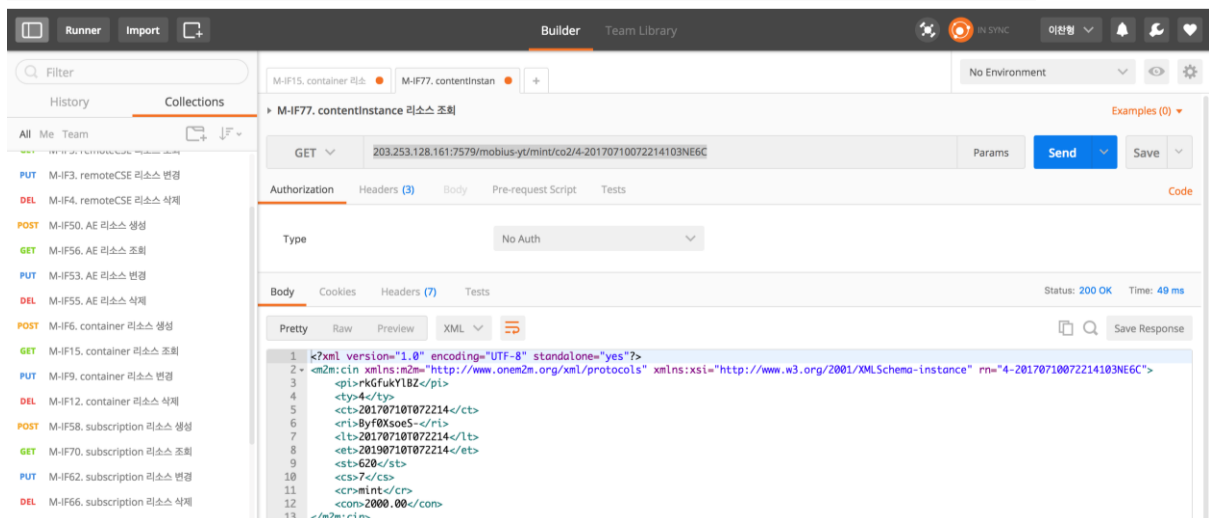
좌측 목록에서 container 리소스 조회 를 선택하고, URL 부분에 다음과 같이 입력하고, 파란색 Send 버튼을 누르면, co2 cnt 내에 생성된 cin 들의 목록을 볼 수 있다.

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



하나의 cin 이름을 복사하여두고, 좌측 목록에서 contentinstance 리소스 조회 를 선택하고, URL 부분에 다음과 같이 입력하고 Send 버튼을 누르면 cin 내의 con 필드 아래에 업로드된 Co2 농도를 확인할 수 있다.

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



RGB-LED 를 제어하기 위해 좌측 목록에서 contentinstance 리소스 생성을 선택하고, URL 부분에 각각 다음과 같이 입력하고 Body 의 con 항목 아래에 적절한 값을 입력하여 Send 버튼을 누르면 RGB-LED 제어가 가능하다.

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

