OCEAN

Interworking Proxy Entity(IPE): LWM2M

Software Version: 1.1.0

-Base on Eclipse Leshan 1.0

Develop Guide v1.0

Document Release Date: July 20, 2017

OCEAN

Contents

1. Introduction

1.1. oneM2M LWM2M IPE

oneM2M is an IoT server platform standard that supports devices as well as cloud server, while LWM2M(Light Weight Machine to Machine) is an IoT service platform that supports devices. IPE(Interworking Proxy Entity) defined in oneM2M enables LWM2M device to work with oneM2M server side platform.

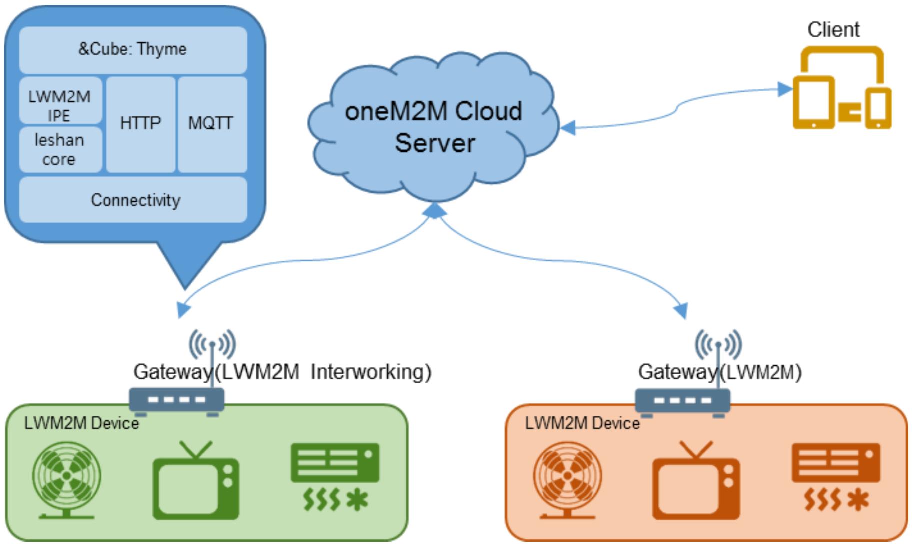


Figure 1 oneM2M LWM2M Interworking

1.2. oneM2M LWM2M IPE Environment

LWM2M IPE uses Eclipse Leshan source code as a basis which is aligned with LWM2M standard implemented in C and Java. A gateway is required to run Eclipse Leshan core and oneM2M device platform which is implemented in Java or Node.js. For instance, IPE needs an OS that can run JVM(Java Virtual Machine) or Node.js and Eclipse Leshan core that can run on the Raspberry pi.

2. IPE Sample Hardware

2.1. Introduction

In this IPE sample case, a LWM2M device by ourselves because it is difficult to find a representative commercial product. So we will use the Arduino UNO, Raspberry Pi, Humiture sensor and LED light to simulate a real LWM2M real device.

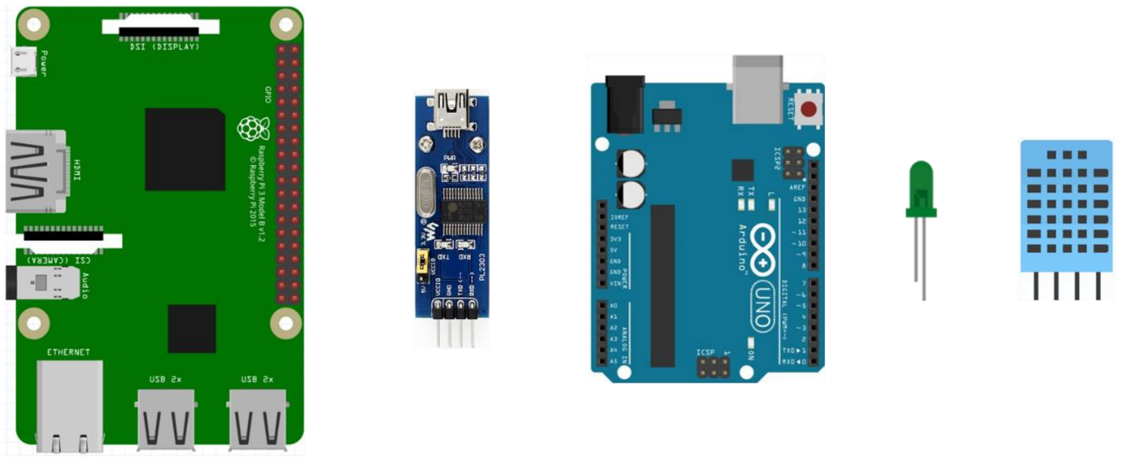


Figure 2 Hardware List

Figure 2 shows hardware devices needed. Raspberry pi runs LWM2M Client and oneM2M IPE(LWM2M Server inside), PL2303 USB UART is used to make Raspberry pi and Arduino UNO transport data with each other through serial port. Arduino UNO collects sensing data from DHT11 sensor and controls LED light ON and OFF.

2.2. Hardware Connectivity

It is easy to connect the each hardware with 3 steps below. But before it you need prepare a 1kΩ resistance for pull up data pin of DHT11 Sensor and 220Ω resistance for protecting LED light.

1. Connect LED light, DHT11 Sensor to Arduino UNO with pins.
2. Connect Arduino UNO to PL2303 USB UART with pins.
3. Connect PL2303 USB UART to Raspberry pi 3 with USB port.

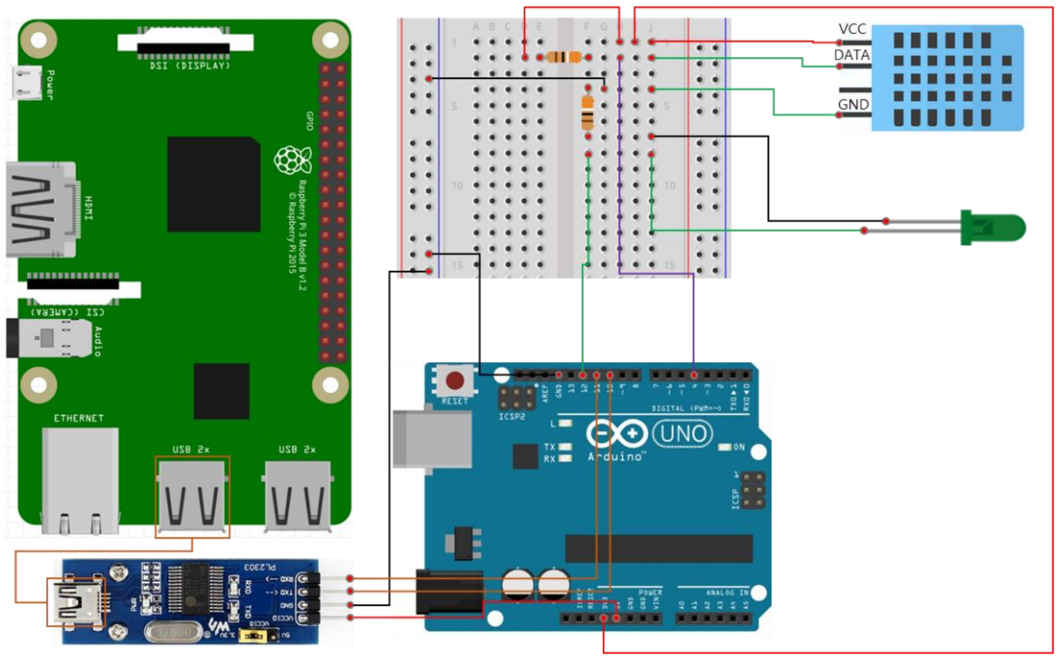


Figure 3 Hardware Connectivity

Figure 3 shows how to connect this hardware together.

3. IPE Sample Software

3.1. Download

1. Download the LWM2M IPE Sample source from IoT Ocean home page.

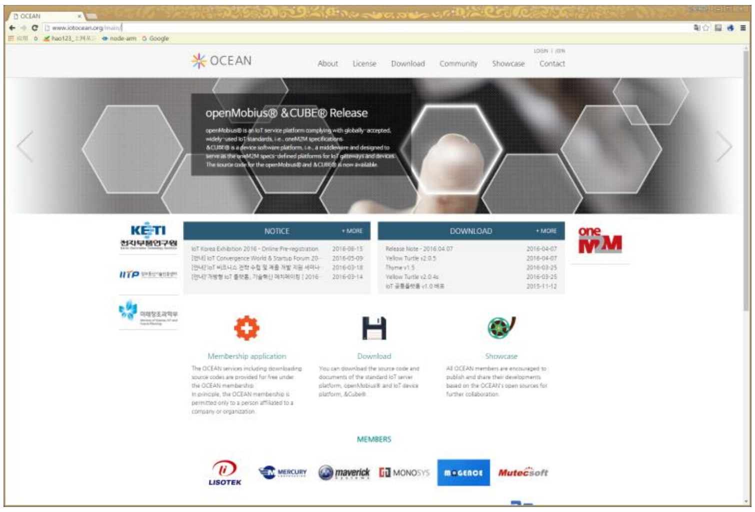
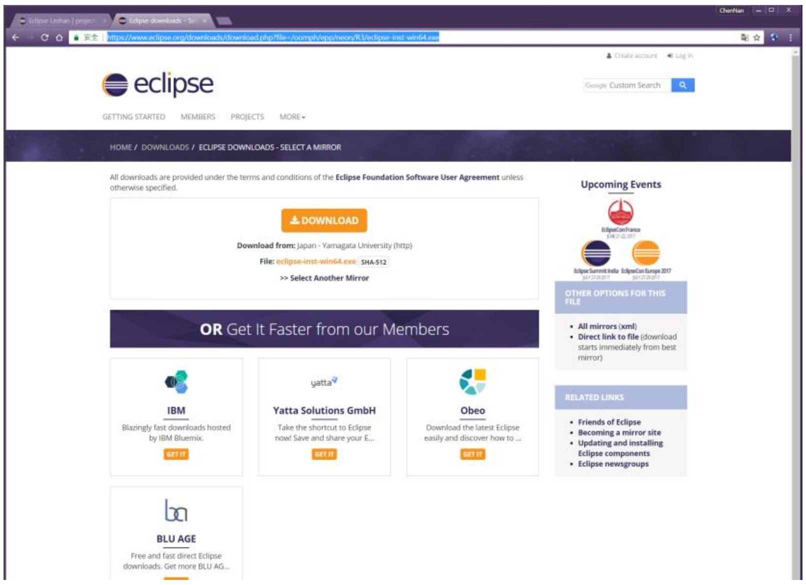


Figure 4 IoT Ocean home page

2. Download the Eclipse Neon from Eclipse home page.


https://www.eclipse.org/downloads/download.php?file=/oomph/epp/neon/R3/eclipse-inst-win64.exe

HOME / DOWNLOADS / ECLIPSE DOWNLOADS - SELECT A MIRROR

All downloads are provided under the terms and conditions of the **Eclipse Foundation Software User Agreement** unless otherwise specified.

Download from: Japan - Yamagata University (http)

File: eclipse-inst-win64.exe SHA-512

>> Select Another Mirror

OR Get It Faster from our Members

IBM Blazingly fast downloads hosted by IBM Bluemix.

Yatta Solutions GmbH Take the shortcut to Eclipse now! Save and share your E...

Obeo Download the latest Eclipse easily and discover how to ...

BLU AGE Free and fast direct Eclipse downloads. Get more BLU AG...

OTHER OPTIONS FOR THIS FILE

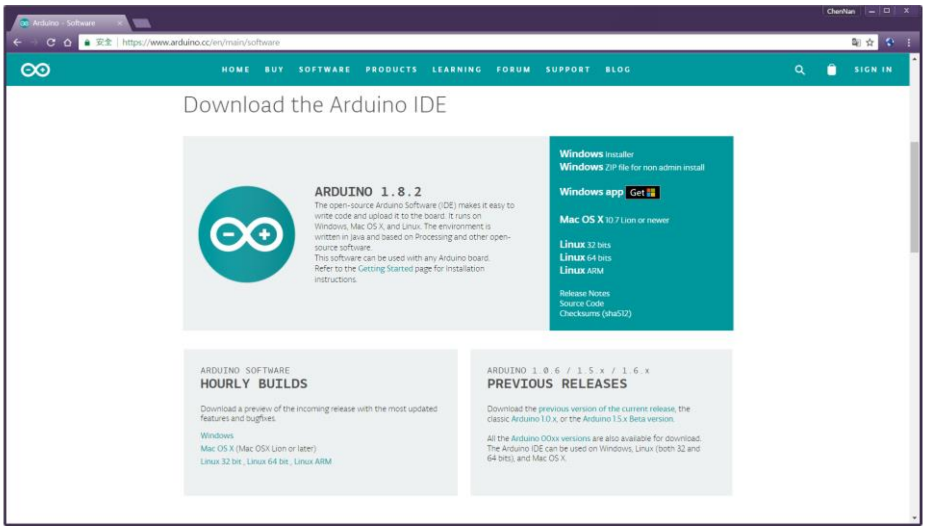
- **All mirrors (xml)**
- **Direct link to file** (download starts immediately from best mirror)

RELATED LINKS

- Friends of Eclipse
- Becoming a mirror site
- Updating and installing Eclipse components
- Eclipse newsgroups

Figure 5 Eclipse home page

3. Download Arduino IDE from Arduino IDE home page.


https://www.arduino.cc/en/main/software

Download the Arduino IDE

ARDUINO 1.8.2

The open-source Arduino Software (IDE) makes it easy to write code and upload it to the board. It runs on Windows, Mac OS X, and Linux. The environment is written in Java and based on Processing and other open-source software.
This software can be used with any Arduino board.
Refer to the Getting Started page for installation instructions.

Windows Installer
Windows ZIP file for non admin install
Windows app Get
Mac OS X 10.7 Lion or newer
Linux 32 bits
Linux 64 bits
Linux ARM
Release Notes
Source Code
Checksums (sha512)

ARDUINO SOFTWARE
HOURLY BUILDS

Download a preview of the incoming release with the most updated features and bugfixes.
Windows
Mac OS X (Mac OSX Lion or later)
Linux 32 bit , Linux 64 bit , Linux ARM

ARDUINO 1.0.6 / 1.5.x / 1.6.x
PREVIOUS RELEASES

Download the previous version of the current release, the classic Arduino 1.0.x, or the Arduino 1.5.x Beta version.
All the Arduino 00xx versions are also available for download. The Arduino IDE can be used on Windows, Linux (both 32 and 64 bits), and Mac OS X.

Figure 6 Arduino IDE home page

4. Install Eclipse, Arduino IDE on windows by double click the installer.

3.2. Import project

1. Import keti_lwm2m_device and keti_lwm2m_ipe project in Eclipse, which can be downloaded from IoT Ocean homepage.

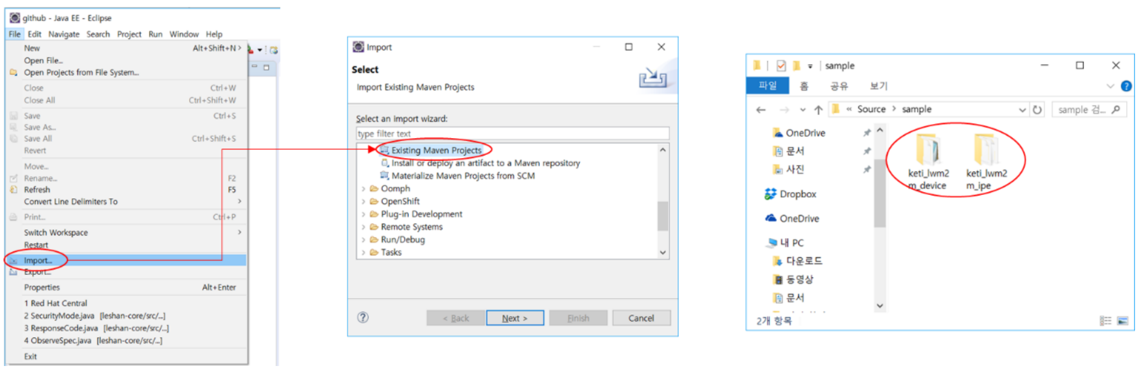
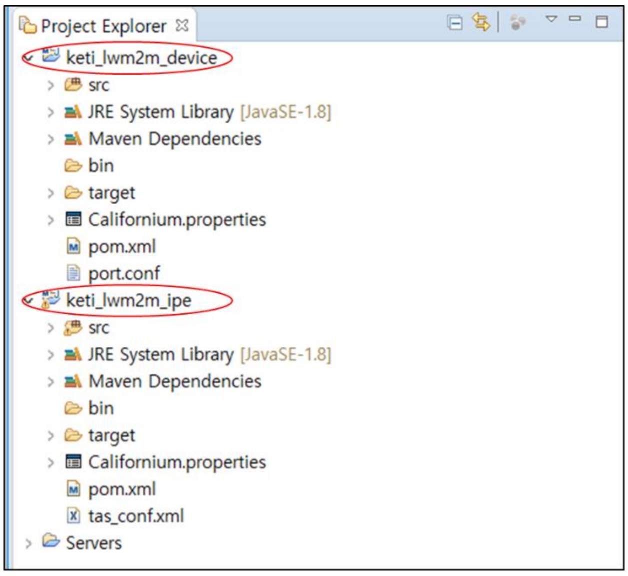


Figure 7 Import maven project

If you are first time to import these maven project maybe it will cost a few minutes for downloading depended libraries. When import finish you can find the keti_lwm2m_device and keti_lwm2m_ipe project in the Project Explorer of Eclipse.

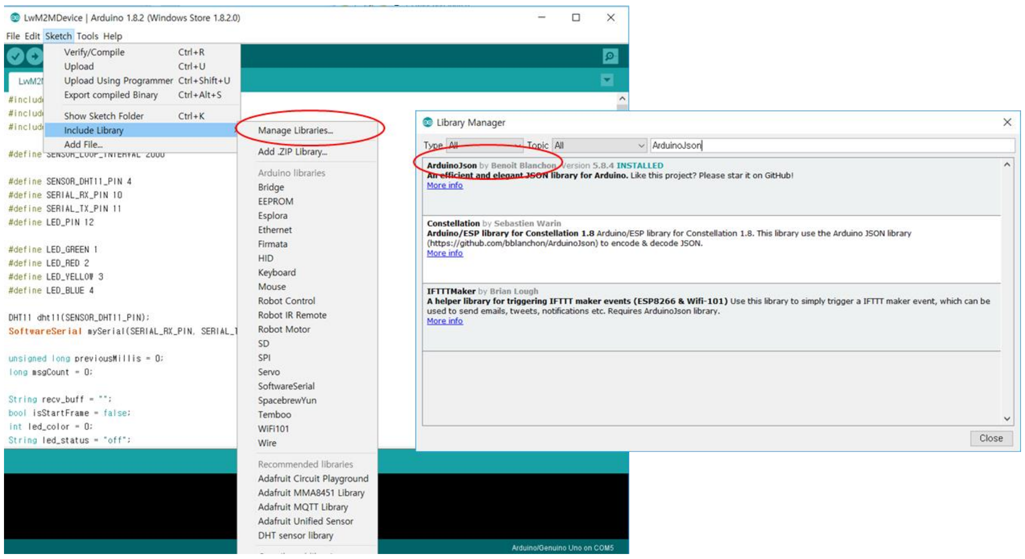


```
Project Explorer
keti_lwm2m_device
    src
    JRE System Library [JavaSE-1.8]
    Maven Dependencies
    bin
    target
    Californium.properties
    pom.xml
    port.conf
keti_lwm2m_ipe
    src
    JRE System Library [JavaSE-1.8]
    Maven Dependencies
    bin
    target
    Californium.properties
    pom.xml
    tas_conf.xml
Servers
```

Figure 8 Eclipse Project Explorer

2. Open "LwM2MDevice.ino" source file using Arduino IDE.

3. Install "ArduinoJson" library by "Library Manager" from Arduino IDE menu.



```
#define SENSOR_DHT11_PIN 4
#define SERIAL_RX_PIN 10
#define SERIAL_TX_PIN 11
#define LED_PIN 12

#define LED_GREEN 1
#define LED_RED 2
#define LED_YELLOW 3
#define LED_BLUE 4

DHT11 dht11(SENSOR_DHT11_PIN);
SoftwareSerial mySerial(SERIAL_RX_PIN, SERIAL_T

unsigned long previousMillis = 0;
long msgCount = 0;

String recv_buff = "";
bool isStartFrame = false;
int led_color = 0;
String led_status = "off";
```

Type All Topic All ArduinoJson

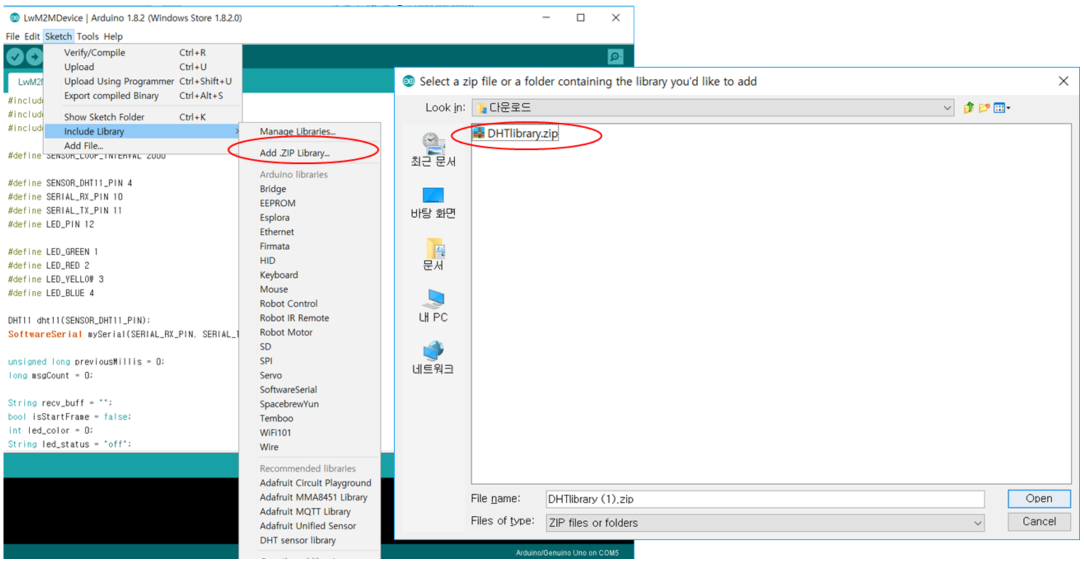
ArduinoJson by Benoit Blanchon Version 5.8.4 INSTALLED
An efficient and elegant JSON library for Arduino. Like this project? Please star it on GitHub!
More info

Constellation by Sebastien Warin
Arduino/ESP library for Constellation 1.8 Arduino/ESP library for Constellation 1.8. This library use the Arduino JSON library (https://github.com/bblanchon/ArduinoJson) to encode & decode JSON.
More info

IFTTTMaker by Brian Lough
A helper library for triggering IFTTT maker events (ESP8266 & Wifi-101) Use this library to simply trigger a IFTTT maker event, which can be used to send emails, tweets, notifications etc. Requires ArduinoJson library.
More info

Figure 9 Install JSON library

4. Download DHT11 library from [here]() and add to Arduino IDE project.



```
#define SENSOR_DHT11_PIN 4
#define SERIAL_RX_PIN 10
#define SERIAL_TX_PIN 11
#define LED_PIN 12

#define LED_GREEN 1
#define LED_RED 2
#define LED_YELLOW 3
#define LED_BLUE 4

DHT11 dht11(SENSOR_DHT11_PIN);
SoftwareSerial mySerial(SERIAL_RX_PIN, SERIAL_T

unsigned long previousMillis = 0;
long msgCount = 0;

String recv_buff = "";
bool isStartFrame = false;
int led_color = 0;
String led_status = "off";
```

Select a zip file or a folder containing the library you'd like to add

Look in: 다운로드

DHTlibrary.zip

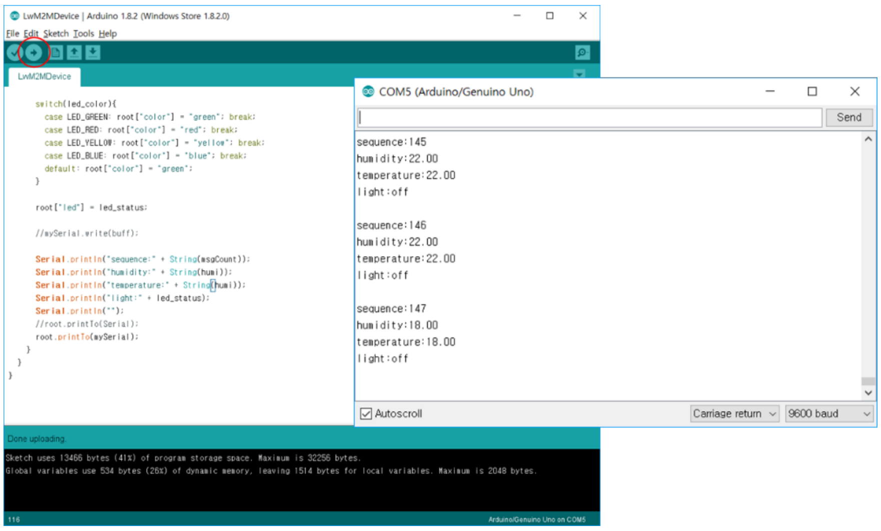
File name: DHTlibrary (1).zip

Files of type: ZIP files or folders

Figure 10 Add DHT11 library

3.3. Execution

1. Connect Arduino UNO to PC with debug USB cable then compile and upload source.


LwM2MDevice | Arduino 1.8.2 (Windows Store 1.8.2.0)
File Edit Sketch Tools Help
LwM2MDevice
COM5 (Arduino/Genuino Uno)
Send
Autoscroll
Carriage return
9600 baud
116
Arduino/Genuino Uno on COM5

```
    switch(led_color){
      case LED_GREEN: root["color"] = "green"; break;
      case LED_RED: root["color"] = "red"; break;
      case LED_YELLOW: root["color"] = "yellow"; break;
      case LED_BLUE: root["color"] = "blue"; break;
      default: root["color"] = "green";
    }

    root["led"] = led_status;

    //mySerial.write(buff);

    Serial.println("sequence:" + String(msgCount));
    Serial.println("humidity:" + String(humi));
    Serial.println("temperature:" + String(humi));
    Serial.println("light:" + led_status);
    Serial.println("");
    //root.printTo(Serial);
    root.printTo(mySerial);
  }
 }
}
```

```
sequence:145
humidity:22.00
temperature:22.00
light:off

sequence:146
humidity:22.00
temperature:22.00
light:off

sequence:147
humidity:18.00
temperature:18.00
light:off
```

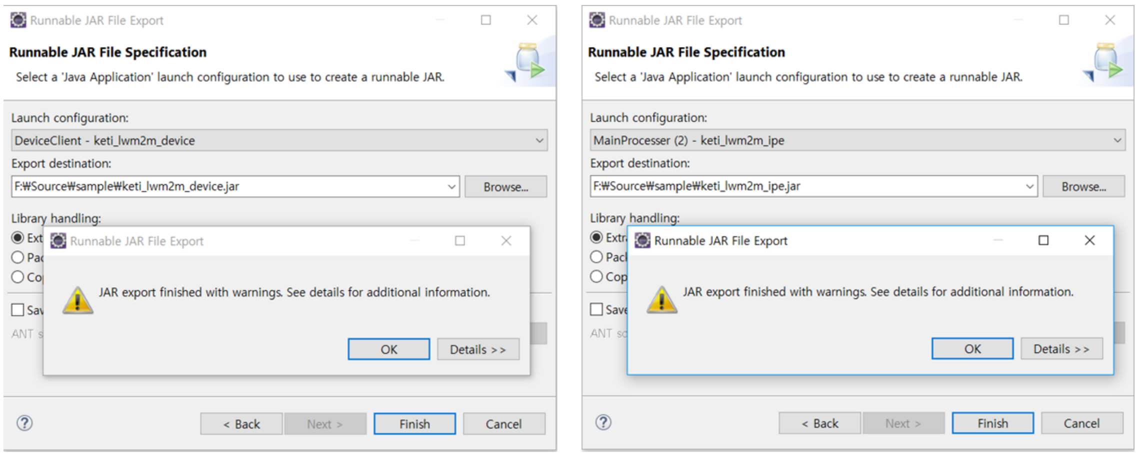
```
Done uploading.
Sketch uses 13466 bytes (41%) of program storage space. Maximum is 32256 bytes.
Global variables use 534 bytes (26%) of dynamic memory, leaving 1514 bytes for local variables. Maximum is 2048 bytes.
```

Figure 11 Compile and upload source

If sensing data is displayed in serial monitor, the device is properly functioning.

2. Export "keti_lwm2m_device" as runnable jar file named "<name>.jar" then copy "<name>.jar" file and "port.conf"(In project) file to raspberry pi.

3. Export "keti_lwm2m_ipe" as runnable jar file named "<name.jar>" then copy "<name>.jar" file and "tas_conf.xml"(In project) file to raspberry pi.



Runnable JAR File Specification

Select a 'Java Application' launch configuration to use to create a runnable JAR.

Launch configuration: DeviceClient - keti_lwm2m_device

Export destination: F:₩Source₩sample₩keti_lwm2m_device.jar

JAR export finished with warnings. See details for additional information.

Runnable JAR File Specification

Select a 'Java Application' launch configuration to use to create a runnable JAR.

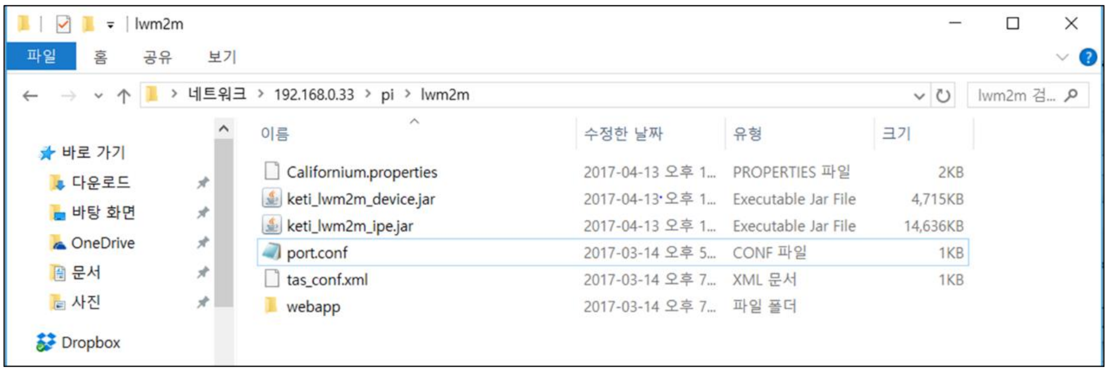
Launch configuration: MainProcesser (2) - keti_lwm2m_ipe

Export destination: F:₩Source₩sample₩keti_lwm2m_ipe.jar

JAR export finished with warnings. See details for additional information.

Figure 12 Export execution jar file

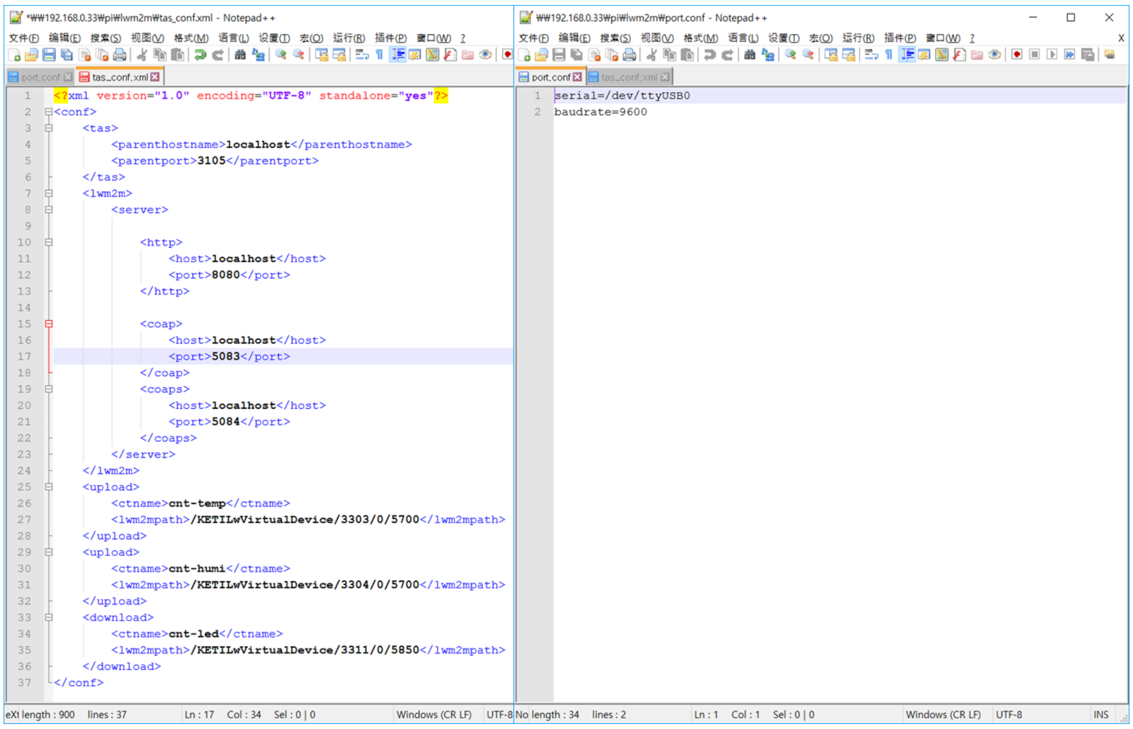
4. Copy webapp folder to raspberry pi jar file path from "keti_lwm2m_device" project.



이름	수정한 날짜	유형	크기
Californium.properties	2017-04-13 오후 1...	PROPERTIES 파일	2KB
keti_lwm2m_device.jar	2017-04-13 오후 1...	Executable Jar File	4,715KB
keti_lwm2m_ipe.jar	2017-04-13 오후 1...	Executable Jar File	14,636KB
port.conf	2017-03-14 오후 5...	CONF 파일	1KB
tas_conf.xml	2017-03-14 오후 7...	XML 문서	1KB
webapp	2017-03-14 오후 7...	파일 폴더	

Figure 13 Copy to Raspberry pi

5. Configure tas_conf.xml and port.conf.


*₩₩192.168.0.33₩pi₩lwm2m₩tas_conf.xml - Notepad++
文件(F) 编辑(E) 搜索(S) 视图(V) 格式(M) 语言(L) 设置(T) 宏(O) 运行(R) 插件(P) 窗口(W) ?
port.conf
tas_conf.xml
eXt length : 900 lines : 37 Ln : 17 Col : 34 Sel : 0 | 0 Windows (CR LF) UTF-8
₩₩192.168.0.33₩pi₩lwm2m₩port.conf - Notepad++
No length : 34 lines : 2 Ln : 1 Col : 1 Sel : 0 | 0 Windows (CR LF) UTF-8 INS

```xml
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<conf>
    <tas>
        <parenthostname>localhost</parenthostname>
        <parentport>3105</parentport>
    </tas>
    <lwm2m>
        <server>

            <http>
                <host>localhost</host>
                <port>8080</port>
            </http>

            <coap>
                <host>localhost</host>
                <port>5083</port>
            </coap>
            <coaps>
                <host>localhost</host>
                <port>5084</port>
            </coaps>
        </server>
    </lwm2m>
    <upload>
        <ctname>cnt-temp</ctname>
        <lwm2mpath>/KETILwVirtualDevice/3303/0/5700</lwm2mpath>
    </upload>
    <upload>
        <ctname>cnt-humi</ctname>
        <lwm2mpath>/KETILwVirtualDevice/3304/0/5700</lwm2mpath>
    </upload>
    <download>
        <ctname>cnt-led</ctname>
        <lwm2mpath>/KETILwVirtualDevice/3311/0/5850</lwm2mpath>
    </download>
</conf>
```

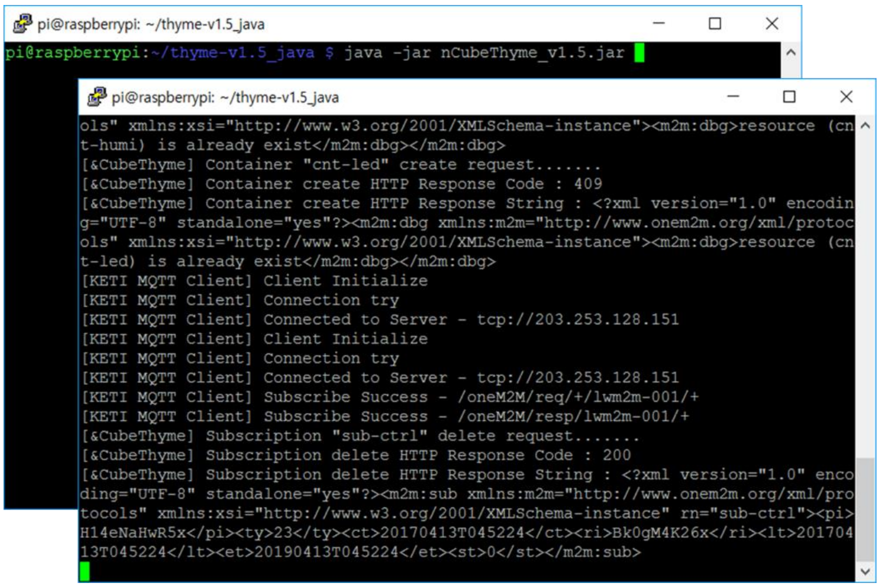
```
serial=/dev/ttyUSB0
baudrate=9600
```

Figure 14 Configure the tas_conf.xml and port.conf file

In tas_conf.xml file, "<tas/parenthostname>" means &Cube(Refer to IoT Ocean Home page) Thyme's IP address. In this case &Cube and IPE will work on the same device. "<lwm2m/server/http>" configure LWM2M Web Server hosting setting. "<lwm2m/server/coap>" and "<lwm2m/server/coaps>" is used to transport with LWM2M device (Do not need to change). "<tas/upload>" is definingoneM2M container resource for collecting sensing data. "<tas/upload>" is defining oneM2M container resource for receive notification (for controlling).

In port.conf file, set serial communication for Arduino board. It contains serial port name and baud rate.

6. Run java thyme 1.5(Refer to Thyme guide document from IoT Ocean Home page).



pi@raspberrypi: ~/thyme-v1.5_java

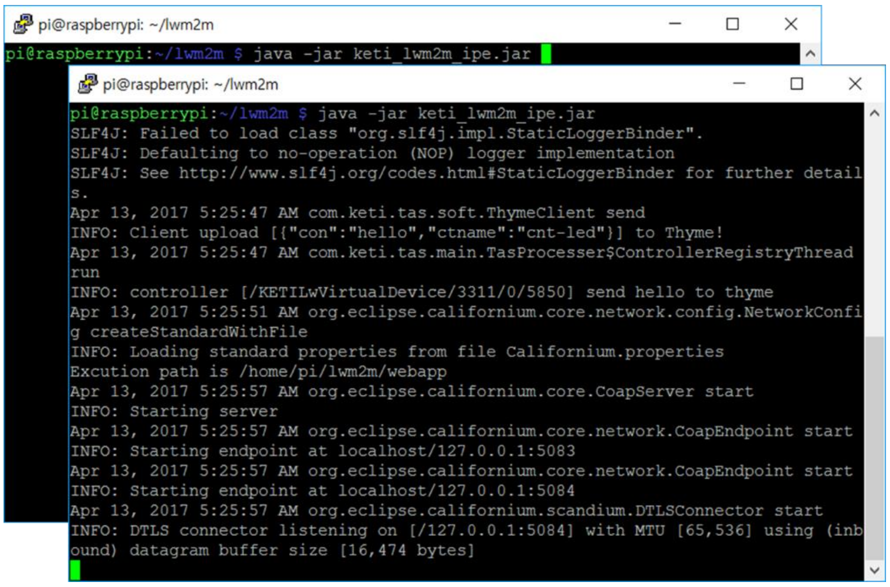
```
pi@raspberrypi:~/thyme-v1.5_java $ java -jar nCubeThyme_v1.5.jar
```

pi@raspberrypi: ~/thyme-v1.5_java

```
ols" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"><m2m:dbg>resource (cn
t-humi) is already exist</m2m:dbg></m2m:dbg>
[&CubeThyme] Container "cnt-led" create request.......
[&CubeThyme] Container create HTTP Response Code : 409
[&CubeThyme] Container create HTTP Response String : <?xml version="1.0" encodin
g="UTF-8" standalone="yes"?><m2m:dbg xmlns:m2m="http://www.onem2m.org/xml/protoc
ols" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"><m2m:dbg>resource (cn
t-led) is already exist</m2m:dbg></m2m:dbg>
[KETI MQTT Client] Client Initialize
[KETI MQTT Client] Connection try
[KETI MQTT Client] Connected to Server - tcp://203.253.128.151
[KETI MQTT Client] Client Initialize
[KETI MQTT Client] Connection try
[KETI MQTT Client] Connected to Server - tcp://203.253.128.151
[KETI MQTT Client] Subscribe Success - /oneM2M/req/+/lwm2m-001/+
[KETI MQTT Client] Subscribe Success - /oneM2M/resp/lwm2m-001/+
[&CubeThyme] Subscription "sub-ctrl" delete request.......
[&CubeThyme] Subscription delete HTTP Response Code : 200
[&CubeThyme] Subscription delete HTTP Response String : <?xml version="1.0" enco
ding="UTF-8" standalone="yes"?><m2m:sub xmlns:m2m="http://www.onem2m.org/xml/pro
tocols" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" rn="sub-ctrl"><pi>
H14eNaHwR5x</pi><ty>23</ty><ct>20170413T045224</ct><ri>Bk0gM4K26x</ri><lt>201704
13T045224</lt><et>20190413T045224</et><st>0</st></m2m:sub>
```

Figure 15 Thyme 1.5 execution

7. Run LWM2M IPE.



pi@raspberrypi: ~/lwm2m

```
pi@raspberrypi:~/lwm2m $ java -jar keti_lwm2m_ipe.jar
```

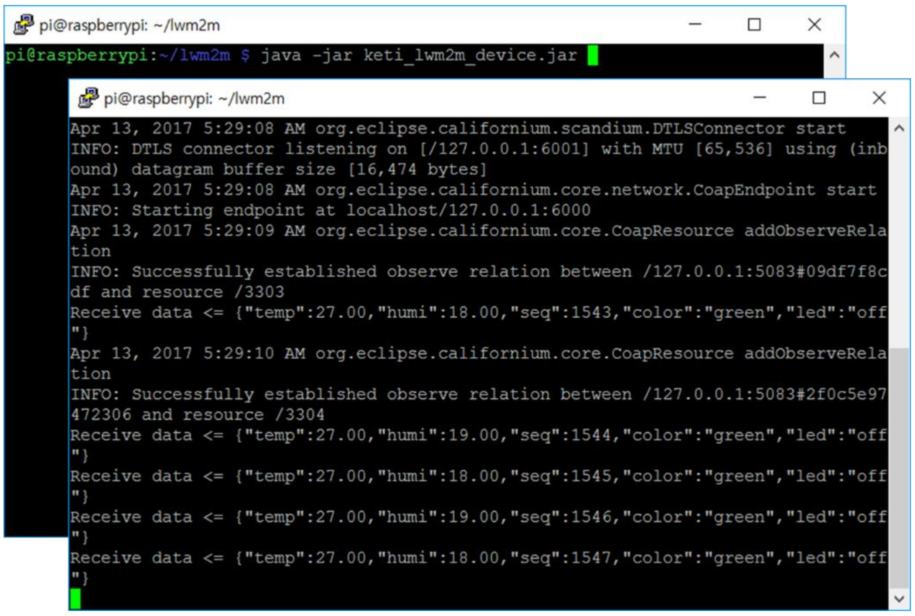
pi@raspberrypi: ~/lwm2m

```
pi@raspberrypi:~/lwm2m $ java -jar keti_lwm2m_ipe.jar
SLF4J: Failed to load class "org.slf4j.impl.StaticLoggerBinder".
SLF4J: Defaulting to no-operation (NOP) logger implementation
SLF4J: See http://www.slf4j.org/codes.html#StaticLoggerBinder for further detail
s.
Apr 13, 2017 5:25:47 AM com.keti.tas.soft.ThymeClient send
INFO: Client upload [{"con":"hello","ctname":"cnt-led"}] to Thyme!
Apr 13, 2017 5:25:47 AM com.keti.tas.main.TasProcesser$ControllerRegistryThread
run
INFO: controller [/KETILwVirtualDevice/3311/0/5850] send hello to thyme
Apr 13, 2017 5:25:51 AM org.eclipse.californium.core.network.config.NetworkConfi
g createStandardWithFile
INFO: Loading standard properties from file Californium.properties
Excution path is /home/pi/lwm2m/webapp
Apr 13, 2017 5:25:57 AM org.eclipse.californium.core.CoapServer start
INFO: Starting server
Apr 13, 2017 5:25:57 AM org.eclipse.californium.core.network.CoapEndpoint start
INFO: Starting endpoint at localhost/127.0.0.1:5083
Apr 13, 2017 5:25:57 AM org.eclipse.californium.core.network.CoapEndpoint start
INFO: Starting endpoint at localhost/127.0.0.1:5084
Apr 13, 2017 5:25:57 AM org.eclipse.californium.scandium.DTLSConnector start
INFO: DTLS connector listening on [/127.0.0.1:5084] with MTU [65,536] using (inb
ound) datagram buffer size [16,474 bytes]
```

Figure 16 LWM2M IPE execution

:

8. Run LWM2M device client.



```
pi@raspberrypi:~/lwm2m $ java -jar keti_lwm2m_device.jar
```

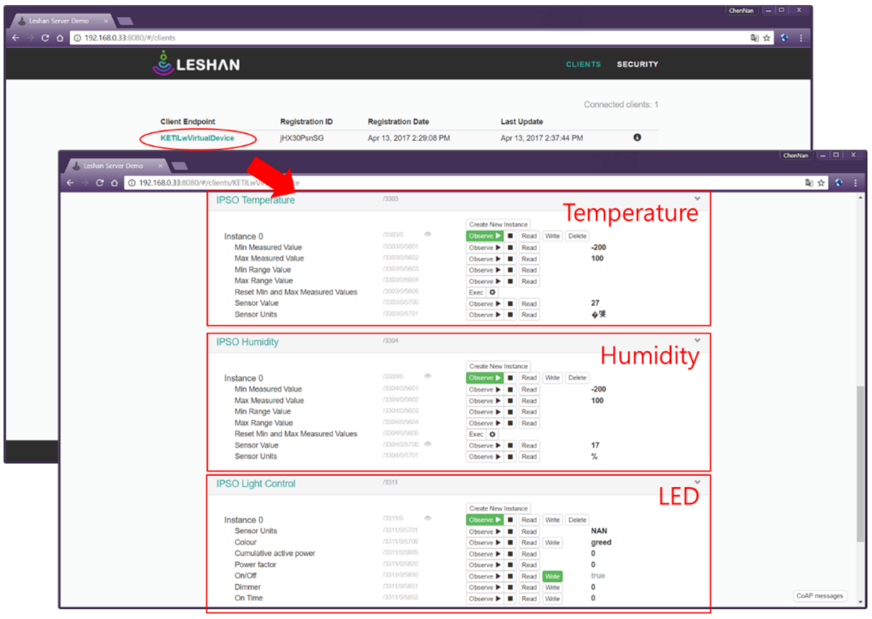
```
Apr 13, 2017 5:29:08 AM org.eclipse.californium.scandium.DTLSConnector start
INFO: DTLS connector listening on [/127.0.0.1:6001] with MTU [65,536] using (inb
ound) datagram buffer size [16,474 bytes]
Apr 13, 2017 5:29:08 AM org.eclipse.californium.core.network.CoapEndpoint start
INFO: Starting endpoint at localhost/127.0.0.1:6000
Apr 13, 2017 5:29:09 AM org.eclipse.californium.core.CoapResource addObserveRela
tion
INFO: Successfully established observe relation between /127.0.0.1:5083#09df7f8c
df and resource /3303
Receive data <= {"temp":27.00,"humi":18.00,"seq":1543,"color":"green","led":"off
"}
Apr 13, 2017 5:29:10 AM org.eclipse.californium.core.CoapResource addObserveRela
tion
INFO: Successfully established observe relation between /127.0.0.1:5083#2f0c5e97
472306 and resource /3304
Receive data <= {"temp":27.00,"humi":19.00,"seq":1544,"color":"green","led":"off
"}
Receive data <= {"temp":27.00,"humi":18.00,"seq":1545,"color":"green","led":"off
"}
Receive data <= {"temp":27.00,"humi":19.00,"seq":1546,"color":"green","led":"off
"}
Receive data <= {"temp":27.00,"humi":18.00,"seq":1547,"color":"green","led":"off
"}
```

Figure 17 LWM2M device client execution

4. Test

4.1. LWM2M test

1. Open http://[raspberry pi IP]:8080 in web browser.



Client Endpoint	Registration ID	Registration Date	Last Update
KETILwVirtualDevice	jHX30PsnSG	Apr 13, 2017 2:29:08 PM	Apr 13, 2017 2:37:44 PM

IPSO Temperature /3303

Instance 0	/3303/0	
Min Measured Value	/3303/0/5601	-200
Max Measured Value	/3303/0/5602	100
Min Range Value	/3303/0/5603	
Max Range Value	/3303/0/5604	
Reset Min and Max Measured Values	/3303/0/5605	
Sensor Value	/3303/0/5700	27
Sensor Units	/3303/0/5701	�햨

IPSO Humidity /3304

Instance 0	/3304/0	
Min Measured Value	/3304/0/5601	-200
Max Measured Value	/3304/0/5602	100
Min Range Value	/3304/0/5603	
Max Range Value	/3304/0/5604	
Reset Min and Max Measured Values	/3304/0/5605	
Sensor Value	/3304/0/5700	17
Sensor Units	/3304/0/5701	%

IPSO Light Control /3311

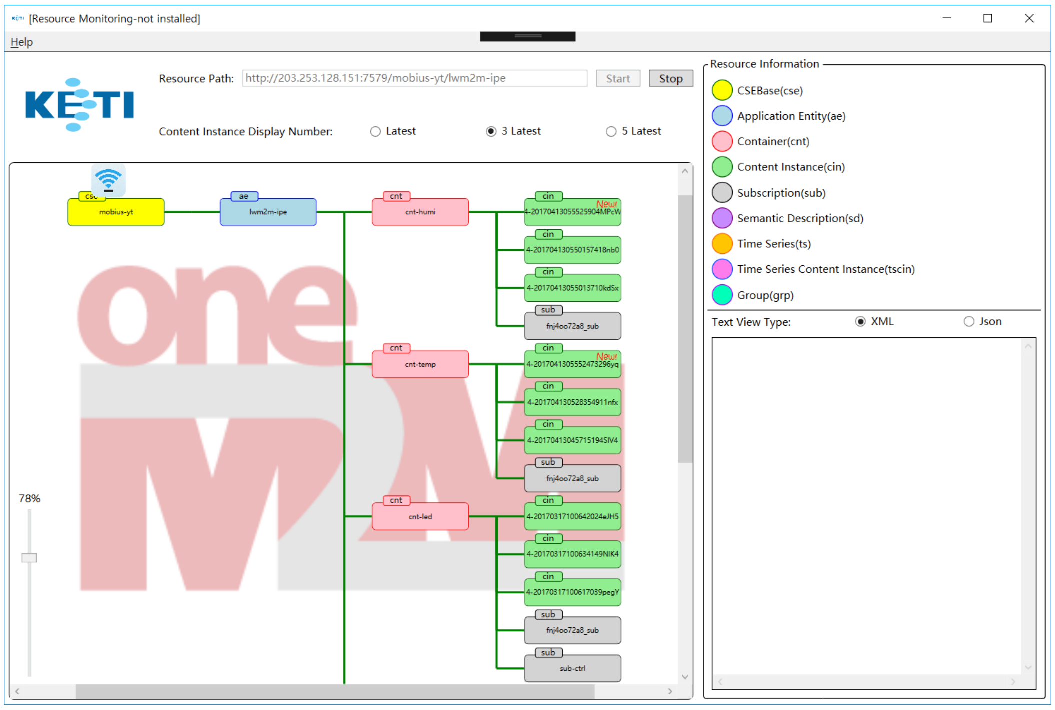
Instance 0	/3311/0	
Sensor Units	/3311/0/5701	NAN
Colour	/3311/0/5706	greed
Cumulative active power	/3311/0/5805	0
Power factor	/3311/0/5820	0
On/Off	/3311/0/5850	true
Dimmer	/3311/0/5851	0
On Time	/3311/0/5852	0

Figure 18 LWM2M service web

If web browser shows the content as Figure 18, LWM2M is properly functioning.

4.2. oneM2M test

1. Run Resource Tree Viewer that it is available from IoT Ocean Site.



[Resource Monitoring-not installed]

Resource Path: http://203.253.128.151:7579/mobius-yt/lwm2m-ipe

Content Instance Display Number: Latest [✓ 3 Latest] 5 Latest

Text View Type: [✓ XML] Json

Resource Information

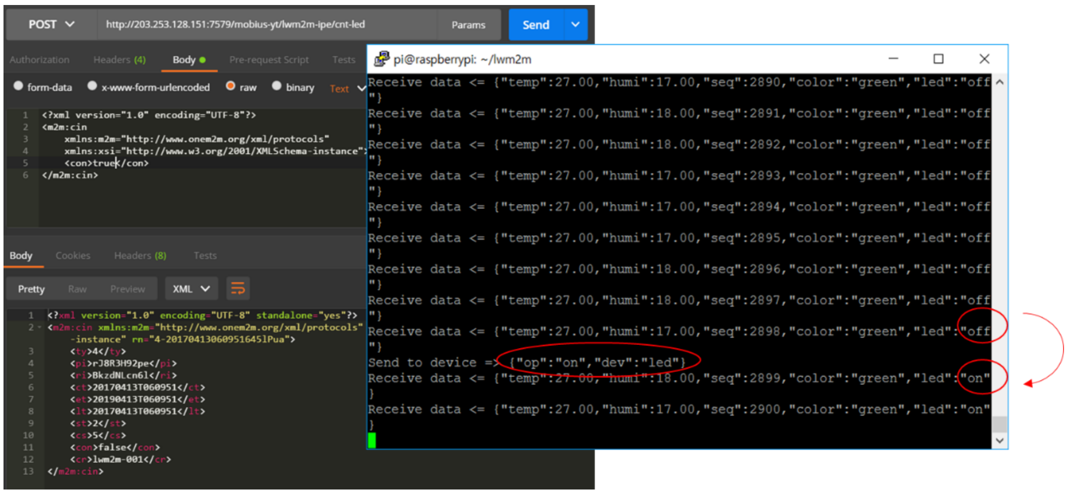
🟡	CSEBase(cse)
🔵	Application Entity(ae)
🔴	Container(cnt)
🟢	Content Instance(cin)
⚪	Subscription(sub)
🟣	Semantic Description(sd)
🟠	Time Series(ts)
🟣	Time Series Content Instance(tscin)
🟢	Group(grp)

Figure 19 Resource Tree Viewer

2. Input AE resource path (depended the thyme conf.json) in textbox and click start then the above LWM2M IPE's resource structure will be displayed. If temperature or humidity value is changed then viewer will add new resource object with "new!" tag.

3. Open post man from google chrome and send content instance create request to mobius-yt as figure 20.

4. If IPE sample build is successfully done, the physical LED light turns on when LWM2M log device "led" status changes "off" to "on".



```
http://203.253.128.151:7579/mobius-yt/lwm2m-ipe/cnt-led
```

```xml
<?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>true</con>
</m2m:cin>
```

```xml
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<m2m:cin xmlns:m2m="http://www.onem2m.org/xml/protocols"
    -instance" rn="4-20170413060951645lPua">
    <ty>4</ty>
    <pi>rJ8R3H92pe</pi>
    <ri>BkzdNLcn6l</ri>
    <ct>20170413T060951</ct>
    <et>20190413T060951</et>
    <lt>20170413T060951</lt>
    <st>2</st>
    <cs>5</cs>
    <con>false</con>
    <cr>lwm2m-001</cr>
</m2m:cin>
```

```
Receive data <= {"temp":27.00,"humi":17.00,"seq":2890,"color":"green","led":"off
"}
Receive data <= {"temp":27.00,"humi":18.00,"seq":2891,"color":"green","led":"off
"}
Receive data <= {"temp":27.00,"humi":18.00,"seq":2892,"color":"green","led":"off
"}
Receive data <= {"temp":27.00,"humi":17.00,"seq":2893,"color":"green","led":"off
"}
Receive data <= {"temp":27.00,"humi":17.00,"seq":2894,"color":"green","led":"off
"}
Receive data <= {"temp":27.00,"humi":17.00,"seq":2895,"color":"green","led":"off
"}
Receive data <= {"temp":27.00,"humi":18.00,"seq":2896,"color":"green","led":"off
"}
Receive data <= {"temp":27.00,"humi":18.00,"seq":2897,"color":"green","led":"off
"}
Receive data <= {"temp":27.00,"humi":17.00,"seq":2898,"color":"green","led":"off
"}
Send to device => {"op":"on","dev":"led"}
Receive data <= {"temp":27.00,"humi":18.00,"seq":2899,"color":"green","led":"on"
}
Receive data <= {"temp":27.00,"humi":17.00,"seq":2900,"color":"green","led":"on"
}
```

Figure 20 LED Control test