# Research Teaser Course on IoT, 2021

## Unit-2-Exercise – 3

**Overview:**

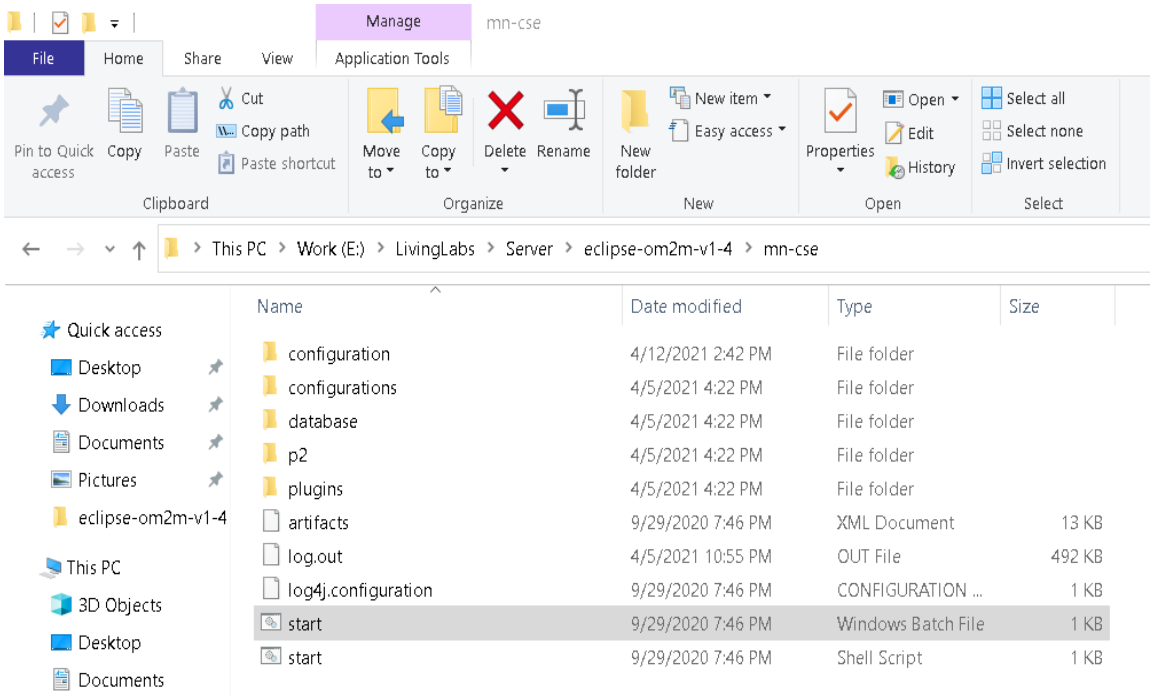
In this tutorial we will be using actuator function of the OM2M server to control the actions performed in microcontroller. Here we will be using the GUI computer provided by the OM2M in our computer for implementing the actuator functionality.

**Prerequisites:**

1. Micro Controller
2. Arduino IDE
3. Servo Motor
4. LDR sensor
5. Working OM2M server

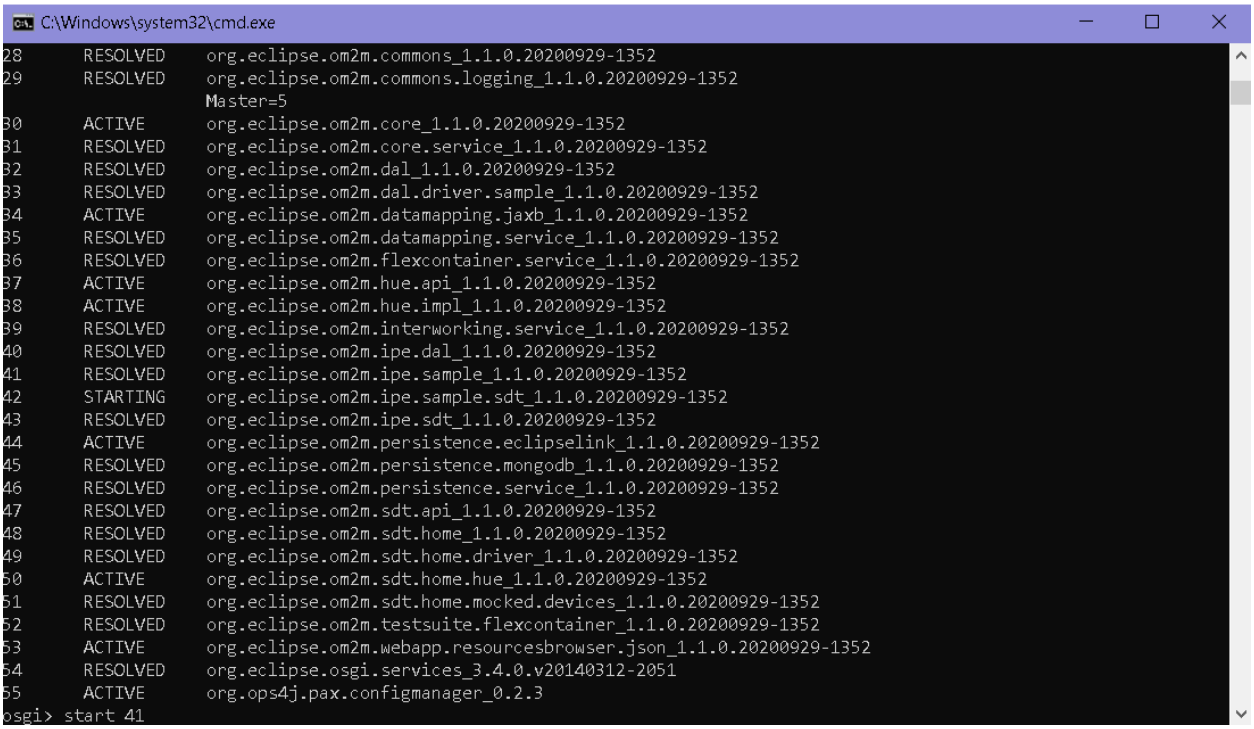
**Part-1: Switching ON the Web Interface plugin in OM2M.**

- After starting the server 'go to the mn-cse folder and start the ms-cse.

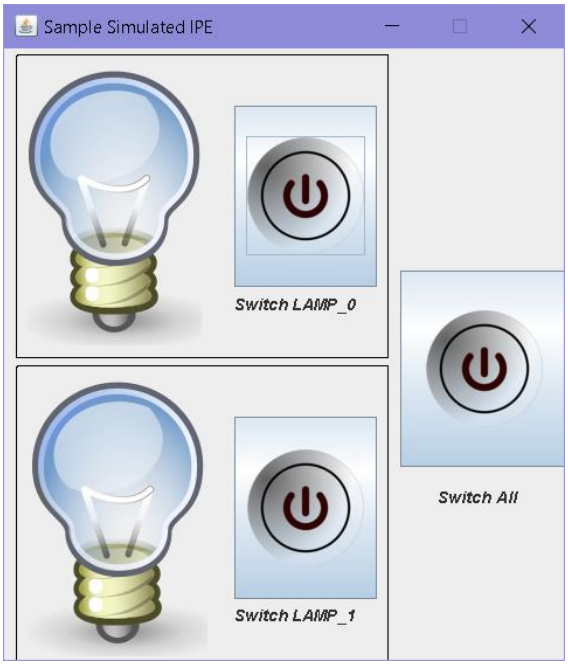



- Now press "ss" to get the list of services which are enabled and can be started.

- Press “start 41”.




- A pop-up window showing the 2 light blub will be displayed as shown in below image.




- It can be observed that every on-off action is being recorded as content instances in the corresponding data containers of the AEs in mn-cse.

Logout

OM2M CSE Resource Tree

http://127.0.0.1:8080/~/in-cse/csr-427531233

- in-name
  - acp_admin
  - SDT_Home_Monitoring_Application_ACP
  - ACP_Device_Admin_1617619573899
  - SDT_Home_Monitoring_Application
  - SDT_IPE
  - mn-name

| Attribute | Value |
|---|---|
| rn | mn-name |
| ty | 16 |
| ri | /in-cse/csr-427531233 |
| pi | /in-cse |
| ct | 20210405T162245 |
| lt | 20210405T162245 |
| acpi | AccessControlPolicyIDs: /in-cse/acp-730458245 |
| poa | Point Of Access: http://127.0.0.1:8282/ |
| cb | //om2m.org/mn-cse |
| csi | /mn-cse |
| rr | true |

Logout

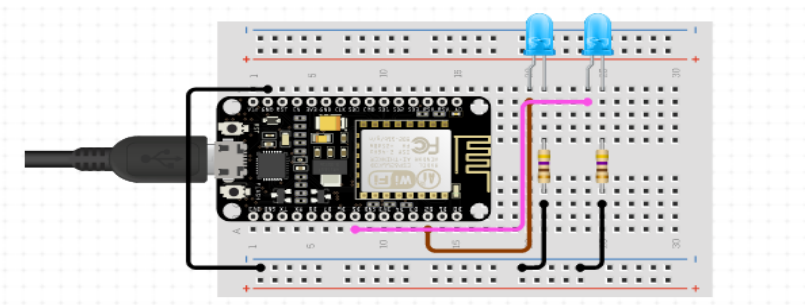
OM2M CSE Resource Tree

http://127.0.0.1:8080/~/mn-cse/cin-937993554

- mn-name
  - acp_admin
  - LAMP_0
    - DATA
      - cin_670355852
      - cin_337407651
      - cin_168899023
      - cin_515680911
      - cin_488056530
      - cin_150897122
      - cin_293731779
      - cin_196913454
      - cin_82543452
      - cin_937993554
    - DESCRIPTOR
  - LAMP_1
  - LAMP_ALL

| Attribute | Value |
|---|---|
| rn | cin_937993554 |
| ty | 4 |
| ri | /mn-cse/cin-937993554 |
| pi | /mn-cse/cnt-357220117 |
| ct | 20210412T144637 |
| lt | 20210412T144637 |
| st | 0 |
| cnf | application/obix:0 |
| cs | 215 |
| con | Attribute / Value: type: LAMP; location: Home; lampId: LAMP_0; state: true |

## Part-2: Implementing the Basic Circuit



**Part-3: Understanding the esp8266 code flow.**

Step-1: Including the required header files

```
#include<ESP8266HTTPClient.h>

#include <ESP8266WiFi.h>
```

Step-2: Defining the required constant variables

```
#define MAIN_SSID "your_ssid"

#define MAIN_PASS "your_password"


#define CSE_IP      "your server ip"

#define CSE_PORT    8080

#define HTTPS     false

#define OM2M_ORGIN    "admin:admin"

#define OM2M_MN     "/~/mn-cse/mn-name/"

#define OM2M_AE_0     "LAMP_0"

#define OM2M_AE_1     "LAMP_1"

#define OM2M_DATA_CONT  "DATA"

#define LISTENER_PORT  8000

WiFiServer listener(LISTENER_PORT);

HTTPClient http;
```

Step-3: defining the sensor pins(here we are using 2 leds)

```
#define LED1 4

#define LED2 14
```

Step-4: Defining the void function which executes only one time.

- This has the Wi-Fi connection part, after a connection is established the esp posts an http request which creates a subscriber object.
- The subscriber content instance constantly sends notification to the esp Ip at port 8000 whenever a new content instance is created in LAMP_0 and LAMP_1 AEs

```
    Serial.begin(115200);
    Serial.println("Connecting to "+String()+MAIN_SSID);
   status = WiFi.begin(ssid, pass);
 if ( status != WL_CONNECTED) {
  Serial.println("Couldn't get a wifi connection");
  // don't do anything else:
  while(true);
 }
    Serial.println("COnnection Successful");
    listener.begin(LISTENER_PORT);
    Serial.println("ESP listener started");
    Serial.println("Ip Address is"+WiFi.localIP());
    delay(500);
   ///Lamp-0 Object
   String server="http://" + String() + CSE_IP + ":" + String() + CSE_PORT +
String()+OM2M_MN;
  http.begin(server + String() +OM2M_AE_0 + "/" + OM2M_DATA_CONT + "/");
  http.addHeader("X-M2M-Origin", OM2M_ORGIN);
  http.addHeader("Content-Type", "application/json;ty=23");
  http.addHeader("Content-Length", "100");
  String req_data = String() + "{\"m2m:sub\": {"
  + "\"rn\": \"led_sub_test\""+","
  + "\"nu\": \"" + "http://"+WiFi.localIP().toString()+":"+LISTENER_PORT + "\","
  + "\"nct\": \"2\""
  + "}}";
 int x=http.POST(req_data);
 http.end();
```

```
////Lamp-1 Object
 http.begin(server + String() +OM2M_AE_1 + "/" + OM2M_DATA_CONT + "/");
 http.addHeader("X-M2M-Origin", OM2M_ORGIN);
 http.addHeader("Content-Type", "application/json;ty=23");
 http.addHeader("Content-Length", "100");
 req_data = String() + "{\"m2m:sub\": {"
  + "\"rn\": \"led_sub_test\""+","
  + "\"nu\": \"" + "http://"+WiFi.localIP().toString()+":"+LISTENER_PORT + "\","
  + "\"nct\": \"2\""
  + "}}";
  x =http.POST(req_data);
  http.end();
  pinMode(LED1,OUTPUT);
  pinMode(LED2,OUTPUT);
```

Step-4: Creating the loop function

- The loop function constantly listens for incoming data at esp_ip:8000
- Whenever data is received in this port, the esp check for status of LAMP_0 &1 and performs the actuation necessary.

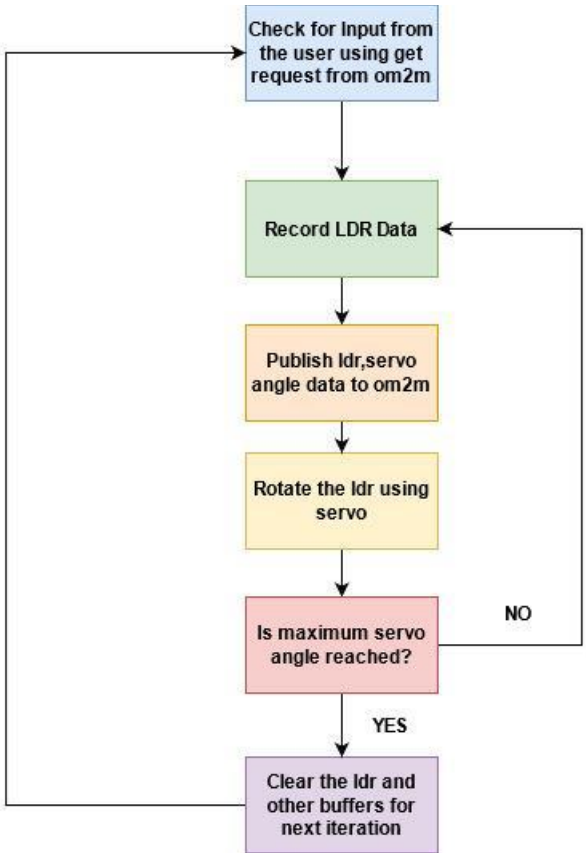
```
listener.begin(LISTENER_PORT);
WiFiClient client = listener.available();
 String w=client.readString();
Serial.println(w);
int a=w.indexOf("false");
int b=w.indexOf("true");
int c=w.indexOf("LAMP_0");
int d=w.indexOf("LAMP_1");

if(b>0)
{
 if(c>0)
 {
 digitalWrite(LED1,HIGH);
```

```
  Serial.println("Light-1 ON");
  }
  else if(d>0)
  {  digitalWrite(LED2,HIGH);
    Serial.println("Light-2 ON");
  }
}
else if(a>0)
{
  if(c>0)
  {
  digitalWrite(LED1,LOW);
  Serial.println("Light-1 OFF");
  }
  else if(d>0)
  {  digitalWrite(LED2,LOW);
    Serial.println("Light-2 OFF");
  }
}
client.flush();
// Send HTTP response to the client
String s = "HTTP/1.1 200 OK\r\n";
client.print(s);
delay(100);
```

**Part-4: Implementing a case study on ambient light variations based on servo action.**

- Studying the ambient light intensity variations is a very important factor for proper implementation of ventilation within a room.
- For implementing such a study at a very basic level we can use the LDR sensor coupled with a servo motor to generate a light map of a given room
- Along with the data logging feature we can also implement an actuation mechanism to avoid generating data which might lead to improper conclusions.
- The basic functionality of this system is a LDR sensor which sits on a servo motor and captures the light intensity variation with a constant rotation.
- The following diagram captures the essence of this system.




**Part-4: References**

- **Data Request formats:**
  1. **GET Request:**

| Field | Value |
|---|---|
| URL | http://127.0.0.1:8080/~/in-cse |
| Method | GET |
| Header | X-M2M-Origin: admin:admin<br>Accept: application/xml |
| Body | (empty) |

2. **Subscriber request:**

| Field | Value |
|---|---|
| URL | http://127.0.0.1:8080/~/in-cse/in-name/LAMP_0/DATA |
| Method | POST |
| Header | X-M2M-Origin: admin:admin<br>Content-Type: application/xml;ty=23 |
| Body | <m2m:sub xmlns:m2m="http://www.onem2m.org/xml/protocols" rn="SUB_MY_SENSOR"><br><nu>http://localhost:1400/monitor</nu><br><nct>2</nct><br></m2m:sub> |

- **Header Files:**
    1. **Servo.h**
    2. **Wifi-header files (As per exercise-2)**
- **Data Sheets:**
    1. **http://www.ee.ic.ac.uk/pcheung/teaching/DE1_EE/stores/sg90_datasheet.pdf**
    2. **https://components101.com/asset/sites/default/files/component_datasheet/LDR%20Datasheet.pdf**