# **[Demo 15: How to build a system to update Price Tag automatically using Arduino ESP32](http://www.iotsharing.com/2017/05/how-to-build-system-to-update-price-tag.html)**

**1. Introduction**

- Imaging you have a mobile store with many mobiles on the shelf. Every mobile has its own price tag. One day you want to update the price tag for them, so it take a lot of time to update price tag for the whole store. In this demo I will show you how to build a system to update price tag automatically using Arduino ESP32. You just sit at one place, open an application, input the new price and click OK.

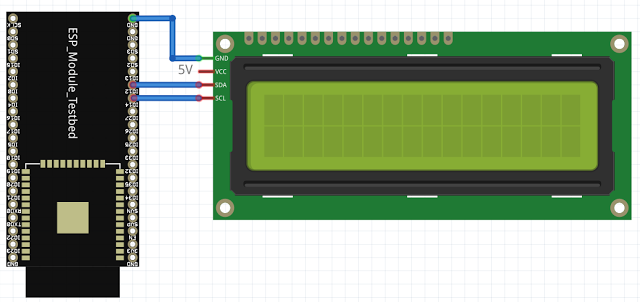
In order to build this system we will refer:

[Demo 4: How to use Arduino ESP32 to display information on I2C LCD](http://www.iotsharing.com/2017/05/how-to-use-arduino-esp32-to-display-i2c-lcd.html" \t "http://www.iotsharing.com/2017/05/_blank)  
[Demo 6: How to use Arduino ESP32 to display information on OLED](http://www.iotsharing.com/2017/05/how-to-use-arduino-esp32-to-display-oled.html" \t "http://www.iotsharing.com/2017/05/_blank)

[Demo 14: How to use MQTT and Arduino ESP32 to build a simple Smart home system](http://www.iotsharing.com/2017/05/how-to-use-mqtt-to-build-smart-home-arduino-esp32.html" \t "http://www.iotsharing.com/2017/05/_blank)

**2. Hardware**

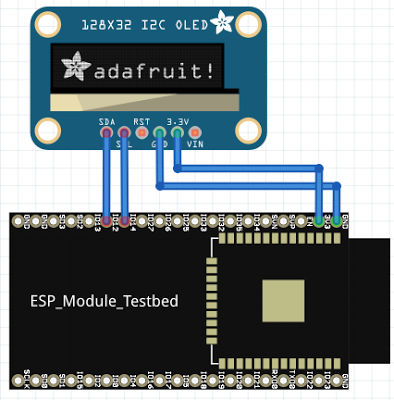
- We will use LCD, OLED to display the Price Tag. First row to display mobile name (iPhone 7). And second row to display the price of it (700USD).  
- In this demo we take 2 nodes (2 ESP32 or ESP8266): first node to display the price tag of iPhone and the second node to display price tag of Samsung S8.  
- For LCD, we connect like [Demo 4](http://www.iotsharing.com/2017/05/how-to-use-arduino-esp32-to-display-i2c-lcd.html" \t "http://www.iotsharing.com/2017/05/_blank)

[](https://4.bp.blogspot.com/-TZ-fcETtG10/WRnDdXBHdlI/AAAAAAAAD00/wyOCTPE-TIAY89QzFWMv2iCU5hw_ShefQCPcB/s1600/lcd1.png)

**Figure: ESP32 connect to I2C SH1106 OLED**

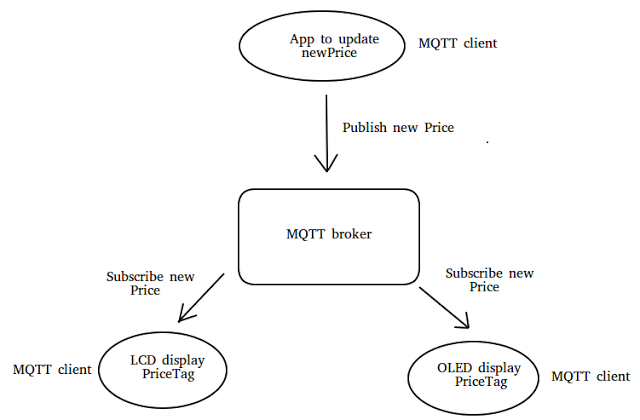
Here we connect:  
[ESP32 GPIO12 - LCM1602 SDA]  
[ESP32 GPIO14 - LCM1602 SCL]  
[ESP32 GND - LCM1602 GND]  
[LCM1602 - 5V]

- For OLED, we connect like [Demo 6](http://www.iotsharing.com/2017/05/how-to-use-arduino-esp32-to-display-oled.html" \t "http://www.iotsharing.com/2017/05/_blank)

[](https://4.bp.blogspot.com/-5dvtnzwS1lE/WRoxcVH2DCI/AAAAAAAAD1A/SDT1P_YW62I0jI9LgT8p1udPvcevydswACPcB/s1600/oled2.png)

**Figure: ESP32 connect to I2C SH1106 OLED**

Here we connect:  
[ESP32 3.3V – OLED VCC]  
[ESP32 GND – OLED GND]  
[ESP32 IO12– OLED SDA]  
[ESP32 IO14 – OLED SCL]  
**3. Software**  
We apply MQTT for our system. Here is the model:

[](https://1.bp.blogspot.com/-6HUgmzFEy1s/WSgwEJxlcfI/AAAAAAAAD8E/CcW2lfaiEdAZhCbtWkaQeKvs0ppz_ijiwCLcB/s1600/pricetag1.png)

**Figure: Price tag system model using MQTT**

- MQTT broker, we use mosquitto

- Two Arduino ESP32 nodes using MQTT library PubSubClient in [Demo 14](http://www.iotsharing.com/2017/05/how-to-use-mqtt-to-build-smart-home-arduino-esp32.html" \t "http://www.iotsharing.com/2017/05/_blank)

- For simple demo, I developed the application to update new price in Python and MQTT client library called [paho-mqtt.](https://pypi.python.org/pypi/paho-mqtt/1.1" \t "http://www.iotsharing.com/2017/05/_blank)

[](https://2.bp.blogspot.com/-H04rDaOm3IM/WSgzoYKWvkI/AAAAAAAAD8Q/Ke2rrjtB-T0Q5BxBH8n43j8h6SgGqoo7ACLcB/s1600/pricetag2.png)

**Figure: Application to update new Price (Python + paho mqtt client)**

- Next we define topic:

**update/pricetag/{unique\_id\_of\_product\_need\_update\_price}**

For example: **update/pricetag/sss8**: here the LCD ESP32 node will display the price tag of Samsung S8 and subscribe the topic **"update/pricetag/sss8"**. And the payload of topic is Self-defined "**samsung8-800USD**". Here **"samsung8"** is new product name, and **"800USD"** is new product price. Both are separated by **"-"**.

- Download and install Python DIY

- To install Paho-mqtt, there are 2 ways:

 + Using pip, from command line type: **pip install paho-mqtt**  
 + Download and unzip [paho-mqtt](https://pypi.python.org/packages/83/96/dacc2b78bc9c5cd83eed178e9ce35d7bceecf2dd38db079c0190423efd4a/paho-mqtt-1.1.tar.gz" \t "http://www.iotsharing.com/2017/05/_blank), go to unzipped folder and type:   
**python setup.py install**  
- This library also support standard interfaces of MQTT like subscribe() and publish() (publish single or multiple topic(s))  
Here is the code for update price application**: pricetag.py**

|  |
| --- |
| #this is to overcome Python2/3 version  try:  #this is style for Python 3 import  from tkinter import \*  except ImportError:  #in case import for Python 3 is error  #use this for Python 2 import  from Tkinter import \*  import sys  #similar to comments above  try:  import paho.mqtt.publish as publish  except ImportError:  import os  import inspect  cmd\_subfolder = os.path.realpath(os.path.abspath(os.path.join(os.path.split(inspect.getfile( inspect.currentframe() ))[0],"../src")))  if cmd\_subfolder not in sys.path:  sys.path.insert(0, cmd\_subfolder)  import paho.mqtt.publish as publish  #for demo I use 2 nodes  data = [  ['sss8', 'sss8.gif', 'SamSung S8', '800USD'],  ['ip7', 'ip7.gif', 'iPhone 7S', '700USD']  ]  #this is the simple GUI part  root = Tk()  root.wm\_title("PriceTag")  rows = []  img = []  for i in range(2):  cols = []  for j in range(4):  if(j == 1):  img.append(PhotoImage(file=data[i][j]))  e = Button(root, image=img[-1])  e.grid(row=i, column=j, sticky=NSEW)  elif(j == 0):  e = Entry(root, relief=RIDGE, justify='center')  e.grid(row=i, column=j, sticky=NSEW)  e.insert(END, data[i][j])  e.config(state=DISABLED)  cols.append(e)  else:  e = Entry(root, relief=RIDGE, justify='center')  e.grid(row=i, column=j, sticky=NSEW)  e.insert(END, data[i][j])  cols.append(e)  rows.append(cols)  #this function will be invoked when the Update button is clicked  def onUpdate():  msgs = []  #we loop through all products and publish new price  for row in rows:  #build topic: update/pricetag/sss8 : {'name': 'samsung8', 'value': '800USD'}  payload = str(row[1].get()) + '-' + str(row[2].get())  #every topic is stored under dictionary like json  pair = {'topic':'update/pricetag/'+row[0].get(), 'payload':payload}  #here we send multiple topics instead of looping through every topic and send  #so we append multiple topics in a list  msgs.append(pair);  #and pass the list of topics to publish.multiple()  publish.multiple(msgs, hostname="localhost")    Button(text='Update', command=onUpdate).grid()  mainloop() |

Here is the Arduino code for LCD display: **esp32lcdpricetag**

|  |
| --- |
| #include <WiFi.h>  #include <PubSubClient.h>  #include <Wire.h>  #include <LiquidCrystal\_I2C.h>  /\* change it with your ssid-password \*/  const char\* ssid = "dd-wrt";  const char\* password = "0000000000";  /\* this is the IP of PC where you installed MQTT Server \*/  const char\* mqtt\_server = "192.168.1.103";  /\* create an instance of PubSubClient client \*/  WiFiClient espClient;  PubSubClient client(espClient);  /\* this is for LCD \*/  LiquidCrystal\_I2C lcd(0x27,16,2);  /\* topics \*/  #define SSS8\_PRICE\_TOPIC "update/pricetag/sss8"  long lastMsg = 0;  char msg[20];  void receivedCallback(char\* topic, byte\* payload, unsigned int length) {  Serial.print("Message received: ");  Serial.println(topic);  int separate = 0;  Serial.print("payload: ");  for (int i = 0; i < length; i++) {  Serial.print((char)payload[i]);  /\* we parse topic by separator '-'\*/  if(payload[i] == '-'){  separate = i+1;  payload[i] = 0;  }  }  lcd.clear();  lcd.setCursor(0,0);  lcd.print((char \*)payload);  /\* set cursor row 1 col 0 \*/  lcd.setCursor(0,1);  lcd.print((char \*)&payload[separate]);  }  void mqttconnect() {  /\* Loop until reconnected \*/  while (!client.connected()) {  Serial.print("MQTT connecting ...");  /\* client ID \*/  String clientId = "ESP32Client";  /\* connect now \*/  if (client.connect(clientId.c\_str())) {  Serial.println("connected");  /\* subscribe topic \*/  client.subscribe(SSS8\_PRICE\_TOPIC);  } else {  Serial.print("failed, status code =");  Serial.print(client.state());  Serial.println("try again in 5 seconds");  /\* Wait 5 seconds before retrying \*/  delay(5000);  }  }  }  void setup() {  Serial.begin(115200);  /\* initialize the lcd with SDA and SCL pins \*/  lcd.begin(12, 14);  /\* Print a message to the LCD.\*/  lcd.backlight();  /\* set cursor row 0 col 0 \*/  lcd.setCursor(0,0);  lcd.print("iotsharing.com");  /\* connect wifi \*/  Serial.print("Connecting to ");  Serial.println(ssid);  WiFi.begin(ssid, password);  while (WiFi.status() != WL\_CONNECTED) {  delay(500);  Serial.print(".");  }  Serial.println("");  Serial.println("WiFi connected");  Serial.println("IP address: ");  Serial.println(WiFi.localIP());  /\* configure the MQTT server with IPaddress and port \*/  client.setServer(mqtt\_server, 1883);  /\* this receivedCallback function will be invoked  when client received subscribed topic \*/  client.setCallback(receivedCallback);    }  void loop() {  /\* if client was disconnected then try to reconnect again \*/  if (!client.connected()) {  mqttconnect();  }  /\* this function will listen for incomming  subscribed topic-process-invoke receivedCallback \*/  client.loop();  } |

Here is the Arduino code for OLED display: **esp32oledpricetag**

|  |
| --- |
| #include <WiFi.h>  #include <PubSubClient.h>  #include <Wire.h>  #include <Adafruit\_SH1106.h>  #define OLED\_SDA 12  #define OLED\_SCL 14  Adafruit\_SH1106 display(12, 14);  /\* change it with your ssid-password \*/  const char\* ssid = "dd-wrt";  const char\* password = "0000000000";  /\* this is the IP of PC where you installed MQTT Server \*/  const char\* mqtt\_server = "192.168.1.103";  /\* create an instance of PubSubClient client \*/  WiFiClient espClient;  PubSubClient client(espClient);  /\* topics \*/  #define IP7\_PRICE\_TOPIC "update/pricetag/ip7"  long lastMsg = 0;  char msg[20];  #define BUF\_SIZE 100  byte rec[BUF\_SIZE];  int separate = 0;  int dispState = 0;    void receivedCallback(char\* topic, byte\* payload, unsigned int length) {  Serial.println(topic);  int i = 0;  Serial.print("payload: ");  for (i = 0; i < length; i++) {  Serial.print((char)payload[i]);  /\* we parse topic by separator '-'\*/  if(payload[i] == '-'){  separate = i+1;  payload[i] = 0;  }  if(i < BUF\_SIZE){  rec[i] = payload[i];  }  }  /\*we got data so backup for processing  if we have enough buffer size, else ignore msg \*/  if(i < BUF\_SIZE){  if(dispState == 0){  dispState = 1;  }  }  }  void mqttconnect() {  /\* Loop until reconnected \*/  while (!client.connected()) {  Serial.print("MQTT connecting ...");  /\* client ID \*/  String clientId = "ESP32Client";  /\* connect now \*/  if (client.connect(clientId.c\_str())) {  Serial.println("connected");  /\* subscribe topic with QoS 1\*/  client.subscribe(IP7\_PRICE\_TOPIC, 1);  } else {  Serial.print("failed, status code =");  Serial.print(client.state());  Serial.println("try again in 5 seconds");  /\* Wait 5 seconds before retrying \*/  delay(5000);  }  }  }  void setup() {  Serial.begin(115200);  /\* while flashing sw detach OLED VCC w ESP32 3.3,  delay here to have enough time to attach  OLED VCC with ESP32 3.3 again \*/  delay(3000);  /\* initialize oled \*/  display.begin(SH1106\_SWITCHCAPVCC, 0x3C);  display.clearDisplay();  /\* set text size, color, cursor position,  set buffer with Hello world and show off\*/  display.setTextSize(2);  display.setTextColor(WHITE);  display.setCursor(2,2);  display.println("iotsharing.com");  display.display();    Serial.print("Connecting to ");  Serial.println(ssid);  WiFi.begin(ssid, password);  while (WiFi.status() != WL\_CONNECTED) {  delay(500);  Serial.print(".");  }  Serial.println("");  Serial.println("WiFi connected");  Serial.println("IP address: ");  Serial.println(WiFi.localIP());  /\* configure the MQTT server with IPaddress and port \*/  client.setServer(mqtt\_server, 1883);  /\* this receivedCallback function will be invoked  when client received subscribed topic \*/  client.setCallback(receivedCallback);  }  void loop() {  /\* if client was disconnected then try to reconnect again \*/  if (!client.connected()) {  mqttconnect();  }  /\* this function will listen for incomming  subscribed topic-process-invoke receivedCallback \*/  client.loop();  /\* because OLED waste time  so we use state machine to display  step by step every loop \*/  switch(dispState)  {  case 1:  display.clearDisplay();  dispState = 2;  break;  case 2:  display.setCursor(2,2);  dispState = 3;  break;  case 3:  display.println((char \*)rec);  dispState = 4;  break;  case 4:  display.setCursor(2,22);  dispState = 5;  break;  case 5:  display.println((char \*)&rec[separate]);  dispState = 6;  break;  case 6:  display.display();  dispState = 0;  memset(rec, 0, BUF\_SIZE);  break;  default:  break;  }  } |

**4. Result**

[](https://3.bp.blogspot.com/-pGoL17Hys7E/WShfynhrkWI/AAAAAAAAD8g/s7UA2oNdK0ECPVvEq-8SobyKRNczARLpACLcB/s1600/pricetag4.jpg)

**Figure: auto update price tag**