Final Project part 1a

```
step 0, (go to the folder that installed MQTT with command;)
brew reinstall mosquitto

step 0b. make changes to the configuration
nano /usr/local/etc/mosquitto/mosquitto.conf

adding:
listener(1883) // subscribes to the esp32's port: -p 1883
allow_anonymous true
```

MQTT-client_subscriber#1

```
step 1. (go to folder that installed MQTT to connect MQTT
server)
1a. cd /usr/local/etc/mosquitto
1b. brew services restart mosquitto


step 2. same terminal:
mosquitto_sub -h "192.168.1.8" -v -t 'esp32/test' -d
(create a MQTT client sends a subscribe packet to MQTT server,
This process to will continue to run, waiting for any published
messages to arrive to the topic string)
-V: connect local MQTT server with version?
-h: subscribes to the ip address
-v: print any published messages verbosely
-t: topic 'esp32/test' that subscribe to
-d enable debug
```

MQTT-client_publisher#1

```
step 3. mosquitto_pub -t 'esp32/test' -m 'Greetings' -d
(open anther terminal to create an MQTT client to publish a
message "Greetings" to the specified topic)
```

MQTT-client_subscriber#1

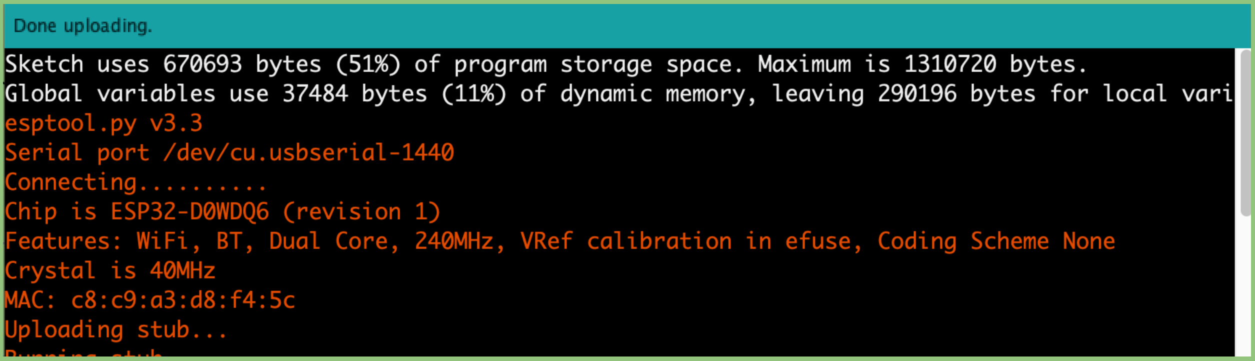
```
step 4. (go back to mosquitto_sub terminal that subscribe the
topic 'esp32/test')
```

see the output as a result of subscription to the specified topic.
esp32/test Greetings

MQTT-Client_publisher#2/subscriber#2
step 5. (arduino IDE: connect MQTT with esp32 under same wifi IP address)
5a. set project manager: arduino UNO --> ESP32 Wrover Module
5b. set library manager: PubSubClient
5c. open monitor serial set baud to 115200
5d. upload sketch code to esp32
source code:

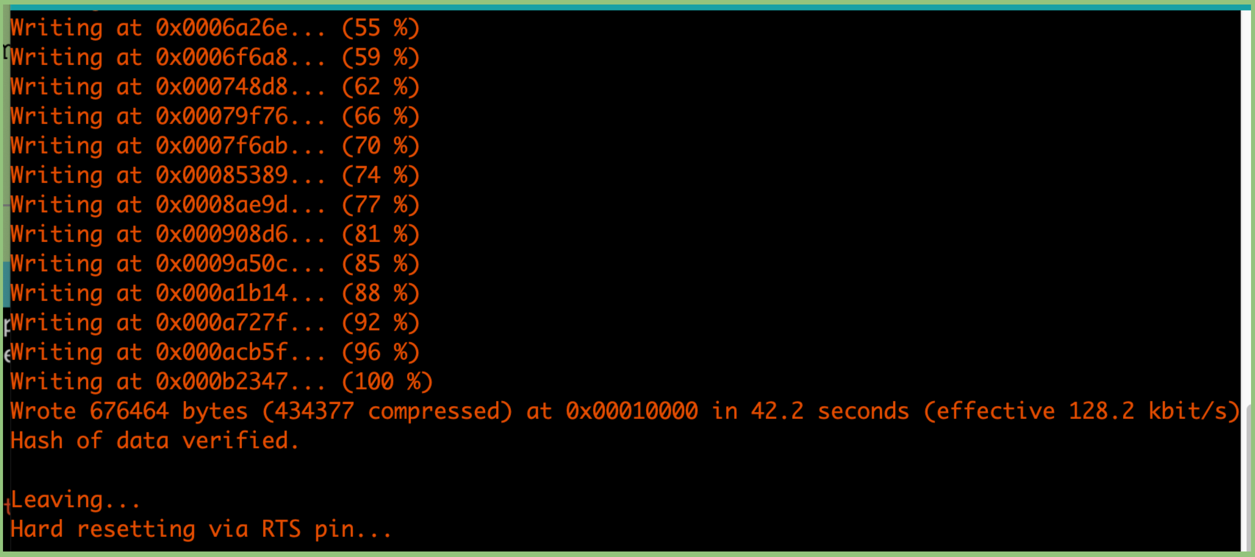
```
#include <WiFi.h>
#include <PubSubClient.h>

// variables for local WiFi
const char *ssid = "Be Cool Honey Bunny"; // WiFi name
const char *password = "7608144989";  // WiFi password

// variables for MQTT Broker(server)
const char *mqtt_broker = "192.168.1.8"; // WiFi local IP
address
const char *topic = "esp32/test";
const char *mqtt_username = "emqx";
const char *mqtt_password = "public";
const int mqtt_port = 1883;


WiFiClient espClient;
PubSubClient client(espClient);


void setup() {

 // 1. Set software serial baud to 115200;
 Serial.begin(115200);

 // 2. esp is connecting to local WiFi network
 WiFi.begin(ssid, password);

 while (WiFi.status() != WL_CONNECTED) {
     delay(500);
     Serial.println("Connecting to WiFi..");
 }
 Serial.println("Connected to the WiFi network");


 // 3. make esp32 to send / receive message via MQTT broker

// 3a. connecting to a mqtt broker
 client.setServer(mqtt_broker, mqtt_port);

// 4. message send back to esp32 (output in serial monitar under
same baud)
 client.setCallback(callback);

// 3b
 while (!client.connected()) {

     String client_id = "[esp32-client]";

     client_id += String(WiFi.macAddress());

     Serial.printf("The client: %s connects to the public mqtt
broker.\n", client_id.c_str());

     // esp connects to MQTT broker succeed
     if (client.connect(client_id.c_str(), mqtt_username,
mqtt_password)) {
         Serial.println("Public emqx mqtt broker connected.");
     }

     // failed
     else {
         Serial.print("failed with state ");
         Serial.print(client.state());
         delay(2000);
     }
 }

 // 5. publish message and subscribe the same topic to receive
message from MQTT server
 client.publish(topic, "Greetings from ESP32 board.");
 client.subscribe(topic);
}

// 4. what you will see in series monitar
void callback(char *topic, byte *payload, unsigned int length) {

 Serial.print("Message arrived in topic: ");
 Serial.println(topic);
```

```
  Serial.print("Message: ");
  for (int i = 0; i < length; i++) {
      Serial.print((char) payload[i]);
  }
  Serial.println();
  Serial.println("-----------------------");
}

// 5. run the loop
void loop() {
  client.loop();
}
```

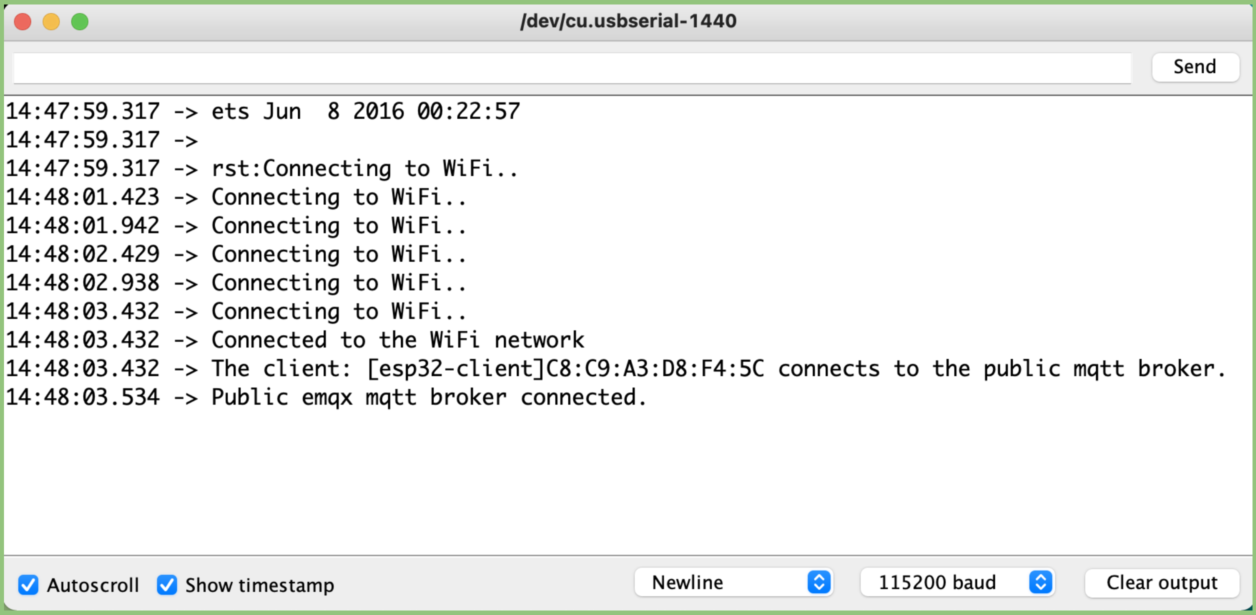


Done uploading.

```
Sketch uses 670693 bytes (51%) of program storage space. Maximum is 1310720 bytes.
Global variables use 37484 bytes (11%) of dynamic memory, leaving 290196 bytes for local vari
esptool.py v3.3
Serial port /dev/cu.usbserial-1440
Connecting..........
Chip is ESP32-D0WDQ6 (revision 1)
Features: WiFi, BT, Dual Core, 240MHz, VRef calibration in efuse, Coding Scheme None
Crystal is 40MHz
MAC: c8:c9:a3:d8:f4:5c
Uploading stub...
```



```
Writing at 0x0006a26e... (55 %)
Writing at 0x0006f6a8... (59 %)
Writing at 0x000748d8... (62 %)
Writing at 0x00079f76... (66 %)
Writing at 0x0007f6ab... (70 %)
Writing at 0x00085389... (74 %)
Writing at 0x0008ae9d... (77 %)
Writing at 0x000908d6... (81 %)
Writing at 0x0009a50c... (85 %)
Writing at 0x000a1b14... (88 %)
Writing at 0x000a727f... (92 %)
Writing at 0x000acb5f... (96 %)
Writing at 0x000b2347... (100 %)
Wrote 676464 bytes (434377 compressed) at 0x00010000 in 42.2 seconds (effective 128.2 kbit/s)
Hash of data verified.

Leaving...
Hard resetting via RTS pin...
```



```
14:47:59.317 -> ets Jun  8 2016 00:22:57
14:47:59.317 ->
14:47:59.317 -> rst:Connecting to WiFi..
14:48:01.423 -> Connecting to WiFi..
14:48:01.942 -> Connecting to WiFi..
14:48:02.429 -> Connecting to WiFi..
14:48:02.938 -> Connecting to WiFi..
14:48:03.432 -> Connecting to WiFi..
14:48:03.432 -> Connected to the WiFi network
14:48:03.432 -> The client: [esp32-client]C8:C9:A3:D8:F4:5C connects to the public mqtt broker.
14:48:03.534 -> Public emqx mqtt broker connected.
```

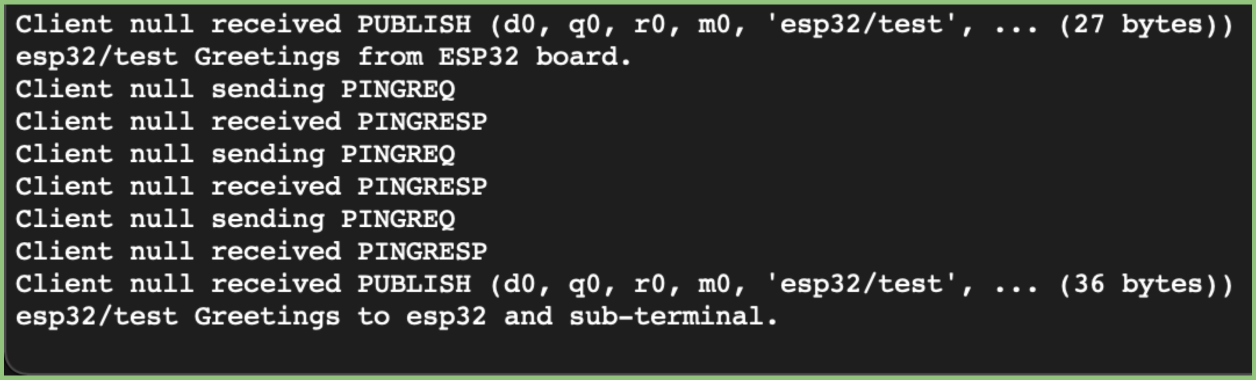
✓ Autoscroll ✓ Show timestamp Newline 115200 baud Clear output

MQTT-client_publisher#1

```
step 6. on pub terminal,
-h: creating a host with local IP;
-t: indicate the topic
-m: publish message to subscribers with the same topic

mosquitto_pub -h "192.168.1.8" -t 'esp32/test' -m 'Greetings to
esp32 and sub-terminal.'
```

MQTT-client_subscriber#1

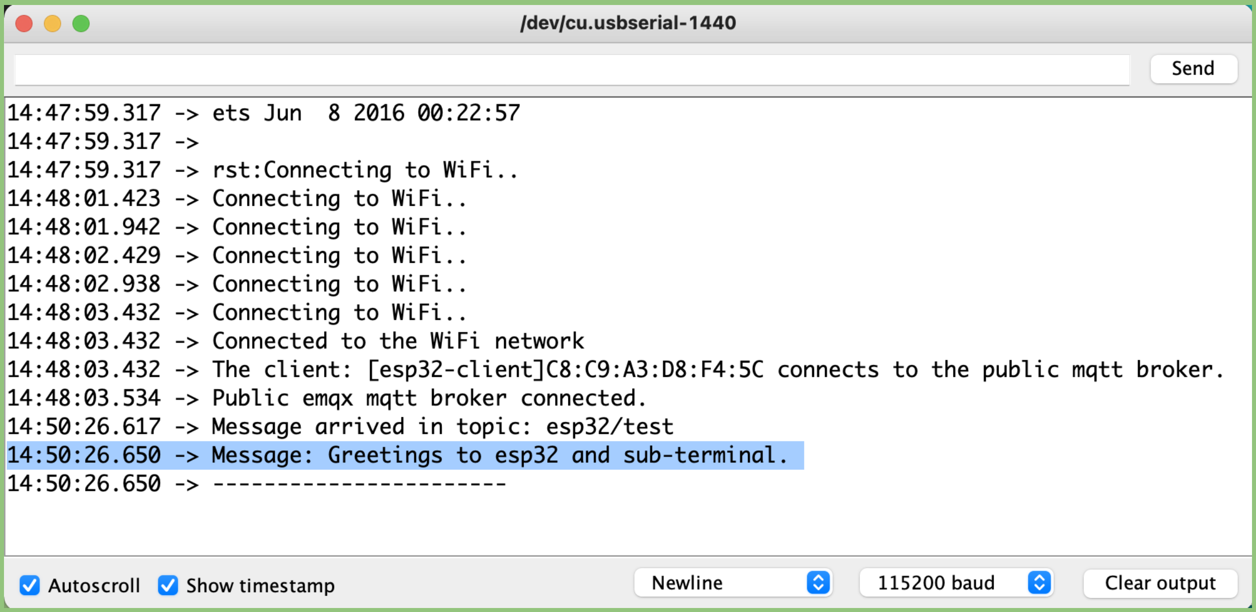
step 7. on sub terminal, will display messages from 2 publishers(esp and terminal)



```
Client null received PUBLISH (d0, q0, r0, m0, 'esp32/test', ... (27 bytes))
esp32/test Greetings from ESP32 board.
Client null sending PINGREQ
Client null received PINGRESP
Client null sending PINGREQ
Client null received PINGRESP
Client null sending PINGREQ
Client null received PINGRESP
Client null received PUBLISH (d0, q0, r0, m0, 'esp32/test', ... (36 bytes))
esp32/test Greetings to esp32 and sub-terminal.
```

MQTT-Client_publisher#2/subscriber#2

on esp serial monitor →



```
14:47:59.317 -> ets Jun  8 2016 00:22:57
14:47:59.317 ->
14:47:59.317 -> rst:Connecting to WiFi..
14:48:01.423 -> Connecting to WiFi..
14:48:01.942 -> Connecting to WiFi..
14:48:02.429 -> Connecting to WiFi..
14:48:02.938 -> Connecting to WiFi..
14:48:03.432 -> Connecting to WiFi..
14:48:03.432 -> Connected to the WiFi network
14:48:03.432 -> The client: [esp32-client]C8:C9:A3:D8:F4:5C connects to the public mqtt broker.
14:48:03.534 -> Public emqx mqtt broker connected.
14:50:26.617 -> Message arrived in topic: esp32/test
14:50:26.650 -> Message: Greetings to esp32 and sub-terminal.
14:50:26.650 -> -----------------------
```

MQTT-client_publisher#1

step 8. stop service to the middle man: MQTT broker(server)

```
brew services stop mosquitto
```