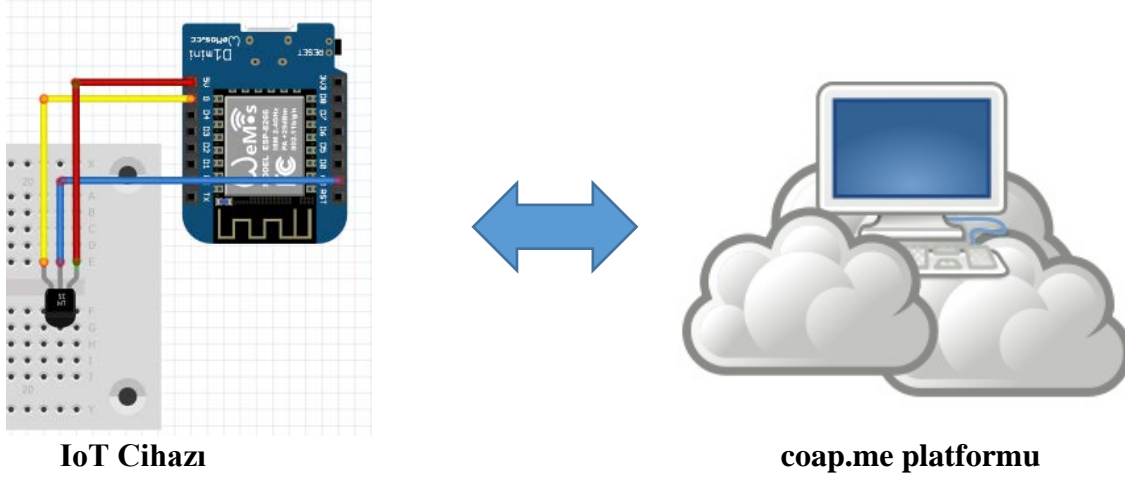


Uygulama Adı: CoAP Protokolü İle IoT Uygulaması

No:

Uygulamanın Tanıtımı:

Esp8266 modülüne sahip Wemos D1 Mini IoT cihazı ile LM35 sensöründen algılanan ortam sıcaklık bilgisini MQTT protokolü ile adafruit IoT platformuna gönderen uygulama.



Şekil 1. Sistem mimarisi

Ekipman Listesi ve Kullanılan Teknolojiler:

- Wemos D1 mini ya da (Arduino + Esp8266 modülü)
- LM35 sıcaklık sensörü
- BreadBord
- Jumper Kablo
- Adafruit IoT platformu
- CoAP protokolü

Kullanılan Teknolojilere Yönelik Teknik Bilgiler:

Wemos D1 Mini

Arduino geliştirme ortamı (IDE), Arduino bootloader (Optiboot), Arduino kütüphaneleri, AVR Dude (Arduino üzerindeki mikrodenetleyici programlayan yazılım) ve derleyiciden (AVR-GCC) oluşur. Arduino yazılımı bir geliştirme ortamı (IDE) ve kütüphanelerden oluşur. IDE, Java dilinde yazılmıştır ve Processing adlı dilin ortamına dayanmaktadır. Kütüphaneler ise C ve C++ dillerinde yazılmıştır ve AVR-GCC ve AVR Libc. ile derlenmiştir.

Wemos D1 kartını Arduino IDE’nde tanımlı kartlar arasına ekleyebilmek için **Dosya > Tercihler** sekmesindeki ekranda **“Ek Devre Kartları Yöneticisi URL’leri”** kutusuna aşağıda verilen linki ekleyiniz.

http://arduino.esp8266.com/stable/package_esp8266com_index.json

Esp8266

Kolayca wireless ağlara bağlanmayı sağlayan modül. esp8266-01'den başlayıp esp8266-12'ye kadar giden versiyonları bulunuyor. Kendi firmware'inizi yazıp yükleyerek başka hiçbir şeye ihtiyaç duymadan uygulama geliştirebiliyoruz. AT+ ile başlayan komutları göndererek bağlanılabilir wi-fi ağlarının listelenmesi, wi-fi adı ve şifresinin gönderilmesiyle ağa bağlanması, ağ üzerinden bir sunucuyla tcp bağlantısı kurup istemci olarak veri alışverişi yapılması, yine tcp üzerinde server olarak kullanılması gibi işlemler yapılabilir.

Wemos D1 mini kartında ESP8266 kütüphanelerini eklemek için Arduino IDE'de **Araçlar > Kart > Kart Yöneticisi** ekranından ESP8266 aratıp, kurunuz.

CoAP

CoAp (Constrained Application Protocol – Kısıtlı Uygulama Protokolü), IETF (Internet Engineering Task Force – İnternet Mühendisliği Görev Gücü) tarafından tasarlanmış bir uygulama katmanı protokolüdür.

Adından da anlaşılacağı gibi birincil amacı kısıtlı kaynaklara sahip cihazlar üzerinde ve kısıtlı bant genişliğine sahip ağlarda çalışmaktır. CoAp, tasarımı basit tutmak için UDP üzerinde çalışır. Makineden makineye veri gönderimi sunar.

Coap Protokolü kullanarak haberleşebilmek için aşağıdaki linkte verilen kütüphaneyi Arduino uygulamamıza **Taslak > library ekle > . ZIP Kitaplığı Ekle** seçeneği ile eklemeliyiz.

<https://github.com/automote/ESP-CoAP>

Kütüphaneyi uygulama olarak açmak için ise **Yeni > Örnekler > Esp-CoAP Simple library > coapclient** seçeneğini tıklamalıyız. Açılan pencerede kodlara birkaç değişiklik yaparak kullanmaya başlayabiliriz.

coap.me IoT Platformu (CoAP test platformu)

Uygulamanın web üzerinden test edilebilmesi için IoT platformu olarak **coap.me** kullanacağız. **coap.me** IoT platformu CoAP uygulamalarımızın kolaylıkla test edilebilmesini sağlamaktadır. Get, Post, Put ve Delete metotlarını destekler. Sunduğu **örnek resourcelar** ile veri gönderme ve çekme işlemleri yapılabilir.

coap.me adresine girdikten sonra **ETSI CoAP#4 test client** altında ilk sıradaki **gönder** butonuna basarak ve ya <http://coap.me/test/coap://coap.me> adresine giderek belirli metotlar ve resource isimleriyle yapılabilen **test içeriklerini** görebilirsiniz.

ETSI test driver

- [TD_COAP_CORE_31](#) ping (should get a RST)
- Simple CON tests:
 - [TD_COAP_CORE_01](#) (GET) (also [TD_COAP_CORE_12](#), as long as tokens not yet touched, and [TD_COAP_CORE_15](#) if lossy)
 - [TD_COAP_CORE_02](#) (DELETE)
 - [TD_COAP_CORE_03](#) (PUT)
 - [TD_COAP_CORE_04](#) (POST) (also [TD_COAP_CORE_18](#))
- Simple NON tests:
 - [TD_COAP_CORE_05](#) (GET)
 - [TD_COAP_CORE_06](#) (DELETE)
 - [TD_COAP_CORE_07](#) (PUT)
 - [TD_COAP_CORE_08](#) (POST)
- Separate response:
 - [TD_COAP_CORE_09](#) (GET) (also [TD_COAP_CORE_16](#) if lossy)
 - [TD_COAP_CORE_17](#) (GET non)
- Simple CON tests with token: (reset token)
 - [TD_COAP_CORE_10](#) (GET w/token) [TD_COAP_CORE_10](#) (GET w/token, large)
 - Separate response: [TD_COAP_CORE_11](#) (GET w/token)
- More CON tests:
 - [TD_COAP_CORE_13](#) (GET w/path)
 - [TD_COAP_CORE_14](#) (GET w/query)
 - [TD_COAP_CORE_14](#) (GET w/query, alternate values)
 - [TD_COAP_CORE_19](#) (POST → query)
- Accept and content format tests:
 - [TD_COAP_CORE_20](#) (GET text/plain)
 - [TD_COAP_CORE_20](#) (GET application/xml)
 - [TD_COAP_CORE_20](#) (GET wrong format)
 - [TD_COAP_CORE_20](#) (GET no pref)
- complex tests:
 - old version: [TD_COAP_CORE_21](#) GET, then use revalidate link in the new window, then switch back to this window, then [PUT](#), then switch forward again and revalidate again (with /test)
 - [TD_COAP_CORE_21](#) GET, then use revalidate link in the new window, then switch back to this window, then [PUT](#), then switch forward again and revalidate again (with /validate)
 - [TD_COAP_CORE_22](#) GET, then copy the etag and replace the if-match in [this PUT](#) then reload
 - [TD_COAP_CORE_23](#) (PUT if-none-match) — test twice — (delete, if necessary)
- Link-Format:
 - [TD_COAP_LINK_01](#) (GET) (also [TD_COAP_LINK_09](#))
 - [TD_COAP_LINK_02](#) (GET ?rt=Type1), also [TD_COAP_LINK_05](#) when testing space-separated attributes
 - [TD_COAP_LINK_03](#) (GET ?rt=*)
 - [TD_COAP_LINK_04](#) (GET ?rt=Type2)
 - [TD_COAP_LINK_05](#) (GET ?if-if*)

Şekil 2: coap.me ETSI test driver

Yukarıdaki resimde görüldüğü üzere Get, Post, Put ve Delete metotları listelenmekte. Bu metotlardan birinin üzerine tıklayınca kullanmanız gerek [resource adını](#) görebilirsiniz.

Örneğin; Simple Con Test altındaki [TD_COAP_CORE_01](#) (GET)’e tıklarsak resource adının **test** olduğunu ve **welcome to the ETSI plugtest!** Gibi bir çıktı vereceğini görebiliriz.

Probing coap://coap.me/test

[coap://coap.me/test](#) [revalidate: [/etag=6668815659478006888/accept=0/coap://coap.me/test](#)] [token: 5371]
:etag 1. 6668815659478006888 = 0x5c8c648267bfb68
welcome to the ETSI plugtest! last change: 2017-09-28 08:50:23 UTC

Şekil 3: coap.me test metot içeriği

Uygulamanın Wemos D1 Mini Kodları

```
# include <ESP8266WiFi.h>
# include "coap_client.h"

//instance for coapclient
coapClient coap;

//WiFi connection info
const char* ssid = "*****"; //Kablosuz internet bağlantı adı
const char* password = "*****"; //Kablosuz internet bağlantı parolası

IPAddress ip(134,102,218,18); // coap.me nin ip adresi
int port = 5683; //CoAP kütüphanesinin varsayılan portu 5683 tür.

// coap client response callback
void callback_response(coapPacket &packet, IPAddress ip, int port);

// coap client response callback
void callback_response(coapPacket &packet, IPAddress ip, int port) {
    char p[packet.payloadlen + 1];
    memcpy(p, packet.payload, packet.payloadlen);
    p[packet.payloadlen] = NULL;

    //response from coap server
    if (packet.type == 3 && packet.code == 0)
    {
        Serial.println("ping ok");
    }

    Serial.println(p);
}

void setup()
{
    Serial.begin(115200);

    WiFi.begin(ssid, password);
    Serial.println(" ");

    // Connection info to WiFi network
    Serial.println();
    Serial.println();
    Serial.print("Connecting to ");
    Serial.println(ssid);
```

```
WiFi.begin(ssid, password);

while (WiFi.status() != WL_CONNECTED)
{
    //delay(500);
    yield();
    Serial.print(".");
}

Serial.println("");
Serial.println("WiFi connected");

// Print the IP address of client
Serial.println(WiFi.localIP());

// client response callback.
// this endpoint is single callback.
coap.response(callback_response);

// start coap client
coap.start();

//get request to server (arguments ip address of server,default port,resource(uri))
int msgid = coap.get(ip, port, "light");

//observe request (arguments ip address of server,default port,resource name,interger(0) )
//int msgid= coap.observe(ip,port,"light",0);

//reset observe cancel
//int msgid=coap.observecancel(ip,port,"resoucenam");
}

int i = 0;
char m[4];

void loop()
{
    bool state;

    sprintf(m, "%d", i);
    // Requests

    //get request
    int msgid = coap.get(ip,port,"hello");
```

```
//put request
//arguments server ip address,default port,resource name, payload,payloadlength
//int msgid =coap.put(ip,port,"resourcenam","0",strlen("0"));

//post request
//arguments server ip address,default port,resource name, payload,payloadlength
//int msgid = coap.post(ip, port, "test", m, 4);

//delete request
//int msgid = coap.delet(ip,port,"resourcenam");

//ping
//int msgid=coap.ping(ip,port);

// int msgid=coap.observe(ip,port,"obs",0);

state = coap.loop();
Serial.print("state=");
Serial.println(state);

if (state == true)
    i = i + 1;

Serial.print("i=");
Serial.println(i);

// if (i == 3)
//{
//Serial.println("cancel observe");
//coap.observeCancel(ip,port,"resourcenam");
//}

Serial.println(msgid);
delay(1000);
}
```

Kaynaklar:

1 — Mehmet Ali Ebleme, “Nesnelerin İnterneti Haberleşme Protokollerinin Başarım Analizi”, Yüksek Lisans Tez Çalışması, Danışman Doç. Dr. Cüneyt BAYILMIŞ 2017.