
lot Dashboard Documentation

electrocoder

14 Eki 2018

1	İçindekiler	1
1.1	Iot Nedir	1
1.2	Kanal Aç	3
1.3	Veri Gönder	4
1.4	Veri Al	20
1.5	Email Besleme	26
1.6	Iot Email Sms Alarm	26
1.7	Iot Mqtt Nedir?	28
1.8	Mqtt Protokolü Nasıldır?	28
1.9	MQTT Temp Test Client	29
1.10	MQTT Test Client Publisher Subscriber	29
1.11	Full Featured MQTT Client	31

1.1 Iot Nedir

1.1.1 Iot Nedir?

Nesnelerin interneti “internet of things” 1999 yılında Kevin Ashton tarafından kullanılan bir kavramdır ve teknolojideki gelişmeler ile birlikte bugünkü haline gelmiştir. RFID teknolojisi için üretilen bu kavram günümüzde tüm elektronik cihazlara uygulanmaktadır.

1.1.2 Iothook nedir?

Iothook internete bağlı nesneler (iot) arasında veri transferi yapan web servis ağı projesidir. Iothook ile Arduino, Raspberry Pi, Android, iOS, Windows Phone, Web Site, Banana Pi, Orange Pi, Beaglebone, ARM, Pic, Windows, Mac OS X, ve Linux tabanlı sistemleri birbirine bağlar.

1.1.3 Niçin Iothook?

- Iothook hızlıdır,
- Sınırsız kanal oluşturabilirsin,
- Sınırsız element ekleyebilirsin,
- Tüm cihazların ile kolayca veri gönderebilirsin (post),
- Tüm iot cihazlarından kolayca veri alabilirsin (get),
- Datalarını gerçek zamanlı takip edebilirsin,
- Dataların için gerçek zamanlı grafik oluşturabilirsin,

1.1.4 Iothook' un sunduğu avantajlar:

- Kanal oluşturma,
- Kanal elementi ekleme,
- Web api,
- Web Sorgu,
- Form api,
- Twit atma,
- SMS atma,
- E-posta,
- Grafik,
- 7/24 destek,

Iothook tüm cihazlarınız arasında kesintisiz veri aktarımı yapan, internete bağlı nesnelere kolayca ulaşabileceği iletişim protokollerini destekler.

Google developer chart apileri ile entegre olarak verileri gerçek zamanlı izleme olanağı sağlar.

1.2 Kanal Aç

Iothook kanal; internete bağlı nesnelere arasında veri iletimini sağlamak için oluşturulmuş kanca sistemidir. Kanal ile iot sistemleri veri paylaşımı yapılabilir, veri gönderim işlemleri tanımlanır.

Iothook web servislerini kullanabilmek için üye olunmalıdır. Üyelik seçenekleri 'Free', 'Student', 'Pro' ve 'Ultra' olmak üzere 4 kullanım planı vardır. Üye olmak için adrese gidiniz.

Üyelik adımından sonra yönetim paneli aracılığı ile 'Kanal Ekle' ekranına girilir.

1.2.1 Kanal Ekle

Kanal ekleme adımları şu şekildedir:

- **Form Metod: Http (Hyper Text Transfer Protocol) de veriler TCP/IP metodu ile iletilmektedir. Http protokolü üzerinden**
 - Post: Verilerin iot cihazda mesaj gövdesine yerleştirilerek gönderilme işlemidir.
 - Get: Verilerin iot cihaz ile sorgulanma ve cevap alınma talebidir.
 - Post/Get: Veri aktarımının iot nesnesi ile server arasında çift taraflı olacağını gösterir.
- Form enctype: "application/x-www-form-urlencoded" ile iot cihazından gönderilen karakterlerin gönderilmeden önce kodlanacağını belirtir. "multipart/form-data" ise verilerin içerisinde ASCII olmayan verilerin olduğunu dosya veya image formatında veri olduğunu belirtir.
- Aygıt türü: Iot cihazın türünü belirler. Arduino, Raspberry Pi... gibi
- Kanal adı: Verilerin toplanacağı kanalın adı.
- Web site: Veriler bir web sitesinde kullanılacak ise web site adresi girilmelidir.
- Email ile haber ver: Veri alındığında kayıtlı olan mail adresine mesaj gönderir. Aktif edilirse 15dk. da bir veri gönderilmesi gerekir.

- Verileri kaydet: Iot nesnesinden gelen verilerin iothook veritabanında saklanması için gereklidir.
- Resim: Kanal tanıtım resmi olarak kullanılır.
- Açıklama: Kanal bilgileri girilmelidir.
- Is public POST: Bu kanal genel kullanıma açık ve veri eklenmesine açıktır.
- Is public GET: Bu kanal genel kullanıma açık ve verilerin okunmasına izin verir.
- Yayındamı: Kanalı aktif et.

1.2.2 Element Ekle

Iot cihazınız için kanal oluşturduktan sonra kanalda bulunmasını istediğiniz veri alanlarını oluşturmalısınız. Bu alanlar veri almaya başlamak için eklenir. Element verilerine POST veta GET metodu ile ulaşabilirsiniz.

1.2.3 Element ayarları:

- Kanal adı: Elementin hangi kanala veri aktaracağı seçilir.
- Grafik türü: Toplanan verilerin çizileceği grafik türünü belirler.
- Element tipi: Verilerin depolanacağı alan tipini belirler. Grafik çizimi sadece “number” veri tipinde yapılmalıdır.
- Kanal adı: Verilerin toplanacağı kanalın adı.
- Element adı: Verilerin tutulacağı element adı.
- Yayındamı: Elementi aktif et.

1.3 Veri Gönder

Veri göndermek için öncelikle kanal ve element eklemeniz gerekir. Kanal oluşturulduğunda size özel “api_key” üretilerek belirlenen erişim metoduna göre (POST, GET, POST/GET) veri işlemi gerçekleştirilir.

Örneğin; Kanalımız ısı, ışık, hareket, bar ve nem değerlerini alan bir yapıda olsun. Kanal içerisinde bulunacak iot cihazlarımız bizlere bu dataları 15 sn. yede bir 100 kere göndersin.

Oluşturulan “API_KEY” Key Yöneticisi sayfasından görülebilir.

1.3.1 Python ile JSON Veri Gönderme

Python ile Json Post Örneği:

Bu örneği <http://bit.ly/2jI1FNQ> sayfasından indirebilirsiniz.

```
# -*- coding: utf-8 -*-  
  
"""  
    Python ile IotHook REST Api Testi  
  
    Kod çalıştırıldığında APIKEY ile doğrulama gerçekleştirilir.  
    Kanal api_key ile ilgili kanal ve element değerleri IotHook a post edilir.  
  
    Bu örnek IotHook servisine veri almak/göndermek için başlangic seviyesinde
```

(continues on next page)

(önceki sayfadan devam)

```
testlerin yapılmasını amaçlamaktadır.

20 Eylül 2017
Sahin MERSIN

Daha fazlası için

http://www.iothook.com
ve
https://github.com/electrocoder/iotHook

sitelerine gidiniz.

Sorular ve destek talepleri için
https://github.com/electrocoder/iotHook/issues
sayfasından veya Meşe Bilişim den yardım alabilirsiniz.

Yayın : http://mesebilisim.com

Licensed under the Apache License, Version 2.0 (the "License").
You may not use this file except in compliance with the License.
A copy of the License is located at

http://www.apache.org/licenses/

"""

import requests
import json
import random
import pprint
import time

headers = {'Content-type': 'application/json'}

url = 'https://iothook.com/api/latest/datas/'

for i in range(100):
    data={
        'api_key': '519ac5d4-95a5-116e185a343eac447b', # demo hesap api_key
        'value_1': i*1,
    }

    data_json = json.dumps(data)

    response = requests.post(url, data=data_json, headers=headers)
    pprint.pprint(response.json())
    time.sleep(15)
```

1.3.2 Python GET Metodu ile Veri Gönderme

IoThook Api v1.3 güncellemesi ile GET metodu ile veri göndermeye izin vermektedir.

Python ile Get metodu kullanarak veri gönderme Örneği:

```
# -*- coding: utf-8 -*-

"""
    Python ile IotHook REST Api Testi

    Kod çalıştırıldığında APIKEY ile gönderim gerçekleştirilir.
    Kanal api_key ile ilgili kanal ve element değerleri IotHook a GET metodu ile
    ↪gönderilir.

    Bu örnek IotHook servisine veri almak/göndermek için başlangıç seviyesinde
    testlerin yapılmasını amaçlamaktadır.

    11 Eylül 2017
    Sahin MERSIN

    Daha fazlası için

    http://www.iothook.com
    ve
    https://github.com/electrocoder/iotHook

    sitelerine gidiniz.

    Sorular ve destek talepleri için
    https://github.com/electrocoder/iotHook/issues
    sayfasından veya Meşe Bilişim den yardım alabilirsiniz.

    Yayın : http://mesebilisim.com

    Licensed under the Apache License, Version 2.0 (the "License").
    You may not use this file except in compliance with the License.
    A copy of the License is located at

    http://www.apache.org/licenses/

    """

import requests
import json
import random
import pprint
import time

headers = {'Content-type': 'application/json'}

API_KEY = '85c4ba5f-96ae-11841415634e983487e'

for i in range(10):
    url = 'https://iothook.com/api/latest/datas/update?api_key=' + API_KEY + '&value_
    ↪1=10&value_2=2&value_3=3'

    response = requests.get(url)
    data = response.json()
    print data
    time.sleep(15)
```

1.3.3 Arduino, ESP8266 POST Metodu ile Veri Gönderme

Bu örnekte Arduino Uno ya RX ve TX ile bağlanmış olan ESP8266 ile iothook a veri gönderme örneği verilmiştir. Örnekte 0-100 arasında rastgele sayı üretilerek iothook da <https://iothook.com/tr/channel/api/public/240> id numaralı cihaz için gönderim gerçekleştirilmiştir. Cihaz datalarını <https://iothook.com/tr/channel/api/public/240> linkinden gerçek zamanlı olarak takip edebilirsiniz.

Bu örneğe https://github.com/electrocoder/IoThook/tree/master/examples/IoThook/v1_3/arduino sayfasından ulaşabilirsiniz.

```
/*
  Arduino ile ESP8266 Wifi Modul Testi

  Kod Arduino ya yuklendiginde Arduino IDE nin Serial Monitor u
  ile ESP8266 arasinda haberlesme gozlenebilir.

  Arduino ile ESP8266 arasindaki iletisim Baud ayari
  115200 olmalidir.

  Arduino 0 ile 100 arasinda uretmis oldugu Random sayiyi iothook a gonderir.

  Bu cihaza ait datalar
  https://iothook.com/tr/channel/api/public/240
  adresinden gercek zamanli olarak izlenebilir.

  Bu ornek IoThook servisine veri gondermek icin baslangic ayarlarinin
  yapilmasini amaclamaktadir.

  24 Eylul 2017
  Sahin MERSIN

  Daha fazlasi icin

  http://www.iothook.com
  ve
  https://github.com/electrocoder/IoThook

  sitelerine gidiniz.
  Sorular ve destek talepleri icin
  https://github.com/electrocoder/IoThook/issues
  sayfasina gidiniz.

  Yayin ve sahiplik http://mesebilisim.com
*/

#include "SoftwareSerial.h"

String ssid = "WIFI_ID";
String password = "WIFI_PASSWORD";

SoftwareSerial esp(10, 11); // RX, TX

String data;

String server = "iothook.com";

String uri = "/api/latest/datas/";
```

(continues on next page)

```
void setup() {

  esp.begin(115200);

  Serial.begin(115200);

  Serial.println("Arduino ile ESP8266 Wifi Modul Testi");
  Serial.println("          www.IoThook.com          ");
  Serial.println("");

  reset();

  connectWifi();

}

void reset() {

  esp.println("AT+RST");

  delay(2000);

  if (esp.find("OK") ) Serial.println("Modul Reset yapildi");
  else Serial.println("Module Reset yapılamadi");

}

void connectWifi() {

  String cmd = "AT+CWJAP=\"" + ssid + "\",\"" + password + "\"";

  esp.println(cmd);

  delay(4000);

  if (esp.find("OK")) {

    Serial.println("ESP8266 Wifi ye baglandi");

  }

  else {

    connectWifi();

    Serial.println("ESP8266 Wifi ye baglanamadi!");

  }

}

void loop () {

  data = "{\"api_key\":\"b4301a9f-9854-11790bdf8d320140da\", \"value_1\":\" +
↳String(random(0, 100)) + "}";
```

(continues on next page)

(önceki sayfadan devam)

```
httppost();

delay(5000);
}

void httppost () {

  esp.println("AT+CIPSTART=\"TCP\", \"" + server + "\",80");

  if ( esp.find("OK")) {

    Serial.println("TCP baglanti hazir");

  }
  else
    Serial.println("TCP baglanti hatali");

  delay(3000);

  String postRequest =

    "POST " + uri + " HTTP/1.0\r\n" +

    "Host: " + server + "\r\n" +

    "Accept: *" + "/" + "*" + "\r\n" +

    "Content-Length: " + data.length() + "\r\n" +

    "Content-Type: application/x-www-form-urlencoded\r\n" +

    "\r\n" + data;

  String sendCmd = "AT+CIPSEND=";

  esp.print(sendCmd);

  esp.println(postRequest.length() );

  delay(1500);

  if (esp.find(">")) {
    Serial.println("Gonderiliyor...");
    esp.print(postRequest);

    if ( esp.find("SEND OK")) {
      Serial.println("Gonderildi :)");

      while (esp.available()) {

        String tmpResp = esp.readString();

        Serial.println(tmpResp);

      }
    }
  }
}
```

(continues on next page)

```
    }

    esp.println("AT+CIPCLOSE");

  }
  else
    Serial.println("Gonderilemedi :(");
}
else
  Serial.println("Gonderim hatasi! ESP hazir degil!");
}
```

1.3.4 Arduino, ESP8266 POST Metodu ile 2 Veri Gönderme

Bu örnekte Arduino Uno ya RX ve TX ile bağlanmış olan ESP8266 ile iothook a veri gönderme örneği verilmiştir. Örnekte 0-100 arasında rastgele 2 sayı üretilerek iothook da <https://iothook.com/tr/channel/api/public/192> id numaralı cihaz için gönderim gerçekleşmiştir. Cihaz datalarını <https://iothook.com/tr/channel/api/public/192> linkinden gerçek zamanlı olarak takip edebilirsiniz.

Bu örneğe https://github.com/electrocoder/IoThook/tree/master/examples/IoThook/v1_3/arduino sayfasından ulaşabilirsiniz.

```
/*
  Arduino ile ESP8266 Wifi Modul Testi

  Kod Arduino ya yuklendiginde Arduino IDE nin Serial Monitor u
  ile ESP8266 arasinda haberlesme gozlenebilir.

  Arduino ile ESP8266 arasindaki iletisim Baud ayari
  115200 olmalidir.

  Arduino 0 ile 100 arasinda uretmis oldugu 2 adet Random sayiyi iothook a gonderir.
  Bu sayilar 'data' deghiskeni icerisinde value_1 ve value_2 degherleridir. Bu degherler
  sensör olarak kullanilmaktadir. Sıcaklık ve Nem gibi sensörlerinizi bu alanlara
  gönderebilirsiniz.

  Bu cihaza ait datalar
  https://iothook.com/tr/channel/api/public/192
  adresinden gercek zamanli olarak izlenebilir.

  Bu ornek IoThook servisine veri gondermek icin baslangic ayarlarinin
  yapilmasini amaclamaktadir.

  24 Eylul 2017
  Sahin MERSIN

  Daha fazlasi icin

  http://www.iothook.com
  ve
  https://github.com/electrocoder/IoThook

  sitelerine gidiniz.
  Sorular ve destek talepleri icin
```

(continues on next page)

(önceki sayfadan devam)

```
https://github.com/electrocoder/IoThook/issues
sayfasina gidiniz.

Yayin ve sahiplik http://mesebilisim.com
*/

#include "SoftwareSerial.h"

String ssid = "WIFI_SSID";
String password = "WIFI_PASSWORD";

SoftwareSerial esp(10, 11); // RX, TX

String data;

String server = "iothook.com";

String uri = "/api/latest/datas/";

void setup() {

    esp.begin(115200);

    Serial.begin(115200);

    Serial.println("Arduino ile ESP8266 Wifi Modul Testi");
    Serial.println("                www.IoThook.com                ");
    Serial.println("");

    reset();

    connectWifi();

}

void reset() {

    esp.println("AT+RST");

    delay(2000);

    if (esp.find("OK") ) Serial.println("Modul Reset yapildi");
    else Serial.println("Module Reset yapılamadi");

}

void connectWifi() {

    String cmd = "AT+CWJAP=\"" + ssid + "\",\"" + password + "\"";

    esp.println(cmd);

    delay(4000);

    if (esp.find("OK")) {
```

(continues on next page)

```
Serial.println("ESP8266 Wifi ye baglandi");
}
else {
    connectWifi();
    Serial.println("ESP8266 Wifi ye baglanamadı!");
}
}

void loop () {

    data = "{\"api_key\": \"5180e8bd-95a5-11cc4ce6cfe4ee481c\", \"value_1\": \" +
↳String(random(0, 100)) + \", \"value_2\": \" + String(random(0, 100)) + \"}";

    httpPost();
    delay(8000);
}

void httpPost () {

    esp.println("AT+CIPSTART=\"TCP\", \"" + server + "\", 80");

    if ( esp.find("OK")) {

        Serial.println("TCP baglanti hazir");

    }
    else
        Serial.println("TCP baglanti hatali");

    delay(3000);

    String postRequest =

        "POST " + uri + " HTTP/1.0\r\n" +

        "Host: " + server + "\r\n" +

        "Accept: *" + "/" + "*" + "\r\n" +

        "Content-Length: " + data.length() + "\r\n" +

        "Content-Type: application/x-www-form-urlencoded\r\n" +

        "\r\n" + data;

    String sendCmd = "AT+CIPSEND=";
```

(continues on next page)

(önceki sayfadan devam)

```

esp.print(sendCmd);

esp.println(postRequest.length() );

delay(1500);

if (esp.find(">")) {
  Serial.println("Gonderiliyor...");
  esp.print(postRequest);

  if ( esp.find("SEND OK")) {
    Serial.println("Gonderildi :)");

    while (esp.available()) {

      String tmpResp = esp.readString();

      Serial.println(tmpResp);

    }

    esp.println("AT+CIPCLOSE");

  }
  else
    Serial.println("Gonderilemedi :(");
}
else
  Serial.println("Gonderim hatasi! ESP hazir degil!");
}

```

1.3.5 Arduino, ESP8266, Nodemcu GET Metodu ile Veri Gönderme

IoThook Api v1.3 güncellemesi ile GET metodu ile veri göndermeye izin vermektedir.

Bu örnekte Arduino, ESP8266 ve NodeMCU ile ile Get metodu kullanarak veri gönderme örneği verilmiştir:

```

// 18.09.2017
// nodemcu ile sicaklik ve nem takibi
// electrocoder@gmail.com
// sahin mersin
// v1

#include <ESP8266WiFi.h>           //https://github.com/esp8266/Arduino

//needed for library
#include <DNSServer.h>
#include <ESP8266WebServer.h>
#include <WiFiManager.h>         //https://github.com/tzapu/WiFiManager

//for LED status
#include <Ticker.h>

#include <ESP8266HTTPClient.h>

```

(continues on next page)

```
#include "DHT.h"

#define DHTPIN 4 // what digital pin we're connected to // D2 - GPIO4
#define DHTTYPE DHT11

DHT dht (DHTPIN, DHTTYPE);

Ticker ticker;

void tick()
{
  //toggle state
  int state = digitalRead(BUILTIN_LED); // get the current state of GPIO1 pin
  digitalWrite(BUILTIN_LED, !state); // set pin to the opposite state
}

//gets called when WiFiManager enters configuration mode
void configModeCallback (WiFiManager *myWiFiManager) {
  Serial.println("Entered config mode");
  Serial.println(WiFi.softAPIP());
  //if you used auto generated SSID, print it
  Serial.println(myWiFiManager->getConfigPortalSSID());
  //entered config mode, make led toggle faster
  ticker.attach(0.2, tick);
}

void setup() {
  // put your setup code here, to run once:
  Serial.begin(115200);

  //set led pin as output
  pinMode(BUILTIN_LED, OUTPUT);

  // start ticker with 0.5 because we start in AP mode and try to connect
  ticker.attach(0.6, tick);

  //WiFiManager
  //Local intialization. Once its business is done, there is no need to keep it around
  WiFiManager wifiManager;
  //reset settings - for testing
  //wifiManager.resetSettings();

  //set callback that gets called when connecting to previous WiFi fails, and enters
  ↪ Access Point mode
  wifiManager.setAPCallback(configModeCallback);

  //fetches ssid and pass and tries to connect
  //if it does not connect it starts an access point with the specified name
  //here "AutoConnectAP"
  //and goes into a blocking loop awaiting configuration
  if (!wifiManager.autoConnect("MeseIoT", "MeseIoT**")) {
    Serial.println("failed to connect and hit timeout");
    //reset and try again, or maybe put it to deep sleep
    ESP.reset();
    delay(1000);
  }
}
```

(continues on next page)

(önceki sayfadan devam)

```

}

//if you get here you have connected to the WiFi
Serial.println("connected...yeey :)");
ticker.detach();
//keep LED on
digitalWrite(BUILTIN_LED, LOW);

dht.begin();
}

void loop() {
  // Wait a few seconds between measurements.
  delay(2000);

  // Reading temperature or humidity takes about 250 milliseconds!
  // Sensor readings may also be up to 2 seconds 'old' (its a very slow sensor)
  float h = dht.readHumidity();
  // Read temperature as Celsius (the default)
  float t = dht.readTemperature();
  // Read temperature as Fahrenheit (isFahrenheit = true)
  float f = dht.readTemperature(true);

  // Check if any reads failed and exit early (to try again).
  if (isnan(h) || isnan(t) || isnan(f)) {
    Serial.println("Failed to read from DHT sensor!");
    return;
  }

  // Compute heat index in Fahrenheit (the default)
  float hif = dht.computeHeatIndex(f, h);
  // Compute heat index in Celsius (isFahreheit = false)
  float hic = dht.computeHeatIndex(t, h, false);

  Serial.print("Humidity: ");
  Serial.print(h);
  Serial.print(" %\t");
  Serial.print("Temperature: ");
  Serial.print(t);
  Serial.print(" *C ");
  Serial.print(f);
  Serial.print(" *F\t");
  Serial.print("Heat index: ");
  Serial.print(hic);
  Serial.print(" *C ");
  Serial.print(hif);
  Serial.println(" *F");

  ///
  HTTPClient http;

  // configure server and url
  http.begin("http://iothook.com/api/latest/datas/update/?api_key=095c75f-9c40-
→11e14084d3e&value_1=" + String(t) + "&value_2=" + String(h) + "");
  //http.begin("192.168.1.12", 80, "/test.html");

```

(continues on next page)

```

Serial.print("[HTTP] GET...\n");
// start connection and send HTTP header
int httpCode = http.GET();
if (httpCode > 0) {
  // HTTP header has been send and Server response header has been handled
  Serial.printf("[HTTP] GET... code: %d\n", httpCode);

  // file found at server
  if (httpCode == HTTP_CODE_OK) {

    // get lenght of document (is -1 when Server sends no Content-Length header)
    int len = http.getSize();

    // create buffer for read
    uint8_t buff[128] = { 0 };

    // get tcp stream
    WiFiClient * stream = http.getStreamPtr();

    // read all data from server
    while (http.connected() && (len > 0 || len == -1)) {
      // get available data size
      size_t size = stream->available();

      if (size) {
        // read up to 128 byte
        int c = stream->readBytes(buff, ((size > sizeof(buff)) ? sizeof(buff) :_
↪size));

        // write it to Serial
        Serial.write(buff, c);

        if (len > 0) {
          len -= c;
        }
      }
      delay(1);
    }

    Serial.println();
    Serial.print("[HTTP] connection closed or file end.\n");

  }
} else {
  Serial.printf("[HTTP] GET... failed, error: %s\n", http.errorToString(httpCode).c_
↪str());
}

http.end();
////
delay(13000);
}

```

1.3.6 GO GET Metodu ile Veri Gönderme

IoThook Api v1.3 güncellemesi ile GET metodu ile veri göndermeye izin vermektedir.

Bu örnekte GO dili ile Get metodu kullanarak veri gönderme örneği verilmiştir:

```
// 04 Eylül 2017
// Sahin MERSIN
// iohook.com
// postman kullanılarak olusturulmustur

package main

import (
    "fmt"
    "net/http"
    "io/ioutil"
)

func main() {

    url := "http://iohook.com/api/latest/datas/update?api_key=22dbb35d-9dd5-
↪113c0200ec44bb9&value_1=10&value_2=2&value_3=3"

    req, _ := http.NewRequest("GET", url, nil)

    req.Header.Add("cache-control", "no-cache")

    res, _ := http.DefaultClient.Do(req)

    defer res.Body.Close()
    body, _ := ioutil.ReadAll(res.Body)

    fmt.Println(res)
    fmt.Println(string(body))
}
```

1.3.7 PHP GET Metodu ile Veri Gönderme

IoThook Api v1.3 güncellemesi ile GET metodu ile veri göndermeye izin vermektedir.

Bu örnekte PHP dili ile Get metodu kullanarak veri gönderme örneği verilmiştir:

```
// 04 Eylül 2017
// Sahin MERSIN
// iohook.com
// postman kullanılarak olusturulmustur

<?php

$request = new HttpRequest();
$request->setUrl('http://iohook.com/api/latest/datas/update');
$request->setMethod(HTTP_METH_GET);

$request->setQueryData(array(
    'api_key' => '22dbb35d-9dd5-12300ec44bb9',
    'value_1' => '10',
    'value_2' => '2',
    'value_3' => '3'
```

(continues on next page)

```
));  
  
$request->setHeaders(array(  
    'postman-token' => '791ba738-7cb8-a920-0e5c-883cfb3e4498',  
    'cache-control' => 'no-cache'  
));  
  
try {  
    $response = $request->send();  
  
    echo $response->getBody();  
} catch (HttpException $ex) {  
    echo $ex;  
}
```

1.3.8 NodeJS GET Metodu ile Veri Gönderme

IoThook Api v1.3 güncellemesi ile GET metodu ile veri göndermeye izin vermektedir.

Bu örnekte NodeJS Native metodu kullanarak veri gönderme örneği verilmiştir:

```
// 04 Eylül 2017  
// Sahin MERSIN  
// iothook.com  
// postman kullanılarak olusturulmustur  
  
var http = require("http");  
  
var options = {  
    "method": "GET",  
    "hostname": "iothook.com",  
    "port": null,  
    "path": "/api/latest/datas/update?api_key=22dbb35d-9dd5-113200ec44bb9&value_1=10&  
↪value_2=2&value_3=3",  
    "headers": {  
        "cache-control": "no-cache",  
        "postman-token": "033da3c8-6196-cd49-f72d-1850a7d18500"  
    }  
};  
  
var req = http.request(options, function (res) {  
    var chunks = [];  
  
    res.on("data", function (chunk) {  
        chunks.push(chunk);  
    });  
  
    res.on("end", function () {  
        var body = Buffer.concat(chunks);  
        console.log(body.toString());  
    });  
});  
  
req.end();
```

1.3.9 Javascript JQuery Ajax GET Metodu ile Veri Gönderme

IoThook Api v1.3 güncellemesi ile GET metodu ile veri göndermeye izin vermektedir.

Bu örnekte NodeJS Native metodu kullanarak veri gönderme örneği verilmiştir:

```
// 04 Eylül 2017
// Sahin MERSIN
// iothook.com
// postman kullanılarak olusturulmustur

var settings = {
  "async": true,
  "crossDomain": true,
  "url": "http://iothook.com/api/latest/datas/update?api_key=22dbb35d-9dd5-
↪113c0342c44bb9&value_1=10&value_2=2&value_3=3",
  "method": "GET",
  "headers": {
    "cache-control": "no-cache",
  }
}

$.ajax(settings).done(function (response) {
  console.log(response);
});
```

1.3.10 Java Unirest GET Metodu ile Veri Gönderme

IoThook Api v1.3 güncellemesi ile GET metodu ile veri göndermeye izin vermektedir.

Bu örnekte NodeJS Native metodu kullanarak veri gönderme örneği verilmiştir:

```
// 04 Eylül 2017
// Sahin MERSIN
// iothook.com
// postman kullanılarak olusturulmustur

HttpResponse<String> response = Unirest.get("http://iothook.com/api/latest/datas/
↪update?api_key=22dbb35d-9dd5-113c0342c44bb9&value_1=10&value_2=2&value_3=3")
  .header("cache-control", "no-cache")
  .asString();
```

1.3.11 Java Unirest GET Metodu ile Veri Gönderme

IoThook Api v1.3 güncellemesi ile GET metodu ile veri göndermeye izin vermektedir.

Bu örnekte NodeJS Native metodu kullanarak veri gönderme örneği verilmiştir:

```
// 04 Eylül 2017
// Sahin MERSIN
// iothook.com
// postman kullanılarak olusturulmustur

OkHttpClient client = new OkHttpClient();
```

(continues on next page)

```
Request request = new Request.Builder()
    .url("http://iothook.com/api/latest/datas/update?api_key=22dbb35d-9dd5-
↪113c03420ec44bb9&value_1=10&value_2=2&value_3=3")
    .get()
    .addHeader("cache-control", "no-cache")
    .build();

Response response = client.newCall(request).execute();
```

1.4 Veri Al

IoT cihazından gönderilen ısı, nem, voltaj, ışık gibi değerleri iothook data merkezinden çekebilmek için öncelikle kanal üye kullanıcı adı ve giriş şifresine ihtiyaç vardır. IoT cihazından gelen veriler, Android, iOS gibi mobil cihazınızdan veya web sitenizden izlenebilir. Kanalınızın kullanımı genel kullanıma açık ise diğer kullanıcılar ile de bu verileri paylaşabilirsiniz.

Örneğin; Kanal adınız “Temperature sensor” olarak belirlenmiş ve kanal içerisinde bulunacak IoT cihazınızdan “temperature” ve “humidity” element verileri gönderiliyor olsun.

1.4.1 Python 2, Python 3 Json ile Veri Alma

Python Json ile Get Örneği:

Bu örneği <http://bit.ly/2jI1FNQ> sayfasından inceleyebilirsiniz.

```
# -*- coding: utf-8 -*-

"""
Python 2, Python 3 ile IoThook REST Api Testi

Kod çalıştırıldığında 'data' değişkenine verilen 'all' değişkeni ile
auth sahipliğindeki tüm veriler alınır.

Bu örnek IoThook servisine veri almak/gondermek için baslangic seviyesinde
testlerin yapılmasını amaçlamaktadır.

10 Mayıs 2017
Sahin MERSIN

Daha fazlası için

http://www.iothook.com
ve
https://github.com/electrocoder/IoThook

sitelerine gidiniz.

Sorular ve destek talepleri için
https://github.com/electrocoder/IoThook/issues
sayfasından veya Meşe Bilişim den yardım alabilirsiniz.

Yayın : http://mesebilisim.com
```

(continues on next page)

(önceki sayfadan devam)

```

Licensed under the Apache License, Version 2.0 (the "License").
You may not use this file except in compliance with the License.
A copy of the License is located at

http://www.apache.org/licenses/

"""

import requests

API_KEY = '511b0173-95a5-11c814c2297e434c06'

url = 'https://iothook.com/api/latest/datas/?api_key=' + API_KEY

response = requests.get(url)
data = response.json()
print data

```

1.4.2 Python 2, Python 3 Json ile İlk Veriyi Alma

Python İlk Veriyi Alma, Json ile Get Örneği:

Bu örneği <http://bit.ly/2jIIFNQ> sayfasından inceleyebilirsiniz.

```

# -*- coding: utf-8 -*-

"""
Python 2 ile IotHook REST Api Testi

Kod çalıştırıldığında 'data' değişkenine verilen 'first' değişkeni ile
auth sahipliğindeki ilk veri alınır. 'channel' değişkeni IotHook dashboard
Kanal oluşturma sırasında otomatik verilen id numarasıdır.

Bu örnek IotHook servisine veri almak/göndermek için başlangıç seviyesinde
testlerin yapılmasını amaçlamaktadır.

10 Mayıs 2017
Sahin MERSIN

Daha fazlası için

http://www.iothook.com
ve
https://github.com/electrocoder/IotHook

sitelerine gidiniz.

Sorular ve destek talepleri için
https://github.com/electrocoder/IotHook/issues
sayfasından veya Meşe Bilişim den yardım alabilirsiniz.

Yayın : http://mesebilisim.com

Licensed under the Apache License, Version 2.0 (the "License").
You may not use this file except in compliance with the License.

```

(continues on next page)

(önceki sayfadan devam)

```
A copy of the License is located at
http://www.apache.org/licenses/

"""

import requests

API_KEY = '511b0173-95a5-11c814c2297e434c06'

url = 'https://iothook.com/api/latest/datas/?data=first&api_key=' + API_KEY

response = requests.get(url)
data = response.json()
print data
```

1.4.3 Python 2, Python 3 Json ile Son Veriyi Alma

Python Son Veriyi Alma, Json ile Get Örneği:

Bu örneği <http://bit.ly/2jI1FNQ> sayfasından inceleyebilirsiniz.

```
# -*- coding: utf-8 -*-

"""
Python 2 ile Iothook REST Api Testi

Kod çalıştırıldığında 'data' değişkenine verilen 'last' değişkeni ile
auth sahipliğindeki en son veri alınır. 'channel' değişkeni Iothook dashboard
Kanal oluşturma sırasında otomatik verilen id numarasıdır.

Bu örnek IotHook servisine veri almak/gondermek için baslangic seviyesinde
testlerin yapılmasını amaçlamaktadır.

10 Mayıs 2017
Sahin MERSIN

Daha fazlası için

http://www.iothook.com
ve
https://github.com/electrocoder/iotHook

sitelerine gidiniz.

Sorular ve destek talepleri için
https://github.com/electrocoder/iotHook/issues
sayfasından veya Meşe Bilişim den yardım alabilirsiniz.

Yayın : http://mesebilisim.com

Licensed under the Apache License, Version 2.0 (the "License").
You may not use this file except in compliance with the License.
A copy of the License is located at
```

(continues on next page)

(önceki sayfadan devam)

```
http://www.apache.org/licenses/  
  
"""  
  
import requests  
  
API_KEY = '516b0073-95a5-11c814c2597e434c06'  
  
url = 'https://iothook.com/api/latest/datas/?data=last&api_key=' + API_KEY  
  
response = requests.get(url)  
data = response.json()  
print data
```

1.4.4 Python 2, Python 3 Json ile Veriye Ait Detay Alma

Python veriye ait detay alma örneği:

Bu örneği <http://bit.ly/2jI1FNQ> sayfasından inceleyebilirsiniz.

```
# -*- coding: utf-8 -*-  
  
"""  
    Python 2, 3 ile IoThook REST Api Testi  
  
    Kod çalıştırıldığında datas url yapısına parametre olarak verilen  
    değer Kanal ve Element içerisinde tanımlı datanın ayrıntılarını getirir.  
  
    Bu örnek IotHook servisine veri almak/gondermek için baslangic seviyesinde  
    testlerin yapılmasını amaçlamaktadır.  
  
    10 Mayıs 2017  
    Sahin MERSIN  
  
    Daha fazlası için  
  
    http://www.iothook.com  
    ve  
    https://github.com/electrocoder/iotHook  
  
    sitelerine gidiniz.  
  
    Sorular ve destek talepleri için  
    https://github.com/electrocoder/iotHook/issues  
    sayfasından veya Meşe Bilişim den yardım alabilirsiniz.  
  
    Yayın : http://mesebilisim.com  
  
    Licensed under the Apache License, Version 2.0 (the "License").  
    You may not use this file except in compliance with the License.  
    A copy of the License is located at  
  
    http://www.apache.org/licenses/
```

(continues on next page)

```
"""  
  
import requests  
  
API_KEY = '516b0073-95a5-11c814c2297e434c06'  
  
url = 'https://iothook.com/api/latest/datas/4545/?api_key=' + API_KEY  
  
response = requests.get(url)  
data = response.json()  
print data
```

1.4.5 C# ile Json Formatında Veri Alma

IoThook kullanıcısının tüm kanallarına ait veriyi alabilmesi için GET metodu ile '?data=last' değişkeninin gönderilmesi gereklidir. Aynı örnekdeki data değişkeninin alabileceği değerler:

- ?data=all : Kullanıcının tüm datalarını getir
- ?data=first : Kullanıcının ilk datasını getir
- ?data=last : Kullanıcının son datasını getir

CSharp örneğinde *HttpRequest* metodu kullanılmıştır.

Bu örneği <http://bit.ly/2jI1FNQ> Github sayfasından inceleyebilirsiniz.

```
/*  
  C# ile IoThook REST Api Testi  
  
  Bu örnek ile CSharp ve Request metodu ile kullanıcının datalarının get metodu ile  
  ↪ alınması gerçekleştirilmiştir.  
  'authorization' ile kullanıcı adı ve parola değeri verilmelidir.  
  
  Bu örnek IoHook servisine veri almak/gondermek için baslangic seviyesinde  
  testlerin yapılmasını amaçlamaktadır.  
  
  29 Temmuz 2017  
  Sahin MERSIN  
  
  Daha fazlası için  
  
  http://www.iothook.com  
  ve  
  https://github.com/electrocoder/iotHook  
  
  sitelerine gidiniz.  
  
  Sorular ve destek talepleri için  
  https://github.com/electrocoder/iotHook/issues  
  sayfasından veya Meşe Bilişim den yardım alabilirsiniz.  
  
  Yayın : http://mesebilisim.com  
  
  Licensed under the Apache License, Version 2.0 (the "License").  
  You may not use this file except in compliance with the License.  
  A copy of the License is located at
```

(continues on next page)

```
    http://www.apache.org/licenses/

*/

using System;
using System.IO;
using System.Net;

namespace Iothook
{
    class Program
    {
        static void Main(string[] args)
        {
            string url = "";
            url = "https://iothook.com/api/v1.2/datas/?data=all"; // for all data

            var webRequest = (HttpWebRequest)WebRequest.Create(url);

            webRequest.Method = "GET";
            webRequest.ContentType = "application/json";
            webRequest.UserAgent = "Mozilla/5.0 (Windows NT 5.1; rv:28.0) Gecko/
↪20100101 Firefox/28.0";
            webRequest.ContentLength = 0;

            string authorization = "USERNAME" + ":" + "PASSWORD";
            byte[] binaryAuthorization = System.Text.Encoding.UTF8.
↪GetBytes(authorization);
            authorization = Convert.ToBase64String(binaryAuthorization);
            authorization = "Basic " + authorization;
            webRequest.Headers.Add("AUTHORIZATION", authorization);

            var webResponse = (HttpWebResponse)webRequest.GetResponse();

            if (webResponse.StatusCode != HttpStatusCode.OK)
                Console.WriteLine(webResponse.Headers.ToString());

            using (StreamReader reader = new StreamReader(webResponse.
↪GetResponseStream()))
            {
                Console.WriteLine(reader.ReadToEnd());
                reader.Close();
                webRequest.Abort();
            }

            Console.ReadLine();
        }
    }
}
```

1.5 Email Besleme

1.5.1 IoT cihazlardan email alma

IoT cihazlardan email almak için 'Kanal Ekle' menüsünden kanal oluşturulurken 'Email feed' seçeneğinin aktif edilmesi gerekir. Kanal oluşturulduktan sonrada email alma seçeneği değiştirilebilir. Güncelleme için 'Kanal Liste' menüsünden 'Düzenle' seçeneği altından yapılabilir.

1.5.2 Email besleme planı

IoT cihazınızdan veri geldiğinde email ile besleme almak için 'STUDENT', 'PRO', veya 'ULTRA' planlardan birisini tercih etmelisiniz.

Plan değişikliği için Ödeme sayfasından size uygun planı seçerek email besleme alabilirsiniz.

1.5.3 Email besleme süresi

- Free plan email besleme süresi: 8 email, ~180 dakika aralık ile
- Student plan email besleme süresi: 10 email, ~144 dakika aralık ile
- Pro plan email besleme süresi: 15 email, ~96 dakika aralık ile
- Ultra plan email besleme süresi: 100 email, ~14 dakika aralık ile

1.6 IoT Email Sms Alarm

1.6.1 Alarm nedir?

IoT Kanal altında oluşturulan Elementlere alarm değeri kurma işlemidir. Alarm değeri kurularak iot cihazdan her veri alındığında operatör ile işlem yapılarak sonuca göre alarm üretilir. Üretilen alarm abonelik tipine göre bir günde en fazla atılabilecek email ve sms planına göre belirlenir.

1.6.2 Operatörler

İşlem operatörleri aşağıdaki gibidir:

- <: Küçüktür operatörü. $a < b$. gelen_deger < alarm_degeri. IoT cihazdan gönderilen değer ile alarm değerini karşılaştırır. İşlem sonucu doğru (True) ise alarm üretilir.
- <=: Küçük eşittir operatörü. $a \leq b$. gelen_deger <= alarm_degeri. IoT cihazdan gönderilen değer ile alarm değerini karşılaştırır. İşlem sonucu doğru (True) ise alarm üretilir.
- ==: Eşittir operatörü. $a == b$. gelen_deger == alarm_degeri. IoT cihazdan gönderilen değer ile alarm değerini karşılaştırır. İşlem sonucu doğru (True) ise alarm üretilir.
- !=: Eşit değil operatörü. $a \neq b$. gelen_deger != alarm_degeri. IoT cihazdan gönderilen değer ile alarm değerini karşılaştırır. İşlem sonucu doğru (True) ise alarm üretilir.
- >=: Büyük eşit operatörü. $a \geq b$. gelen_deger >= alarm_degeri. IoT cihazdan gönderilen değer ile alarm değerini karşılaştırır. İşlem sonucu doğru (True) ise alarm üretilir.

- >: Büyüktür operatörü. $a > b$. gelen_deger > alarm_degeri. Iot cihazdan gönderilen değer ile alarm değerini karşılaştırır. İşlem sonucu doğru (True) ise alarm üretilir.

Örnek operatör işlemleri:

- < Küçüktür operatörü python örnek:

```
a = 5
b = 7
a < b
True
```

```
a = 9
b = 7
a < b
False
```

- <= Küçük eşittir operatörü python örnek :

```
a = 5
b = 7
a <= b
True
```

```
a = 7
b = 7
a <= b
True
```

- == Eşittir operatörü python örnek :

```
a = 5
b = 7
a == b
False
```

```
a = 7
b = 7
a == b
True
```

- != Eşit değil operatörü python örnek :

```
a = 5
b = 7
a != b
True
```

```
a = 7
b = 7
a != b
False
```

- >= Büyük eşit operatörü python örnek :

```
a = 5
b = 7
```

(continues on next page)

(önceki sayfadan devam)

```
a >= b
False
```

```
a = 7
b = 7
a >= b
True
```

- > Büyüktür operatörü python örnek :

```
a = 5
b = 7
a > b
False
```

```
a = 9
b = 7
a > b
True
```

1.6.3 Email Alarm nedir?

IoT Kanal/Element alarm işlemi uygulandığında gelen değer ile alarm değeri mantıksal operatör işlem sonucuna göre kanal yöneticisine email gönderilir. Kayıt olur iken kullanılan email adresi geçerli email adresidir. Günlük (24 saat) email gönderilme sayısı üyelik planına göre hesaplanır.

1.6.4 SMS Alarm nedir?

IoT Kanal/Element alarm işlemi uygulandığında gelen değer ile alarm değeri mantıksal operatör işlem sonucuna göre kanal yöneticisine sms gönderilir. Sms mesaj gönderilebilmesi için kanal yöneticisinin cep telefonunun onaylı olması gerekir. Günlük (24 saat) sms gönderilme sayısı üyelik planına göre hesaplanır.

1.7 Iot Mqtt Nedir?

MQTT Message Queuing Telemetry Transport kelimelerinin baş harfleri ile tanıdığımız bu teknoloji mesajın karşı tarafa ulaştırılması için kullanılan haberleşme protokolüdür. Haberleşme için mesaj yayınlayan, mesaja abone olan ve mesaj trafiğini kontrol eden yöneticiden oluşmaktadır.

Mesaj trafiğini kontrol eden yöneticiye **BROKER**, mesaj yayınına **PUBLISH** ve aboneye **SUBSCRIBE** denir. Mesaj alışverişi publisher dan subscriber lara doğru yani yayıncılardan abonelere doğru olmaktadır.

1.8 Mqtt Protokolü Nasıldır?

MQTT de asenkron haberleşme protokolü kullanılmaktadır. Mesaj yayıncıları ve mesaj alıcılar arasında eşzamanlı olarak veri taşınmaktadır. Diğer haberleşme yapılarına göre basit oluşu ve minimum kaynak tüketmesi sebebiyle “machine-to-machine” (M2M) makineden makineye veri iletiminde ve (IOT) “Internet of Things” İnternete bağlı nesnelerin mesajlaşmasında tercih edilmektedir.



Şekil 2: MQTT

1.9 MQTT Temp Test Client

Iot MQTT Temp Test Client Mosquitto Brokera websocket ile gelen temp/random başlığını dinler.

Bu örnek 'test.mosquitto.org' sitesinden alınmıştır. MQTT Temp örneğinin orjinal kaynağına 'desert-home.com' adresinden ve Github üzerinden ulaşabilirsiniz. MQTT Brokera nasıl mesaj gönderebilirim?

Iothook MQTT brokerına veri göndermek için "temp/random" başlığı gönderilmelidir. Gönderilen değer -20 ile +50 aralığında kayar noktalı (float) veya tamsayı (int) formatında olmalıdır.

MQTT Broker kimlik doğrulama ile çalışır.

Örnek -> Mesaj yayınlama: mosquitto_pub -h iothook.com -p 1883 -t "temp/random" -m "6" -u pub_user -P iothook_pub_user

Örnek -> Mesaja abone olma: mosquitto_sub -h iothook.com -p 1883 -t "temp/random" -u pub_user -P iothook_pub_user

MQTT Brokera için Test Kullanıcıları:

Kullanıcı Adı : pub_user Şifre : iothook_pub_user

Kullanıcı Adı : sub_user Şifre : iothook_sub_user

Kullanıcı Adı : pub_client Şifre : iothook_pub_client

Kullanıcı Adı : sub_client Şifre : iothook_sub_client

Temp client sayfasına <https://iothook.com/mqtt/mqtt-temp-test/> adresinden ulaşabilirsiniz.

1.10 MQTT Test Client Publisher Subscriber

MQTT Brokera Mesaj Gönderme ve Abone Olma Mosquitto Brokera websocket ile gelen temp/random başlığını dinler.

IoT MQTT Temp Test Client

Mosquitto Brokera websocket ile gelen temp/random başlığını dinler



Son gelen mesajlar

Henüz değil!

Şekil 3: MQTT Temp Test

Iothook MQTT brokerına veri göndermek için “temp/random” başlığı gönderilmelidir. Gönderilen veri kayar noktalı (float), tamsayı (int) veya string (text) formatında olabilir.

MQTT Broker kimlik doğrulama ile çalışır.

Örnek -> Mesaj yayınlama: `mosquitto_pub -h iothook.com -p 1883 -t “temp/random” -m “6” -u pub_user -P iot-hook_pub_user`

Örnek -> Mesaja abone olma: `mosquitto_sub -h iothook.com -p 1883 -t “temp/random” -u pub_user -P iot-hook_pub_user`

MQTT Brokera için Test Kullanıcıları:

Kullanıcı Adı : `pub_user` Şifre : `iothook_pub_user`

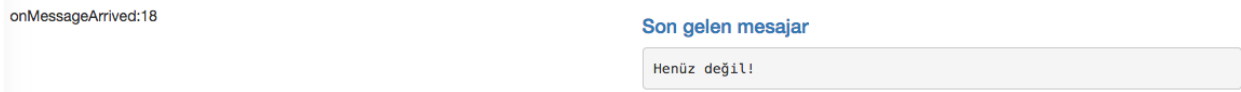
Kullanıcı Adı : `sub_user` Şifre : `iothook_sub_user`

Kullanıcı Adı : `pub_client` Şifre : `iothook_pub_client`

Kullanıcı Adı : `sub_client` Şifre : `iothook_sub_client`

MQTT Brokera Mesaj Gönderme ve Abone Olma

Mosquitto Brokera websocket ile gelen temp/random başlığını dinler



Şekil 4: *MQTT Test Publisher Subscriber*

Temp client sayfasına <https://iothook.com/mqtt/mqtt-temp-test-pub-sub/> adresinden ulaşabilirsiniz.

1.11 Full Featured MQTT Client

MQTT Brokera Mesaj Gönderme ve Alma

MQTT Broker kimlik doğrulama ile çalışır.

Örnek -> Mesaj yayınlama: `mosquitto_pub -h iothook.com -p 1883 -t “temp/random” -m “6” -u pub_user -P iot-hook_pub_user`

Örnek -> Mesaja abone olma: `mosquitto_sub -h iothook.com -p 1883 -t “temp/random” -u pub_user -P iot-hook_pub_user`

MQTT Brokera için Test Kullanıcıları:

Kullanıcı Adı : `pub_user` Şifre : `iothook_pub_user`

Kullanıcı Adı : `sub_user` Şifre : `iothook_sub_user`

Kullanıcı Adı : `pub_client` Şifre : `iothook_pub_client`

Kullanıcı Adı : `sub_client` Şifre : `iothook_sub_client`

Websockets Client Uygulaması Apache License Version 2.0 ile HiveMQ <http://www.hivemq.com/> tarafından dağıtılmaktadır. Lisans hakkında ayrıca bilgi alınır.

Temp client sayfasına <https://iothook.com/mqtt/full-featured-mqtt-client/> adresinden ulaşabilirsiniz.