MQTT 与移动应用打造物联网系统：Ionic + ESP8266

Phodal Huang

October 24, 2017

目录

步骤 0：系统架构 3
步骤 1：材料准备 4
硬件 4
软件 4
步骤 2：组装电路 4
步骤 3：准备软件 5
步骤 4：深入代码 6
REST API Server + MQTT Client 6
ESP8266 Arduino 代码 8
Ionic 移动应用 8
步骤 5：结论 9

原文链接：https://www.wandianshenme.com/play/esp8266-mqtt-ionic-build-iot-application

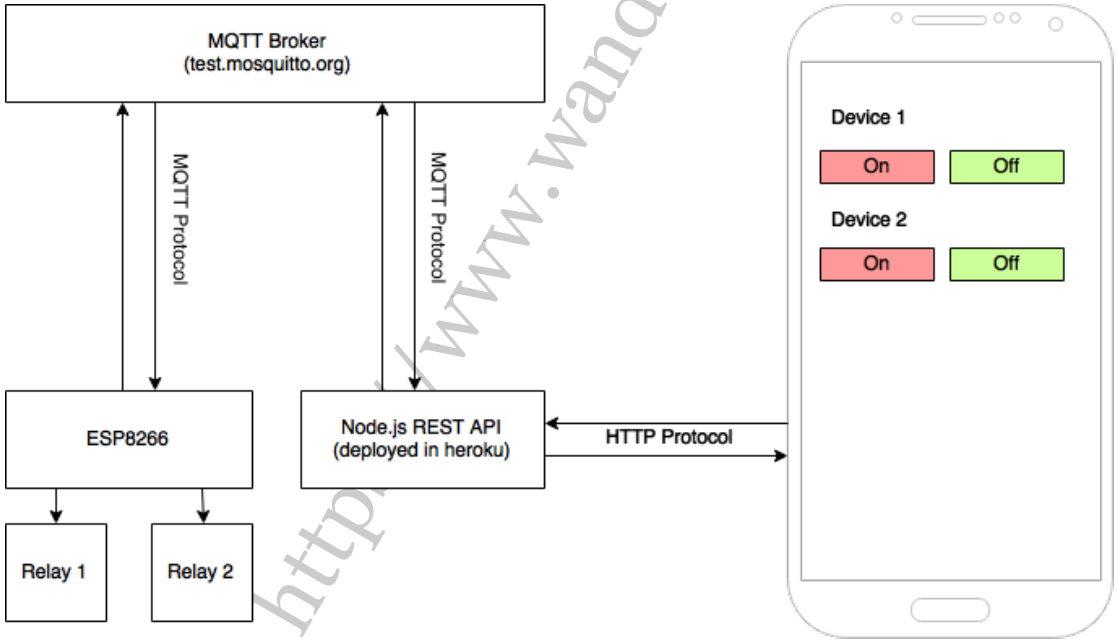
本文的目标是，创建一个非常简单的低级 IoT 应用程序。文章更侧重于如何从头开始构建它，而非需使用现有的 IoT 物联网平台，如 AWS IoT，IBM Bluemix，Samsung Artik 等服务。

该系统基于一个非常便宜的、只要 $3 的 WiFi 模块 - ESP8266，它通过 Mosquitto 的免费在线 MQTT 代理（test.mosquitto.org）连接到 Node.js REST API 服务器。Android Mobile 应用程序的是，基于 Angular.js 的 Ionic Framework 构建。

REST API 服务器托管在云服务器（heroku）上，可以通过互联网在线控制中继。

步骤 0：系统架构

下图显示了系统的连接流程。ESP8266 Wifi 模块通过 WiFi 连接到互联网，以便让的移动设备工作。



本文介绍了，如何使用自己的 REST API 服务器，为 ESP8266 及您的移动应用程序，来通过 MQTT 进行通信。您可以使用部署在 heroku（http://esp8266-relays.herokuapp.com ）中的现有 REST API 服务器，也可以部署自己的 REST 服务器。如果您不想使用“ test.mosquitto.org ”作为您的 MQTT 代理，您可以通过下载并安装 Mosquitto，来进行自己的操作。

步骤 **1**：材料准备

硬件

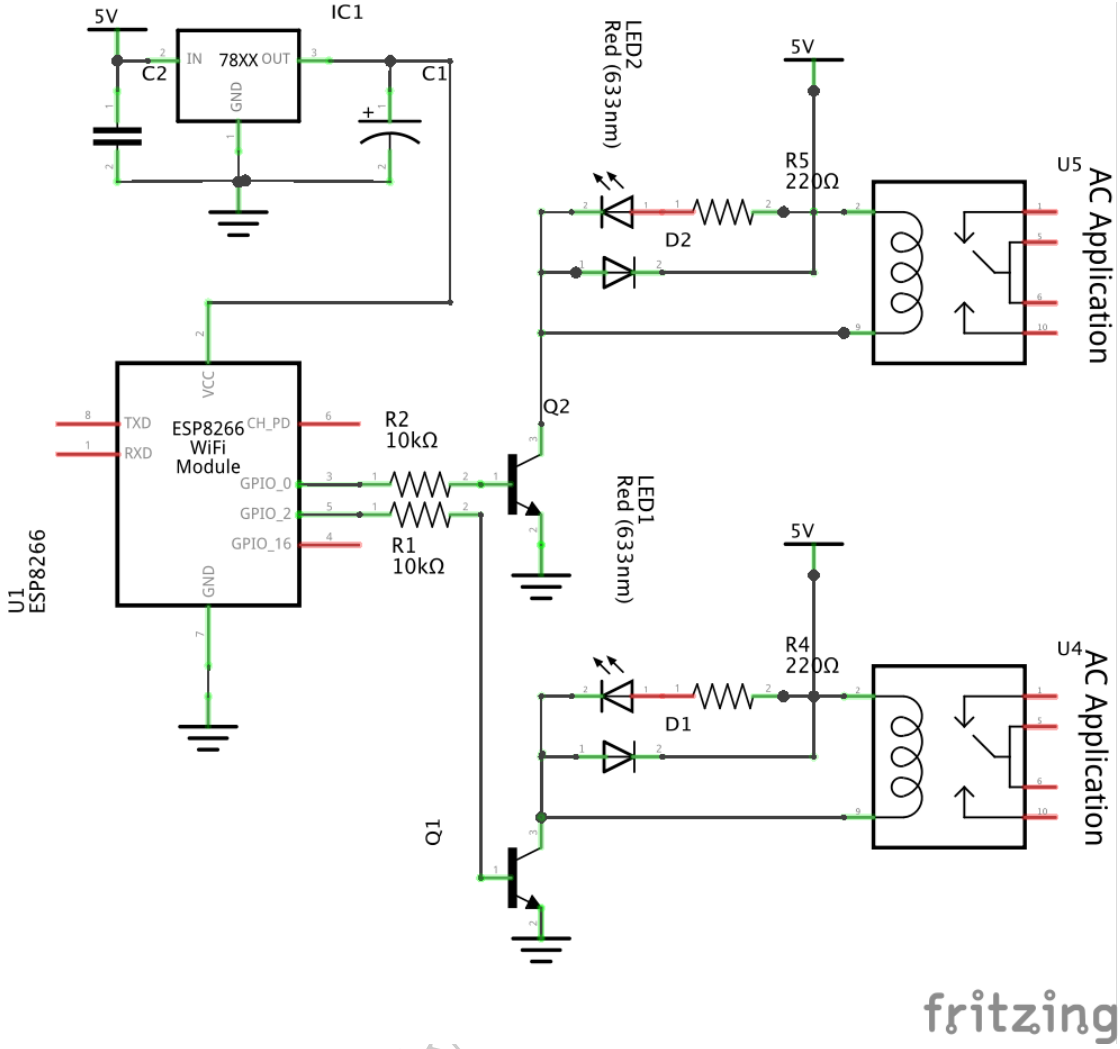
- ESP8266
- 5V 电源（至少 1A）
- 3.3V 稳压器
- 5V 继电器
- 2N2222A 晶体管
- 电阻器
- 电容器
- 1N4108 二极管
- 发光二极管

软件

- Node.js
- Ionic Framework
- Arduino IDE

步骤 **2**：组装电路

ESP8266 芯片模块不是使用 5V 电压，因此您必须添加 3.3V 稳压器。我在这个项目中使用了 KIA78R25API。它是一个 1A 4 端口的低电压稳压器，内置 ON/OFF 控制端口。晶体管用于将 3.3V 信号从 ESP8266 的升压到 5V，以便能够控制继电器。



步骤 **3**：准备软件

1). 从 Node.js 的官网 https://nodejs.org/download/ 下载 Node.js，并依据你的系统来安装 Node.js。

(如果你使用的是 Ubuntu，可以通过这个教程来安装 Node.js：https://elizarpepino.com/install-latest-nodejs-on-ubuntu-box/)

要测试 Node.js 是否已正确安装，您应该可以运行这些命令并显示当前版本。

```
1 node -v
2 npm -v
```

2). 从 Ionic 的官网 http://ionicframework.com/getting-started/ 安装 Ionic 框架。Ionic 框架依赖于 Node.js，因此您必须先安装 Node.js。

3). 设置用于 ESP8266 的 Arduino IDE，可以参见：《智能家居声控：Amazon Echo

+ NodeMCU（ESP8266 模拟 Wemo）控制 LED»中的『步骤 2：安装 Arduino ESP8266』。

步骤 4：深入代码

在这里一共有三个代码模块是分别是：

- MQTT Client + REST API Server，https://github.com/vynci/MQTT-REST-API
- ESP8266 Arduino，https://github.com/vynci/esp8266-relay
- Ionic Mobile App Framework，https://github.com/vynci/esp8266-ionic

你可以从这些 GitHub 上获得代码。

REST API Server + MQTT Client

这个第一个模块是部署到 heroku 的服务器（http://esp8266-relays.herokuapp.com）。您可以在本地创建自己的服务器，也可以在云端部署。此节点应用程序需要两个模块，Hapi.js 是一个 REST API 框架，而 MQTT.js 是一个 mqtt 客户端。

该块初始化 REST API 服务器，并建立与 MQTT 代理服务器的连接。

```
var Hapi = require('hapi');
var mqtt = require('mqtt');

var server = new Hapi.Server();
var port = Number(process.env.PORT || 4444);

server.connection({ port: port, routes: { cors: true } });

var client  = mqtt.connect('mqtt://test.mosquitto.org:1883');
```

此功能用于向 MQTT 代理发布消息：

```
var mqttPublish = function(topic, msg){
  client.publish(topic, msg, function() {
    console.log('msg sent: ' + msg);
  });
}
```

此块代码用于创建 `/device/control` 路由的 POST 方法，如果此路由被调用，它将执行 `mqttPublish` 函数。`deviceInfo` 变量包含了 ESP8266 转换的消息，其中的

dev1-on 用于打开继电器 1，dev1-off 用于关闭，dev2-on 和 dev2-off 也是类似的。

```
server.route([
  {
    method: 'POST',
    path: '/device/control',
    handler: function (request, reply) {
      var deviceInfo = 'dev' + request.payload.deviceNum + '-' +
          request.payload.command;
      reply(deviceInfo);
      mqttPublish('device/control', deviceInfo, {
        'qos' : 2
      });
    }
  }
]);

server.start();
```

mqttPublish 函数中的中的第一个参数：device / control，是我们的 ESP8266 监听的主题，那么第二个参数是消息 deviceInfo 则是被传递到的设置信息。qos 是指服务质量，即发送方和接收方之间有一个关于传递消息保证的消息的协议级别。要了解更多信息，可以在这里阅读：MQTT Essentials Part 6: Quality of Service 0, 1 & 2。

从 GitHub（https://github.com/vynci/MQTT-REST-API）下载 Node.js 应用程序后，可以运行它：

```
npm install
node index.js
```

它将创建一个 REST API 服务器，移动应用程序可以连接到这个服务器。它也建立了与 MQTT 代理服务器（test.mosquitto.org ）的连接。

如果你想在 heroku 中部署自己的服务器，可以从这个教程里：Getting Started on Heroku with Node.js 了解。

ESP8266 Arduino 代码

该代码块通过指定 SSID 和密码，将 ESP8266 连接到 WiFi 网络。该代码块还建立了与 MQTT 代理的连接。

```
#include <PubSubClient.h>
#include <ESP8266WiFi.h>

const char* ssid = "your-wifi-ssid";
const char* password = "your-wifi-passwd";

char* topic = "device/control";
char* server = "85.119.83.194"; // IP of test.mosquitto.org

WiFiClient wifiClient;
PubSubClient client(server, 1883, callback, wifiClient);
.....
```

要使用 Arduino IDE 将该代码，烧录到 ESP8266 中，可以按照我之前的文章中的步骤进行操作。

Ionic 移动应用

下面的文件（services.js）在这里连接到 REST API 服务器，执行 POST 方法：

```
angular.module('starter.services', [])

.factory('Devices', function($http) {

  var ipServer = 'http://esp8266-relays.herokuapp.com';

  return {
    deviceCommand: function(data) {
      console.log(data);
      return $http.post(ipServer + '/device/control', data);
    }
  };
});
```

您可以通过在项目的根文件夹中执行下面命令，然后通过浏览器运行此移动应用程序：

```
ionic serve
```

如果要构建并安装到您的移动设备，可以运行：

```
ionic platform add android
ionic run android
```

我不会解释，如何安装必要的库来构建 android/ios 应用程序的过程。您可以按照 Ionic Framework 网站的文档指南中的说明进行操作。

步骤 5：结论

本文介绍了一个基本的物联网（IoT）设备的架构，它连接到 MQTT 代理服务器，并通过 Node.js REST API 服务器侦听传入数据的通道。移动应用程序然后连接到服务器，使其能够将命令发送到 ESP8266，然后开关继电器。

该实现仅用于教育目的，并且不包含任何授权和认证层，因此它没有被保护。因此有很多事情仍然可以改进。例如，通过 Websockets 而不是 HTTP 与 MQTT 通信。您可以使用从长远来看更安全有效的 TRIACS，而不是使用继电器。并且通过为 ESP8266 添加一个管理仪表板，您可以添加一个动态保存 WIFI SSID 和密码的表单，以便您无需重新编程芯片，只需更改 WiFi 凭据即可。

ESP8266 是非常便宜的、非常强大的 WiFi 模块。你可以做很多，这篇文章只是显示一个 “Hello World” 等效的应用程序。我希望你已经从这篇文章学习很多内容，并能建立你自己的伟大的 IoT 设备！

原文链接：https://steemit.com/technology/@vynci/mobile-app-controlled-relays-through-esp8266-via-mqtt-http

原文链接：https://www.wandianshenme.com/play/esp8266-mqtt-ionic-build-iot-application