

INF1805



Felipe Vieira Côrtes

Fernando Homem da Costa

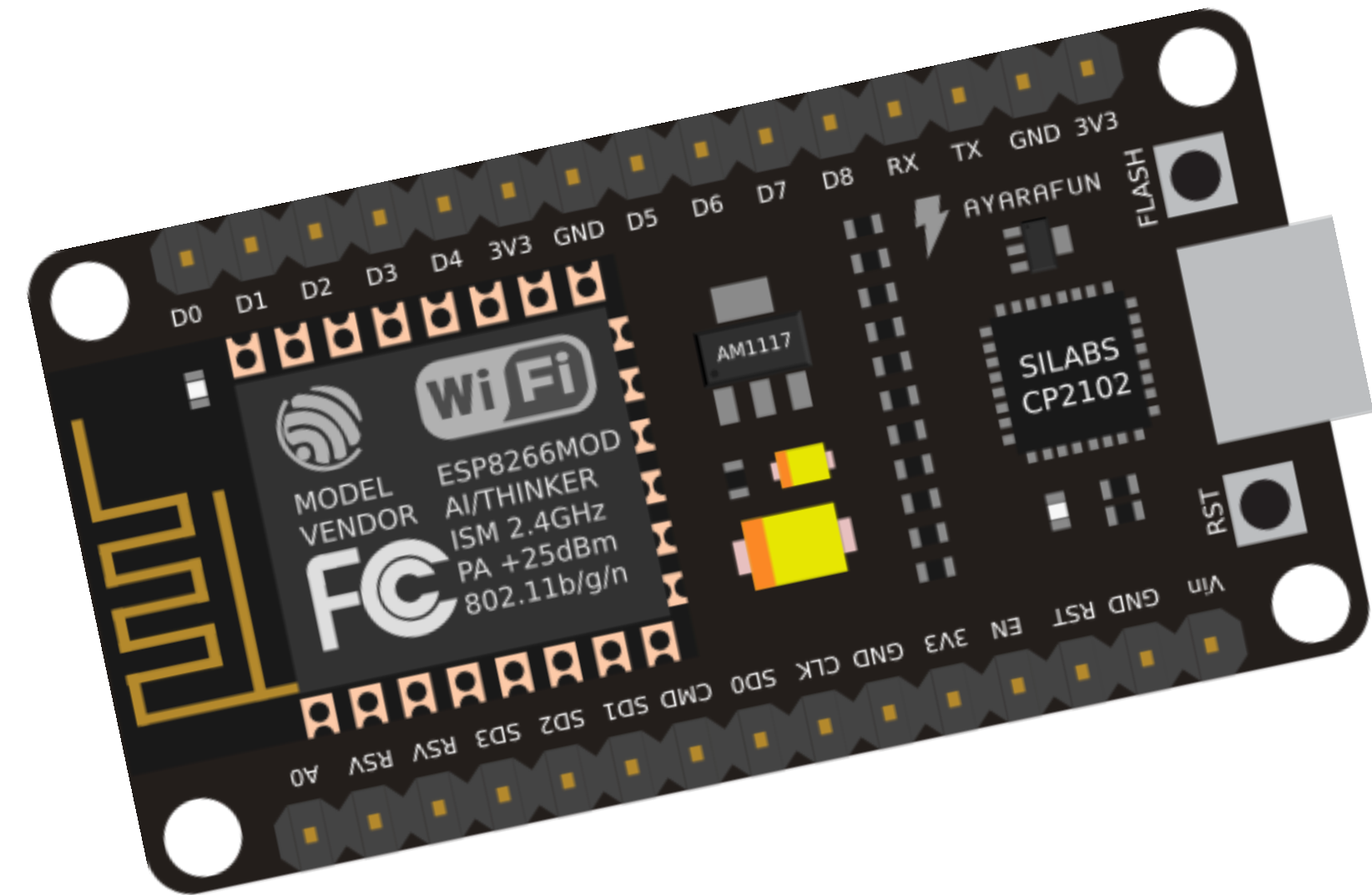
www.github.com/nandohdc/inf1805

OBJETIVO

- Desenvolver um projeto utilizando NodeMCU e a linguagem Lua.
- Utilizando conceitos aprendidos em aula:
 - Wi-Fi - Modo STA
 - Servidor
 - MQTT - Broker & Client
 - Timers & Triggers

PROJETO

- Simular um sistema de reservas de lugares.
- Cada nodeMCU representa um lugar, quando um aluno desejar utilizar aquele lugar, deverá apertar o botão de ocupado presente no nodeMCU.
- Essa informação será transmitida através do MQTT até o outro nodeMCU que é um servidor web.
- O servidor deverá interpretar as mensagens e gerar páginas htmls para disponibilizar as informações.



CÓDIGO

--Configuracoes do nodemcu para ser identificado numa rede

```
nodemcu = {  
  ID = 0,  
  wificonfig = {  
    --Colocar em SSID a rede desejada para conectar  
    ssid = "Valinor",  
    pwd = "bateria123",  
    save = false  
  },  
  MQTT_SERVER = "192.168.43.35",  
  Status = "free"  
}
```

CÓDIGO

```
function reageBotao(pin,c)
    led_livre.desliga()
    led_ocupado.desliga()
    -- readtemp()
    print("reacting...")
    if(pin == button1) then
        c:publish("infos",nodemcu.ID.." occupied "..TEMP,0,0,
            function(c) print("done") end)
        led_ocupado.liga()
        nodemcu.Status = "occupied"
    else
        c:publish("infos",nodemcu.ID.." free "..TEMP,0,0,
            function(c) print("done") end)
        led_livre.liga()
        nodemcu.Status = "free"
    end
end
function conectado (client)
    local envia = publica(client)
    local inscrito = inscreve(client)
    print("Conectado")
    inscrito.novaInscricao()
    recebeControle(client)
    envia.message()
    timer:register(15000,tmr.ALARM_AUTO,function()
        client:publish("infos",nodemcu.ID.." "..nodemcu.Status.." "..TEMP,0,0,function()
            print("sent from alarm")
            end)
        end)
    timer:start()
    gpio.trig(button1,"down", function () reageBotao(button1,client) end)
    gpio.trig(button2,"down", function () reageBotao(button2,client) end)
end
```


CÓDIGO

```
function trataTopico(c, t, m)
  print("topico : "..t)
  if(t == "connect") then
    print ("mensagem ".. msgsConnect .. ", topico: ".. t .. ", dados: " .. m)
    list_Clients[m] = {ID= m, Temp= nil, Status= nil}
    table.insert(client_table, list_Clients[m])
    msgsConnect = msgsConnect + 1
    vals.QtdDevices = msgsConnect
  elseif(t == "infos") then
    print ("mensagem ".. msgsInfos .. ", topico: ".. t .. ", dados: " .. m)
    local split = splitString(m)
    list_Clients[split[1]].Status = split[2]
    list_Clients[split[1]].Temp = split[3]
    updateMaxMinTemp()
    updateStatus()
    msgsInfos = msgsInfos + 1
  elseif(t == "disconnect") then
    print ("mensagem ".. msgsInfos .. ", topico: ".. t .. ", dados: " .. m)
    for i in pairs(client_table) do
      local id = client_table[i].ID
      if (id == m) then
        table.remove(client_table, i)
        print("Node '"..id..' removed from client_table")
        print("new size : "..#client_table)
        updateMaxMinTemp()
        updateStatus()
        vals.QtdDevices = vals.QtdDevices - 1
      end
    end
  end
end
```

CÓDIGO

```
elseif(t == "commands") then
    if(m == "list ip") then
        for i in pairs(client_table) do
            print("> "..client_table[i].ID)
        end
    elseif(m == "list free") then
        for i in pairs(client_table) do
            if(client_table[i].Status == "free") then
                print("> "..client_table[i].ID)
            end
        end
    elseif(m == "list occ") then
        for i in pairs(client_table) do
            if(client_table[i].Status == "occupied") then
                print("> "..client_table[i].ID)
            end
        end
    elseif(m == "hot spot") then
        print("> "..hotSpot)
    elseif(m == "cold spot") then
        print("> "..coldSpot)
    else
        print("Command '"..m.."'" not recognized")
    end
end
end
end
```

DIFICULDADES ENCONTRADAS

- Reunir o servidor com mqtt
 - Lendo a documentação do nodeMCU
- Manter a lista de nodes sempre atualizada
- Interpretar as mensagens que chegam nos diferentes canais
 - Criamos funções auxiliares para quebrar a mensagem
- Sincronizar as callbacks para que nenhuma acesse valor nil
- Realizar testes
 - Criamos testes automatizados com bash.

Perguntas?



Obrigado!