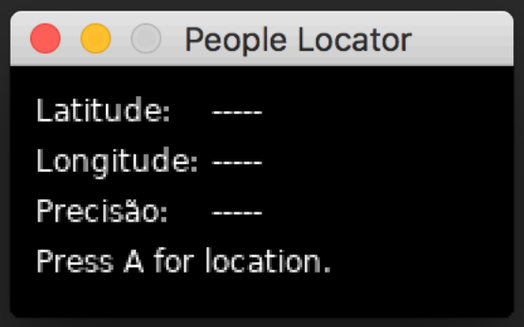
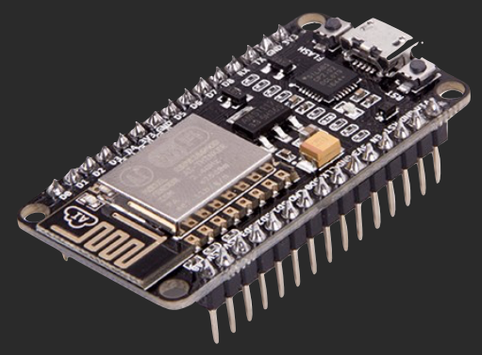
MiniProjeto III - Geolocation no NodeMCU

INF1805 - Sistemas Reativos
Matheus Cunha
Victor Meira

Funcionamento

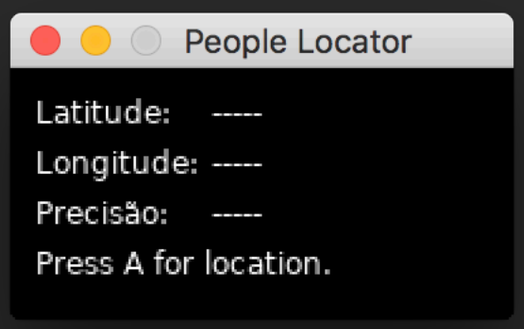


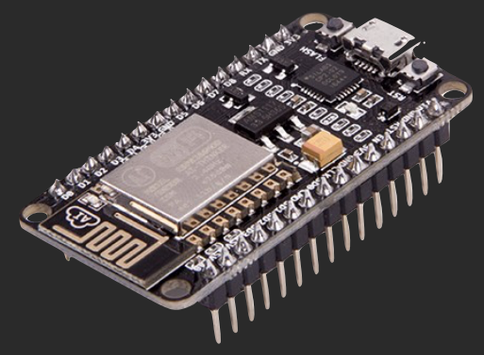


Cliente LÖVE

Cliente NodeMCU

Funcionamento

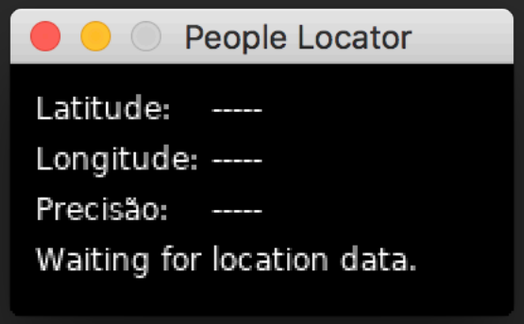


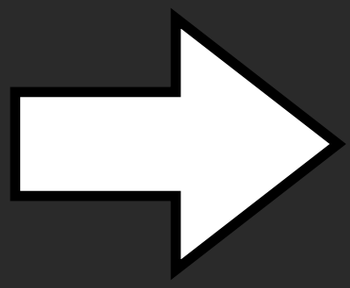


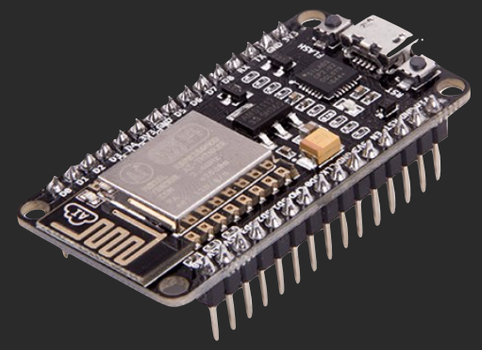
Cliente LÖVE

Cliente NodeMCU

Funcionamento





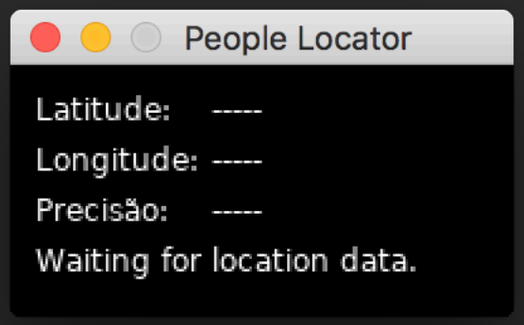


Requesting location

Cliente LÖVE

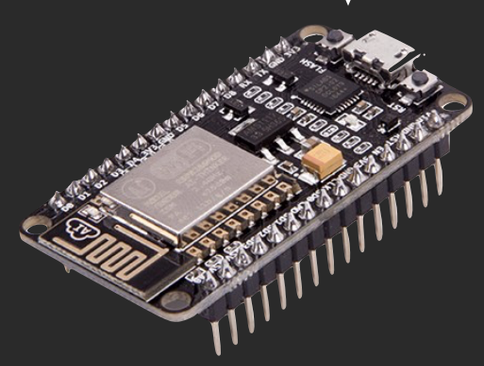
Cliente NodeMCU

Funcionamento



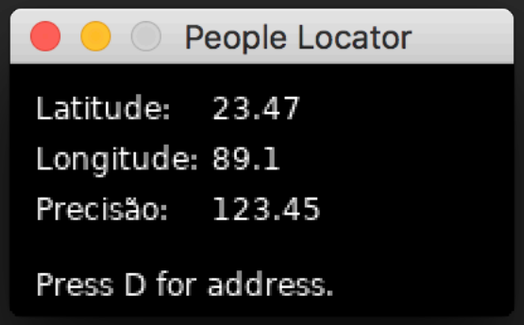
Cliente LÖVE

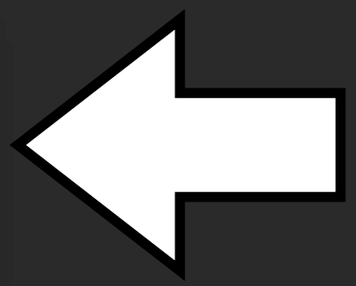
Pressionando botão 1



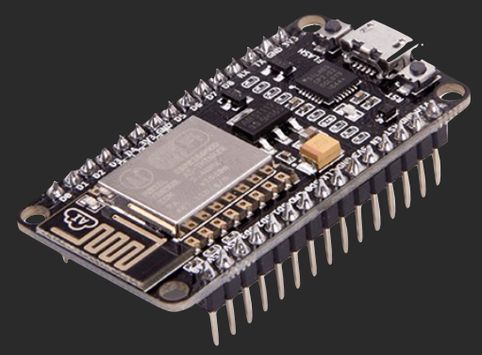
Cliente NodeMCU

Funcionamento





<LAT>;<LONG>;<ACC>

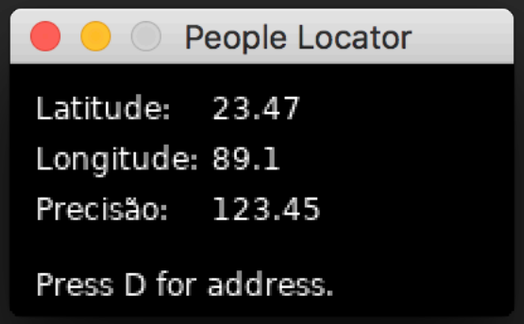


Calcula Latitude e Longitude
via Google Geolocation API

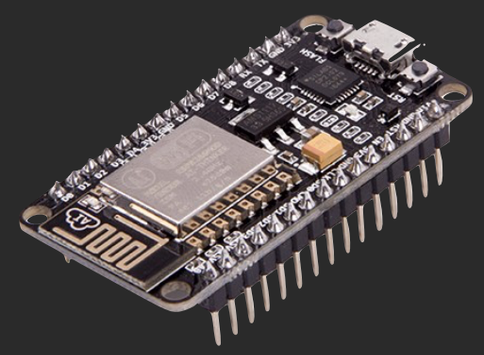
Cliente LÖVE

Cliente NodeMCU

Funcionamento

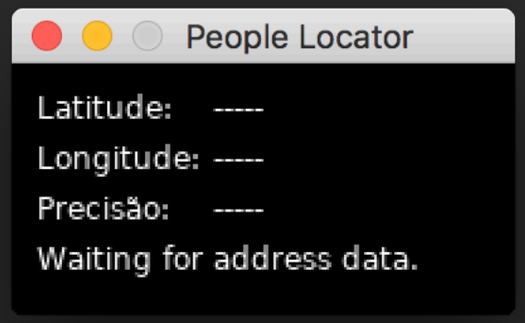


Cliente LÖVE



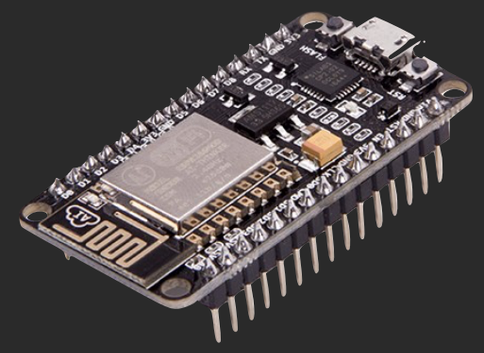
Cliente NodeMCU

Funcionamento



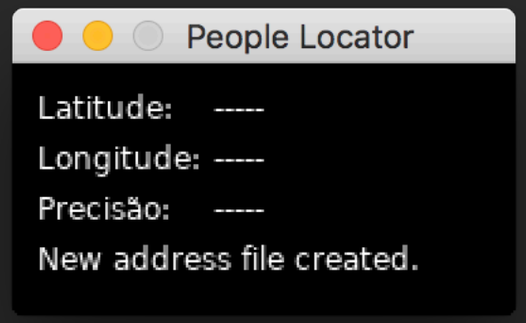
Calcula o endereço via Google Reverse Geocoding API

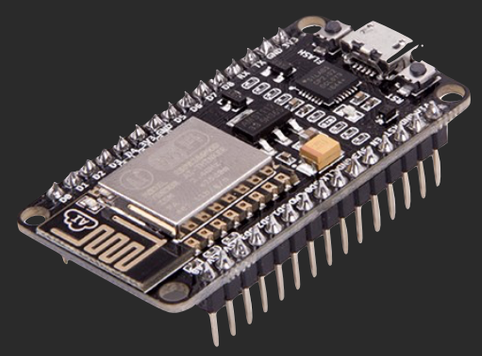
Cliente LÖVE

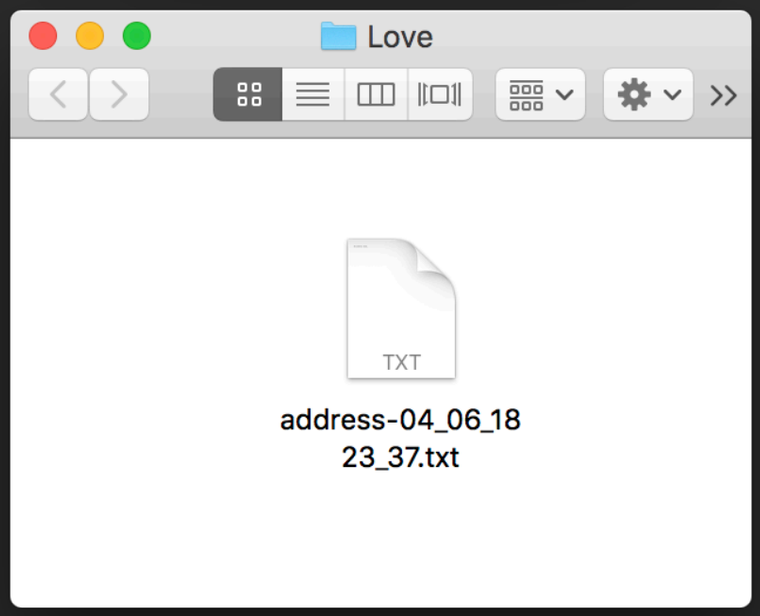


Cliente NodeMCU

Funcionamento



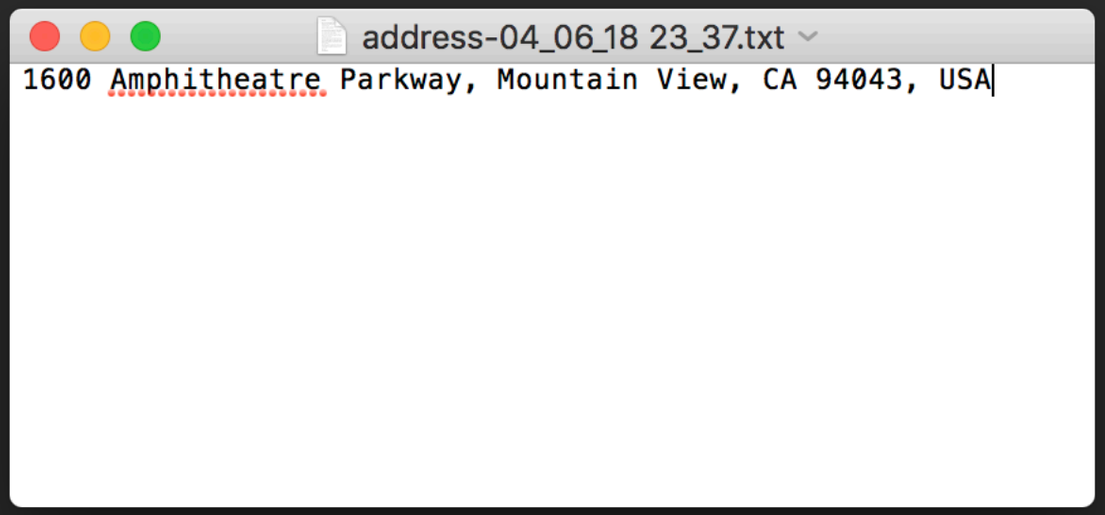


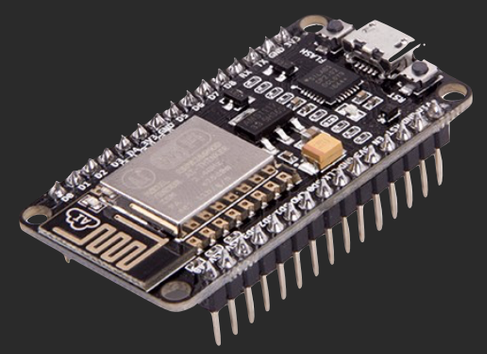


Cliente LÖVE

Cliente NodeMCU

Funcionamento





Cliente LÖVE

Cliente NodeMCU

Implementação - NodeMCU

```
function buttonpressed ()
    local delay = 500000
    local last = 0

    return
    function (level, timestamp)
        local now = tmr.now()

        -- debouncing
        if now - last < delay then return end

            last = now

            locationJSON = ""

            gpio.write  (led1, gpio.HIGH)
            wifi.sta.getap(listap)


    end
end
gpio.trig   (sw1, "down", buttonpressed())
```

Implementação - NodeMCU

```
function listap(t) -- (SSID : Authmode, RSSI, BSSID, Channel)
    local i = 0

    for ssid,v in pairs(t) do
        local authmode, rssi, bssid, channel = string.match(v, "
            ([^,]+),([^,]+),([^,]+),([^,]+)")

        listdeap[i] = [[{ "macAddress":  "]] .. bssid .. [[]] .. [[
            ","signalStrength": ]] .. rssi .. [[,"channel": ]] .. channel .. [[}
            ]]
        if (i == 6) then break end
        i = i + 1
    end

    json = [[{"wifiAccessPoints": [ ]]
    json = json .. table.concat(listdeap,",")

    json = json .. [[]}]]

    print(json)

    --empty table for next call
    for k in pairs (listdeap) do
        listdeap [k] = nil
    end

    local numberOfTries = 5

    local function callbackPost(code,data)
```

Implementação - NodeMCU

```
local function callbackPost(code,data)
        if (code < 0) then
            print("HTTP request failed :", code)
            numberOfTries = numberOfTries - 1
            print("Number of remaining tries: " .. numberOfTries)
            tmr.delay(1000000)
            if(numberOfTries > 0) then
               http.post('https://www.googleapis.com/geolocation/
                v1/geolocate?key=AIzaSyDKcCPg4oxcRCVu-sYs97V1VHNYCdVKn_o'
                ,
                'Content-Type: application/json\r\n',json,callbackPost)
            else
                m:publish("locationData", "Failure", 0,  0,
              function(client, reason) print("Failure to get location
                published.") end)
            end
        else
            print(code, data)
            locationJSON = parseLocationData(data)
            print(locationJSON)
            publishLocationData()
        end
end
```

Implementação - NodeMCU

```
function publishLocationData()
    if(string.len(locationJSON) > 0) then
        print("Publishing Location Data.")
        m:publish("locationData", locationJSON, 0,  0,
                  function(client, reason) print("Location Data Published.") end
                      )
    else
        print("Publish failed. Data was empty")
    end
end

--topic te diz a fila que recebeu e message a string
function messageReceivedCallback(client)

    local function messageTreatment(userdata, topic, message)
            if(message == "Requesting location") then
                print("Received location request.")
                locationJSON = ""
                wifi.sta.getap(listap)
            end
    end

    client:on("message",messageTreatment)
end
```

Implementação - LÖVE

```
function love.draw()
  love.graphics.print("Latitude: ", 10,10)
  love.graphics.print("Longitude: ", 10,30)
  love.graphics.print("Precisão: ", 10,50)

  if(mode == 1) then
    love.graphics.print(latitude, 80,10)
    love.graphics.print(longitude, 80,30)
    love.graphics.print(precisao, 80,50)
  else
    love.graphics.print("-----", 80,10)
    love.graphics.print("-----", 80,30)
    love.graphics.print("-----", 80,50)
  end

  if(mode == -1) then
    love.graphics.print("Press A for location.", 10,70)
  elseif(mode == 1) then
    love.graphics.print("Press D for address.", 10,80)
  elseif(mode == 0) then
    love.graphics.print("Waiting for location data.", 10,70)
  elseif(mode == 2) then
    love.graphics.print("Waiting for address data.", 10,70)
  elseif(mode == -2) then
    love.graphics.print("Request failed.", 10,70)
  elseif(mode == 3) then
    love.graphics.print("New address file created.", 10,70)
  end

end
```

Implementação - LÖVE

```
function love.keypressed(key)
  if key == 'a' then
    mode = 0
    mqtt_client:publish("requestData", "Requesting location")
  end
  if key == 'd' then
    mode = 2
    findAddress()
  end

end
```

Implementação - LÖVE

```
function split(s, delimiter)
    result = {};
    for match in (s..delimiter):gmatch("(.-)"..delimiter) do
        table.insert(result, match);
    end
    return result;
end


function split_message(message)
  local list_1 = {}

  --Splita a string message pelo ;
  list_1 = split(message, ";")

  latitude = list_1[1]
  longitude = list_1[2]
  precisao = list_1[3]

end

function mqttcb(topic, message)
  if(message == "Failure") then
    mode = -2
  else
    split_message(message)
  end

end
```

Implementação - LÖVE

```
function findAddress()

  url = 'https://maps.googleapis.com/maps/api/geocode/json?latlng=' .. latitude
    .. ',' .. longitude ..'
    &key=AIzaSyA5Z5xai0SkBDbbjHueWLggvXvV_rLMG5E&result_type=street_address'

  local body, code, headers, status = https.request(url)

  address_json = body

  local addStart, addEnd = string.find(body,"formatted_address")
  local geoStart, geoEnd = string.find(body,"geometry")

  if(addStart == nil or geoStart == nil) then
    address_json = "No address found."
  else
    address_json = string.sub(body, addEnd + 6, geoStart - 14)
  end

  filename = "address-".. os.date('%d_%m_%y %H_%M.txt')
  -- Opens a file in write mode
  file = io.open(filename, "w")

  -- sets the default output file as the address file
  io.output(file)

  -- writes the adress to the file
  io.write(address_json)

  -- closes the open file
  io.close(file)

  modo = 3

end
```

Principais Dificuldades

- Conseguir fazer com que a API do Google funcionasse confiavelmente
- Conseguir fazer com que o MQTT funcionasse confiavelmente
- Fazer o Reverse GeoCoding API de tal forma que retornasse o endereço completo de forma correta
- Fazer no Node MCU a transformação das informações do WiFi para JSON de tal forma a enviar pro Geolocation API