Controlando um LED via Web Com Python e Wemos D1

Duodécimo Fernandes, Dez/2017.

O objetivo deste tutorial é demonstrar como podemos enviar comandos a partir de um programa gráfico em nosso computador, programado em *python* 3, para uma placa de prototipagem *Wemos* D1 (um clone do *Arduino*). A placa *Wemos* D1 já vem com um módulo esp8266 embutido.

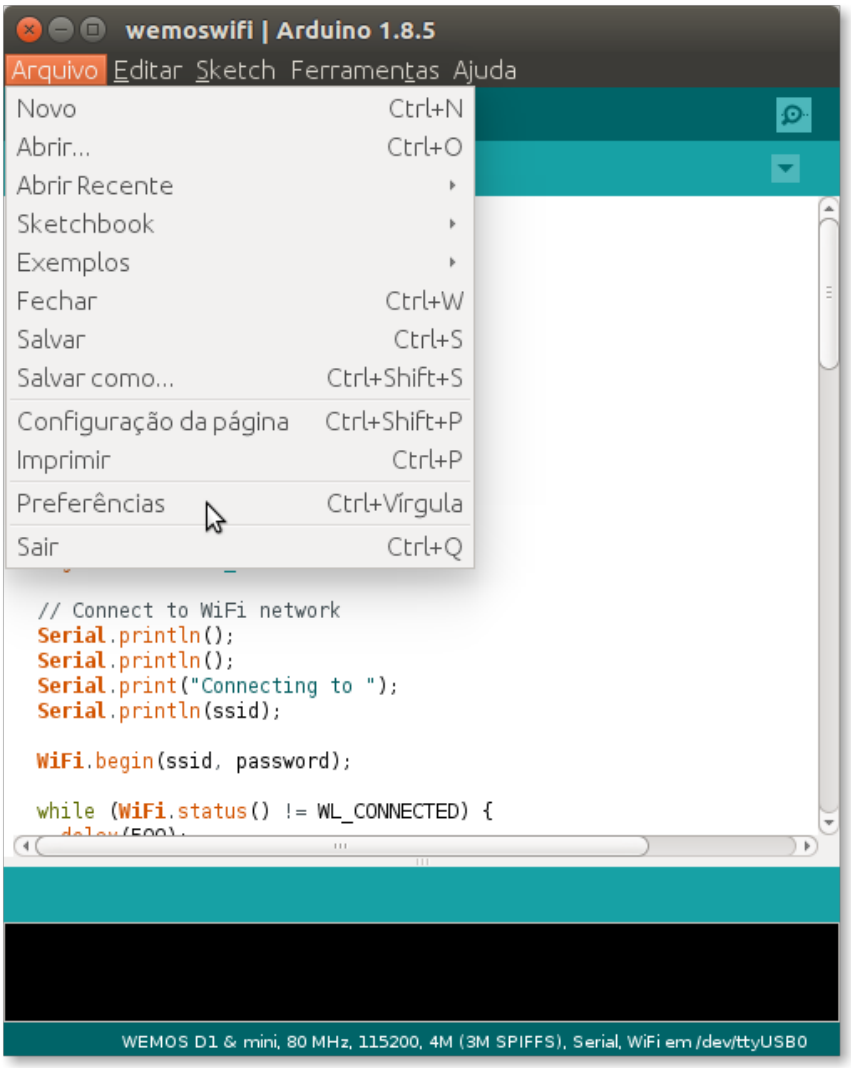
Neste tutorial vamos utilizar o ubuntu 16.04 LTS como ambiente de programação.
Vamos utilizar o *IDE* do *Arduino* para controlar a placa WEMOS D1. A versão da IDE disponibilizada pelo repositório oficial do ubuntu 16.04 não possui as facilidades que desejamos para adicionar as configurações de uma placa. Por isso, foi necessário instalar uma versão mais nova.

Baixamos a versão Arduino 1.8.5 do site do Arduino (https://www.arduino.cc/en/Main/Software). Usando o navegador de arquivos, localizamos o arquivo baixado (arduino-1.8.5-linux64.tar.xz), e clicando com o botão direito do mouse escolhemos descompactar aquí. Em seguida, utilizando o terminal, logamos na pasta opt do ubuntu (cd /opt). Em seguida, movemos a pasta da nova IDE descompactada para a pasta opt (se, por exemplo, tiver sido descompactada na pasta Dowloads do usuário: sudo mv ~/Downloads/arduino-1.8.5 ./). Por último, fizemos que esta versão seja chamada no lugar da anterior (bom, isto é válido porque já tinhamos a versão do repositório do ubuntu isntalada). Logamos em /usr/bin (cd /usr/bin). E fizemos: sudo mv arduino oldArduino; sudo ln -s /opt/arduino-1.8.5/arduino arduino.

Precisamos em seguida disponibilizar para a IDE o software do módulo esp8266 (é recomendável usar a versão mais atual). Abrimos a nova IDE (para garantir que alguns caminhos fossem criados, caso não existissem). No terminal, logamos na pasta do usuário do arduino, normalmente chamada de Arduino, fica dentro da pasta pessoal do usuário (cd ~/Arduino). Criamos a subpasta hardware (mkdir hardware), logamos nela, (cd hardware), criamos a pasta esp8266com (mkdir esp8266com), logamos nela, e clonamos o projeto esp8266 do github:
git clone https://github.com/esp8266/Arduino.git.
Neste momento possuíamos a pasta esp8266, dentro esp8266com, dentro de hardware, dentro da pasta do usuário do Arduino.

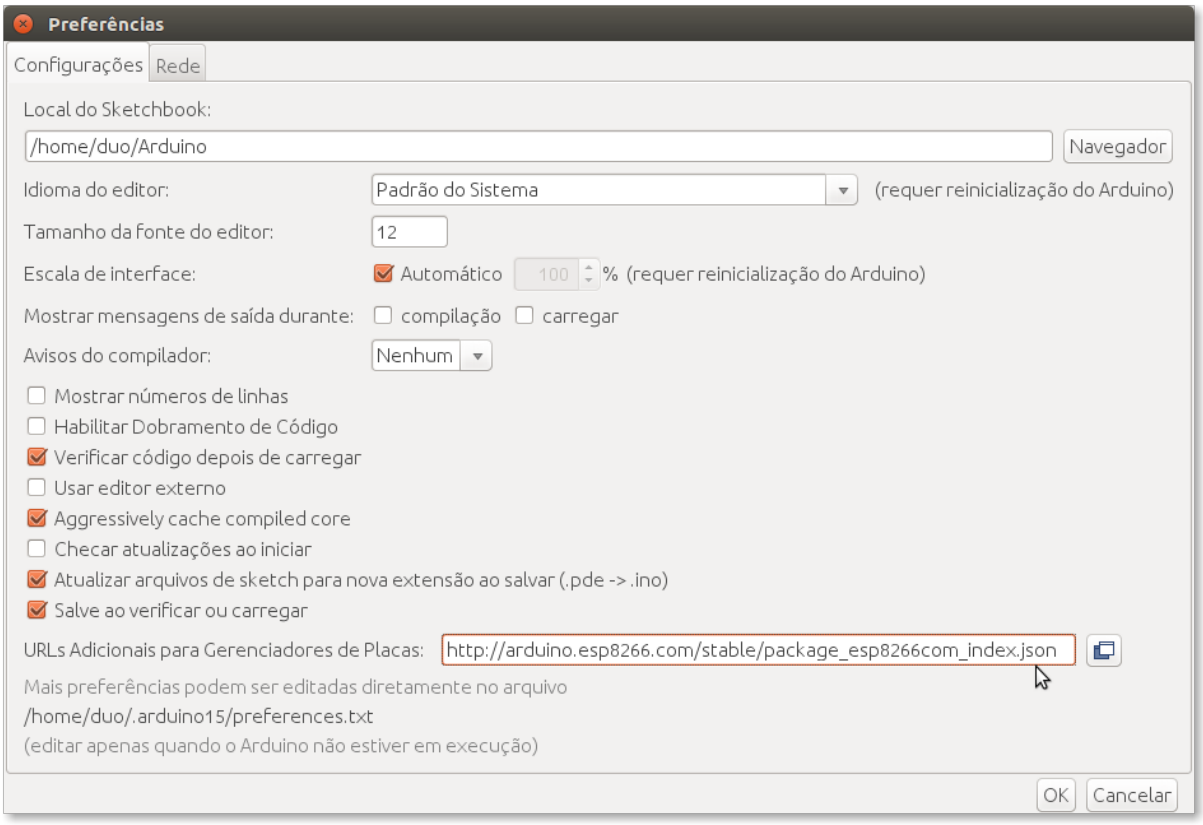
No menu Arduino IDE, abrimos “Arquivo”, e escolhemos a opção “Preferências”, conforme mostra a Figura 1.

Figura 1: Preferências do Arduino IDE



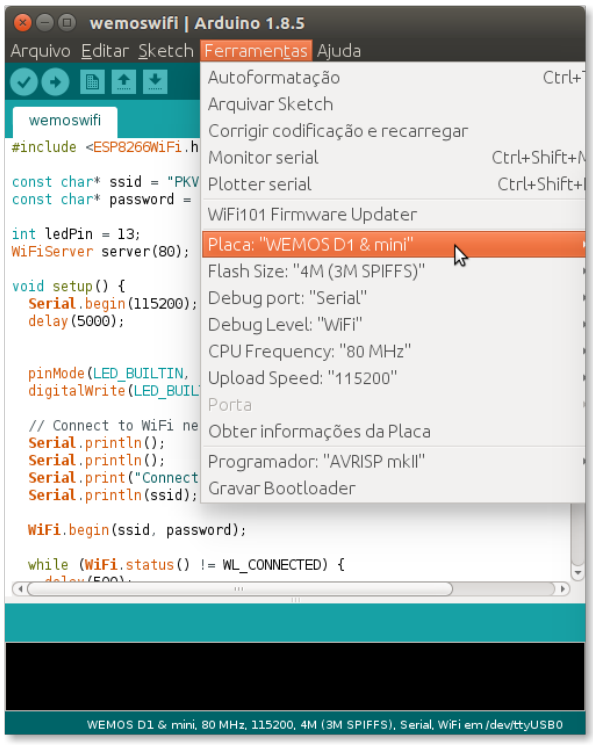
Na janela de Preferências, na aba “Configurações”, na caixa URLs Adicionais para Gerenciadores de Placas, adicionamos a URL:
“http://arduino.esp8266.com.stable/package_esp8266com_index.json”, conforme mostra a Figura 2.

Figura 2: Inserindo uma URL para adicionar novas placas.



Fechamos e reabrimos o Arduino IDE. Neste ponto, podemos selecionar a placa Wemos D1 no menu da IDE, em “Ferramentas”, conforme mostra a Figura 3.

Figura 3: A placa Wemos D1 selecionada no Arduino IDE.



Estávamos prontos para codificar. O código utilizado na placa WEMOS D1 é o seguinte:

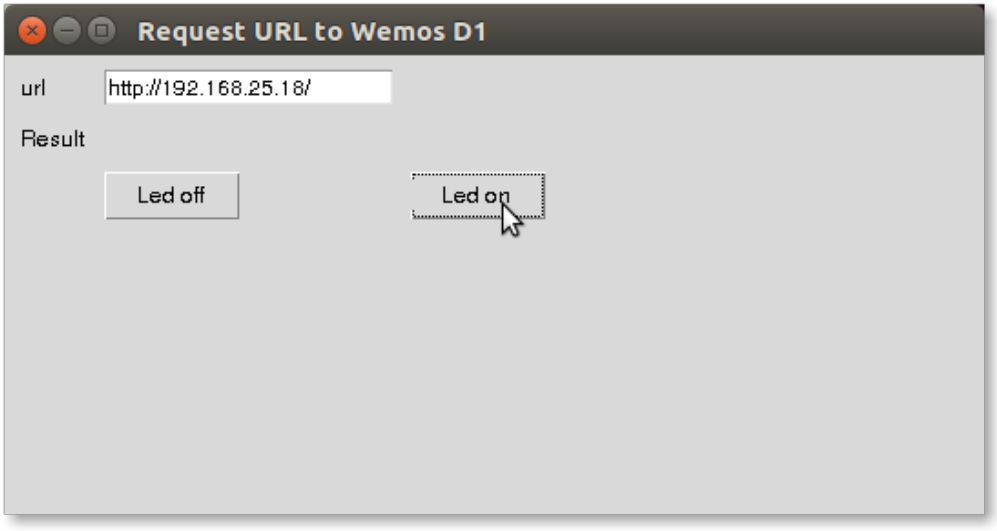
```
#include <ESP8266WiFi.h>

const char* ssid = "<seu SSID>";
const char* password = "<Sua Senha>";

int ledPin = 13;
WiFiServer server(80);

void setup() {
  Serial.begin(115200);
  delay(5000);


  pinMode(LED_BUILTIN, OUTPUT);
  digitalWrite(LED_BUILTIN, LOW);

  // Connect 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);
    Serial.print(".");
  }
  Serial.println("");
  Serial.println("WiFi connected");

  // Start the server
  server.begin();
  Serial.println("Server started");

  // Print the IP address
  Serial.print("Use this URL : ");
  Serial.print("http://");
  Serial.print(WiFi.localIP());
  Serial.println("/");

}

void loop() {
  // Check if a client has connected
  WiFiClient client = server.available();
  if (!client) {
    return;
  }

  // Wait until the client sends some data
  Serial.println("new client");
  while(!client.available()){
    delay(1);
  }

  // Read the first line of the request
  String request =
client.readStringUntil('\r');
  Serial.println(request);
  client.flush();

  // Match the request

  int value = LOW;
  if (request.indexOf("/LED=ON") != -1) {
    digitalWrite(LED_BUILTIN, LOW);
    value = HIGH;
  }
  if (request.indexOf("/LED=OFF") != -1){
    digitalWrite(LED_BUILTIN, HIGH);
    value = LOW;
  }



  // Return the response
  client.println("HTTP/1.1 200 OK");
  client.println("Content-Type: text/html");
  client.println(""); //  do not forget this
one
  client.println("<!DOCTYPE HTML>");
  client.println("<html>");

  client.print("Led pin is now: ");

  if(value == HIGH) {
    client.print("On");
  } else {
    client.print("Off");
  }
  client.println("<br><br>");
  client.println("Click <a
href=\"/LED=ON\">here</a> turn the LED
BUILTIN ON<br>");
  client.println("Click <a
href=\"/LED=OFF\">here</a> turn the LED
BUILTIN OFF<br>");
  client.println("</html>");

  delay(1);
  Serial.println("Client disconnected");
  Serial.println("");

}
```

Em seguida, o código utilizado no Python 3:

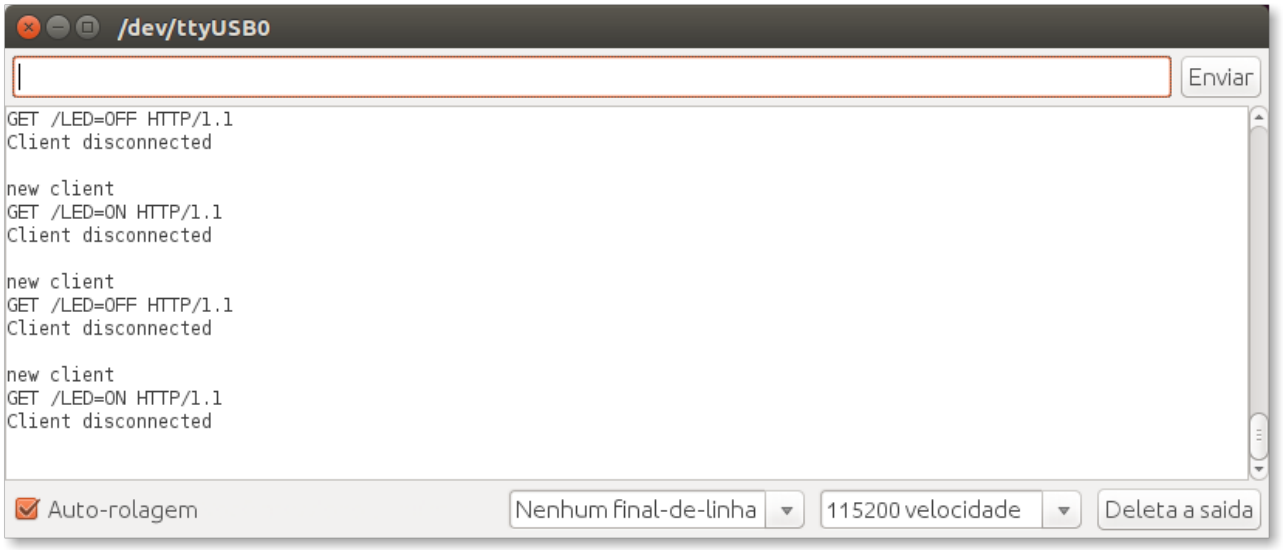
```
import urllib.request
from tkinter import *
from tkinter import ttk

def placeCall(cmd):
  try:
    x = urllib.request.urlopen(url.get() + cmd)
    result.set(x)
  except urllib.error.URLError as e:
    ##Show user an error
    self.update_text(str(e))

root = Tk()
root.title("Request URL to Wemos D1")

mainframe = ttk.Frame(root, padding="3 3 12 12")
mainframe.grid(column=0, row=0, sticky=(N, W, E, S))
mainframe.columnconfigure(0, weight=1)
mainframe.rowconfigure(0, weight=1)

url = StringVar()
result = StringVar()
url_entry = ttk.Entry(mainframe, width=22, textvariable=url)
url_entry.grid(column=2, row=1, sticky=(W, E))

ttk.Button(mainframe, text="Led off", command = lambda :
placeCall("LED=OFF")).grid(column=2, row=3, sticky=W)
ttk.Button(mainframe, text="Led on", command = lambda :
placeCall("LED=ON")).grid(column=3, row=3, sticky=W)

ttk.Label(mainframe, text="url").grid(column=1, row=1, sticky=W)
ttk.Label(mainframe, text="Result").grid(column=1, row=2, sticky=E)
ttk.Label(mainframe, text=result.get()).grid(column=3, row=3, sticky=W)

for child in mainframe.winfo_children(): child.grid_configure(padx=5, pady=5)

url_entry.focus()
root.bind('<Return>', placeCall)

root.mainloop()
```

Conforme mostra a Figura 4, observemos o código em Python 3 rodando:

Figura 4: Interface gráfica para controlar via WEB um Led na placa Wemos D1.



A url a ser informada deve ser observada na janela serial do Arduino IDE assim que o código é carregado no Wemos D1.



Vai tentar? Bom divertimento!

Para ver um breve vídeo que mostra o projeto funcionando: clique aquí para vídeo no youtube.