



## Fundamentos de Internet das Coisas

---

# Atividade Formativa – Conectividade do Módulo Wi-Fi Integrado - ESP32

*Semana 5*

---



## Atividade Formativa

---

Olá, para que você realize a atividade formativa desta semana é um pré-requisito dispor do ambiente de desenvolvimento Linux criado durante as semanas anteriores, nele exploraremos a programação em micropython de rotinas específicas de uso da rede Wi-Fi pelo microcontrolador, através do Thonny IDE em seu Linux. À partir dessa programação o ESP32 terá a conectividade necessária, para que através do módulo Wi-Fi, ele comunique-se principalmente com a internet.

A atividade é dividida em duas etapas: a primeira etapa consiste em programar uma rotina em linguagem python que permitirá o próprio microcontrolador ESP32 de forma automatizada estabelecer a conectividade necessária entre o dispositivo a uma rede Wi-Fi que concederá acesso à internet. Isso será realizado em um arquivo único, com as duas ações. Já a segunda etapa consiste em desmembrar a função conecta do programa testewifi.py e utilizá-la como uma biblioteca de Wi-Fi em diferentes programações que necessitem de conectividade à internet.

Atenção, as ações a serem realizadas durante esta atividade formativa farão parte da próxima etapa de avaliação, a atividade somativa 2. Portanto, como já foi citado nas atividades formativas anteriores, é importante que realize todas as etapas descritas, sem pular a ordem de qualquer uma delas.



## ATIVIDADE FORMATIVA

ATIVIDADE FORMATIVA: Conectividade do Módulo Wi Fi Integrado - ESP32

Estudante:

Etapa 1: Esta atividade consiste em programar uma rotina em linguagem python que permitirá o próprio microcontrolador ESP32 de forma automatizada estabelecer a conectividade necessária entre o dispositivo e uma rede Wi-Fi.

**Objetivo:** Permitir o microcontrolador ESP32 estabelecer a conectividade entre o dispositivo e uma rede Wi-Fi.

**Resolução da etapa:**

Vamos ao passo a passo dos ajustes necessários:

Inicialmente use o comando **help()** para verificar se o seu ESP32 está conectado corretamente ao Thonny IDE. Se os comandos de controle forem apresentados, tudo está ok, caso contrário refaça os passos.

```

Shell
>>> help() ←
Welcome to MicroPython on the ESP32!

For generic online docs please visit http://docs.micropython.org/

For access to the hardware use the 'machine' module:

import machine
pin12 = machine.Pin(12, machine.Pin.OUT)
pin12.value(1)
pin13 = machine.Pin(13, machine.Pin.IN, machine.Pin.PULL_UP)
print(pin13.value())
i2c = machine.I2C(scl=machine.Pin(21), sda=machine.Pin(22))
i2c.scan()
i2c.writeto(addr, b'1234')
i2c.readfrom(addr, 4)

Basic WiFi configuration:

import network
sta_if = network.WLAN(network.STA_IF); sta_if.active(True)
sta_if.scan()                                # Scan for available access points
sta_if.connect("<AP_name>", "<password>") # Connect to an AP
sta_if.isconnected()                          # Check for successful connection

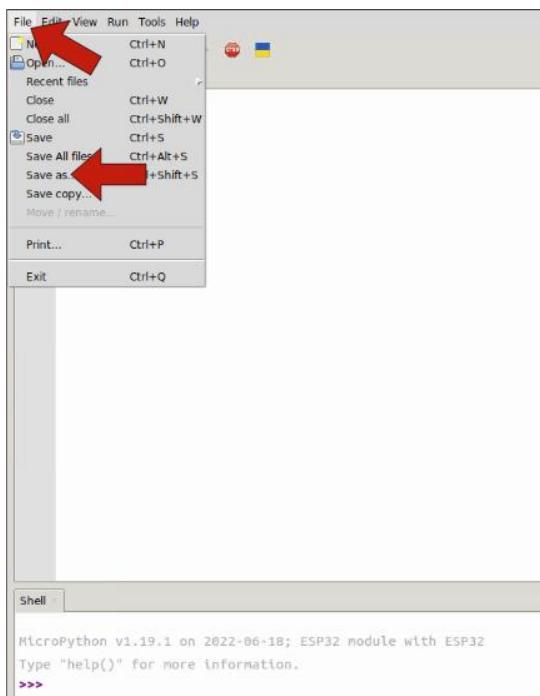
Control commands:
CTRL-A      -- on a blank line, enter raw REPL mode
CTRL-B      -- on a blank line, enter normal REPL mode
CTRL-C      -- interrupt a running program
CTRL-D      -- on a blank line, do a soft reset of the board
CTRL-E      -- on a blank line, enter paste mode

For further help on a specific object, type help(obj)
For a list of available modules, type help('modules')
>>> |

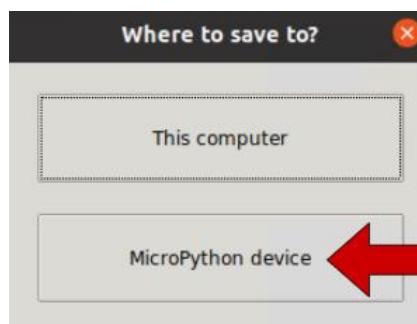
```



Antes de qualquer passo a seguir, salve o arquivo .py a ser utilizado.

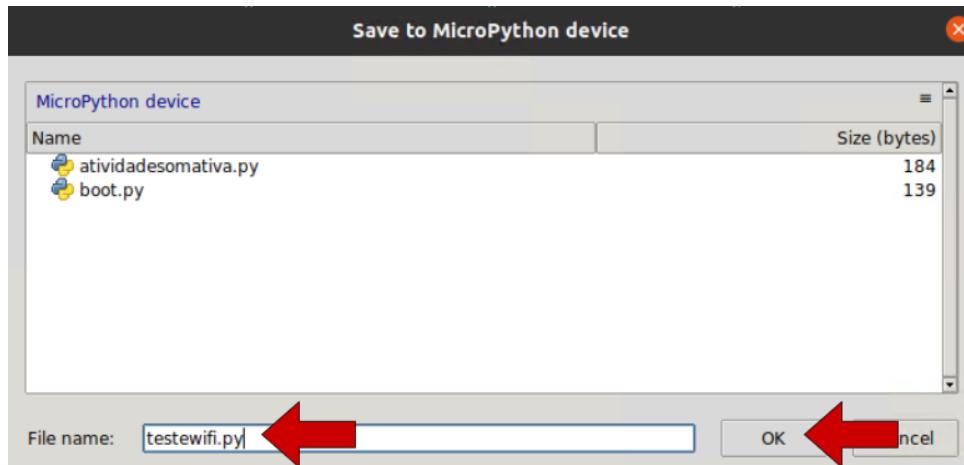


Escolha o local de salvamento como sendo o microcontrolador.



Escolha o nome testewifi.py como o nome a ser utilizado.

Atenção neste ponto: não se esqueça de salvar com a extensão .py



Crie, na área de programação do arquivo testewifi.py, uma **função** chamada **conecta**, com os parâmetros **ssid** e **senha**

```
1 def conecta(ssid, senha):
```

Para estabelecer a comunicação do ESP32 com a rede Wi-Fi copie as funções que já usamos e testamos anteriormente: `import network, station = network.WLAN(network.STA_IF)`, `station.active(True)` e `station.connect("NOME-DA-SUA-REDE-WIFI", "SUA-SENHA-REDE-WIFI")` neste caso utilize os parâmetros escolhidos para a função: **ssid** e **senha** no lugar do **NOME-DA-SUA-REDE-WIFI** e **SUA-SENHA-REDE-WIFI**

Repare que **import time** não existia, mas será utilizada logo a frente no desenvolvimento.

```
1 def conecta(ssid, senha):
2     import network
3     import time
4     station = network.WLAN(network.STA_IF)
5     station.active(True)
6     station.connect(ssid, senha)
```

Para reforçar o processo de automatização, a seguir vamos criar um **laço for** para realizar diversas tentativas de conexão, neste caso 50 tentativas a cada 0.1 segundo. Se em algum momento das tentativas de conexão a condição for satisfeita então a ação será encerrada e já fora do **laço for** usaremos o **return** `station` onde o próprio objeto `station.isconnected` será consultado e uma vez conectado finalizará a nossa função `conecta`.

Repare que para utilizarmos o `time.sleep` para verificar a cada 0.1 segundo, é necessário utilizar a biblioteca `time`, motivo pelo qual a importamos antes, pelo **import time**



```

1 def conecta(ssid, senha):
2     import network
3     import time
4     station = network.WLAN(network.STA_IF)
5     station.active(True)
6     station.connect(ssid, senha)
7     for t in range(50):
8         if station.isconnected():
9             break
10        time.sleep(0.1)
11    return station
12

```

Com a conexão entre ESP32 e rede Wi-Fi estabelecida, podemos nos concentrar em testar a conectividade com a internet. Assim como usamos e testamos copie para o seu programa: **import urequests, station = conecta("teste\_wifi", "senha\_teste\_wifi"), response = urequests.get("SITE-A-ACESSAR") e print(response.text)** você deverá implementá-los. \*Atenção: lembre-se que **teste\_wifi** e **senha\_teste\_wifi** são dados de sua rede Wi-Fi.

Desta vez, além de implementá-los você também deverá imprimir as mensagens que descrevem as ações de conexão, como, 1<sup>a</sup> condição: o ato da tentativa (se ela está sendo realizada), 2<sup>a</sup> condição: o ato da conexão (se ela foi estabelecida ou não) e 3<sup>a</sup> e última condição se o ato do acesso à página da internet solicitada teve êxito. Essa descrição tornará a visualização dos resultados mais amigável.

```

1 def conecta(ssid, senha):
2     import network
3     import time
4     station = network.WLAN(network.STA_IF)
5     station.active(True)
6     station.connect(ssid, senha)
7     for t in range(50):
8         if station.isconnected():
9             break
10        time.sleep(0.1)
11    return station
12
13 import urequests
14 print("Conectando...") ← 1a
15 station = conecta("teste_wifi", "senha_teste_wifi")
16 if not station.isconnected():
17     print("Nao conectado!...") ← 2a - não estabelecida
18 else:
19     print("Conectado!...") ← 2a - estabelecida
20     print("Acessando o site...") ← 3a
21     response = urequests.get("https://www.ppgia.pucpr.br")
22     print("Pagina acessada:") ← 3a - sucesso
23     print(response.text)
24     station.disconnect()
25

```

Com a conexão entre ESP32 e rede Wi-Fi estabelecida, podemos nos concentrar em testar a conectividade com a internet. Assim como usamos e testamos copie para o seu programa: **import urequests, response = urequests.get("SITE-A-ACESSAR") e print(response.text)** você deverá implementá-los.



Desta vez, além de implementá-los você também deverá imprimir as mensagens que descrevem as ações de conexão, como, 1<sup>a</sup> condição: o ato da tentativa (se ela está sendo realizada), 2<sup>a</sup> condição: o ato da conexão (se ela foi estabelecida ou não) e 3<sup>a</sup> e última condição se o ato do acesso à página da internet solicitada teve êxito. Essa descrição tornará a visualização dos resultados mais amigável.

Salve e execute

```
File Edit View Run Tools Help
[ testewifi.py ] * 
def conecta(ssid, senha):
    import network
    import time
    station = network.WLAN(network.STA_IF)
    station.active(True)
    station.connect(ssid, senha)
    for t in range(50):
        if station.isconnected():
            break
        time.sleep(0.1)
    return station

import urequests
print("Conectando...")
station = conecta("teste_wifi", "senha_teste_wifi")
if not station.isconnected():
    print("Nao conectado!")
else:
    print("Conectado!")
    print("Acessando o site...")
    response = urequests.get("https://www.ppgla.pucpr.br")
    print("Pagina acessada:")
    print(response.text)
    station.disconnect()
```



Observe o resultado desejado

The screenshot shows a code editor window with a Python script named `testewifi.py` and a terminal window below it.

**Code Editor (testewifi.py):**

```
File Edit View Run Tools Help
[ testewifi.py ]
1 def conecta(ssid, senha):
2     import network
3     import time
4     station = network.WLAN(network.STA_IF)
5     station.active(True)
6     station.connect(ssid, senha)
7     for t in range(50):
8         if station.isconnected():
9             break
10        time.sleep(0.1)
11    return station
12
13 import urequests
14 print("Conectando...")
15 station = conecta("teste_wifi", "senha_teste_wifi")
16 if not station.isconnected():
17     print("Nao conectado!...")
18 else:
19     print("Conectado!...")
20     print("Acessando o site...")
21     response = urequests.get("https://www.ppgia.pucpr.br")
22     print("Pagina acessada:")
23     print(response.text)
24     station.disconnect()
```

**Terminal (Shell):**

```
>>> %Run -c SEDITOR_CONTENT
Conectando...
Conectado!...
Acessando o site...
Pagina acessada:
<html>
<META HTTP-EQUIV="Refresh" content="0;URL=https://www.ppgia.pucpr.br/pt">
</html>
```



Etapa 2: Esta atividade consiste em desmembrar a função conecta do programa testewifi.py e utilizá-la como uma biblioteca de Wi-Fi em diferentes programações que necessitem de conectividade à internet.

---

**Objetivo:** Criar e permitir o uso por outros programas .py de uma biblioteca Wi-Fi para a conexão internet.

Entregas a serem realizadas:

- Crie um novo arquivo nomeado de wifi\_lib.py;
- Salve o wifi\_lib.py no microcontrolador;
- Recorte a função conecta do arquivo testewifi.py para o novo arquivo wifi\_lib.py;
- No arquivo testewifi.py adicione às importações o import **from wifi\_lib import conecta**;
- Salve os arquivos;
- Execute o programa testewifi.py;
- O resultado obtido deve ser idêntico ao da etapa anterior.



PROFESSOR-AUTOR  
Adilson Galiano Filho

