Wlan

Der **ESP32** hat **Wlan** an Board. Nun wollen wir uns einmal ansehen, wie man den M5Stick C Plus mit einem Wlan verbinden kann.

Dazu gibt es sowohl die Methode die Micropython mitbringt, als auch die Methode die M5Stack in seinem Micropython implementiert hat. **Wir werden uns hier auf die Methode von Micropython beschränken.**

Wlan

Die benötigten Dinge finden wir im Modul **network**. Das sind vor allem die Klasse **WLAN** und die Constanten **AP_IF** und **STA_IF**.

```
>>> import network

>>> dir(network)

['__class__', '__init__', '__name__', 'AP_IF', 'AUTH_MAX', 'AUTH_OPEN',
'AUTH_WEP', 'AUTH_WPA2_PSK', 'AUTH_WPA_PSK', 'AUTH_WPA_WPA2_PSK', 'MODE_11B',
'MODE_11G', 'MODE_11N', 'PHY_IP101', 'PHY_LAN8720', 'PHY_TLK110',
'STAT_ASSOC_FAIL', 'STAT_BEACON_TIMEOUT', 'STAT_CONNECTING', 'STAT_GOT_IP',
'STAT_HANDSHAKE_TIMEOUT', 'STAT_IDLE', 'STAT_NO_AP_FOUND',
'STAT_WRONG_PASSWORD', 'STA_IF', 'WLAN', 'phy_mode']
```

Wlan

Zuerst muss eine Instanz der Klasse WLAN angelegt werden. Dabei muss auch festgelegt werden, ob es sich um eine Station oder einen Accesspoint handeln soll.

STA_IF = Station

AP_IF = Accesspoint

```
>>> wlan = network.WLAN(network.STA_IF)
>>> wlan.active(True)
True
>>> wlan.scan()
[(b'FRITZ!Box 7560 KV', b'|\xffMj\xe1\x95', 1, -53, 3, False), (b'Kapest-Fritz_01', b'\x001\
x92Wbf', 6, -58, 4, False), (b'WILLY.TEL-71IU1DHQNW', b'DNm\xc2\x04\xcc', 1, -64, 3, False),
(b'Kapest-Fritz', b'$e\x11f\xf11', 6, -66, 4, False), (b'ESP_A06851', b'P\x02\x91\xa0hQ', 1, -87,
0, False), (b'Brian', b'DNm\xde\xb1\xdc', 1, -94, 3, False)]
>>>
```

Wlan

Die SSID’s der Wlan’s werden als bytestring empfangen. Deshalb ist die Umwandlung in einen String erforderlich:

```
[(b'FRITZ!Box 7560 KV', b'|\xffMj\xe1\x95', 1, -53, 3, False), (b'Kapest-
Fritz_01', b'\x001\x92Wbf', 6, -58, 4, False), (b'WILLY.TEL-71IU1DHQNW', b'DNm\
xc2\x04\xcc', 1, -64, 3, False), (b'Kapest-Fritz', b'$e\x11f\xf11', 6, -66, 4,
False), (b'ESP_A06851', b'P\x02\x91\xa0hQ', 1, -87, 0, False), (b'Brian',
b'DNm\xde\xb1\xdc', 1, -94, 3, False)]


>>> [x[0].decode('utf-8') for x in wlan.scan()]

['FRITZ!Box 7560 KV', 'Kapest-Fritz_01', 'WILLY.TEL-71IU1DHQNW', 'Kapest-
Fritz', 'WLAN-372270', 'WILLY.TEL-1DACHUM3UY', 'ESP_A06851']
```

Wlan

Ein bischen Spielkram.

```
>>> [print(x[0].decode('utf-8')) for x in wlan.scan()]

Kapest-Fritz_01
WILLY.TEL-71IU1DHQNW
FRITZ!Box 7560 KV
Kapest-Fritz
ESP_A06851
WILLY.TEL-1DACHUM3UY
WLAN-372270
[None, None, None, None, None, None, None]
>>>
```

Wlan

Mit einem Wlan verbinden.

Hierfür benötigen wir die **SSID** und das **Passwort** des Wlan mit dem wir uns verbinden wollen.

```
>>> SSID = 'Attraktor'
>>> PW = 'blafablafa'

>>> wlan.connect(SSID, PW)

>>> wlan.isconnected()
False                                   # Bin gerade zu Hause

>>> wlan.connect(SSID, PW)
>>> wlan.isconnected()
True                                    # Mit den Daten meines Wlan
```

Wlan

Die Daten unserer Wlan Verbindung ansehen / einstellen.

Dafür stehen die Methoden **.config()** und **.ifconfig()** zur Verfügung. **.config()** funktioniert nur mit **‘mac‘.**

```
>>> wlan.config('mac')
b'\x94\xb9~\x8d6\xec'    # Das Decodieren diese Zeichenfolge klappte nicht.
>>>                      # Original: 94b97e8d36ec
for x in mac:
    print(hex(x))

0x94                     # Aber so geht’s
0xb9
0x7e
0x8d
0x36
0xec
```

Wlan

Die Daten unserer Wlan Verbindung ansehen / einstellen.

Mit **.ifconfig()** klappt es aber besser. Hier werden Strings zurückgegeben:

```
>>> wlan.ifconfig()
    ('192.168.5.98', '255.255.255.0', '192.168.5.1', '192.168.5.1')
    #  IP-Adresse      Netzmaske          Gateway      DNS-Server

>>> wlan.ifconfig()[0]      # IP-Adresse
'192.168.5.98'
```

Wlan

Eine Verbindung zu einem Wlan kann auch wieder getrennt werden:

```
>>> wlan.connect(SSID, PW)
>>> wlan.isconnected()
True
>>> wlan.disconnect()
>>> wlan.isconnected()
False
```

Wlan

Beim experimentieren mit **Wlan** ist mir gelegentlich **Thonny abgestürzt** und musste neu gestartet werden. Damit ich nicht immer alles wieder eintippen musste, habe das nebenstehende **Script** geschrieben. Das starte ich zuerst, und schon bin ich wieder im Wlan.

```
# connect.py

import network

SSID = 'xxx'
PW = 'xxx'

wlan = network.WLAN(network.STA_IF)
wlan.active(True)

wlan.connect(SSID, PW)

while wlan.isconnected() == False:
    time.sleep(1)

print(wlan.ifconfig()[0])
```

Der Kampf mit der MAC-Adresse

```
# '94:b9:7e:8d:36:ec'

>>> wlan.config('mac')
b'\x94\xb9~\x8d6\xec' # bytestring

>>> mac = wlan.config('mac')
>>> str(mac)
"b'\\x94\\xb9~\\x8d6\\xec'"

>>> for i in mac:
        print(hex(i))
0x94
0xb9
0x7e
0x8d
0x36
0xec
```

```
>>> chr(0x7e)      ord('~')
'~'

>>> chr(0x36)
'6'

>>> for i in mac:
        mac_liste += hex(i)[2:]

>>> mac_liste
'94b97e8d36ec'
```

Der Kampf mit der MAC-Adresse

```
>>> for i in mac:
        mac_liste += ':'
        mac_liste += hex(i)[2:]

>>> mac_liste
':94:b9:7e:8d:36:ec'

>>> mac_liste = ''
>>> for i in mac:
        mac_liste = mac_liste + ':' + hex(i)[2:]

>>> mac_liste
':94:b9:7e:8d:36:ec'
```

```
>>> mac_liste = mac_liste[1:]

>>> mac_liste
'94:b9:7e:8d:36:ec'
```