

Micropython - octopus() tahák

Toto není výuka Micropythonu, pouze orientační shrnutí a pomůcka pro workshopy.

Syntaxe taháku:

```
uPy - zkratka pro Micropython
-> vrácená hodnota (neuvádíme pokudě)
# poznámka nebo vysvětlení / upřesnění
[text] napíšeme slovo "text" a odešleme ENTERem
...víceřádkový
...kód
```

Základy Pythonu

```
>>> toto je takzvaný "prompt", terminálová výzva,
abychom tam něco napsali, příkaz nebo posloupnost
příkazů:
```

```
>>> a = 123 -> 123 # přiřazení hodnoty do proměnné
(prompt už dále neuvádíme)
import math # načtení modulu math (knihovna)
math. a po tečce stisknout TAB > vypíše metody
math.log10(1000) -> 3
math.pi
```

```
def suma(x, y): # definice vlastní funkce
    return x + y # nezapomenout odsazení
suma (1, 2) -> 3
```

```
while True: # nekonečná smyčka
    print("loop")
```

ESP32

```
import esp32 # načtení modulu esp32
esp32.raw_temperature() -> teplota u procesoru
esp32.hall_sensor() -> hallova sonda, mg.pole
```

Instalace

```
octopus_initial.setup() # je součástí octopus-uPy
[w] # wifi sub menu
[a] # add
"ssid" # zadání vaší wifi
"pass" # zadání hesla
[cw] # connect to wifi
[sd] # system download
```

Update

```
octopus() # inicializace octopus, vždy na začátku
.u() # metoda update - provede:
from util.setup import deploy
deploy("https://octopusengine.org/download/micropython/stable.tar")
```

Nastavení

```
setup() # hlavní metoda vyvolá menu > zvolíme:
[sd] # device - volba "desky" / hw modulu
[ios] # input / output setup - periferie
[si] # system info
...
```

octopus() jednopísmenné metody a inicializace:

```
h() # o_help() = nápověda
i() # o_info() = info o systému
c() # clt() clear terminal
w() # w_connect() = připojení k wifi
u() # system update / upgrade
r() # reset() = system reset

d7 = disp7_init() # sedmisegmetový Led display
d8 = disp8_init() # matice 8x8
oled = oled_init() # oled 128x64 nebo 128x32 px
d2 = lcd2_init() # i2c LCD 2 nebo 4 řádky
time_init() # nastavení času z web-api
s1 = servo_init() # servo
temp = temp_init() # teploměr
b0 = button_init(0) # tlačítko
db = database_init() # databázová btree nadstavba
wc = ap_init # wifi > access point
...
```

```
*_init() # obecná inicializace, příprava, většinou
vytváří instanci objektu, se kterou dále pracujeme
(instance má metody a vlastnosti definované
objektem), možno použít "komplexní init" a report
o inicializovaných objektech:
octopus_init() # provede *_init a test se všemi
# nastavenými periferiemi
```

Blikáme - pípáme

```
from util.led import Led
led = Led(2)
from util.rgb import Rgb
ws = Rgb()
ws.color(RED) # RED, GREEN, BLUE, BLACK = tma
ws.simple_test() # projede R G B
```

```
piezzo = Buzzer(18)
piezzo.beep()
```

Displeje

```
d7 = disp7_init()
d7.show(123)
```

```
oled = oled_init()
oled.fill(1) # vyplnění bílou
oled.show() # přesun na displ.
oled.hline(x, y, h, color) # horizontal
oled.vline(x, y, h, color) # vertical
```

Webový server + IDE a simple control

```
wc = w() # připojení k nastavené WiFi
wc = ap_init() # nebo spuštění "access point"
web_server() # z prohlížeče na IP adrese
```

Vstupy

```
B0 = button_init(0) # tlačítko
button(B0)
if button(B0)[0]: # vrací dvě hodnoty (0 a 1)
    ... # "debounce" proti zámkitu
t = temp_init() # dallas teploměr
get_temp(t[0],t[1]) # vrací z pole všech
get_temp(*t) # alternativa
```

```
d7.show(get_temp(*t))
```

```
from util.rgb import wheel
ws.color(wheel( int((get_temp(t[0],t[1])-20)*20)))
```

```
from util.analog import Analog
photo = Analog(36) # AD převodník na pinu
a = photo.read() # čte analog. hodnotu
photo.get_adc_aver(20) # průměr z 20-ti načtených
```

Mechatronika

```
from util.servo import Servo
s1 = servo_init()
s1.set_degree(90)
```

Examples

```
ls("examples") # výpis všech ukázek
import examples.clock # spuštění hodin
file_copy("examples.clock") # hodiny do main.py
```

```
from examples.twitter_rgb import twitter_rgb
ws.color(twitter_rgb())
```

octopus() - starší verze ver: 0.90- 2019/09
připravujeme: o = octopus()

zdrojové kódy -> <https://github.com/octopusengine/octopuslab/tree/master/esp32-micropython>

Struktura Env. s globálními hodnotami a pointry
na instance objektů

