Commentaires

```
# ceci est un commentaire
Cnt = 10 # init variable

''' 
Cette section est commentée
'''
```

Opérateurs de comparaison

```
==      if val == input         # si val égal ...
!=      if val != input # si val différend ...
>       if val > input  # si val plus grand ...
>=      if val >= input         # si val plus grand ou égal
<       if val < input  # si val plus petit ...
<=      if val <= input         # si val plus petit ou égal
```

Opérateurs

```
=       # a = 3
+=      # a+=2          (a=a+2)
-=      # a-=1          (a=a-1)
*=      # a*=3          (a=a*3)
/=      # a/=3          (a=a/3)
**=     # a**=2         (a=a^2)
```

Opérateurs de bits

```
&       # AND
|       # OR
^       # ExOR
~       # NOT
<<      # décal. à gauche
>>      # décal. à droite
```

Fonction

```
def Fonction(a,b) :
        opérations ...
        return(variable)
```

If then else

```
if condition :
        opération1
        opération2
else :
        opération 3
        opération 4
```

while

```
while condition :
        opération1
        opération2
```

for

```
for variable in sequence :
    opérations...
for x in range(0,5) :
        Y = x+z
for lettre in "club"
   print (lettre)          # 'c'   'l'   'u'   'b'
for x in range(0,15,5)
   print (x)               # '0'  '5'  '10'
```

continue

```
while condition :
        opération1
        if condition :
                continue
        opération2
# si condition, ne fait pas
        opération2
```

break

```
while condition :
        opération1
        if condition :
                break
        opération2
# si condition, quitte le
while
```

Représention

```
Décimal : 23
Hexa : 0x35
Binaire : 0b0011
```

Variables numériques

```
int       : entier signé : -200
long : long entier: 24356279L
float : virgule flottante : 2.45
complex : nb complexe 3.3+2.1j
```

Conversions numériques

```
int(x)          int(3.14) -> 3
float(x)        float(3) -> 3.0
abs(x)          abs(-3) -> 3
bin(x)          bin(24) ->'0b10111'
hex(x)          hex(24) -> '0x17'
math.ceil(x)    ceil(2.4) -> 3
math.floor(x)   floor(2.8) -> 2
round(x)        round(3.1) -> 3
                round(3.8) -> 4
round(x,n)      round(3.145,2) ->3.14
pow(x,y)        pow(2,3) ->2^3=8
sqrt(x)         racine carrée de x
```

Version 02.2018, L. Francey

Chaînes de caractères

```
string chaine = "Microclub"
S = chaine[0]          # s = M
S = chaine[2:4]        # s = cr
S = chaine[5:]         # club
S = chaine*2           #MicroclubMicroclub
S = chaine + '2018' #Microclub2018
S = chaine[1:2] + 'gros'      #Migros

\n      nouvelle ligne
\r      retour à la ligne (0x13)
\       form feed (0x10)
\t      tab
\a      beep
```

Fonctions de chaînes de caractères s= "Federer", c=" CLUB2 "

```
count(str,deb,fin)    # s.count('e',0,6) -> 3
find(str,deb,fin)     # s.find('e',0,6) -> 1 (index du premier)
                      # s.find('x',0,6) -> -1
len(str)              # len(s) -> 7
Isalpha()             # s.isapha() -> True
Islower()             # c.islower() -> False
Isupper()             # c.isupper() -> True
lower()               # s.lower() -> 'federer'
upper()               # s.upper() -> 'FEDERER'
lstrip()              # c.lstrip() -> 'CLUB2  '
rstrip()              # c.rstrip() -> '  CLUB2'
```

Fonctions de listes lst = ['A','B','C','D']

```
del([i:j])        # efface les éléments de i à j-1, del(lst[2]) -> ['A','B','D']
list.append       # ajoute l'élément à la fin de la liste
cmp(n1,n2)        # compare les éléments n1,n2)
len(li)           # longueur de la liste li
max(li)           # retourne l'élément valeur max de la liste
min()             # retourne l'élément valeur min de la liste
list.count(x)     # retourne le nb de foix que x figure dans liste
list.index(x)     # =index de la première occuence de x
list.insert(i,x) # insert x à la position i de la liste
list.remove(x) # supprimer l'élément spécifié
list.revers()     # inverse la liste
list.sort()       # trie la liste
```

import time

```
time.localtime()      # (YYYY, MM, DD, hh, mm,
                        ss, week day, year day)
time.sleep(x) # pause x secondes
time.sleep_ms(x)    # pause x milli secondes
time.sleep_us(x)    # pause x micro secondes
```

Fichiers

Open
Close
Read
Write
R
R+
W
W+
A
a+

import os (librairie pour fichiers)

Rmdir :
Chdir
Mkdir
Getcwd
Remove
Listdir
Rename
Stat
statvfs

import machine (reset)

Ret = machine.reset() : reset le module
Ret = machine.PWRON.RESET
machine.HARD_RESET
machine.WDT_RESET
machine.DEEPSLEEP_RESET
machine.SOFT_RESET
machine.sleep() stop le CPU mais pas WLAN
machine.unique_id() retourne S/N

import network

```
net = network.WLAN(network.STA_IF)
                    STA IF ou AP_IF
net.isconnected()
net.disconnect()
net.active()
net.connect("SSID","Pass")
net.ifconfig()
```

import machine (Pin)

machine.Pin(id, mode, pull, value, drive, alt)
id : no de la pin
mode : Pin.IN ou Pin.OUT
pull : Pin.None / PULL_UP / PULL_DOWN
value : valeur lors de l'initialisation
drive : Pin.LOW_POWER/ MED_POWER/HIGH_POWER
alt : spécifie une fonction alternative de la pin
Pin.value([x]) : lit ou écrit la valeur sur une pin
Pin.out_value() :
Pin.toggle() :
Pin.id() :
Pin.mode([mode])

interrupt

Pin.irq(handler, trigger, prority, wake)
Handler fonction appellée lors de l'interuption
Trigger : Pin.IRQ_FALLING / IRQ_RISING/
IRQ_LOW_LEVEL / IRQ_HIGH_LEVEL
Priority : niveau de priorité
Wake : machine.IDLE / SLEEP / DEEPSLEEP

ADC

ADC(pin) : initialisation pin analog.
Read() : lecture de la valeur analog.

Version 02.2018, L. Francey