

Procés per a posar en marxa l'ESP8266 amb l'Arduino Nano

Quan apareix `Sending command: AT` s'ha de fer reset a l'ESP8266 tocant el terminal, de la resistència de valor 2K2, més proper al logotip d'electronic.cat

```
jordi@eCat:~/Documents/electronics.cat/conferencies/20151028/code/python$ python esp8266server.py
/dev/ttyUSB0 ONOF5AC 2445418455
Sending command: AT
[Vendor:www.ai-thinker.com Version:0.9.2.4]
```

```
ready
Command result: ready
Sending command: AT+CWMODE=1
no change
Command result: no change
Sending command: AT+CWLAP
+CWLAP:(2,"MOVISTAR_9FCA",-89,"f8:8e:85:d9:9f:cb",1)
+CWLAP:(2,"MOVISTAR_F73D",-91,"f8:8e:85:e6:f7:3e",1)
+CWLAP:(4,"ONOF5AC",-55,"c0:3f:0e:c2:f5:ac",2)
+CWLAP:(4,"Valledupar",-77,"84:9c:a6:47:a7:33",2)
+CWLAP:(4,"INT_U9T7H9",-80,"10:fe:ed:9c:16:6a",2)
+CWLAP:(0,"_AUTO_ONOWiFi",-56,"c2:3f:0e:c2:f5:ad",2)
+CWLAP:(0,"_ONOWiFi",-56,"c2:3f:0e:c2:f5:ae",2)
+CWLAP:(3,"TRAC",-93,"28:94:0f:f9:be:b6",6)
+CWLAP:(4,"Orange-C341",-84,"88:03:55:aa:c3:43",7)
+CWLAP:(2,"vodafoneppjpl",-81,"20:2b:c1:37:22:c3",8)
+CWLAP:(3,"Ninecols",-85,"20:c9:d0:1b:2d:69",11)
+CWLAP:(4,"InOutTravel",-93,"0a:18:d6:0b:61:f1",11)
+CWLAP:(0,"Kubi_CityWifi_Fast_Internet",-76,"a4:6c:2a:68:ea:71",11)
+CWLAP:(0,"Barcelona WiFi",-76,"a4:6c:2a:68:ea:73",11)
+CWLAP:(1,"TechSales",-59,"e0:cb:4e:59:d9:23",13)
```

```
OK
Command result: OK
Sending command: AT+CWJAP="ONOF5AC","2445418455"
Command result: AT+CWJAP="ONOF5AC","2445418455"
```

```
Sending command: AT+CIFSR
busy p...
```

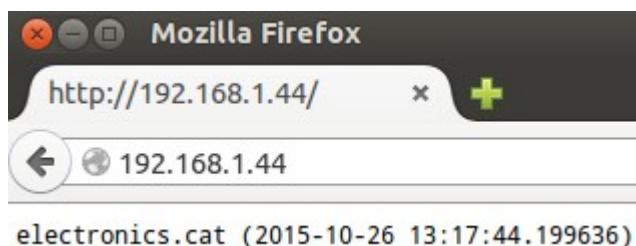
```
OK
Command result: OK
Sending command: AT+CIPMUX=1
```

```
OK
Command result: OK
Sending command: AT+CIPSERVER=1,80
```

```
OK
Command result: OK
Sending command: AT+CIFSR
192.168.1.44
```

```
OK
Command result: OK
```

Anant al navegador (una sola crida, per a que torni a funcionar s'ha de repetir el procés):



Codi per Arduino **serialPassWiFi.ino** :

```
#include <SoftwareSerial.h>

SoftwareSerial mySerial(10, 11); // RX, TX

void setup(){
  Serial.begin(9600);
  while (!Serial) {
    ; // wait for serial port to connect. Needed for Leonardo only
  }
  mySerial.begin(9600);
}

void loop(){
  if (mySerial.available())
    Serial.write(mySerial.read());
  if (Serial.available())
    mySerial.write(Serial.read());
}
```

Codi en Python **esp8266server.py** :

```
#!/usr/bin/python
# -*- coding: iso-8859-15 -*-

import logging
import sys, serial
from time import *
import datetime, string

def enum(**enums):
    return type('Enum', (), enums)

Status = enum(ERR=['ERROR', 'Fail'], OK=['OK', 'ready', 'no change', 'SEND OK'], BUSY='busy', LINK='Link')

def send_cmd( sCmd, waitTm=1, retry=5, delay=1):
    lp = 0
    ret = ""

    print( "Sending command: %s" % sCmd )

    for i in range(retry):
        ser.flushInput()
        ser.write( sCmd + "\r\n" )
        ret = ser.readline()      # Eat echo of command.
        sleep( 0.2 )
        while( lp < waitTm or 'busy' in ret):
            while( ser.inWaiting() ):
                ret = ser.readline().strip( "\r\n" )
                print( ret )
                lp = 0
            if( ret in Status.OK ): break
            #if( ret == 'ready' ): break
            if( ret in Status.ERR ): break
            sleep( delay )
            lp += 1

        sleep(delay)
        if( ret in Status.OK ): break

    print( "Command result: %s" % ret )
    return ret

def send_response(response, cid='0'):
    # no need to wait for response between these commands (or retry), so just send them out
    # ser.flushInput()
    ser.write( "AT+CIPSEND=" + cid + "," + str(len(response)) + "\r\n" )
    sleep(0.3)
    ser.write( response + "\r\n" )
    sleep(0.3)

    send_res = False
```

```

        for i in range(100):
            while( ser.inWaiting() ):
                ret = ser.readline().strip( "\r\n" )
                # print ret
                if( ret == Status.OK[3] ):
                    # print "send ok!"
                    send_res = True
            if send_res: break
            sleep(0.1)

        sleep(0.3)
        ser.write( "AT+CIPCLOSE=" + cid + "\r\n" )
        sleep(0.3)

def process_request(response):
    has_link = False
    cid = '0'
    while( ser.inWaiting() ):
        ret = ser.readline().strip( "\r\n" )
        # print ret
        if (ret in Status.LINK):
            has_link = True
        ipd_str = '+IPD,'
        if (ipd_str in ret):
            cid = ret[ret.find(ipd_str) + len(ipd_str)]

    if has_link:
        # process response
        send_response(response, cid)

if len(sys.argv) != 4:
    print "Usage: esp8266test.py port ssid password"
    sys.exit()

port = sys.argv[1]
speed = 9600
ssid = sys.argv[2]
pwd = sys.argv[3]
p = 80

ser = serial.Serial(port,speed)
if ser.isOpen():
    ser.close()
ser.open()
ser.isOpen()

send_cmd( "AT" )
send_cmd( "AT+CWMODE=1" ) # set device mode (1=client, 2=AP, 3=both)
send_cmd( "AT+CWLAP", 30) # scan for WiFi hotspots
send_cmd( "AT+CWJAP=\""+ssid+"\", \""+pwd+"\"", 5 ) # connect
send_cmd( "AT+CIFSR", 5) # check IP address

send_cmd( "AT+CIPMUX=1" ) # multiple connection mode
send_cmd("AT+CIPSERVER=1," + str(p))
send_cmd( "AT+CIFSR", 5) # check IP address

# process requests
while (1):
    process_request("electronics.cat (" + str(datetime.datetime.now()) + ")")
    sleep(0.3)

ser.close()

```