SUBIR PROGRAMAS MICROPYTHON

Conectando nuestro micropython remotamente

MICROPYTHON Y WIFI

- Veremos como subir programas desde nuestro PC al disco iterno del esp8266
- Describiremos dos metodos para subir: mediante comandos externos o accediaendo wia WEB y WIFI.
- Veremos que incluso via inalambrica la console remota se comporta exactamente de la misma forma.
- Veremos como ejecutar estos programas una vez subidos..

MATERIAL REQUERIDO.

[product id=»12333″]	 Un cable USB a microUSB, como el de cualquier teléfono móvil

SUBIENDO PROGRAMAS MICROPYTHON

Hasta ahora hemos visto algunas instrucciones básicas con **MicroPython** y cómo usarlas en el intérprete interactivo o **REPL**, que sin duda resulta muy interesante pero poco práctico si queremos almacenar un programa en nuestro módulo **Esp8266**.

Por ese motivo tenemos que ver como volcar estos programas que vamos haciendo, al disco interno del **ESP8266** de modo que podamos ejecutarlos sin necesidad de escribir las instrucciones una a una.

En nuestros PC la cosa es muy fácil, basta con guardar el programa en disco y llamarlo después. Pero con el módulo NodeMCU/esp8266 el tema es un poco más complicado, porque no disponemos de un sistema operativo que nos ayude.

Para empezar no disponemos de un editor como el que disponemos en el IDLE de Python, y tampoco tenemos acceso desde el exterior a ese disco interno que veíamos en alguna sesión previa. Por eso vamos a necesitar una herramienta externa que nos permita volcar los programas y además usar un editor de programas externo.

Resumiendo, que vamos a necesitar un editor externo por una parte y una forma de volcar los programas que hagamos al interior del disco interno para ejecutarlos.

EL EDITOR EXTERNO

La opción más obvia para editar fuera tus programas es usar cualquiera de los editores habituales como el propio incluido en el Python 2.7 que ya instalamos.

Es cómodo y rápido porque si ya has usado Python estás acostumbrado a él y no necesitas nada para empezar. Pero hay muchos otros disponibles si por cualquier razón no te apetece usarlo, como por ejemplo el notead++ (https://notepad-plus-plus.org/download/v7.5.1.html) , que es sencillo, gratis y de dominio público.

Pero imagínate, volviendo a nuestro problema original que queremos escribir un programa que haga el blinking led en nuestro **NodeMCU** (*Que asombrosa imaginación*)

Podríamos escribir algo así en nuestro editor elegido:

```
import machine, time

def toggle(p):
    p.value(not p.value())

pin = machine.Pin(2, machine.Pin.OUT)
while True:
    toggle(pin)
    time.sleep_ms(300)
    print("Running...")
```

Lo guardamos en disco con el nombre nada original de toggle.py

El problema ahora es que tenemos este programa en nuestro disco duro y tenemos que pasarlo al módulo esp8266, pero nos surge un pequeño problema ¿*Cómo hacerlo*?

VOLCANDO PROGRAMAS AL ESP8266

No hay un modo directo de volcar estos programas al módulo, pero la buena noticia es que algunos benefactores de la humanidad han escrito unos programas que nos permiten hacerlo.

La mala es que son muy al estilo **Linux** (*Con ventanas de comandos y todo eso que tanto os gusta)* que suelen horrorizar a los usuarios promedio de Windows que piensan con convencimiento que en el principio los ordenadores ya nacieron con ventanas gráficas y que tienen dificultades en comprender porque alguien sigue usando un sistema tan oscuro y rarito.

Por tanto, hay básicamente dos soluciones, una es usar una herramienta de volcado (*Con comandos y esas cosas*) y otra más fácil usando WIFI y haciendo el volcado con un navegador web directamente.

No dudo de que la mayoría os habréis apuntado enseguida a esta segunda opción (*Que os conozco)* pero las pruebas me han demostrado que los comandos son más fiables que el modo **WERPL** vía WIFI ya que aparentemente solo me funciona los días pares y me deja siempre con un regusto raro en la boca.

- *Tenéis que entender que todo este asunto del **MicroPython** está aún en fase beta y no todo funciona como nos gustaría, pero también es un poco esa la diversión, ir de pioneros mola.*
- *Pero también conviene recordar que a los pioneros se les reconoce por los flechazos que los indios les clavan en la espalda*.

Así que vamos a empezar por las utilidades de línea de comandos que nos permiten volcar programas al **NodeMCU**

VOLCANDO CON COMANDOS

La primera herramienta que intente usar fue **Ampy** (https://learn.adafruit.com/micropython-basics-load-files-and-run-code/install-ampy)de nuestros amigos de Adafruit. Esta gente me gusta. Tienen un compromiso con la comunidad general y dedican mucho esfuerzo a proporcionarnos herramientas y librerías gratuitas, así que os animo a que les apoyéis siempre que podáis.

El problema que me encontré fue que no fui capaz de hacer correr Ampy , porque todo eran problemas que aparentemente solo me pasaban a mí.

Tened en cuenta que el desarrollo de todas estas herramientas se suele hacer desde **Linux** y por eso los port a Windows no están todo lo finos que debieran (*Y gracias porque se molestan*)

Así que al final me decante por **mpfshell** (https://github.com/wendlers/mpfshell)que me funcionó casi a la primera y sigue funcionándome hoy y esta es la versión que vamos a usar.

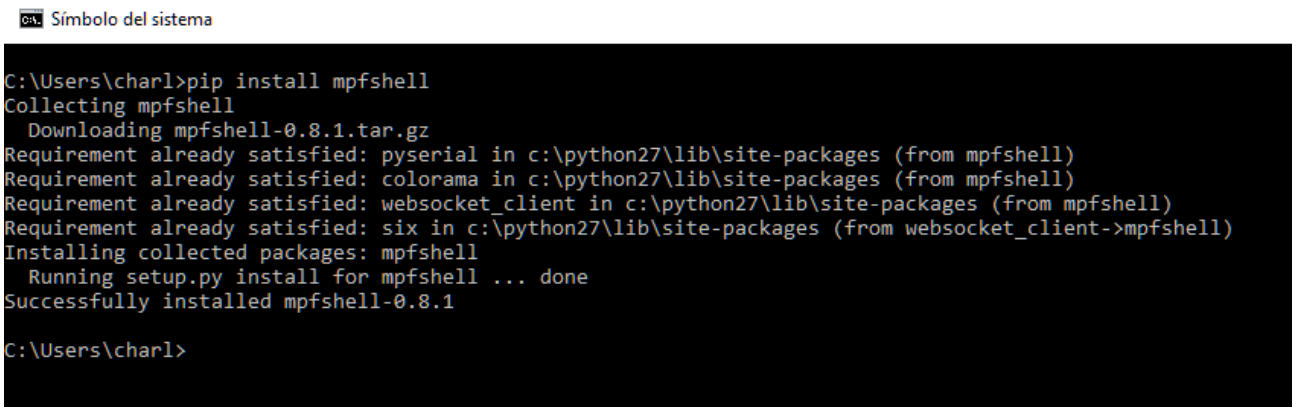
Vamos a empezar con un ejemplo de todo esto. Cread un programa Python, con vuestro editor favorito, llamado toggle.py que contenga estas líneas *(Creo que ya lo pusimos arriba)*:

```
import machine, time

def toggle(p):
    p.value(not p.value())

pin = machine.Pin(2, machine.Pin.OUT)
while True:
    toggle(pin)
    time.sleep_ms(300)
    print("Running...")
```

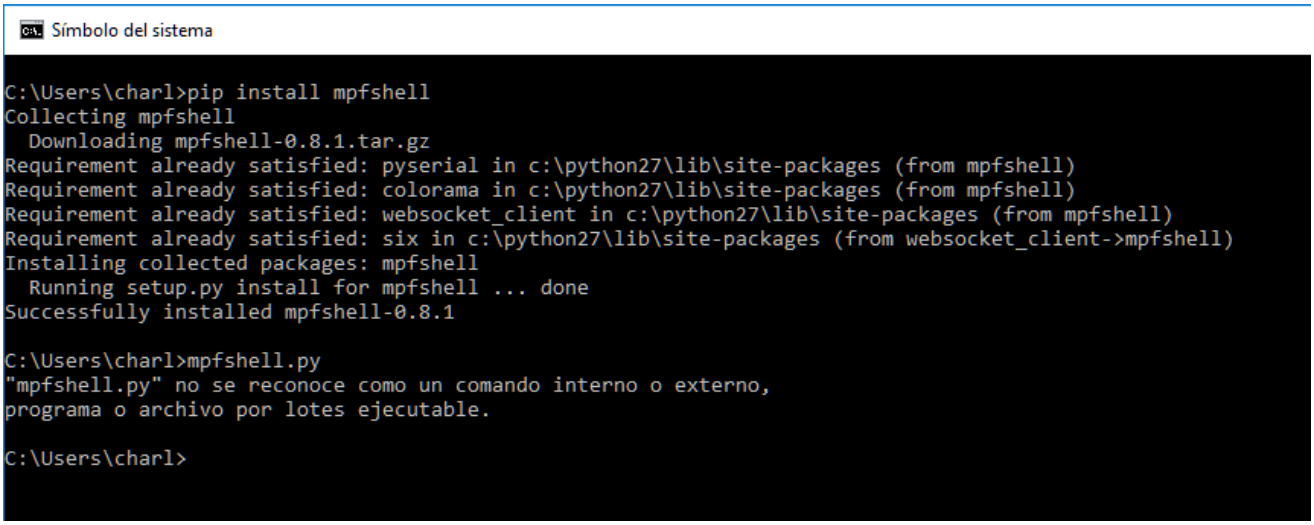
Simplemente harán encender y apagar la luz interna cada medio segundo. Vamos a ver como lo volcamos y ejecutamos en el **NodeMCU**.

Lo primero es instalar el mpfshell, para ello abre una ventana de comandos y escribe:

```
pip install mpfshell
```



Como veis tenemos un mensaje de éxito en la instalación, pero si invocáis el **mpfshell** ahora encontrareis un bonito error:



El motivo es que ha instalado correctamente el programa en un directorio que no está en el path y no lo encuentra. Se puede arreglar sin más que copiar el mpfshell.py desde donde está ahora al directorio scripts de Python que si está en el path:

```
copy c:\Python27\Lib\site-packages\mp\mpfshell.py \Python27\Scripts
```

Y ahora sí que lo puedes ejecutar y verás los colorines del Shell:

```
C:\Users\charl>copy c:\Python27\Lib\site-packages\mp\mpfshell.py \Python27
        1 archivo(s) copiado(s).

C:\Users\charl>mpfshell.py

** Micropython File Shell v0.8.1, sw@kaltpost.de **
-- Running on Python 2.7 using PySerial 3.4 --

mpfs [/]>
```

Pulsa [CTRL] + C para salir. Ahora ya podemos volcar nuestro humilde toggle.py

Vamos a hacer el ejemplo completo de volcar el programa toggle.py a nuestro modulo. Para ello lo primero será que te vayas al directorio donde tienes el fichero. En mi caso c:\python27 y desde allí llamamos al mpfshell:

✔ ✔ *Fijate que hay que escribir mpfshell.py y no simplemente mpfshell..*

El Shell nos ofrece toda una colección de ayudas para tratar con el disco interno del Esp8266. EL comando help te da una lista de ordenes disponibles. No es el momento de entrar a hablar de ellas pero vamos a usar algunas.

Para empezar tenemos que conectar con el modulo a través de la puerta com correspondiente en mi caso el com7, pero vosotros reemplazad esto por lo que corresponda:

```
open com7
ls
```

```
Símbolo del sistema - mpfshell.py

C:\Python27>mpfshell.py

** Micropython File Shell v0.8.1, sw@kaltpost.de **
-- Running on Python 2.7 using PySerial 3.4 --

mpfs [/]> help

Documented commands (type help <topic>):
========================================
EOF  cd     exec  get   lcd  lpwd  md    mput  mrm   put  repl
cat  close  exit  help  lls  ls    mget  mpyc  open  pwd  rm

mpfs [/]> open com7
Connected to esp8266
mpfs [/]> ls

Remote files in '/':

       boot.py

mpfs [/]>
```

En mi caso solo hay un fichero llamado boot.py que si recordáis lo que comentamos en la ultima sesión se ejecuta siempre en el arranque del modulo. *¿Quieres ver lo que contiene?* Nada mas fácil:

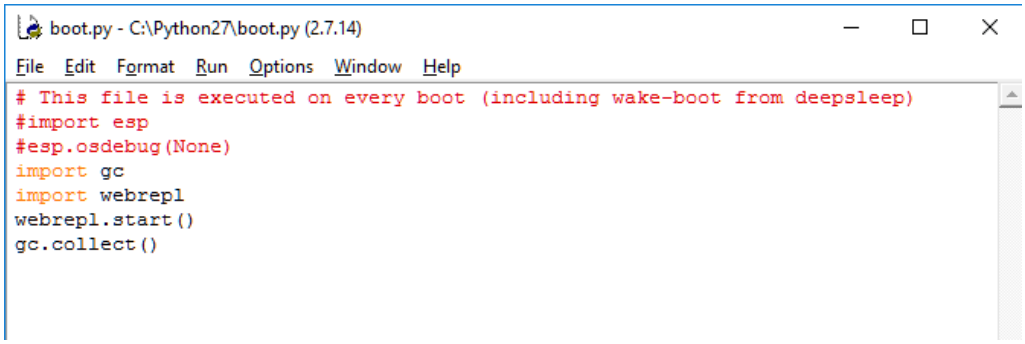
```
get boot.py
```

Esto te copiara el fichero boot.py a tu directorio actual, que en mi caso es \python27

```
mpfs [/]> open com7
Connected to esp8266
mpfs [/]> ls

Remote files in '/':
```

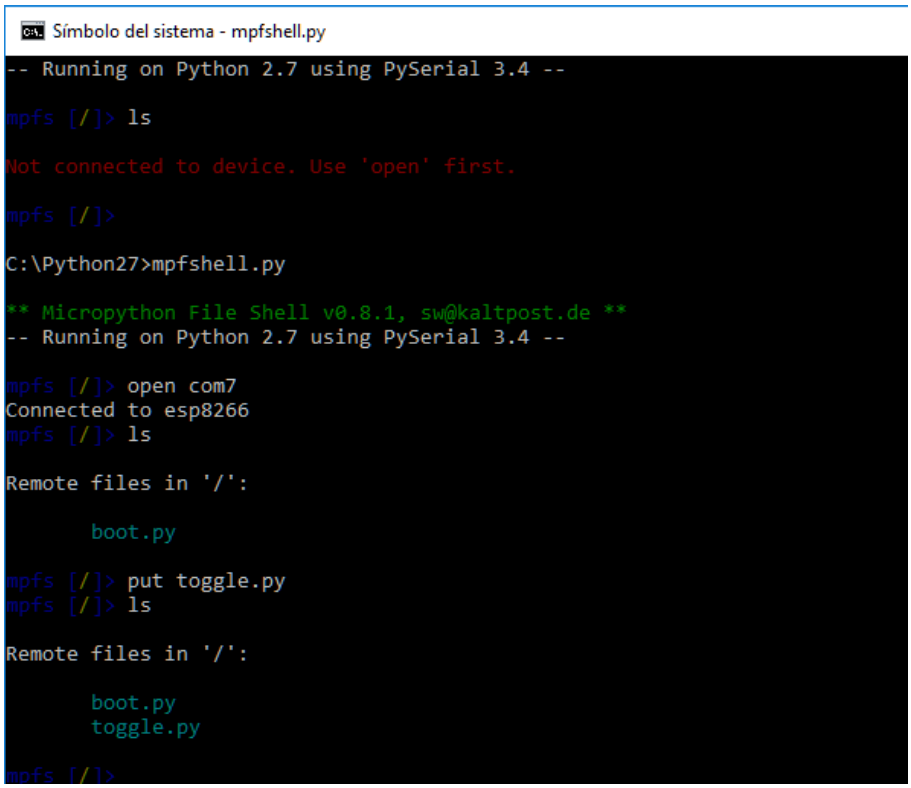
```
      boot.py

mpfs [/]> get boot.py
mpfs [/]>
```

Si quieres ver su contenido basta con que lo edites desde tu pc:



No está mal pero lo queríamos es subir programas no bajarlos, dir mas de uno. Pues igual de fácil:

```
put toggle.py
ls
```



El comando put, nos pasa el fichero en el directorio actual al disco interno. Tenemos que hacer un par de consideraciones aquí:

- *No he conseguido traspasar ficheros mas que del directorio actual, es decir aquel directorio del pc desde el que invocamos el mpfshell*
- *Cuando intento hacer un put con un path de Windows, el Shell me lo deniega tranquilamente diciendo que es un path invalido (Esperaremos a la siguiente versión) y por eso conviene invocarlos desde el directorio desde el que contiene el fichero.*

Una vez transferido el fichero toggle.py, ya podemos verlo desde el **REPL** del modulo (*Siempre y cuando cierres antes el mpfshell, porque sino va a bloquear el uso de la puerta serie del modulo*) y normalmente conviene resetear el modulo.

```
>>>
>>>
>>> import os
>>> os.listdir()
['boot.py', 'toggle.py']
>>>
```

Para ejecutar toggle.py basta con que lo invoques con un import. Cargara el programa y lo ejecutará de inmediato, el led interno comenzara a parpadear (*Que es lo único que hace nuestro programa, ademas de sacar un mensaje repetitivo.*)

```
import toggle
```

```
COM7 - Tera Term VT
File  Edit  Setup  Control  Window  Help
>>>
>>> import os
>>> os.listdir()
['boot.py', 'toggle.py']
>>> import toggle
Running...
Running...
Running...
Running...
Running...
```

El programa está en un ciclo eterno de imprimir “Running” y hacer parpadear el led. Si quieres pararlo basta con que interrumpas la ejecución con [CTRL] + C.

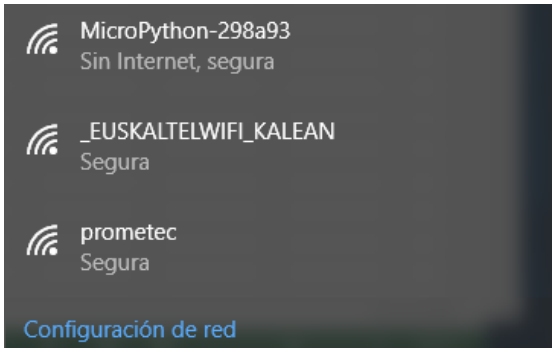
```
Running...
Running...
Running...
Running...
Running...
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
  File "toggle.py", line 11, in <module>
KeyboardInterrupt:
>>>
```

- *Si queremos volver a ejecutar el programa lo tenemos mal, porque aunque repitas el import, micropython considera que ya esta cargado y por tanto no lo vuelve a cargar ni ejecutarlo y no conozco forma de volver a lanzarlo, sino es cn un reset pulsando el botoncito en la placa y repitiendo la operación anterior.[/fancy-*ul]

CONEXIÓN VIA WIFI

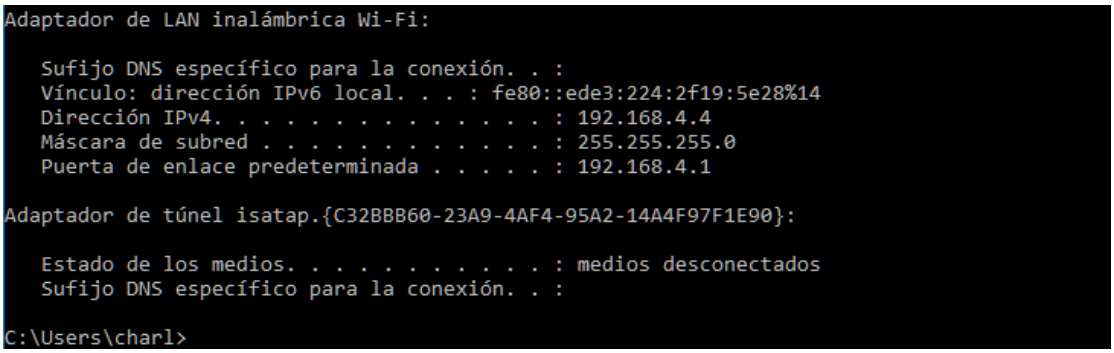
Vamos con la parte que os gusta, sin cables y a lo loco. Recordad que ya hemos comentado antes, que por defecto **MicroPython** arranca el modulo en modo ACCES POINT, es decir que podemos conectarnos a él como si fuera un Router, y de hecho es lo primero que vamos a hacer.

Buscad con la WIFI un punto de acceso que se llame algo así como MicroPython-xxxx y conéctate:



Te pedirá una clave de acceso. Si es la primera vez, la clave es: micropythoN (*Con la última N mayúscula, atento*). Para comprobar que te has conectado abre una ventana de comandos y escribe:

```
ipconfig
```

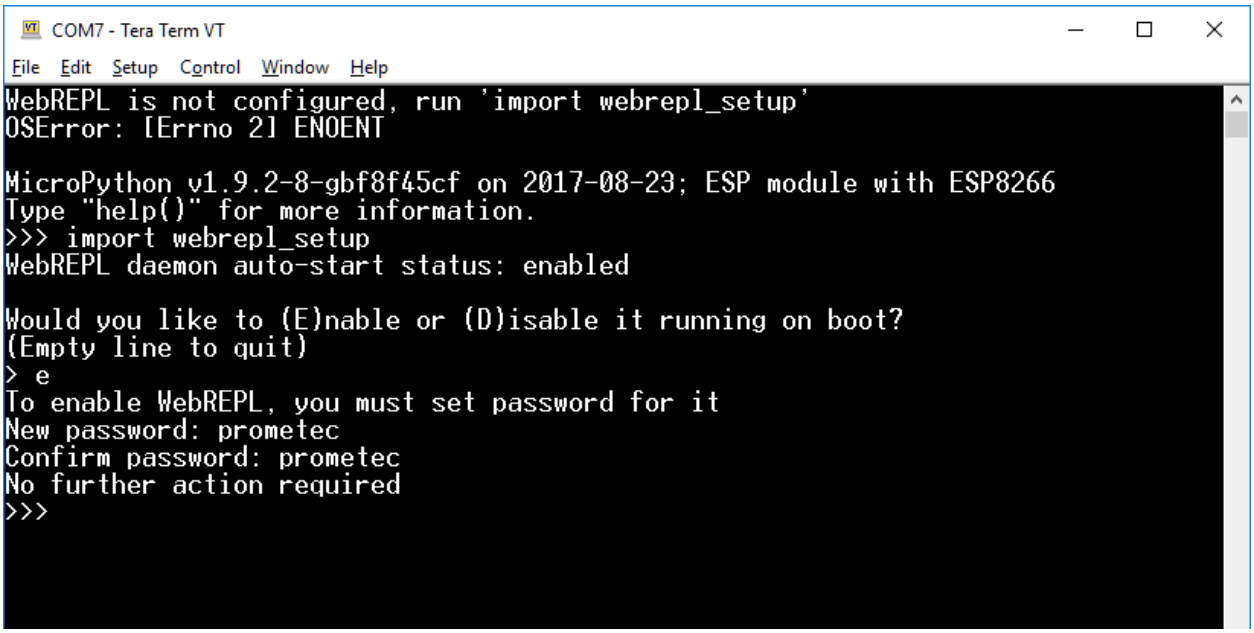


Veras que tu Router por defecto es 192.168.4.1 y tu IP una en el mismo rango. Es buena señal. Para conectar usando el navegador necesitamos descargar y configurar el WEREPL

Para ello ejecuta desde micropython, el siguiente comando:

```
import webrepl_setup
```

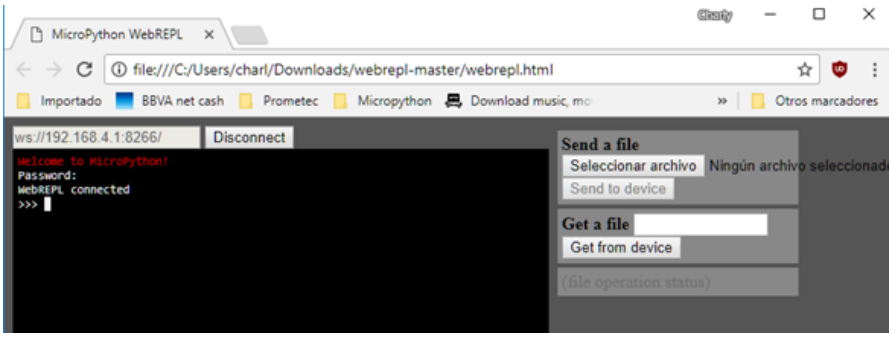
Esto lanza una utilidad de configuración del acceso vía WIFI, que te pide autorizar el asunto (Con E de Enable) y después te pedirá una clave de acceso:



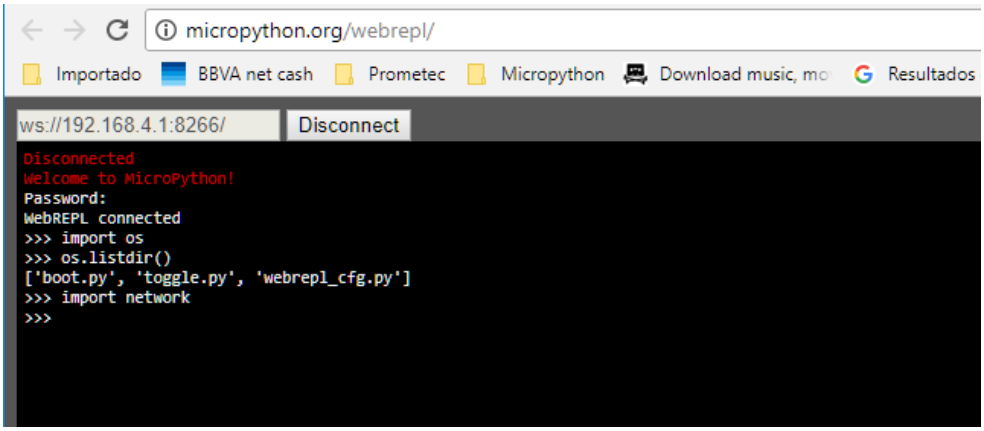
Y ahora solo queda ya descargar la versión local para acceder vía web aquí, o bien descargarte la versión local aquí (https://github.com/micropython/webrepl).

Si arrancas cualquiera de las dos versiones desde el navegador, te aparecerá una página como las que ves a continuación y veras que en la barra de direcciones apunta a ws://192.168.4.1:8266, que es la dirección por defecto que adopta el **micropython** y el puerto 8266 (*Difícil de recordar*) para acceder remotamente.

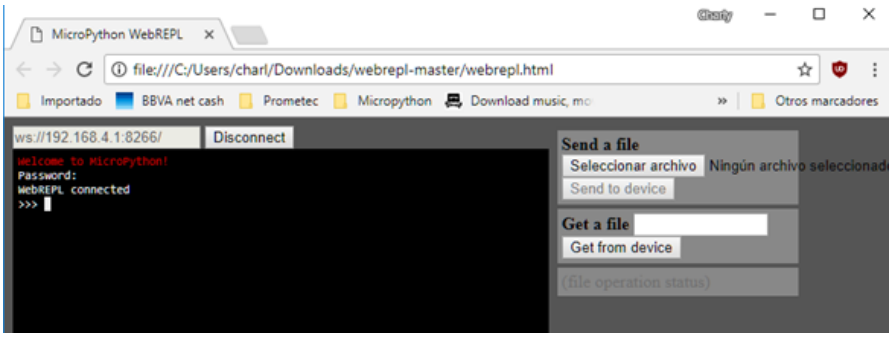
Cuando des al botón Connect:



Te pedirá el password que has definido antes y una vez superado el trámite, estás en la misma situación que cuando nos conectábamos por la puerta serie. Puedes usar los mismos comandos y responderá exactamente del mismo modo.



Fíjate ahora en la parte de la derecha de la página. Veras los botones de seleccionar archivo (*Que te permite buscar ficheros en cualquier directorio de tu PC)* y el botón Send to device, que una vez seleccionado el archivo, te lo sube tranquilamente al disco del **Nodemcu / ESP8266**, y el botón Get from device, que te lo copia del disco interno a tu disco local si lo necesitas



Así que de este modo ya estamos condiciones de subir nuestros programas al disco interno del **Esp8266** cuando lo necesitemos.

RESUMEN DE LA SESIÓN

- Hemos visto como subir programas al disco interno de nuestro nodemcu/esp8266.
- Vimos que se puede hacer mediante comando como mpfshell o bien mediante wifi y una pagina web.
- Podemos asi subir nuestros programas y ejecutarlos, incluso desde una consola remota.

Anterior ()

Siguiente ()

(https://www.facebook.com/prometecnet-1541207239447373/timeline/)