Comprobar que el ESP8266 está a 9600 baudios por defecto

Tal y como se aconseja **aquí**, el ESP8266 permite trabajar la wifi a velocidades de 9600 a 115200 baudios pero para evitar problemas se aconseja bajar a 9660.

Esta comprobación no es necesario hacerlo siempre, con una sola vez es suficiente, el ESP8266 se queda grabado a esa velocidad.

Comunicarnos con el ESP8266

El ESP8266 se comunica con nuestro Arduino por un puerto serie. Nosotros ya utilizamos un puerto serie en el Arduino, el que nos permite la comunicación de Arduino con nuestro ordenador por el puerto COM de nuestro ordenador y por D0 y D1 de nuestro Arduino. Con la librería **SoftwareSerial.h** es posible crear otro puerto serie adicional. Luego vamos a grabar este código extraído de https://docs.arduino.cc/tutorials/communication/SoftwareSerialExample y nos permitirá una vez creado el puerto serie adicional, comunicarnos con él

Tenemos que:

1. descargar el programa Arduino IDE de https://www.arduino.cc/en/software
2. Instalarlo y ejecutarlo
3. seleccionar la placa Arduino Uno
4. seleccionar el puerto donde está conectado

Para estos 4 pasos, te recomendamos que vea esta página.

Pega este código

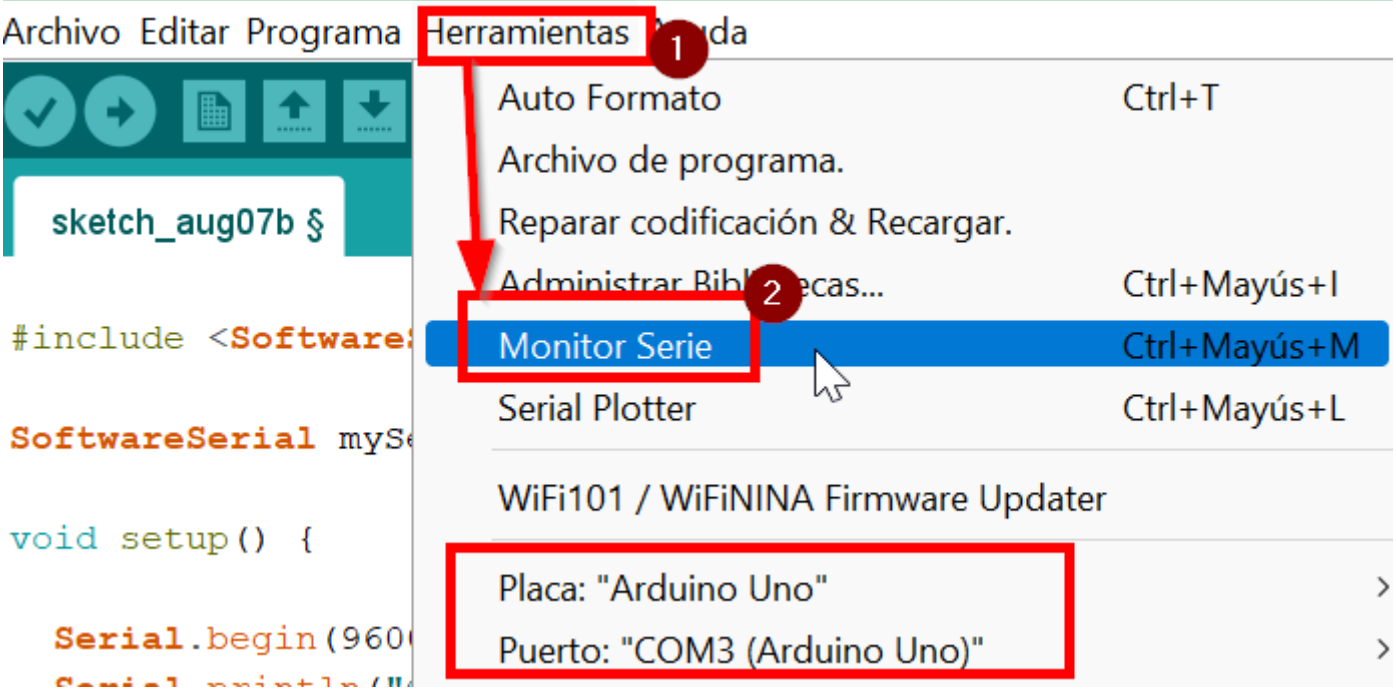
```
#include <SoftwareSerial.h>
```

```
SoftwareSerial mySerial(3, 5); // Esto crea un nuevo puerto serie en RX, TX en los pines 3 y 5



void setup() {

  Serial.begin(9600);    // pone la velocidad a 9600 baudios DEL PUERTO SERIE COM no el que vamos a crear nuevo
  Serial.println("Cónsola serie. Teclear el comando AT en la línea de arriba y pulsa Enviar... a ver si sale OK");

  // set the data rate for the SoftwareSerial port
  mySerial.begin(9600);     //esto fija a 9600 la comunicación entre ESP01 y tu ordenador, no la velocidad wifi del ESP01

}

void loop() { // run over and over
  if (mySerial.available()) {
    Serial.write(mySerial.read());
  }
  if (Serial.available()) {
    mySerial.write(Serial.read());
  }

}
```

Y lo subimos al Arduino

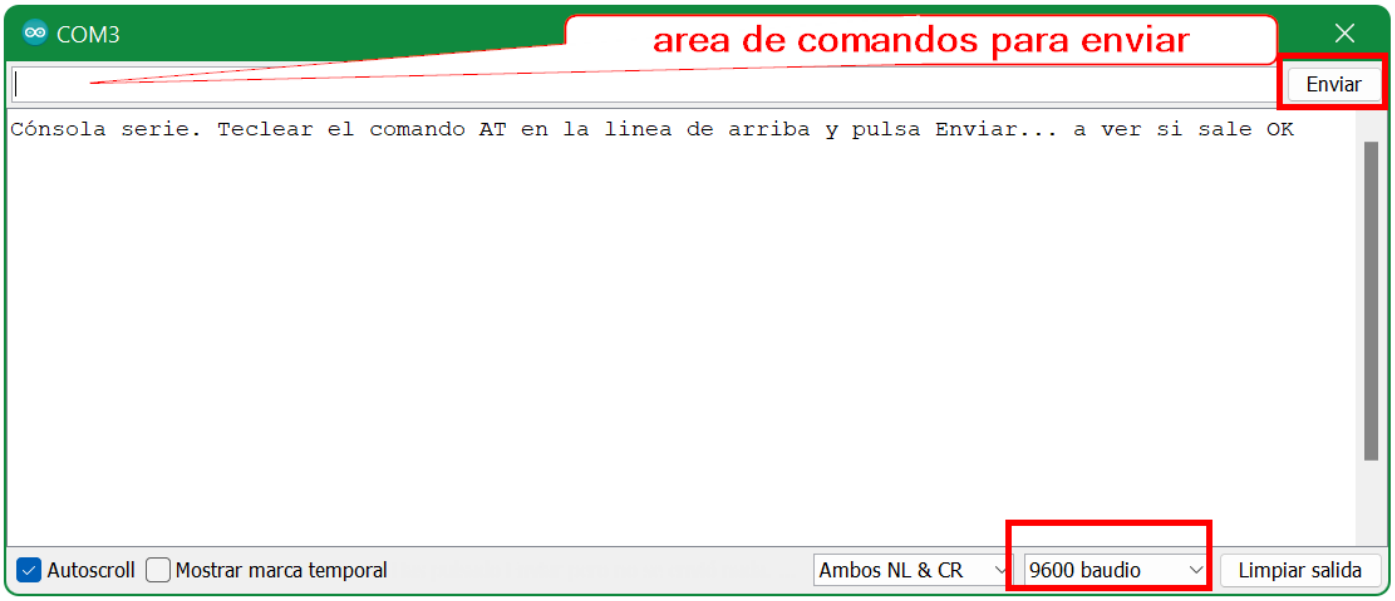
> Este programa **NO NOS HA CONFIGURADO EL ESP8266 A 9600 BAUDIOS** simplemente este programa nos permite comunicarnos con el ESP8266 por la consola serie

Luego abrimos la monitorización del puerto serie:

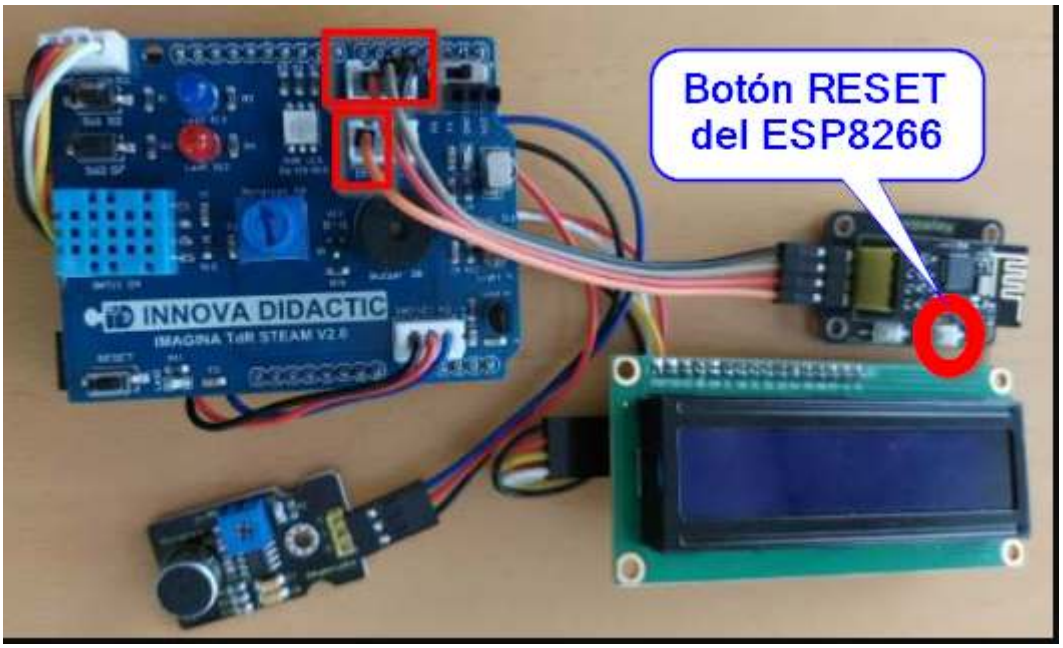


Y nos sale esta pantalla, una zona de entrada de comandos hacia el Arduino y la zona de respuesta del Arduino:

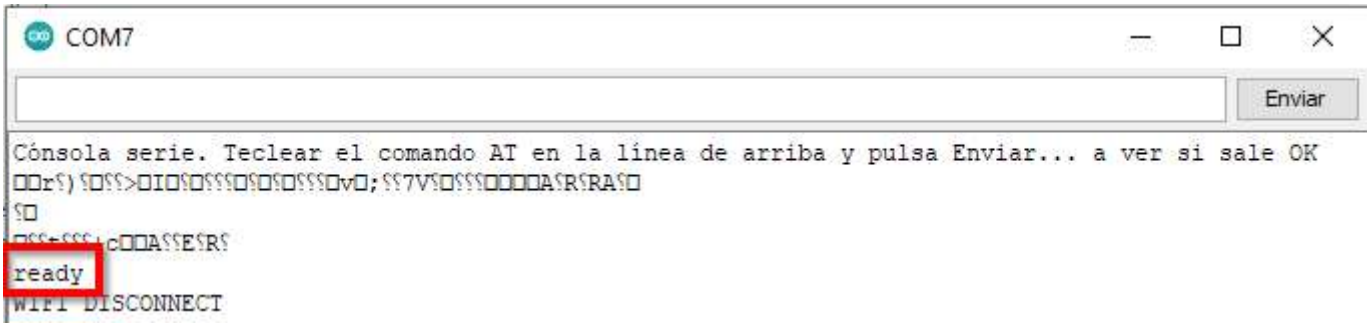
> Asegúrate que tienes la comunicación a **9600 baudios**



Apretamos el botón reset del ESP8266



Si en el monitor serie aparece al final la palabra **ready** es que la comunicación con el ESP8266 se realiza sin problemas



Si no sale la palabra **ready** tendrás que configurarlo a 9600 con los comandos AT:

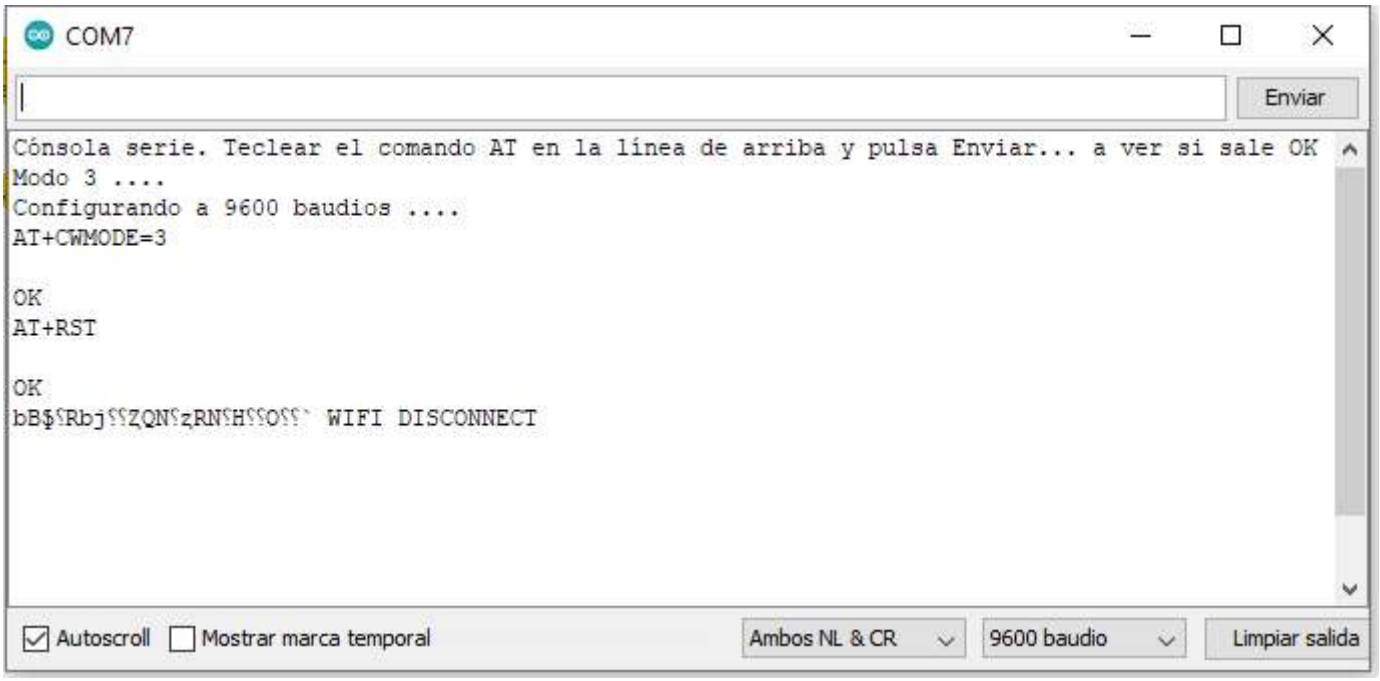
Configurando a 9600

Si no te sale la palabra **ready**, carga este programa que ejecuta la instrucción **AT+UART_DEF=9600,8,1,0,0** para configurar tu ESP8266 a 9600 baudios

```
#include <SoftwareSerial.h>


SoftwareSerial mySerial(3, 5); // Esto crea un nuevo puerto serie en RX, TX en los pines 3 y 5
```

```
void setup() {

  Serial.begin(9600);    // pone la velocidad a 9600 baudios DEL PUERTO SERIE COM no el que vamos a crear nuevo
  Serial.println("Cónsola serie. Teclear el comando AT en la línea de arriba y pulsa Enviar... a ver si sale OK");

  // set the data rate for the SoftwareSerial port
  mySerial.begin(9600);     //esto fija a 9600 la comunicación entre ESP01 y tu ordenador, no la velocidad wifi del ESP01

   delay(1000);

   Serial.print("Modo 3 ....\n");
   mySerial.write("AT+CWMODE=3\r\n");
   delay (1000);
   mySerial.write("AT+RST\r\n");
   delay (1000);
   Serial.print("Configurando a 9600 baudios ....\n");
   mySerial.write("AT+UART_DEF=9600,8,1,0,0\r\n");
   delay (1000);
   mySerial.write("AT+RST\r\n");
   delay (1000);
}

void loop() { // run over and over
  if (mySerial.available()) {
    Serial.write(mySerial.read());
  }
  if (Serial.available()) {
    mySerial.write(Serial.read());
  }

}
```

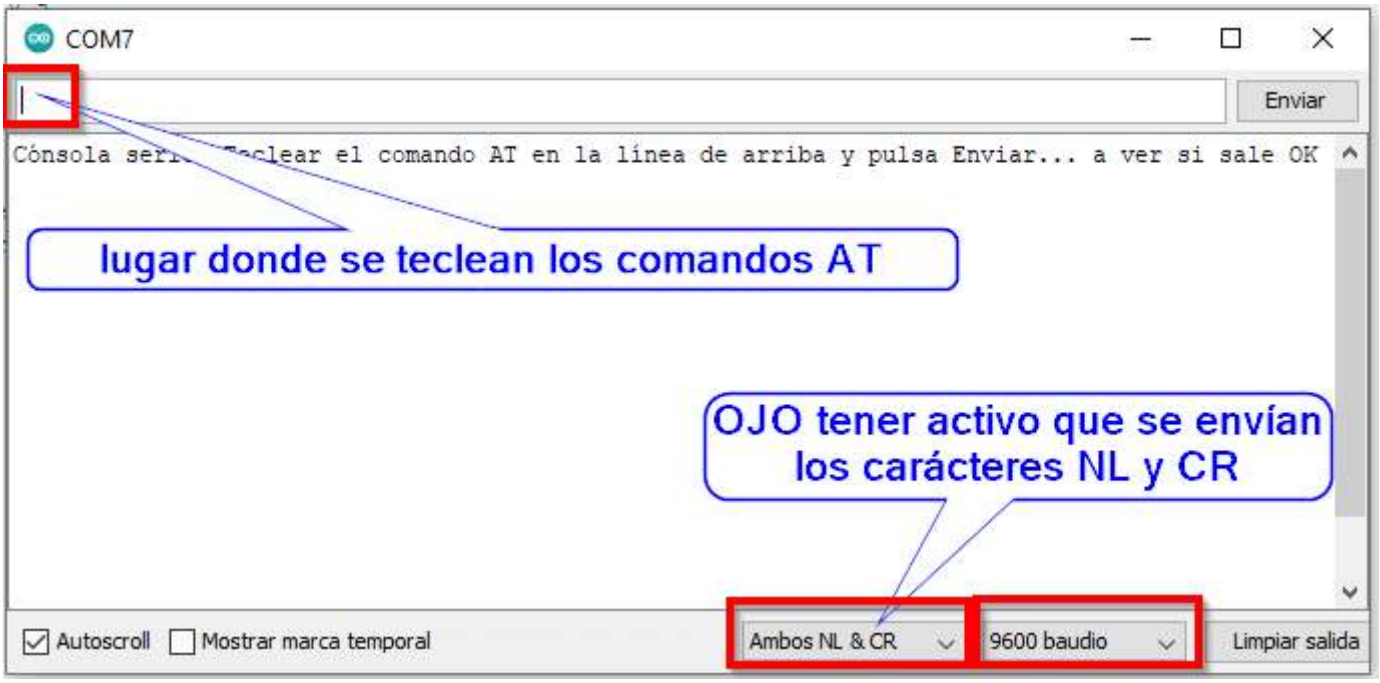
Te tiene que salir algo así:



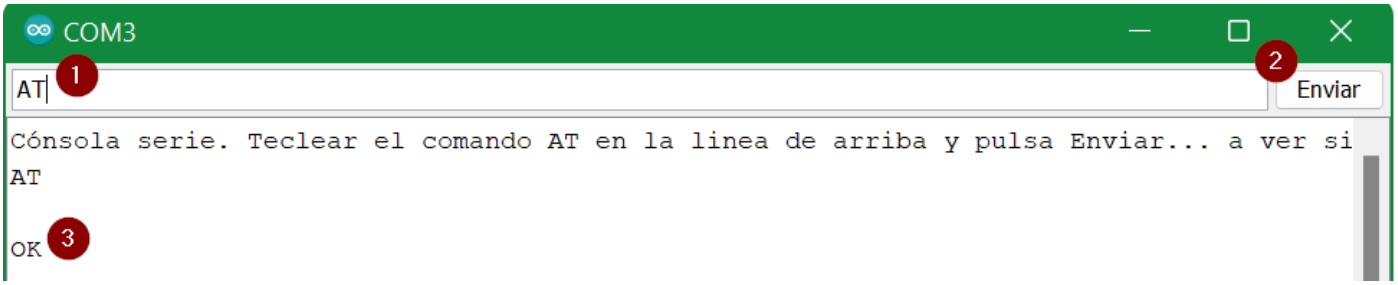
Experimenta con los comandos AT

Los comandos AT son simplemente instrucciones de texto que se envían por el puerto serie, En este caso utilizando Arduino y el puerto creado con SoftwareSerial.h como intermediarios entre tu ordenador y el ESP8266.

OJO, los comandos AT necesitan que terminen con los caracteres **nueva línea** y **carry return** para eso, tienes que tener activo que se envían estos caracteres en esta lista desplegable de la ventana del monitor serie



Teclea **AT** y pulsa **Enviar**, te tiene que salir OK esto significa que hay comunicación con el ESP8266 con los comandos AT

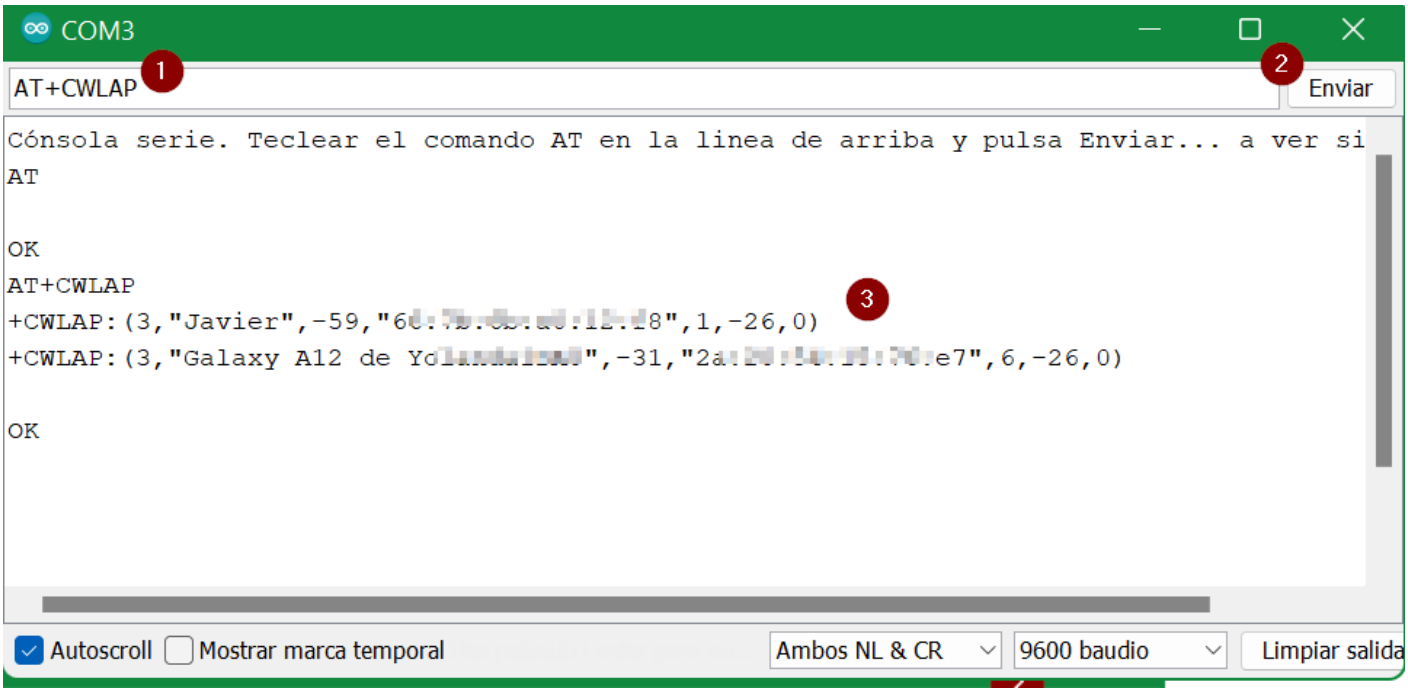


> Verás en Internet programas específicos en Arduino IDE para ejecutar los comandos AT.
> NO ES NECESARIO como ves se puede hacer desde la consola del monitor serie.
> Pero en el Arduino tiene que estar el programa cargado de crear el nuevo puerto y de visualizar los comandos.

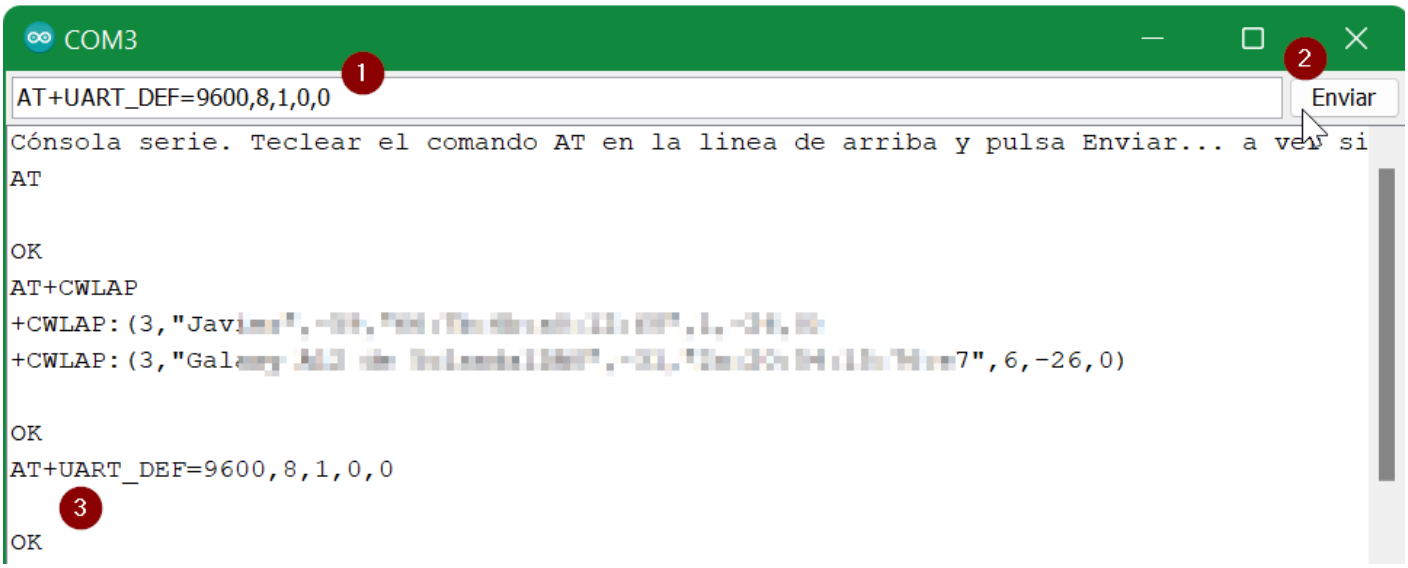
Ponlo en modo normal 3 = station+softAp es la más versátil **AT+CWMODE=3**

Luego reinicialo o con el botón de reset o con el comando **AT+RST**

Teclea **AT+CWLAP** y verás las wifis disponibles



La instrucción **AT+UART_DEF=9600,8,1,0,0** configura tu ESP8266 a 9600 baudios



Conectarse a una red wifi concreta **AT+CWSAP="ssid","password",3,0**

Ver la IP del 8266 **AT+CIFSR**

Para saber más comandos AT visita esta página de Luis Llamas

Mentirijillas

He dicho que esto sólo se hace una sóla vez, que no es necesario hacerlo en la programación habitual con ESP8266 añadir puertos serie. Es una mentirijilla. Realmente siempre hay que usar el **SoftwareSerial.h** y crear un nuevo puerto con **SoftwareSerial mySerial(3, 5)** para poder manejar el ESP8266, pero esto **lo hace Arduinoblocks por nosotros** de forma transparente.

Cuando ponemos en Arduinoblocks la instrucción para conectarnos a la wifi con el ESP8266 :

Si vemos en el código, ya pone estas instrucciones :

```
#include <SoftwareSerial.h>
#include "ABlocksIOTMQTTESP8266.h"

String s_LED_TXT;
boolean b_conectado;
const char mqtt_broker[]="io.adafruit.com";
const int mqtt_port=1883;
const char mqtt_user[]=".......";
const char mqtt_pass[]=".....";
const char mqtt_clientid[]="cualquiercosa";
const char mqtt_wifi_ssid[]="...";
```

```
const char mqtt_wifi_pass[]="......";
SoftwareSerial mqtt_esp8266_serial(3,5);
ESP8266 mqtt_esp8266_wifi(&mqtt_esp8266_serial);

etc....
```

Financiado por el Ministerio de Educación y Formación Profesional y por la Unión Europea - NextGenerationEU

Revision #15
Created 19 August 2022 09:44:12 by Javier Quintana
Updated 31 January 2023 11:11:22 by Javier Quintana