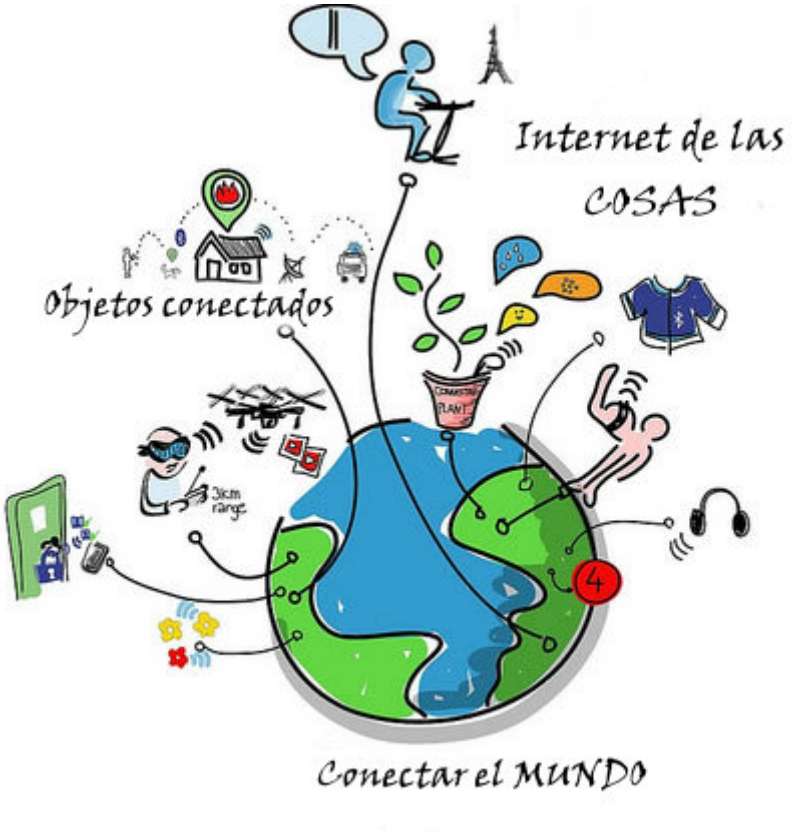
IoT-Wifi

- ¿Qué es Internet de las cosas IoT?
- ESP8266 o ESP01
- Conexión ESP8266
- Comprobar que el ESP8266 está a 9600 baudios por defecto

¿Qué es Internet de las cosas IoT?

> El **Internet de las cosas** (Internet of Thing IoT) describe objetos físicos —o grupos de estos— con sensores, capacidad de procesamiento, software y otras tecnologías que se conectan e intercambian datos con otros dispositivos y sistemas a través de internet u otras redes de comunicación. El Internet de las cosas se ha considerado un término erróneo porque los dispositivos no necesitan estar conectados a la Internet pública. Sólo necesitan estar conectadas a una red y ser direccionables individualmente

Fuente Wikipedia IoT Internet de las cosas CC-BY-SA



De Drawed by Wilgengebroed on FlickrTranslated by Prades97 CC BY-SA 3.0

Estamos hablando de dispositivos que se conectan a internet de forma desatendida, por vía hardware (o mejor dicho firmware) a diferencia de un ordenador, tablet o móvil, donde tienes que configurar por software el dispositivo y hay un diálogo entre usuario y dispositivo sobre el uso de Internet (el software solicita tal página web, tales datos etc por voluntad del usuario o por diálogo

con el usuario) Aquí los dispositivos están ya configurados de los datos que se comunican. Es decir "conectar y olvidar".

Piensa en la diferencia entre un enchufe inteligente y un ordenador, el primero es lo que se considera dentro de IoT

Las formas "desatendidas" son un avance en la sociedad pero también puede generar problemas muy serios a nivel mundial, ver el caso Mirai

> Las cosas claras. ¿asíncrono o síncrono?

Hay muchas herramientas IoT

- **Blynk:** lo que nos gusta de esta herramienta es que es casi "instantánea" o "síncrona". Esto es imprescindible con ciertos robots como el **Rover Marciano con Arduino**. Necesitamos que "gire" para evitar un obstáculo, no podemos esperar !!!. Veremos con **BLYNK** un protocolo que entre el dispositivo electrónico (nuestro robot) y nosotros (en ordenador, en una APP en el móvil) la comunicación es instantánea, gracias a un servidor que hará de intermedio, que puede ser local (BLYNK LEGACY) o en Internet (BLYNK IoT).
  - **Blynk legacy** es la que se va a trabajar en
    - **Rover Marciano con Arduino**
    - **Arduinoblocks en el aula**
    - **ESP32 en el aula**
  - **Blynk IoT** es la que se va a trabajar con
    - **En ESP32 en el aula**
- **MQTT** El emisor envía datos, se almacenan en un servidor, y cuando puede, lo vuelca al cliente. Cliente y emisor pueden ser el dispositivo electrónico y nosotros o viceversa. Veremos que esto es lo que hace el protocolo **MQTT** y está tremendamente extendido por lo barato y fácil que es. Hace que los servidores no estén tan ocupados, por lo tanto hay varios proveedores que ofrecen este servicio gratuitamente. Hay robots como los que tienen la placa **TDR STEAM IMAGINA** que envía datos de temperatura, humedad, .. y pueden recibir datos pero no precisan de esta exigencia instantánea como un rover.
  - ESP32 EN EL AULA
- **Arduino cloud IoT**
  - **Arduino Alvik**
- **Cyberpi y mBot2**
  - **Iot con Cyberpi**

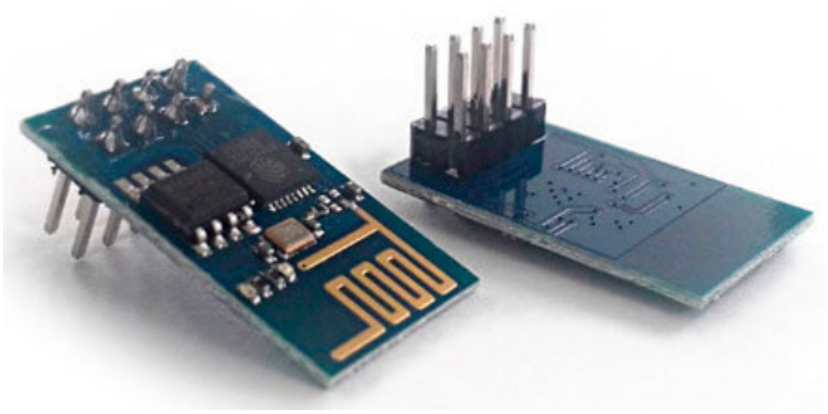
Financiado por el Ministerio de Educación y Formación Profesional y por la Unión Europea - NextGenerationEU

ESP8266 o ESP01

Arduino no tiene wifi, y es importante esto para conseguir hacer prácticas IoT. Hay shields que permiten una conexión Ethernet o Wifi pero resultan caras. Otra opción es utilizar la versión del Arduino MKR1000 pero también resulta cara. Lo mejor es utilizar el ESP8266 para que a través de él nuestro Arduino pueda volcar o recibir datos a través de una Wifi.

Resumiendo brevemente, el **ESP8266** es un chip microcontrolador, es decir, no es un sensor, no es un actuador, no es una entrada/salida del Arduino sino que es, igual que el Arduino, es una placa electrónica montado en un módulo que tiene un microcontrolador (Tensilica L106 de 32bits) capaz de hacer cosas pero que tiene una característica importante: **Que tiene Wif**i, pero no lo veas como un módulo Wifi para Arduino, sino como una placa electrónica completa, como el Arduino, incluso es su competencia.

El chip **ESP8266** se vende montado en un módulo, el más vendido es el **ESP01** y es el que se proporciona en el kit del curso **Arduinoblocks en el aula** de CATEDU.



Fuente Luis Llamas CC-BY-NC-SA ver

> Por eso se habla indistintamente ESP8266 o el ESP01

Sus entradas y salidas son comunicaciones por puerto serie que está ocupado en el Arduino con las comunicaciones a nuestro ordenador, esto hay que tenerlo en cuenta en la programación añadiendo un nuevo puerto serie. Arduinoblocks lo hace automáticamente por nosotros.

Si quieres saber más de este chip y su zócalo te recomendamos

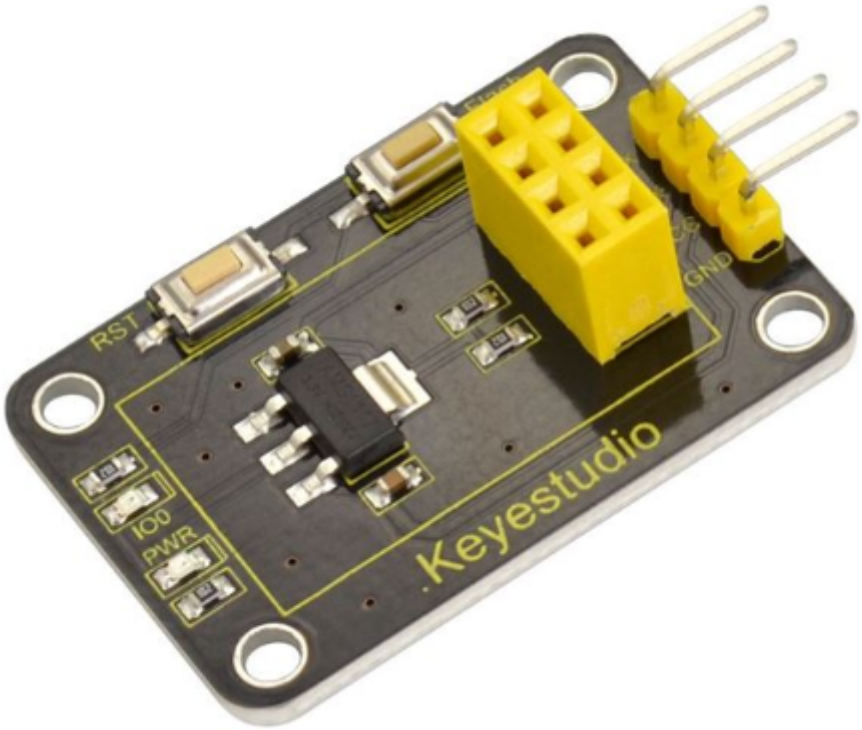
https://www.luisllamas.es/esp8266/

Su bajo precio y su software libre permitió al mundo maker utilizar este chip. No sólo se puede utilizar para que el Arduino tenga acceso a Internet, sino también se han desarrollado multitud de módulos con el ESP8266, como veremos más adelante, el más interesante en su evolución es el módulo ESP-12E o el ESP32.

Pero sigamos con el ESP8266 montado en el módulo ESP01. Tiene unas pegas... no se diseñó para montarlo en el Arduino : **PRIMERO** La alimentación es 3V a 3.6V con picos de 200mA por lo que no puede conectarse directamente a la alimentación 3.3V y 50mA de Arduino. **SEGUNDO** Consecuencia del máximo de 3.6V es que las entradas y salidas del ESP8266 no conviene conectarlas directamente a las entradas y salidas del Arduino que van a 5V.

En conclusión, NO SE PUEDE CONECTAR DIRECTAMENTE. Verás en Internet muchos tutoriales que conectan el ES01 directamente al Arduino pero a mi personalmente no me ha funcionado ninguno.

En conclusión: tenemos que usar una alimentación externa como se aconseja aquí o utilizar un **zócalos específico** como el proporcionado en el kit de Arduino en el Aula del fabricante Keyestudio.



Fuente página web del fabricante Keyestudio

> **NUESTRO CONSEJO**

Si piensas trabajar con Internet y el Arduino, es mejor usar otras placas que puedan usar el lenguaje IDE de Arduino, pero que tengan integrada una versión del ESP8266 con un módulo adaptado a su integración

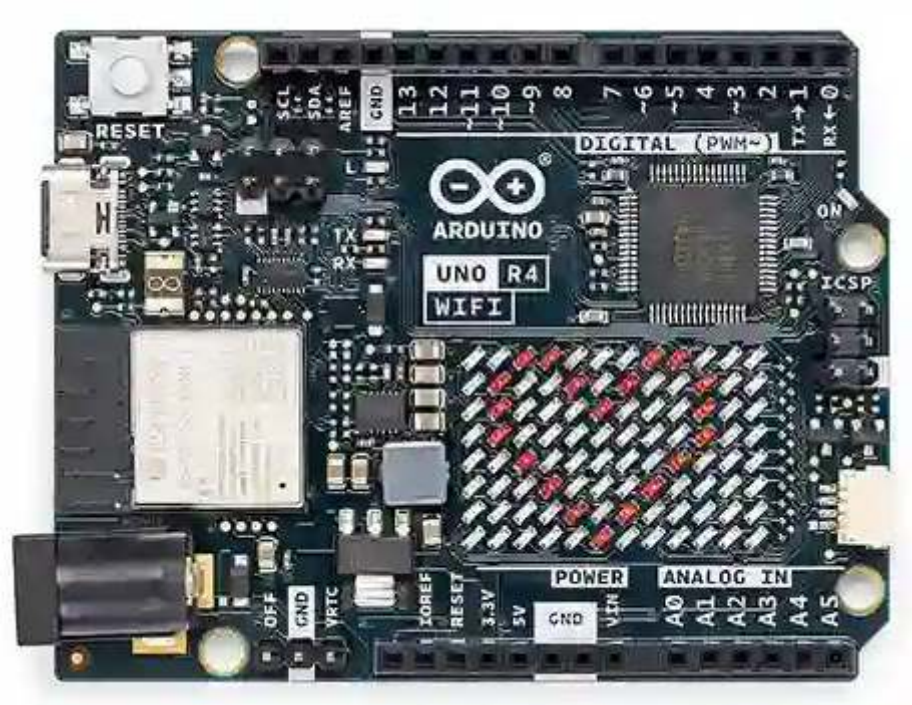
- **NodeMCU** que utiliza el módulo **El ESP-12E** y se ha utiliza en el curso **ROVER MARCIANO CON ARDUINO.**

Módulo ESP-12E Fuente Luis Llamas CC-BY-NC-SA ver

- **ESP32 en el Aula** que utiliza la placa ESP32 de Innovadidactic y es un serio competidor al Arduino: Más barato y más potente y se utiliza en el curso **ESP32 EN EL AULA**

Módulo ESP32 Fuente Luis Llamas CC-BY-NC-SA https://www.luisllamas.es/esp32/

- **ARDUINO UNO R4 WIFI** lanzado en junio23 integra un ESP32 dentro del Arduino para obtener la Wifi, curioso, metes a tu competidor dentro: *si no puedes con tu enemigo...* ¿lo veis en la foto? Incorpora Bluetooth y una matriz de leds



Arduino UNO R4 Wifi Fuente Luis Llamas CC-BY-NC-SA *https://www.luisllamas.es/arduino-uno-r4-minima-wifi/*

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

Financiado por la Unión Europea NextGenerationEU
Plan de Recuperación, Transformación y Resiliencia
GOBIERNO DE ARAGON

Conexión ESP8266

Vamos a conectar el ESP8266 con nuestro KIT TDR STEAM IMAGINA. Si no tuvieras la shield TDR STEAM tienes que conectarlo directamente en el Arduino, en esta página tienes un esquema de cómo se hace.

En primer lugar **insertamos el ESP8266 con el zócalo** que nos facilitará la conexión al Arduino tal y como hemos visto anteriormente.

Luego utilizamos **el cable de 3 hilos** que proporciona el Kit al zócalo del ESP8266 de tal forma que :

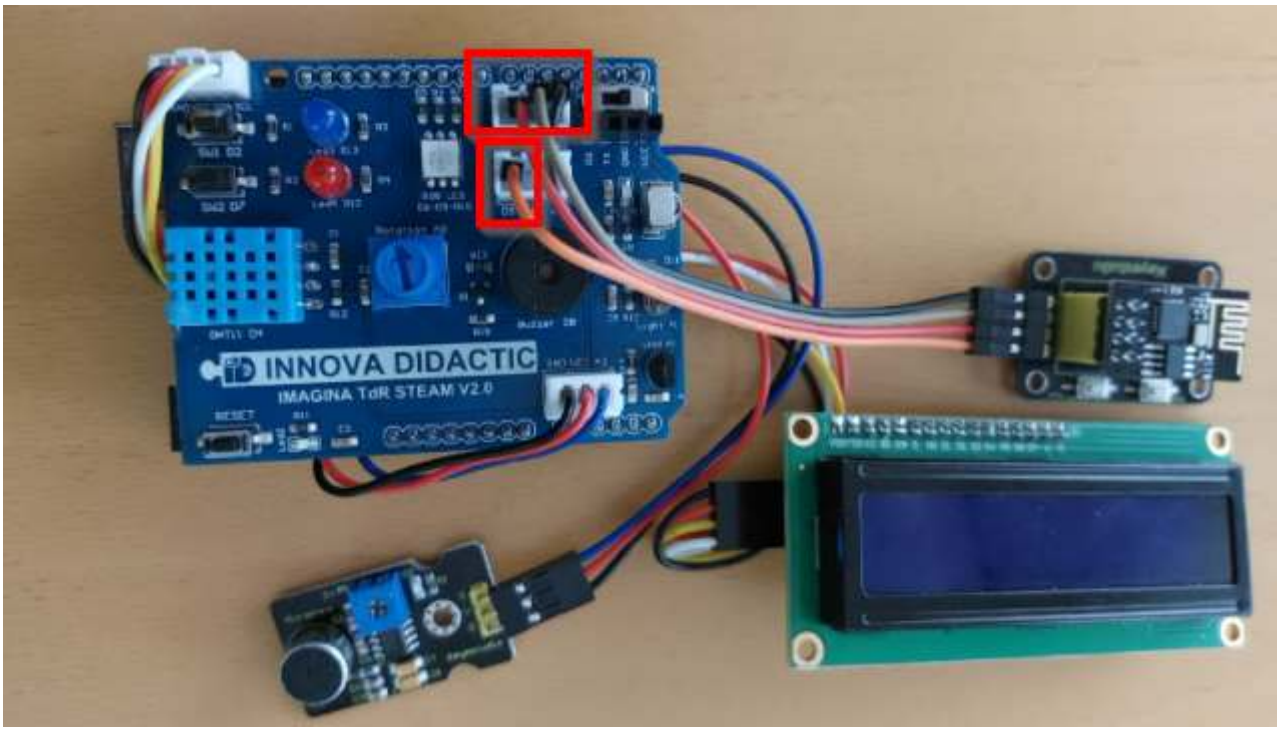
- Cable negro a GND
- Cable rojo a Vcc
- Cable azul a TX del zócalo ESP8266

El otro extremo del cable de 3 hilos a la Shield TDR Imagina, tenemos dos posibilidades, lo conectaremos al slot del D3

Luego utilizaremos **un cable extra: Dupont Hembra - Hembra** (en la foto de color marrón) que lo conectaremos un extremo al RX del zócalo ESP8266 y el otro extremos al D5.

De esta manera tenemos alimentado el zócalo ESP8266 y

- RX a D5
- TX a D3



Otra posibilidad es al revés, utilizar el otro slot D5 en el cable de tres hilos y conectar el Dupont H-H al D3 pero en este libro utilizaremos el criterio expuesto por elegir uno.

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

Financiado por la Unión Europea NextGenerationEU
Plan de Recuperación, Transformación y Resiliencia
GOBIERNO DE ARAGON

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

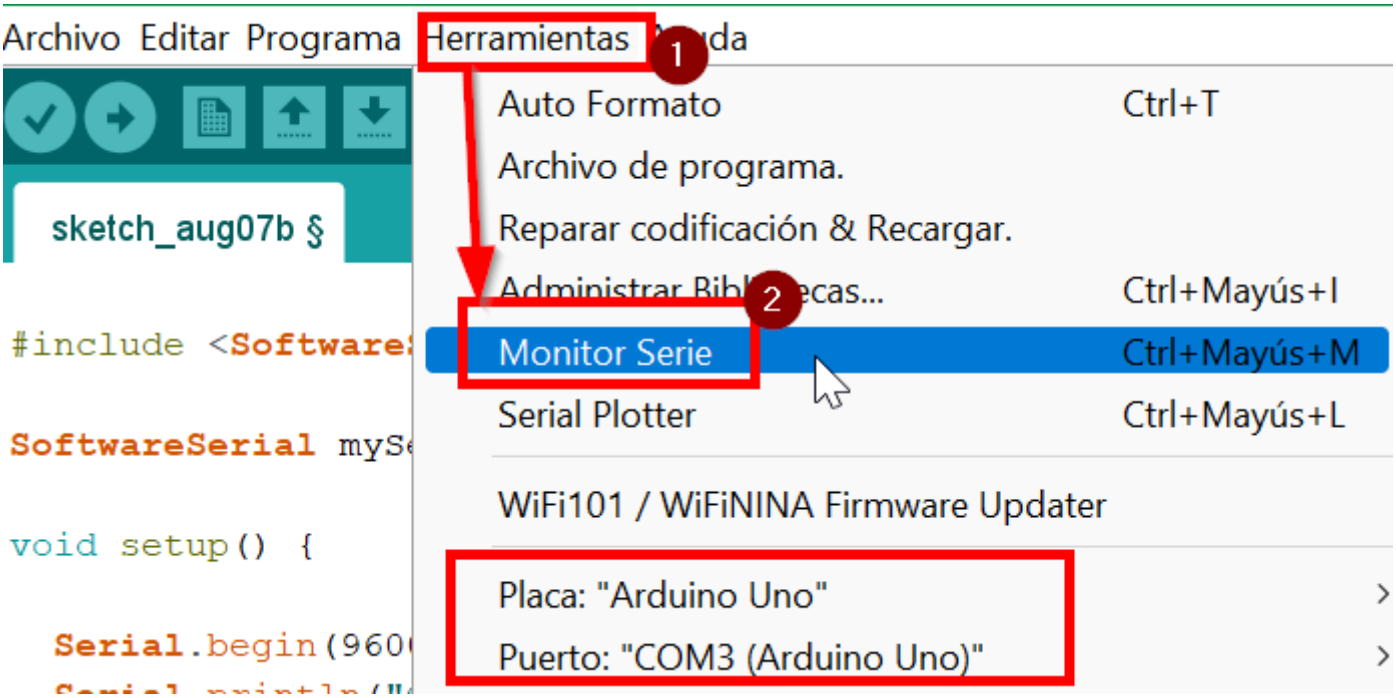
```
#include <SoftwareSerial.h>

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

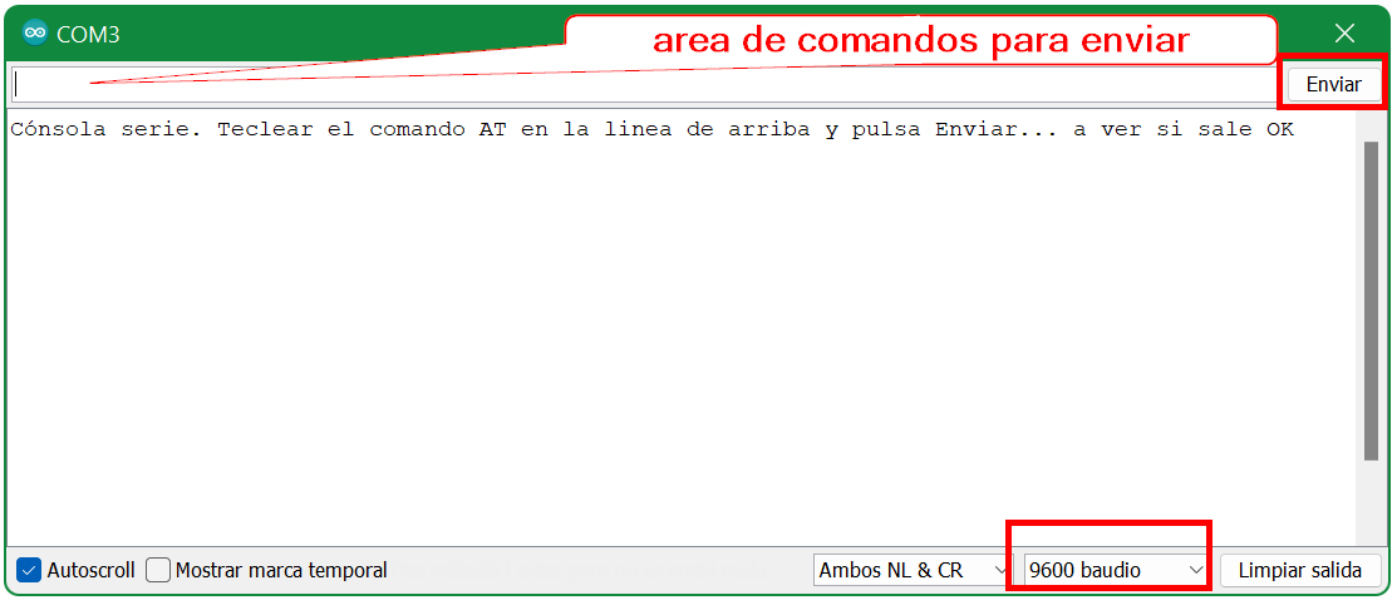
> Este programa **NO NOS HA CONFIGURADO EL ESP8266 A 9600 BAUDIOS** simplemente este programa nos permite comunicarnos con el ESP8266 por la consola serie

Luego abrimos la monitorización del puerto serie:

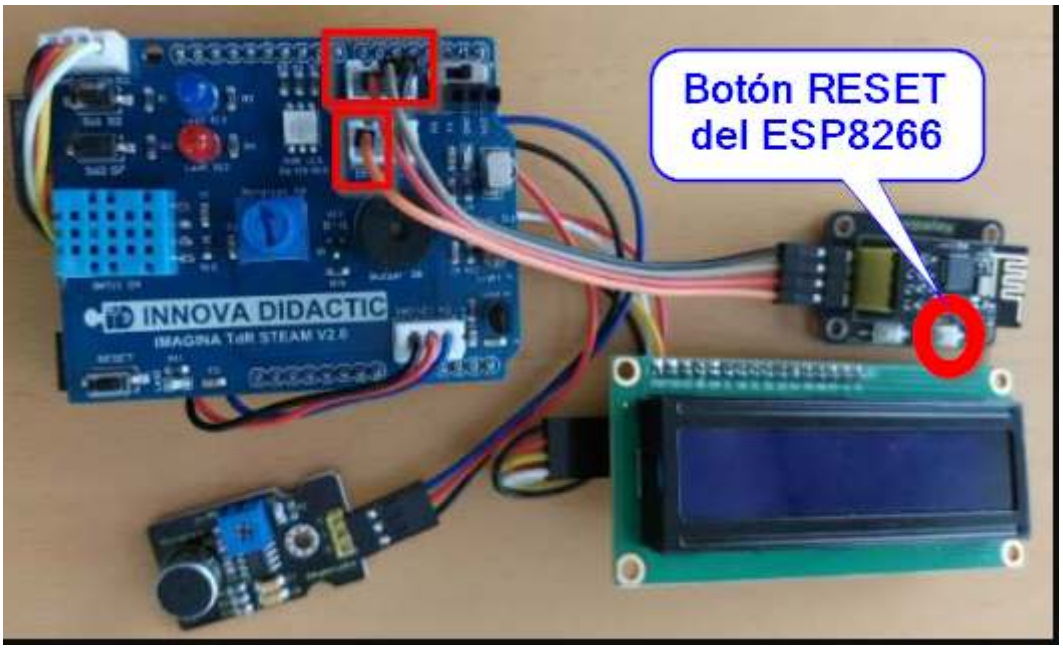


Y nos sale esta pantalla, una zona de entrada de comandos hacia el Arduino y la zona de respuesta del Arduino:

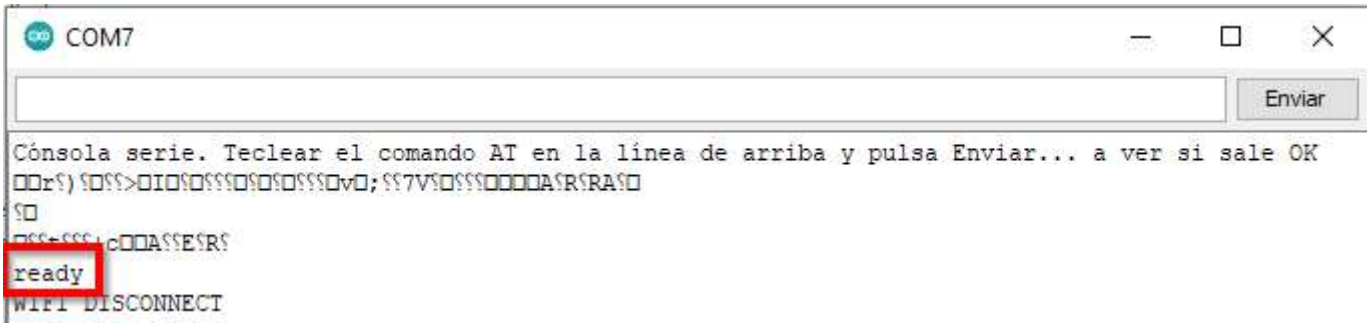
> Asegúrate que tienes la comunicación a **9600 baudios**



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:

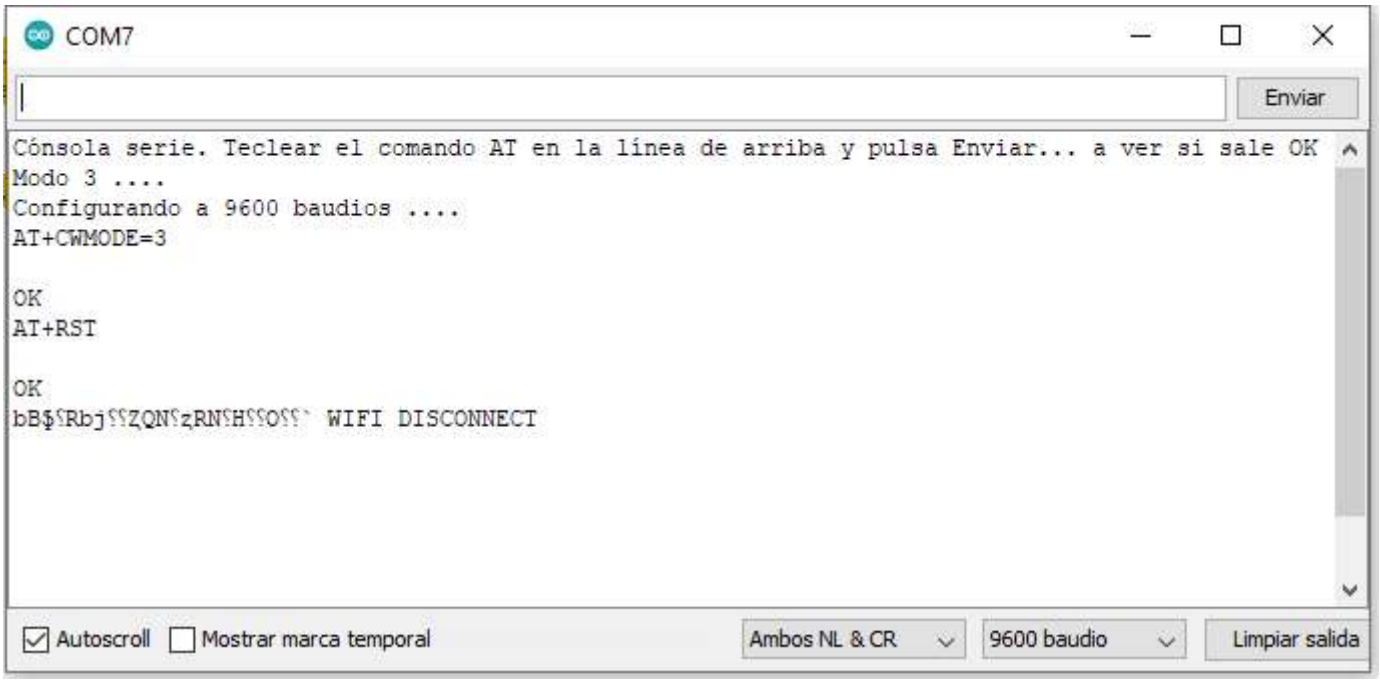
Configurando a 9600

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());
  }

}
```

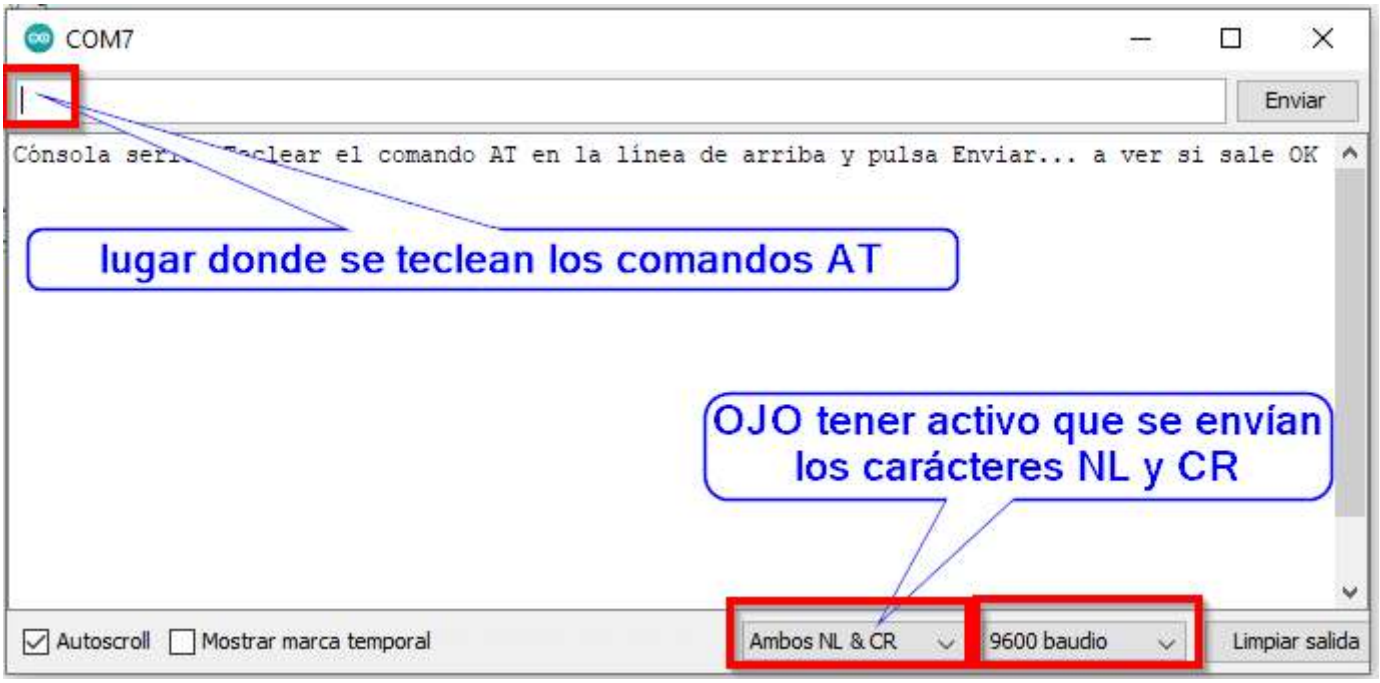
Te tiene que salir algo así:



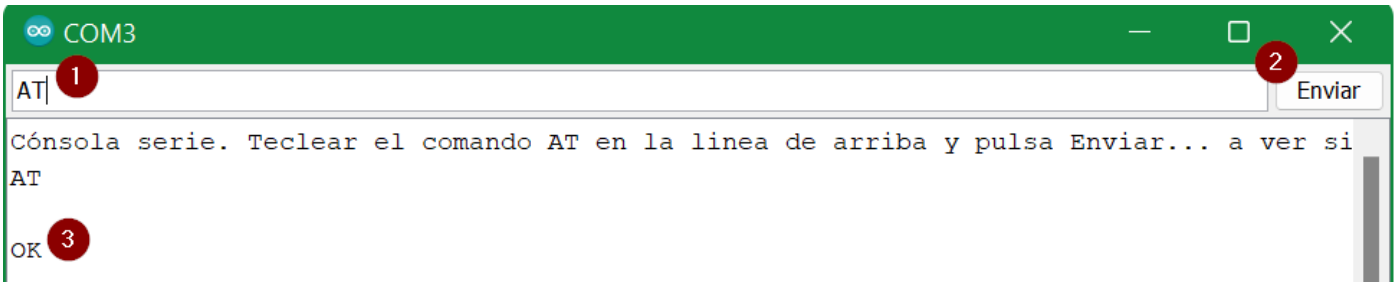
Experimenta con los comandos AT

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

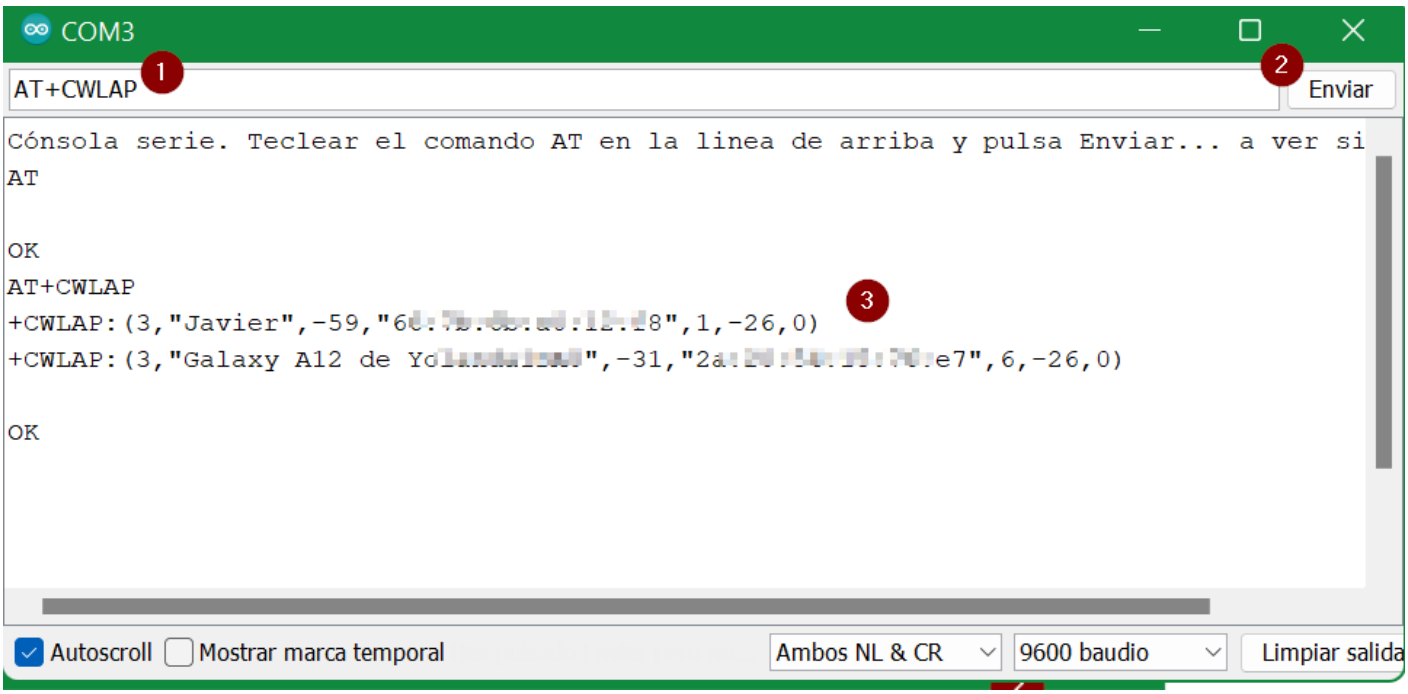


> Verás en Internet programas específicos en Arduino IDE para ejecutar los comandos AT.
> NO ES NECESARIO como ves se puede hacer desde la consola del monitor serie.
> Pero en el Arduino tiene que estar el programa cargado de crear el nuevo puerto y de visualizar los comandos.

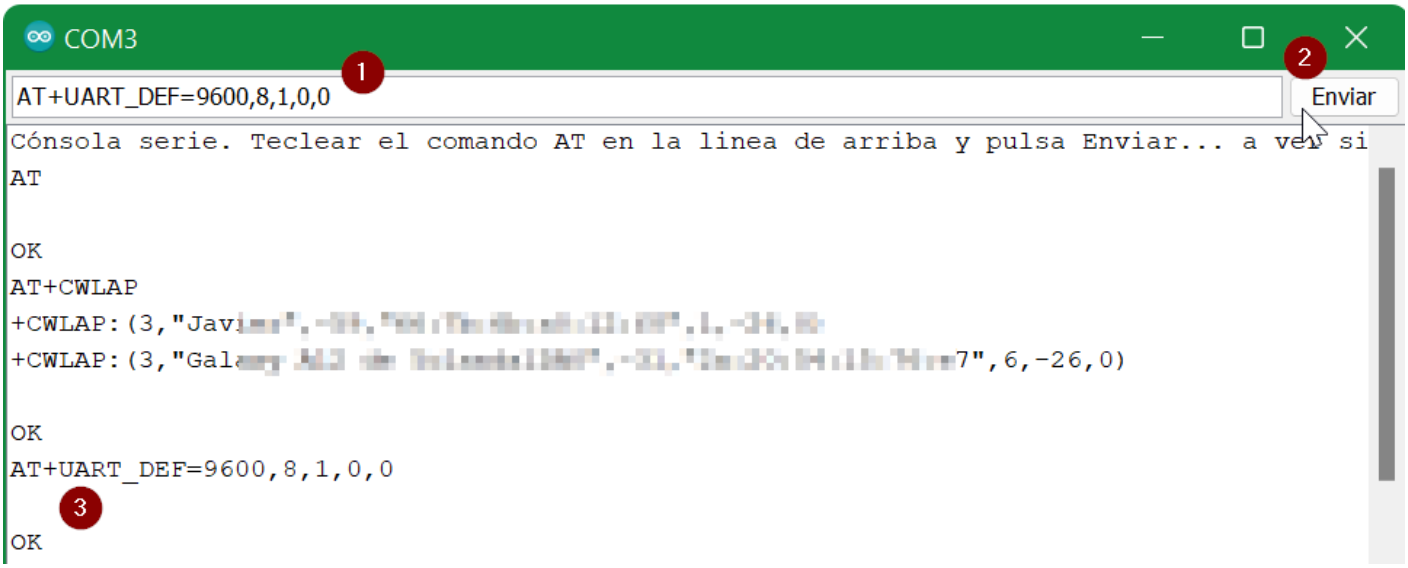
Ponlo en modo normal 3 = station+softAp es la más versátil **AT+CWMODE=3**

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