Módulo 5. Prácticas con sensores analógicos

- Actividad-07. El potenciómetro y Plotter
- Actividad-08. La fotorresistencia o LDR
- Actividad-09. El sensor de temperatura LM35D
- Actividad-10. El sensor de temperatura y humedad DHT-11
- Actividad-11. Emisor y receptor de infrarrojos

Copyright 2026

Actividad-07. El potenciómetro y Plotter

Conceptos generales

Un **potenciómetro** sirve para regular la cantidad de corriente o el voltaje que pasa por un circuito electrónico de forma manual

¿Cómo funciona? Es un componente que actúa como una resistencia variable.Tiene una perilla, un botón o un deslizador que puedes mover.Al moverlo, cambias la distancia que recorre la electricidad por dentro, lo que sube o baja la resistencia.

¿Para qué se utiliza? Controlar el volumen en radios, guitarras y equipos de sonido.Regular el brillo de pantallas o luces.Ajustar la velocidad de motores pequeños.Calibrar o controlar parámetros en aparatos electrónicos.

Un **potenciómetro con el plotter** (o graficador serial de un entorno de programación como Arduino IDE) sirve para visualizar en tiempo real y de forma gráfica los cambios de voltaje que ocurren al girar o deslizar la perilla del componente.

> Página extraída de Federico Coca Notas sobre ESP32 STEAMakers CC-BY-SA

Enunciado

Introducir el uso de las entradas analógicas para visualizar los cambios que hagamos con el potenciómetro en el terminal serie o consola serie. Cuando hablamos de lecturas analógicas suele ser muy útil el uso del mapeo, concepto que también vamos a introducir y que veremos en la segunda parte del reto.

Teoría

Antes de nada ya hemos indicado que el potenciómetro va conectado a un pin analógico y es en este momento cuando vamos a establecer la diferencia entre los conceptos de analógico y de digital.

Una clasificación de los circuitos electrónicos es dividirlos en dos grandes categorías: digitales y analógicos.

- La electrónica digital utiliza magnitudes con dos valores discretos conocidos como 0 - 1, alto - bajo, on - off, etc, y que se corresponden con la presencia o no de tensión en un determinado punto.
- La electrónica analógica emplea magnitudes con valores continuos. En concreto en las placas En la ESP32 Plus STEAMakers, las entradas analógicas pueden tener 2^12 valores (12 bits de resolución = 4.096 valores), es decir, valores comprendidos entre 0 y 4.095. Estos valores se pueden expresar de forma numérica o de forma porcentual, correspondiendo el 0 a un 0% y el 4.095 al 100%.

El potenciómetro

En la imagen siguiente vemos el aspecto de un modelo concreto de potenciómetro así como los símbolos habituales del mismo. La numeración indica lo siguiente:

Terminales 1 y 3 son los contactos unidos a los extremos de la resistencia fija o resistencia total del potenciómetro.
Terminal 2 es el contacto que va unido al cursor o parte móvil que se desliza sobre la resistencia fija haciendo que la resistencia entre un terminal y el cursor varíe en función de la posición de este.

Imagen Federico Coca Notas sobre ESP32 STEAMakers CC-BY-SA

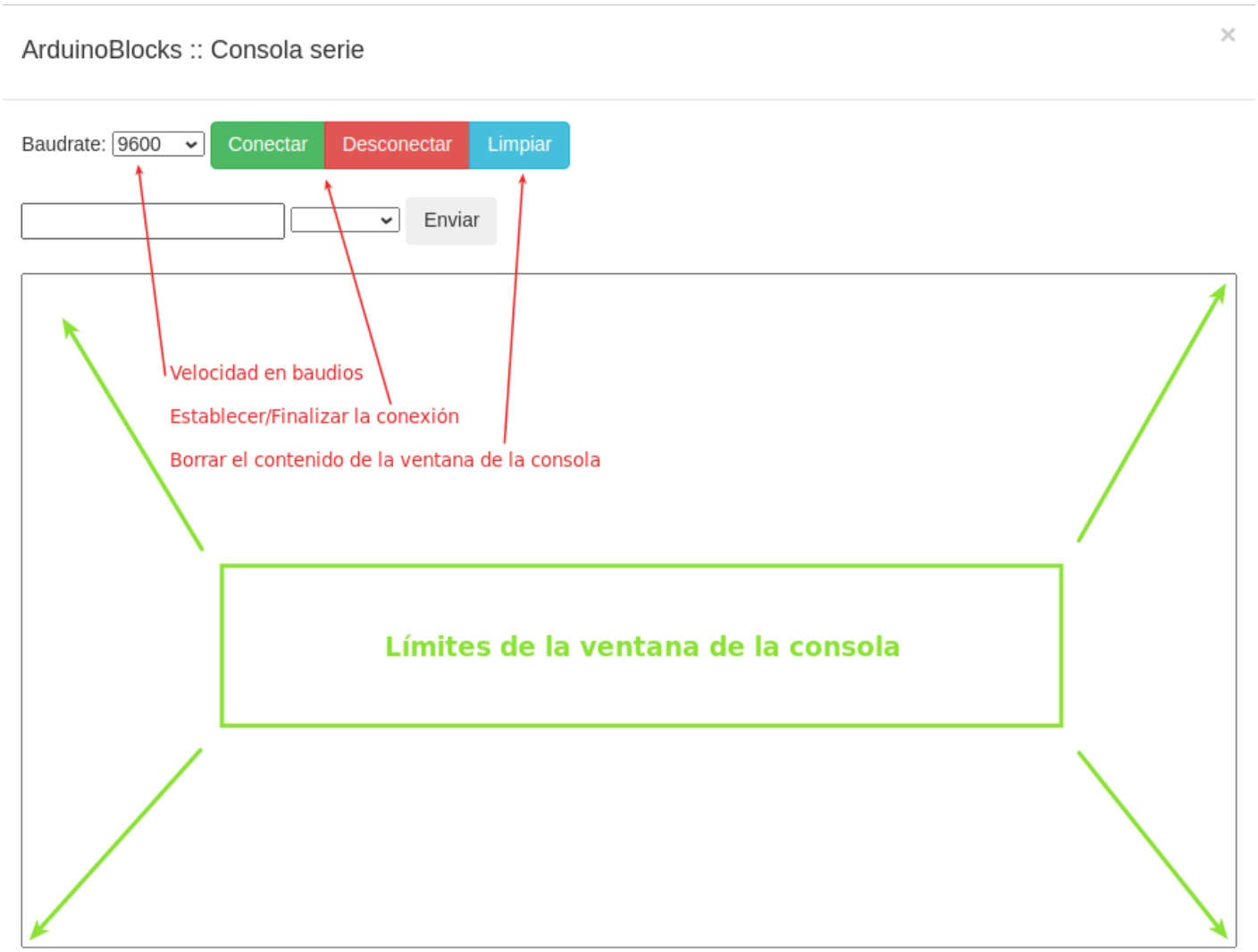
La posición del cursor se determina de forma mecánica y son adecuados para usarlos como elementos de control de tensión (conexión en serie) o de corriente (conexión en paralelo). Los potenciómetros del tipo que estamos viendo (existen de otros muchos tipos) tienen un funcionamiento en forma de rotación con un ángulo de unos 270 grados entre los puntos mas extremos.

El terminal o consola serie

Sirve para visualizar en un ordenador los datos recibidos a través del puerto serie y en realidad es una aplicación que controla las comunicaciones bidireccionales a través de la UART integrada en el microcontrolador.

Es muy importante tener siempre presente que el puerto serie es el que se utiliza para "subir" el firmware o programa a la placa, tarea que se realiza a través de una aplicación pregrabada en el microcontrolador y que se denomina "bootloader".

En ArduinoBlocks es posible usar la consola serie solamente si tenemos instalada y en funcionamiento la aplicación ArduinoBlocks-Connector que es la encargada de establecer las comunicaciones locales de nuestra placa con las remotas de la aplicación. En la imagen siguiente vemos el aspecto de la consola.



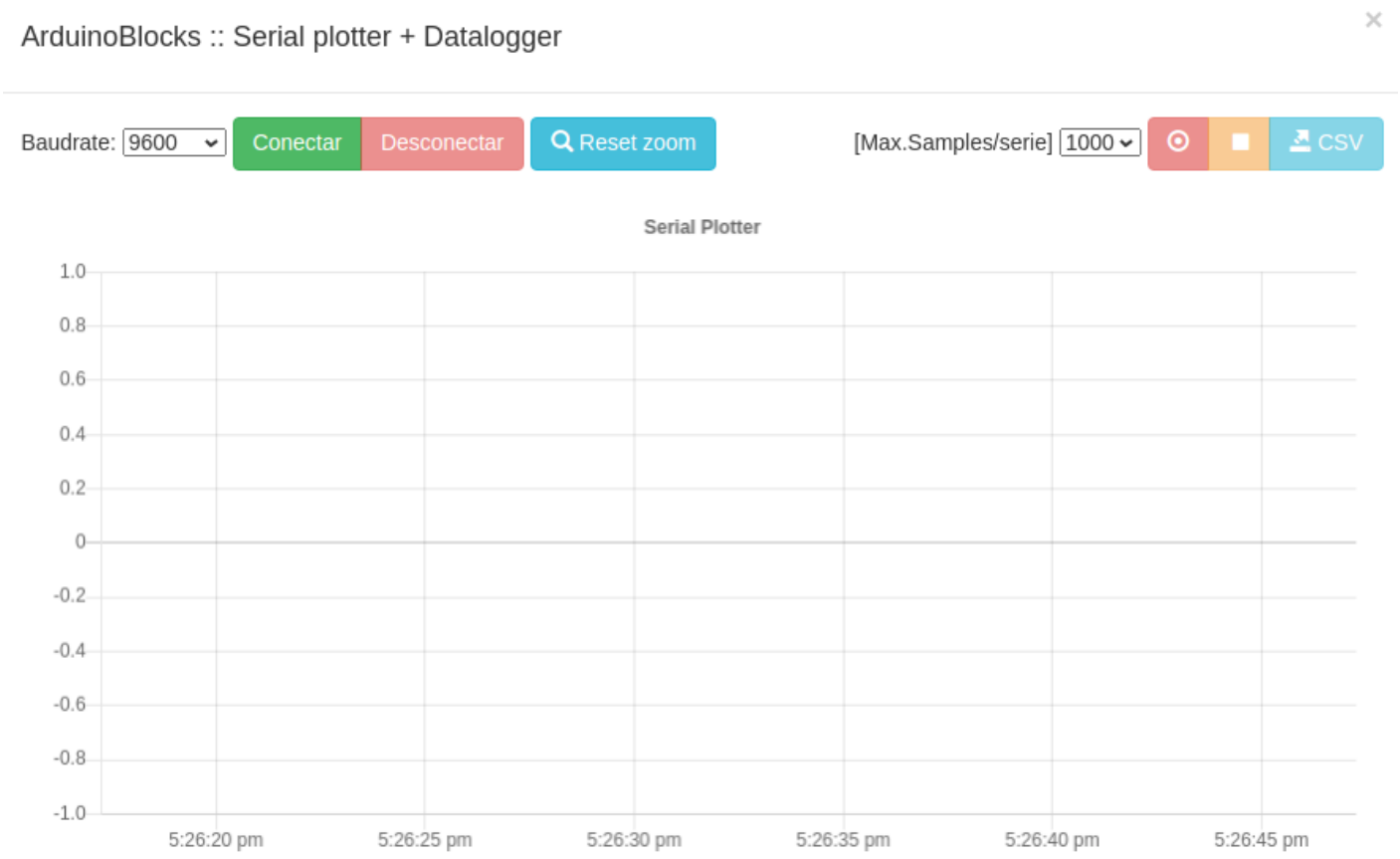
Aspecto de la Consola en AB [Imagen Federico Coca Notas sobre ESP32 STEAMakers](#) CC-BY-SA

A la hora de establecer una conexión serie los dos extremos que intervienen en la conexión (placa ESP32 STEAMakers y ordenador) deben establecer el mismo valor en la velocidad de la conexión. Por defecto esta velocidad es de 115200 baudios o bits por segundo en el bloque inicializar y otras velocidades comunmente utilizadas son: 4800, 9600, 19200, 38400, 57600. Es por lo tanto

imprescindible incluir en el bloque "Inicializar" el bloque "Iniciar" y establecer la velocidad de comunicación.

Serial Plotter - Datalogger

Es otra funcionalidad relacionada con la comunicación serie que nos permite visualizar información en forma de gráfica en tiempo real. Además el “Serial Plotter” implementa un sencillo datalogger con el que podemos ir grabando los datos para exportarlos posteriormente. En ArduinoBlocks existen bloques que nos permiten trabajar con el serial plotter. El serial plotter + datalogger se activa haciendo clic en la flecha a la derecha de Consola y tiene el aspecto que vemos en la imagen siguiente:



Serial Plotter + Datalogger [Imagen Federico Coca Notas sobre ESP32 STEAMakers](#) CC-BY-SA

Mapeo

Hemos indicado que las entradas analógicas trabajan con 12 bits, o sea valores comprendidos entre 0 y 4095 (2^12) y también que las salidas digitales trabajan con 8 bits, o sea valores entre 0 y 255 (28), por lo que si queremos combinar en nuestro programa entradas analógicas con salidas digitales debemos realizar un ajuste de escala en los datos. A este ajuste se le conoce como

"mapear" y es un bloque disponible en el menú Matemáticas con el aspecto de la imagen siguiente:

Imagen Federico Coca Notas sobre ESP32 STEAMakers CC-BY-SA

En la TdR STEAM

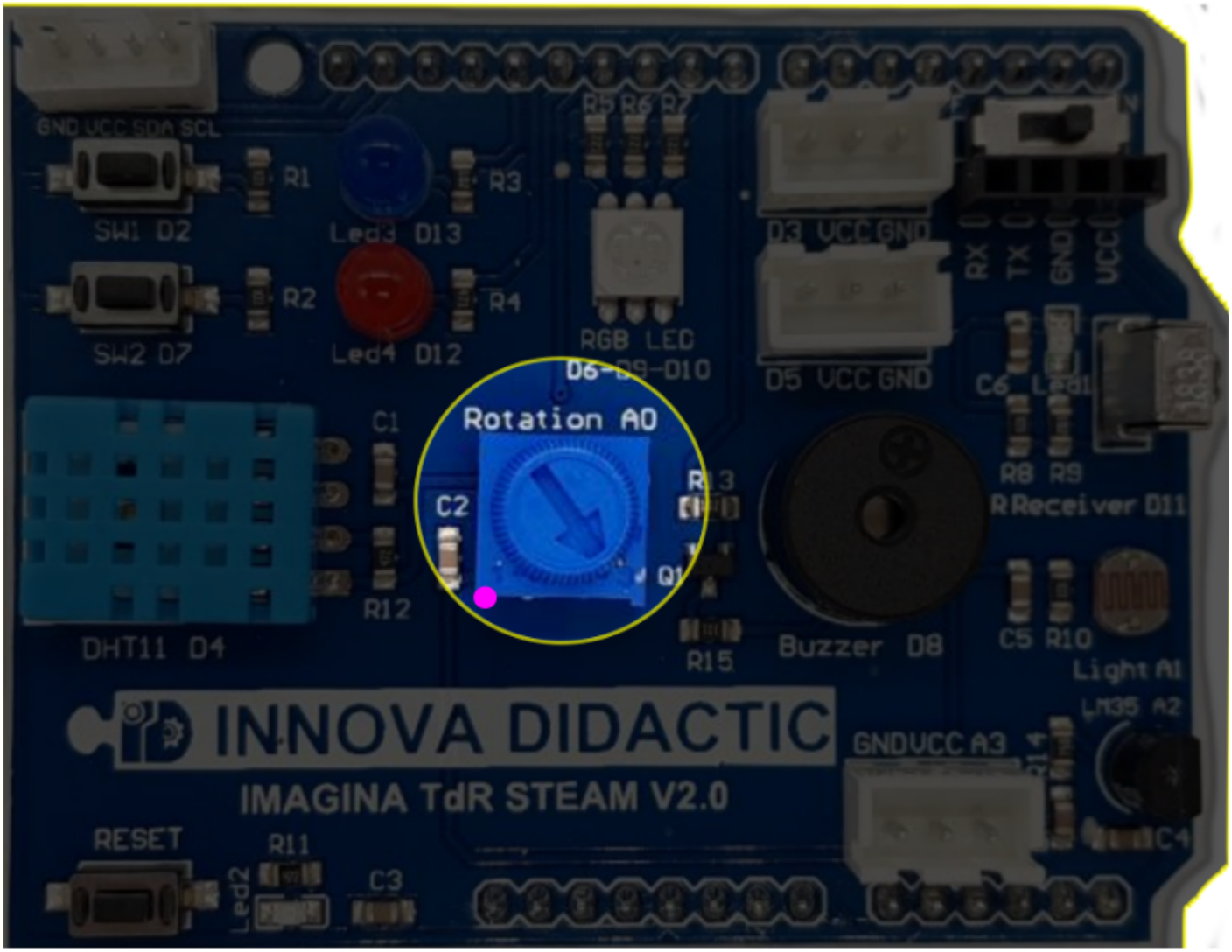


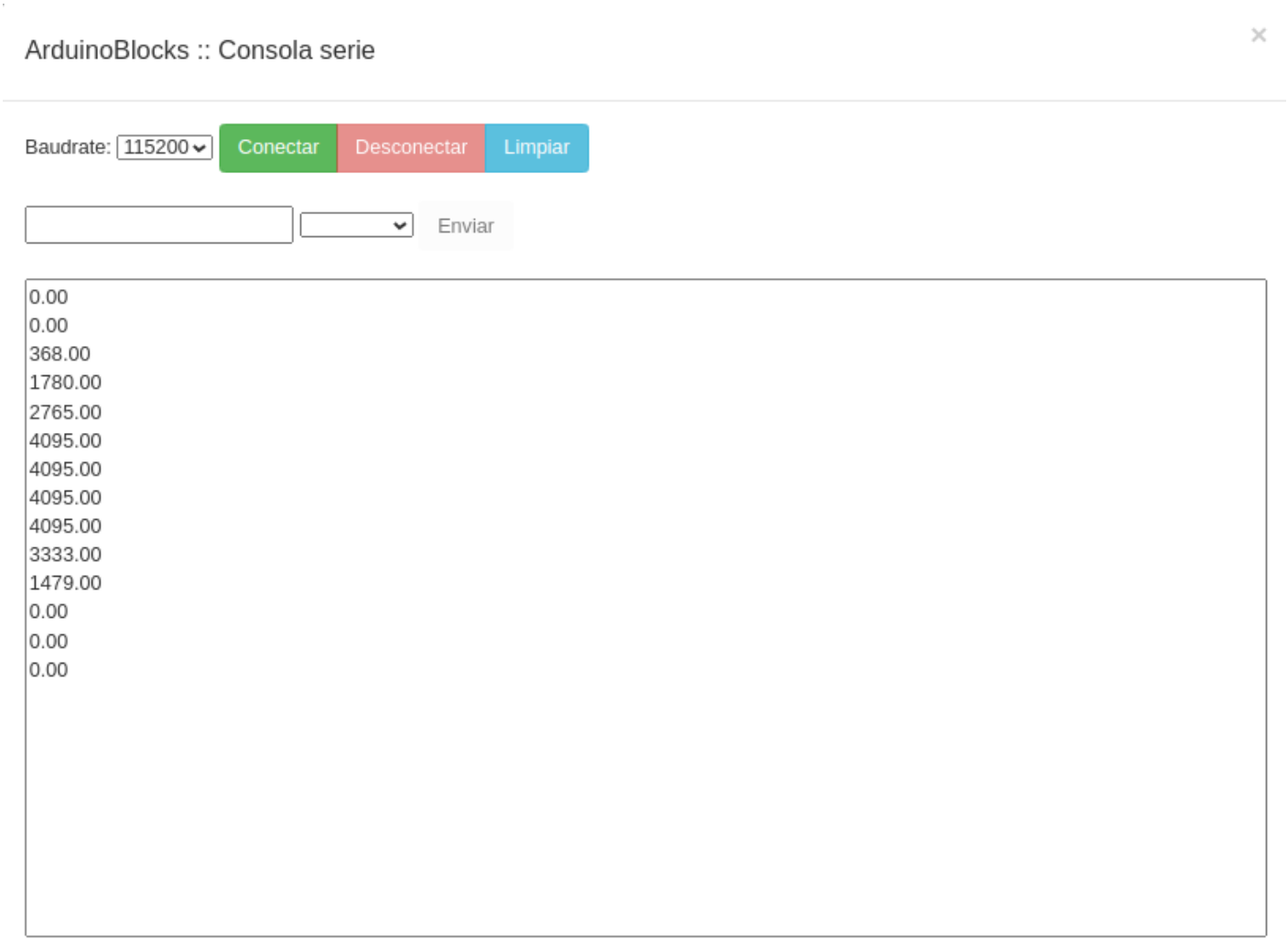
Imagen Federico Coca Notas sobre ESP32 STEAMakers CC-BY-SA

Programando la actividad

Parte 1. Vamos a guardar los datos leídos del potenciómetro en una variable y mostrarlos a través de la consola serie. La solución la tenemos disponible en Actividad-07: Parte 1 que es el programa que vemos en la imagen siguiente:

Imagen Federico Coca Notas sobre ESP32 STEAMakers CC-BY-SA

Si cargamos el programa en nuestra placa y posteriormente activamos la consola y conectamos ArduinoBlocks con nuestro ordenador podemos ver un resultado similar al de la imagen siguiente que se corresponde con variaciones de extremo a extremo del potenciómetro.



Aspecto de la consola [Imagen Federico Coca Notas sobre ESP32 STEAMakers](#) CC-BY-SA

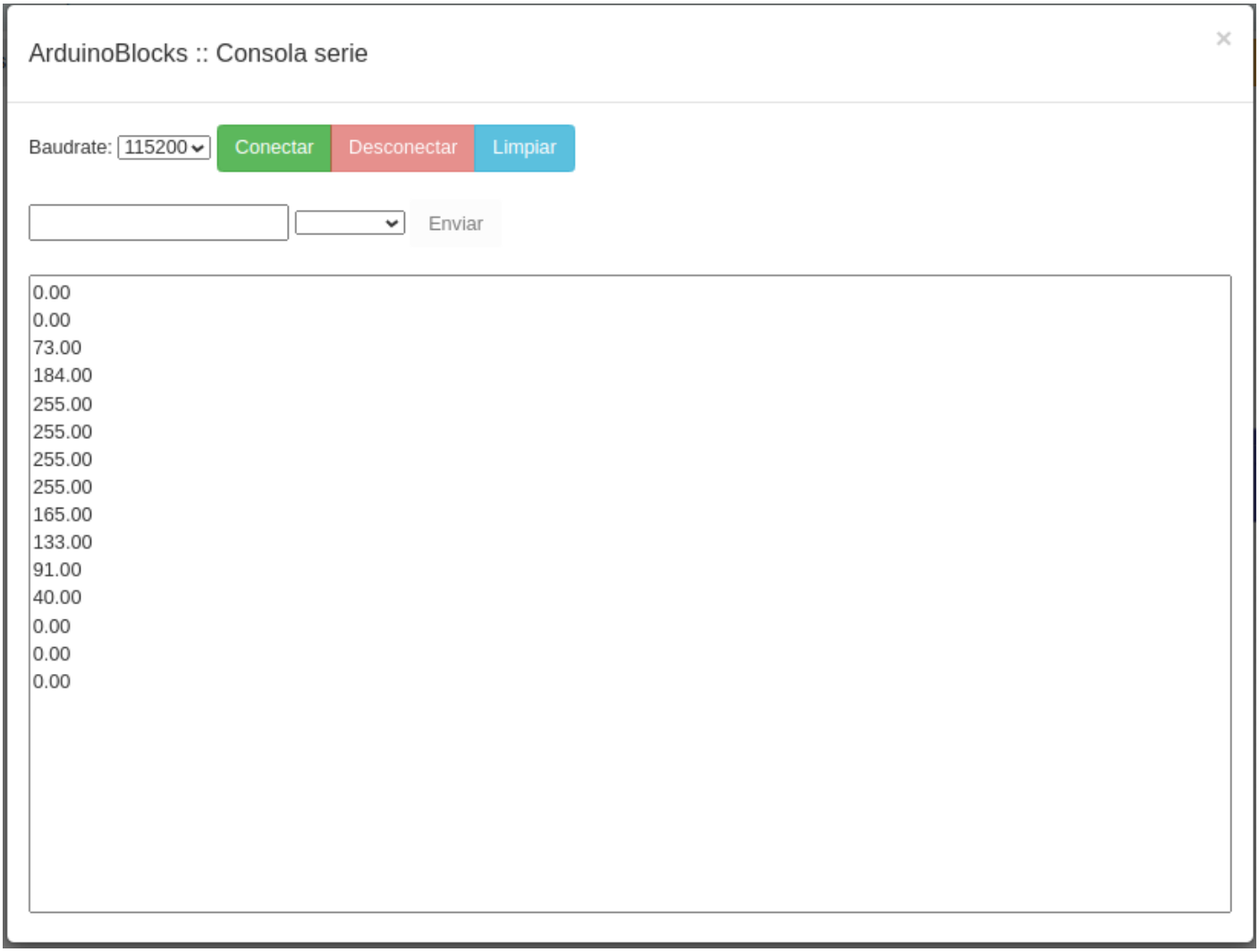
Para entender el significado del check "salto de linea" del bloque "Enviar" podemos desactivarlo y volver a hacer el mismo proceso.

> **¡MUY IMPORTANTE!** cuando terminemos de manipular el potenciómetro que el mismo esté totalmente girado a la posición izquierda (hacia el reset), ya que comparte la conexión A0 (GPIO02) con el sistema de grabación del programa.
> Si no está en esa posición se producirá un error en el envío del siguiente programa.

Parte 2. Vamos a modificar el programa de la parte 1 del reto para mapear los datos antes de enviarlos a la consola serie. La solución la tenemos disponible en Actividad-07: Parte 2 que es el programa que vemos en la imagen siguiente:

Parte 2 de la actividad 7 *Imagen Federico Coca Notas sobre ESP32 STEAMakers* *CC-BY-SA*

El resultado ahora lo vemos en la imagen siguiente:



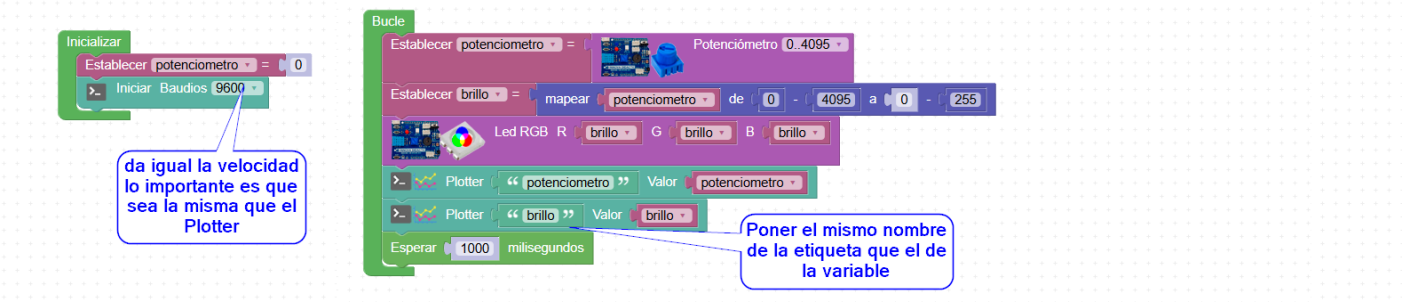
Aspecto de la consola [Imagen Federico Coca Notas sobre ESP32 STEAMakers](#) CC-BY-SA

ENVIO DE DATOS SERIE PERO EN VEZ DE CONSOLA POR PLOTTER

(Autor Javier Quintana)

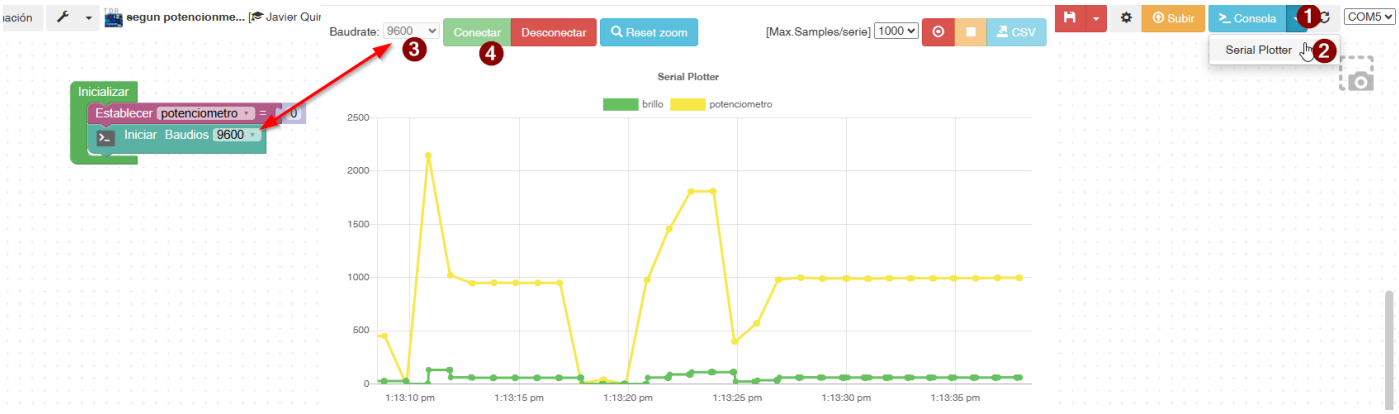
Ahora vamos a cambiar el "**Enviar**" serie por "**Plotter**"

Y además vamos a mapear el valor del potenciómetro de 0-4095 a 0-255 y lo almacenaremos en una variable "**brillo**" que nos servirá para que el RGB se ilumine más o menos:



> **Es importante** poner un retardo (en este caso 1000 mseg) si no lo ponemos, saturamos el puerto serie

Abrimos el plotter y jugamos:



> **¡MUY IMPORTANTE!** cuando terminemos de manipular el potenciómetro que el mismo esté totalmente girado a la posición izquierda (hacia el reset), ya que comparte la conexión A0 (GPIO02) con el sistema de grabación del programa.
> Si no está en esa posición se producirá un error en el envío del siguiente programa.

Retos de ampliación

- **A7.R1.** Utilización del bloque "crear texto con ..." del menú "Texto". Utilizaremos cualquiera de los programas vistos en la actividad y configuraremos el nuevo bloque para que en cada línea nos muestre el mensaje "El valor del potenciometro es: ", a continuación nos muestre el valor y tras el mismo el símbolo % porque configuraremos la lectura en porcentaje. Dejaremos transcurrir un tiempo de 3s entre cada muestra de salida por consola.
- **A7.R2.** Control del LED RGB con el potenciometro. Vamos a dividir el rango total en 8 partes y asignarle a cada una de ellas uno de los colores RGB según vemos en la tabla siguiente:

Color	Rango	R	G	B
Rojo	0 a 512	255	0	0
Verde	513 a 1023	0	255	0
Azul	1024 a 1536	0	0	255
Amarillo	1537 a 2048	255	255	0
Cian	2049 a 2560	0	255	255
Magenta	2561 a 3072	255	0	255
Blanco	3073 a 3584	255	255	255
Naranja	3584 a 4096	255	127	0

- **A7.R3.** Repetir el reto 2 de esta actividad pero ahora además mostrando por consola el valor del potenciometro.

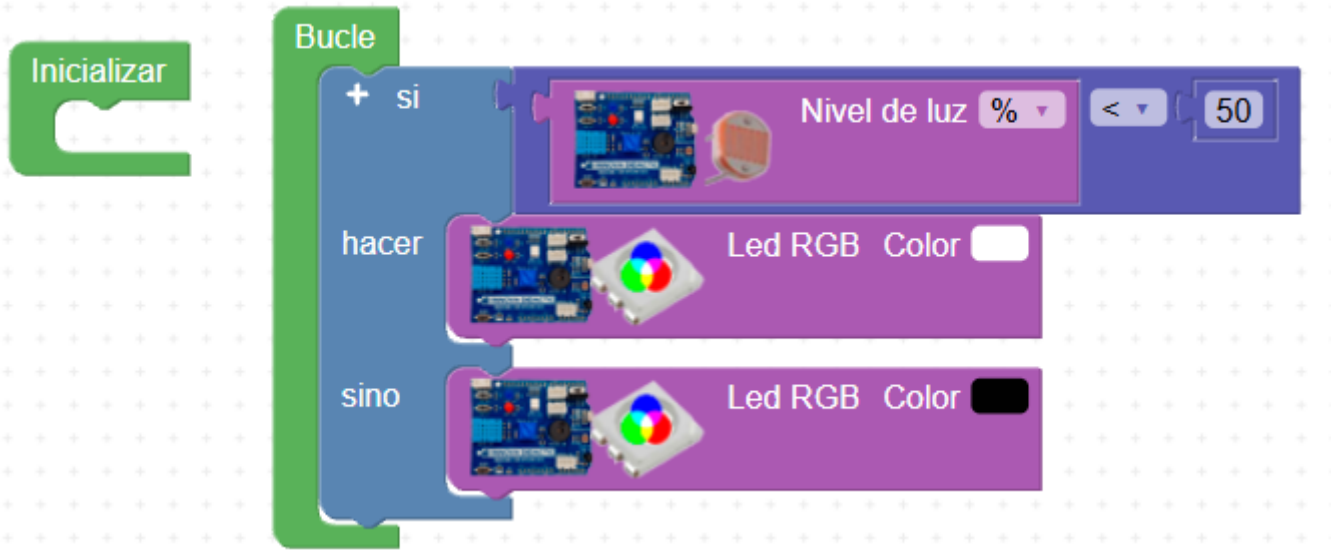
Copyright 2026

Actividad-08. La fotorresistencia o LDR

Interruptor Crepuscular

Para ver cómo funciona, qué mejor ejemplo que hacer un interruptor crepuscular, es decir "Que se encienda la luz cuando se haga de noche"

Para ello ponemos el nivel de luz en % y definimos qué es día y qué es noche con un valor umbral, en este ejemplo hemos puesto 50%



https://www.youtube.com/embed/1yza0ayy9wg

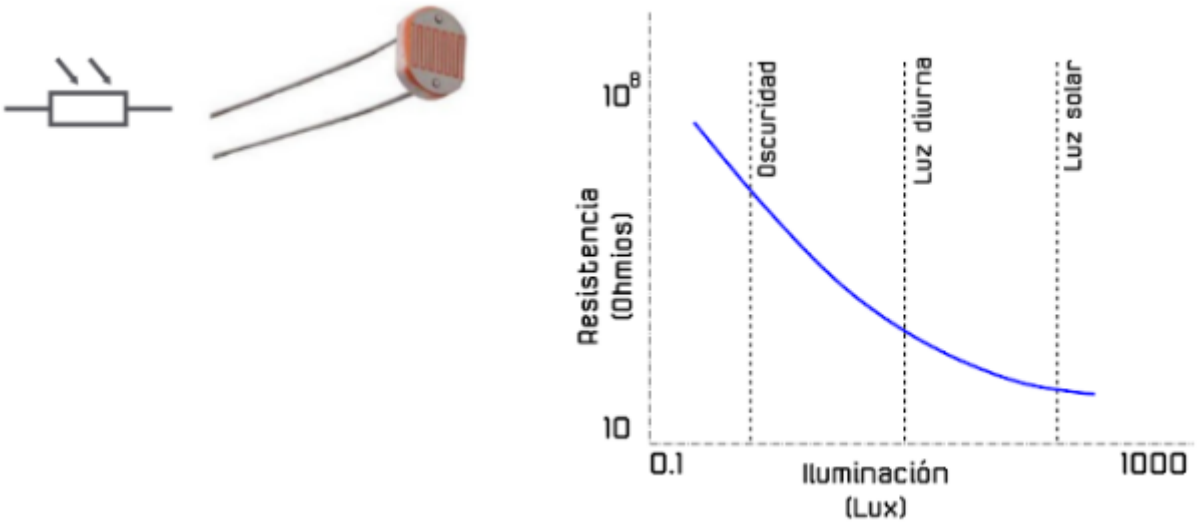
> Página extraída de Federico Coca Notas sobre ESP32 STEAMakers CC-BY-SA

Enunciado

Utilizaremos la consola serie para mostrar los valores de luz detectados por la resistencia LDR o fotocélula.

Teoría

Una fotorresistencia o fotorresistor es un componente electrónico cuya resistencia disminuye de forma exponencial con el aumento de la intensidad de luz incidente. Las siglas LDR vienen de su nombre en inglés, que es Light Dependent Resistor. En la imagen siguiente tenemos el símbolo, el aspecto real de una LDR y su curva característica de variación de resistencia con la iluminación.



Símbolo y aspecto de la LDR y curva característica *Imagen Federico Coca [Notas sobre ESP32 STEAMakers](#) CC-BY-SA*

En la TdR STEAM

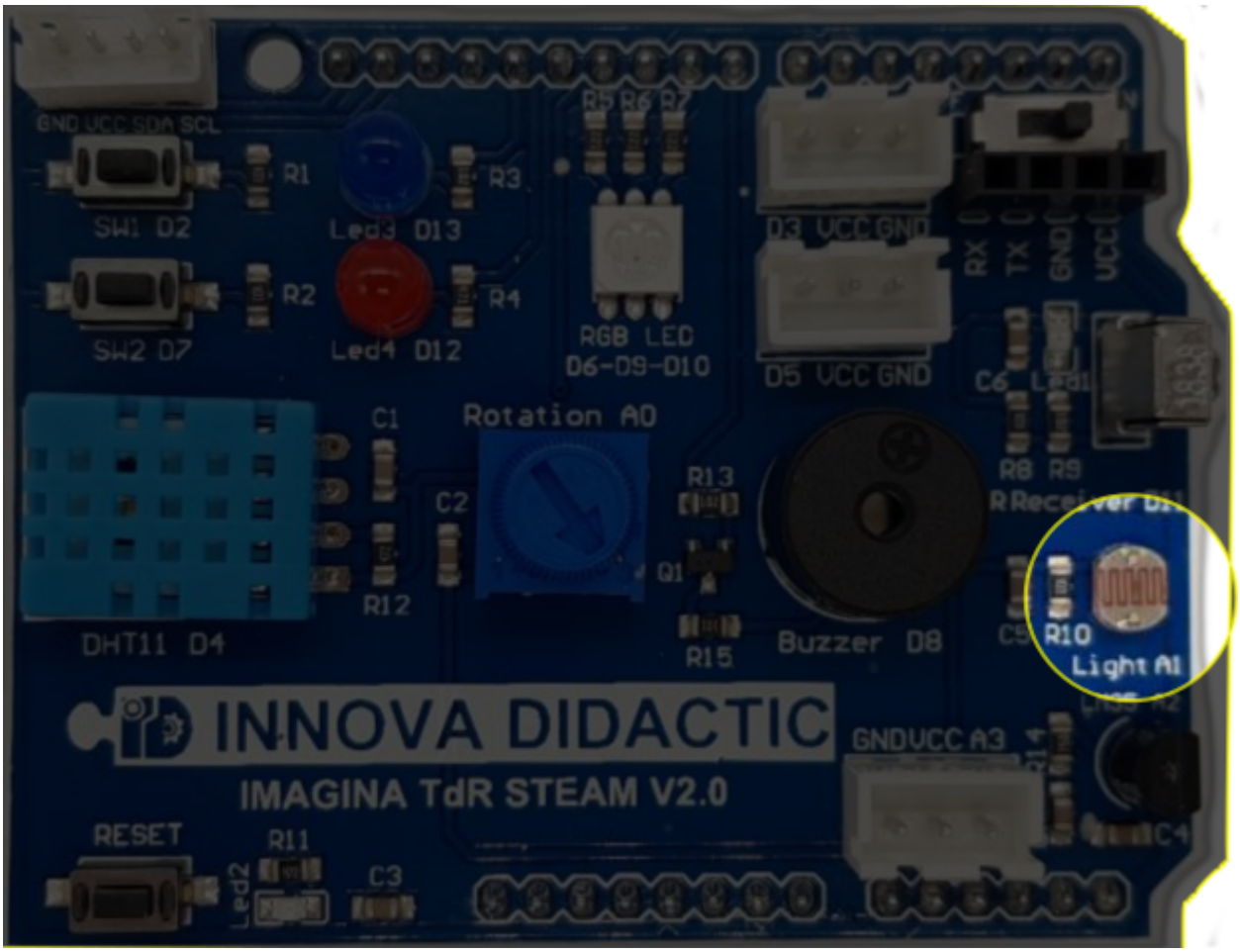


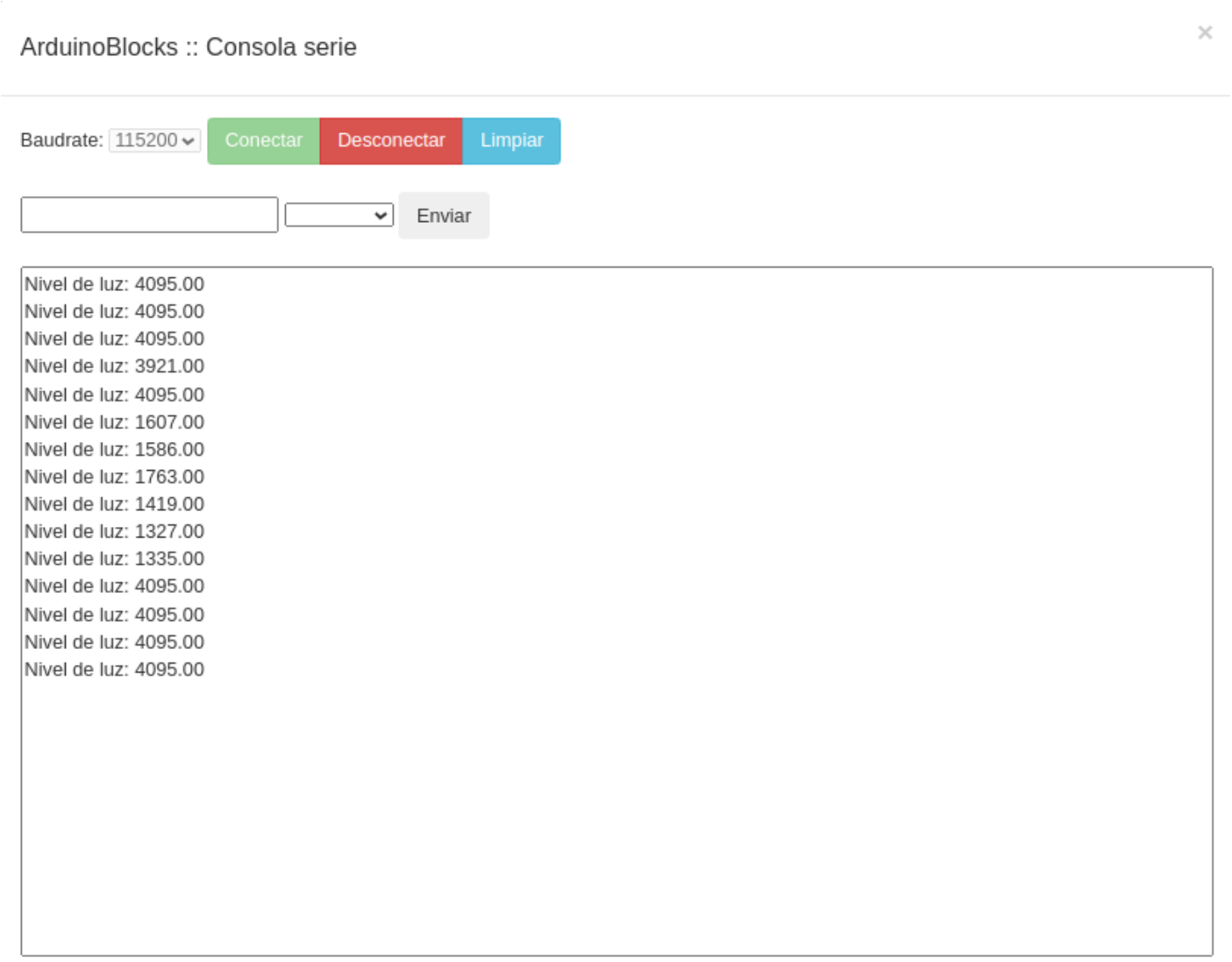
Imagen Federico Coca [Notas sobre ESP32 STEAMakers](#) CC-BY-SA

Programando la actividad

Dado que en el menú TDR STEAM de ArduinoBlocks tenemos disponible un bloque que nos devuelve el nivel de luz en porcentaje o de forma numérica, resolver la actividad es sumamente sencillo. La solución la tenemos disponible en ESP32-SM-Actividad-08 que es el programa que vemos en la imagen siguiente:

Actividad-08 *Imagen Federico Coca* [*Notas sobre ESP32 STEAMakers*](#) *CC-BY-SA*

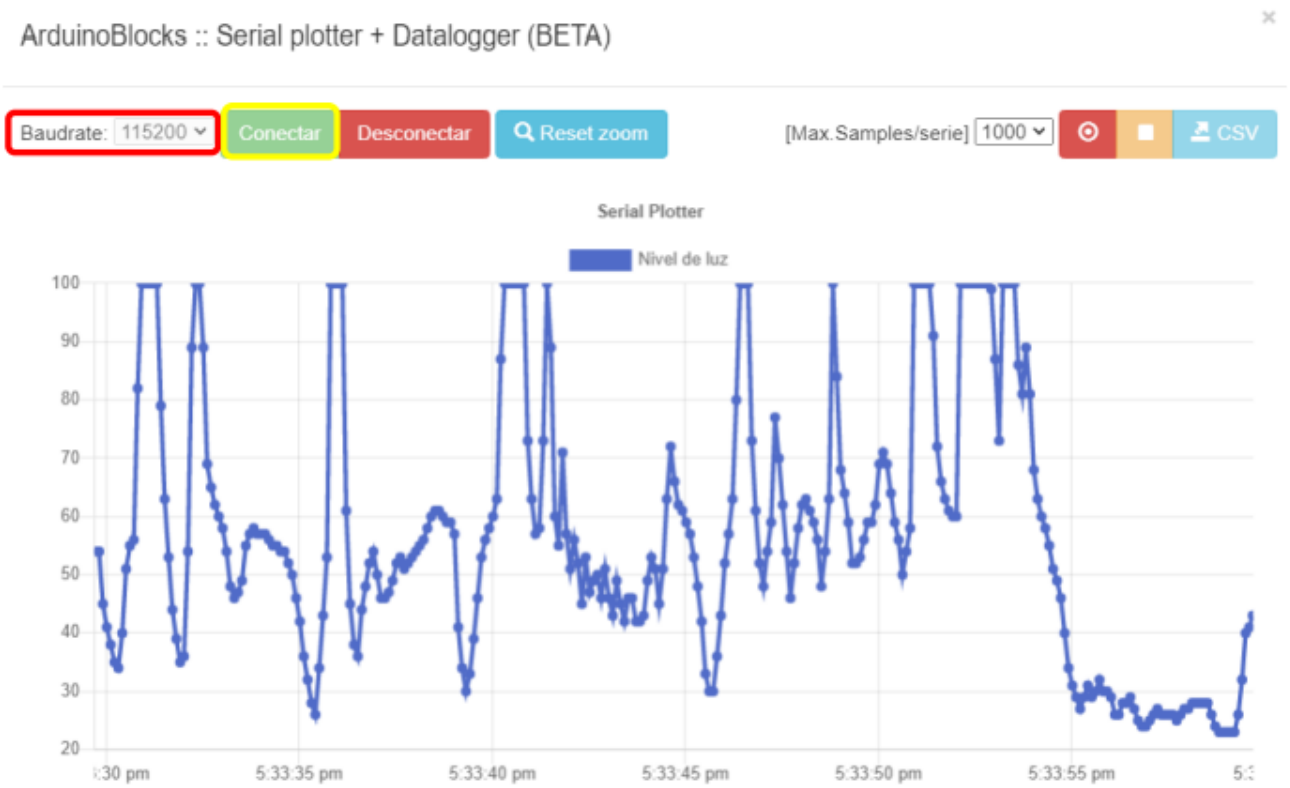
Esto nos generará algo como lo de la imagen siguiente en la consola:



Consola que produce la actividad 8 *Imagen Federico Coca [Notas sobre ESP32 STEAMakers](#) CC-BY-SA*

Opción Plotter gráfico

Es más visual y dinámico



Extraido de la página 167 Manual Steam Maker ESP32

- Descarga desde libros.catedu..es Manual Actividades ESP32 SteaMakers 2022_Junio.pdf

- Descarga desde el drive del autro tutorial Manual de Actvidades ESP32 SteamMaker 2022_Junio

Retos de ampliación

A8.R1. Detectar el nivel de iluminación de la sala y condiciones de iluminación en las que estemos utilizando la LDR y enviar el resultado a la consola.

A8.R2. Programar un interruptor crepuscular utilizando la LDR y uno de los LEDs para simular el farol. El nivel de luz mínimo permitido antes de encender el farol dependerá del resultado obtenido en la reto 1, de forma que con un valor menor o igual se encienda el LED y con un valor mayor permanezca apagado.

A8.R3. Hacer el interruptor crepuscular utilizando como farol en LED RGB que se encenderá en color blanco.

Actividad-09. El sensor de temperatura LM35D

> Página extraída de Federico Coca Notas sobre ESP32 STEAMakers CC-BY-SA

Enunciado

Utilizaremos este sensor de temperatura para medir la temperatura de la habitación en la que estemos resolviendo el reto mostrando el resultado en la consola.

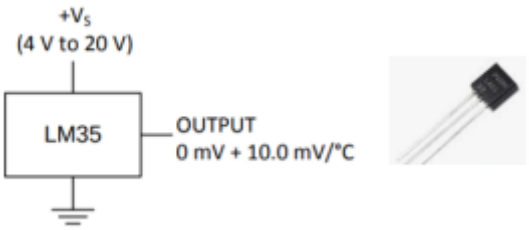
Teoría

LM35

El LM35 es un circuito integrado de medida de temperatura de precisión con una tensión de salida lineal y proporcional a la temperatura en grados centígrados. Algunas de las características que se pueden extraer de la hoja de datos del LM35 son:

- Directamente calibrado en grados Celsius (Centígrados)
- Factor de escala lineal de +10-mV/°C
- Resolución asegurada de 0.5°C
- Rango de medida de −55°C a 150°C

En la imagen siguiente vemos su representación en circuito y el aspecto físico que tiene en uno de sus encapsulados mas usual.



Representación esquemática y aspecto *Imagen Federico Coca Notas sobre ESP32 STEAMakers CC-*

BY-SA

Bloques de tiempo

Para esta actividad vamos a explicar los bloques de tiempo que se implementan en ArduinoBlocks. Las funciones de tiempo o retardo nos permiten realizar pausas y obtener información sobre el tiempo transcurrido dentro del microcontrolador.

1.- **Bloques esperar.** Realizan una pausa del tiempo que establezcamos hasta seguir con la ejecución del siguiente bloque. Se corresponden con las instrucciones delay(1000) y delayMicroseconds(1000). Tenemos disponibles los dos de la imagen siguiente:

Imagen Federico Coca [Notas sobre ESP32 STEAMakers](#) CC-BY-SA

2.- Tiempo transcurrido. Obtiene un valor con el tiempo transcurrido desde el inicio o reset del microcontrolador de la placa ESP32 STEAMakers. Se trata de las funciones millis() y micros(). El valor puede ser en milisegundos o microsegundos, tal y como vemos en la imagen siguiente:

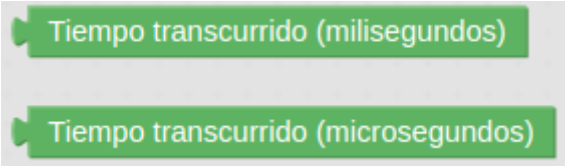


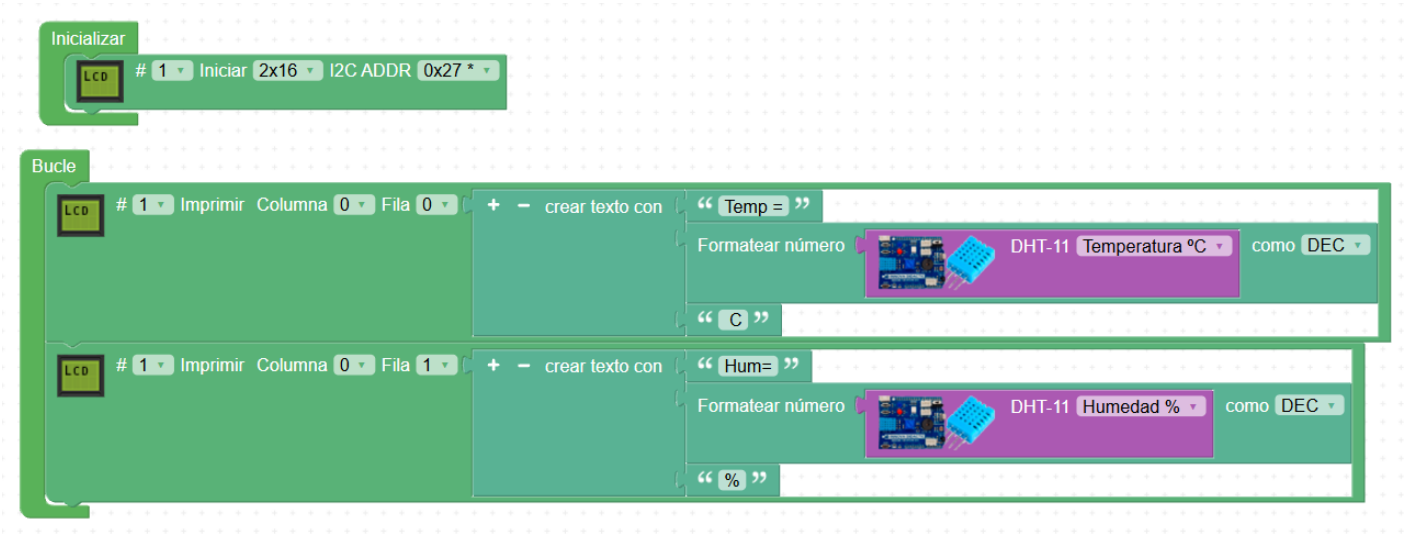
Imagen Federico Coca [Notas sobre ESP32 STEAMakers](#) CC-BY-SA

Actividad-10. El sensor de temperatura y humedad DHT-11

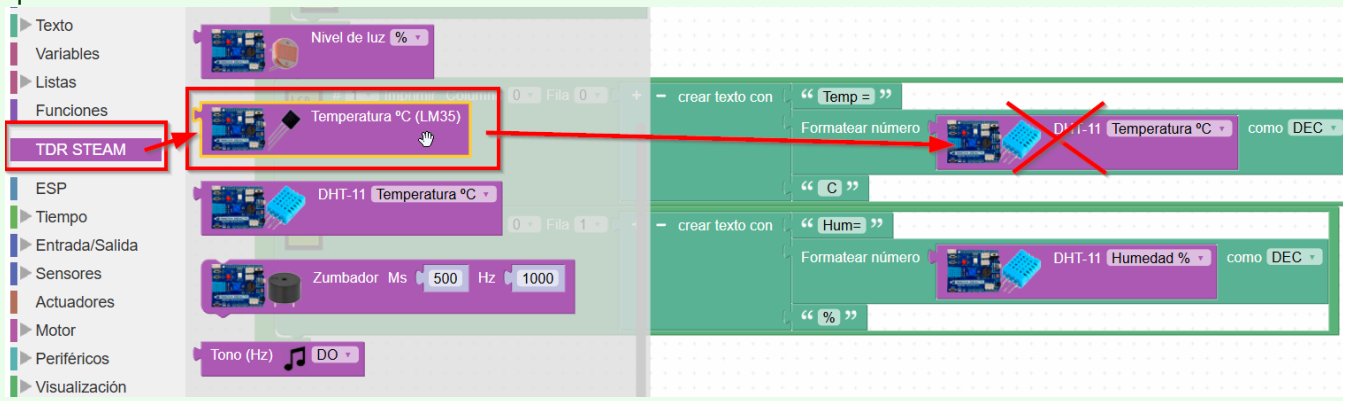
Visualización humedad y temperatura por el display LCD

El clásico es visualizar por el LCD la temperatura y la humedad instantánea.

- Más adelante veremos el LCD y por qué hay que inicializarlo con la dirección 0x27
- Fíjate que hay que formatear el valor que devuelve la función DHT-11 como decimal, la explicación lo vimos aquí en sensores DHT11



Haz un pequeño experimento:
Sustituye la instrucción de temperatura DHT11 por el LM35 ¿Sale el mismo valor?¿Cual crees que es el verdadero?



Convierte el ESP32 EN ESTACIÓN METEROLÓGICA
En la página 158 del Manual actividades ESP32 tienes un ejemplo de grabar la temperatura y la humedad en un fichero texto en una tarjeta microsim (la placa ESP32 incorpora un lector)

- Descarga desde libros.catedu..es Manual Actividades ESP32 SteaMakers 2022_Junio.pdf
- Descarga desde el drive del autro tutorial Manual de Actvidades ESP32 SteamMaker 2022_Junio

Página extraída de Federico Coca Notas sobre ESP32 STEAMakers CC-BY-SA

Enunciado

Realizar un programa básico de medida de temperatura y humedad relativa de la habitación en la que estamos trabajando con nuestra TdR STEAM.

Teoría

DHTxx

El DHT11 es un modelo de sensor que permiten realizar la medición simultánea de temperatura y humedad. Dispone de un procesador interno que es el encargado de realizar la medición entregando la información mediante una señal digital.

Se presenta en un encapsulado plástico típico de color azul. Sus principales características son:

- Rango de temperatura: 0 a 50ºC
- Precisión de la medida de temperatura: ±2ºC
- Rango de humedad: 20 a 80%
- Precisión en la medida de humedad: ±5%.
- Frecuencia de muestreo: 1 muestra por segundo (1 Hz)

El DHT11 es un sensor bastante limitado que podemos usar con fines de formación, pruebas, o en proyectos que realmente no requieran una medición precisa.

Si necesitamos mayor precisión y rango podemos recurrir al DHT22 que es de la misma familia y lo único que cambia es sus características y el precio. Sus características son:

- Rango de temperatura: -40 a 125ºC
- Precisión de la medida de temperatura: ±0.5ºC
- Rango de humedad: 0 a 100%
- Precisión en la medida de humedad: ±2 a 5%
- Frecuencia de muestreo: 2 muestras por segundo (2 Hz)

En la imagen siguiente vemos el aspecto de ambos sensores:

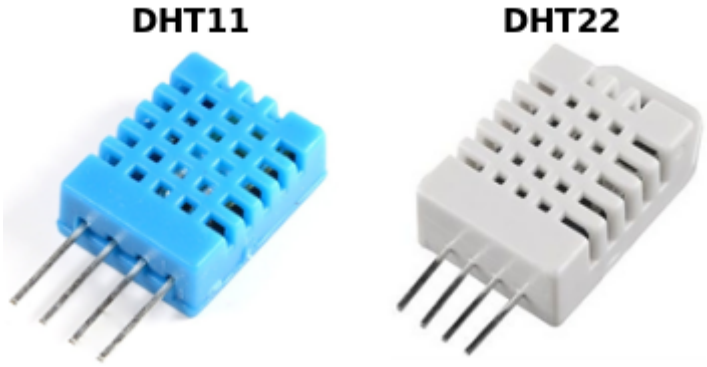


Imagen Federico Coca [Notas sobre ESP32 STEAMakers](#) CC-BY-SA

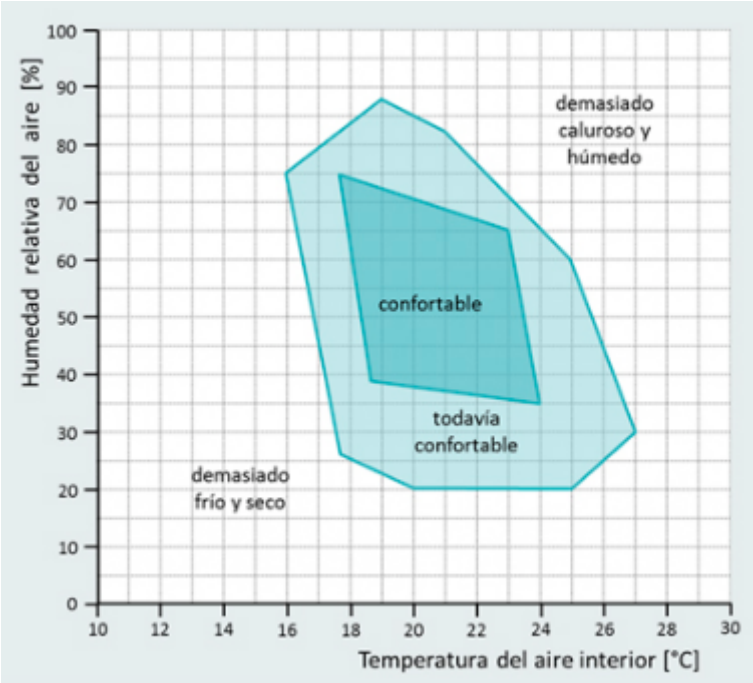
Estado de confort

En la web [ARQUITECTURA & ENERGÍA](#) podemos encontrar un artículo donde se nos explica con bastante profundad el tema del confort térmico.

Puede definirse confort térmico, o más propiamente comodidad higrotérmica, como la ausencia de malestar térmico. En fisiología, se dice que hay confort higrotérmico cuando no tienen que

intervenir los mecanismos termorreguladores del cuerpo para una actividad sedentaria y con una indumentaria ligera. Esta situación puede registrarse mediante índices que no deben ser sobrepasados para que no se pongan en funcionamiento los sistemas termorreguladores (metabolismo, sudoración y otros).

En la imagen siguiente vemos los valores de temperatura y humedad que delimitan las zonas de confortabilidad.

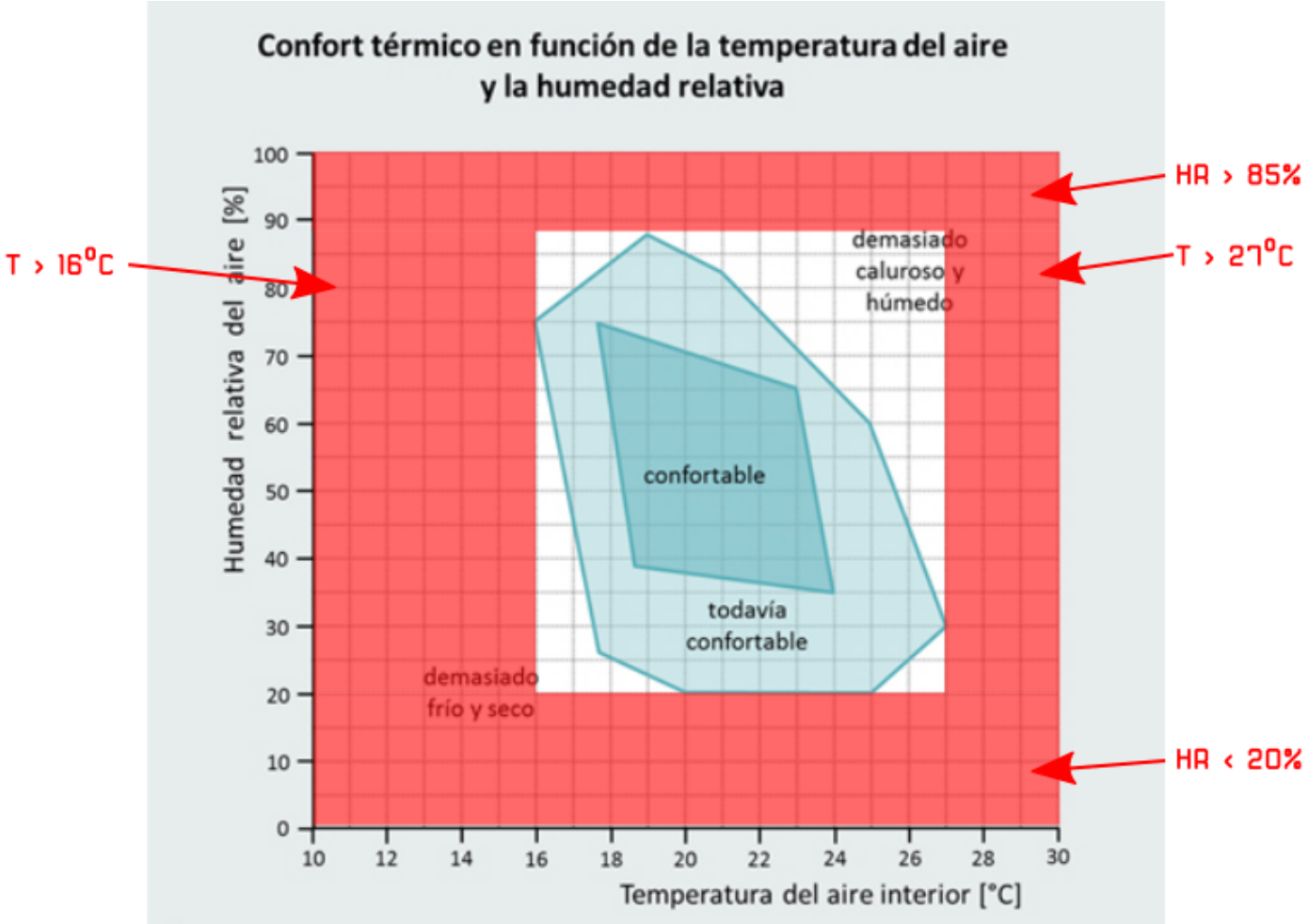


Confort térmico en función de temperatura y humedad Imagen Federico Coca [Notas sobre ESP32 STEAMakers](#) CC-BY-SA

Sobre el gráfico vamos a delimitar zonas de temperatura y humedad para establecer su color. Por motivos de simplicidad lo vamos a hacer delimitando zonas rectangulares, pero comprobamos que no cometemos grandes errores y para nuestro propósito nos sirve.

1.- Zona Roja: en la imagen siguiente tenemos delimitadas las zonas:

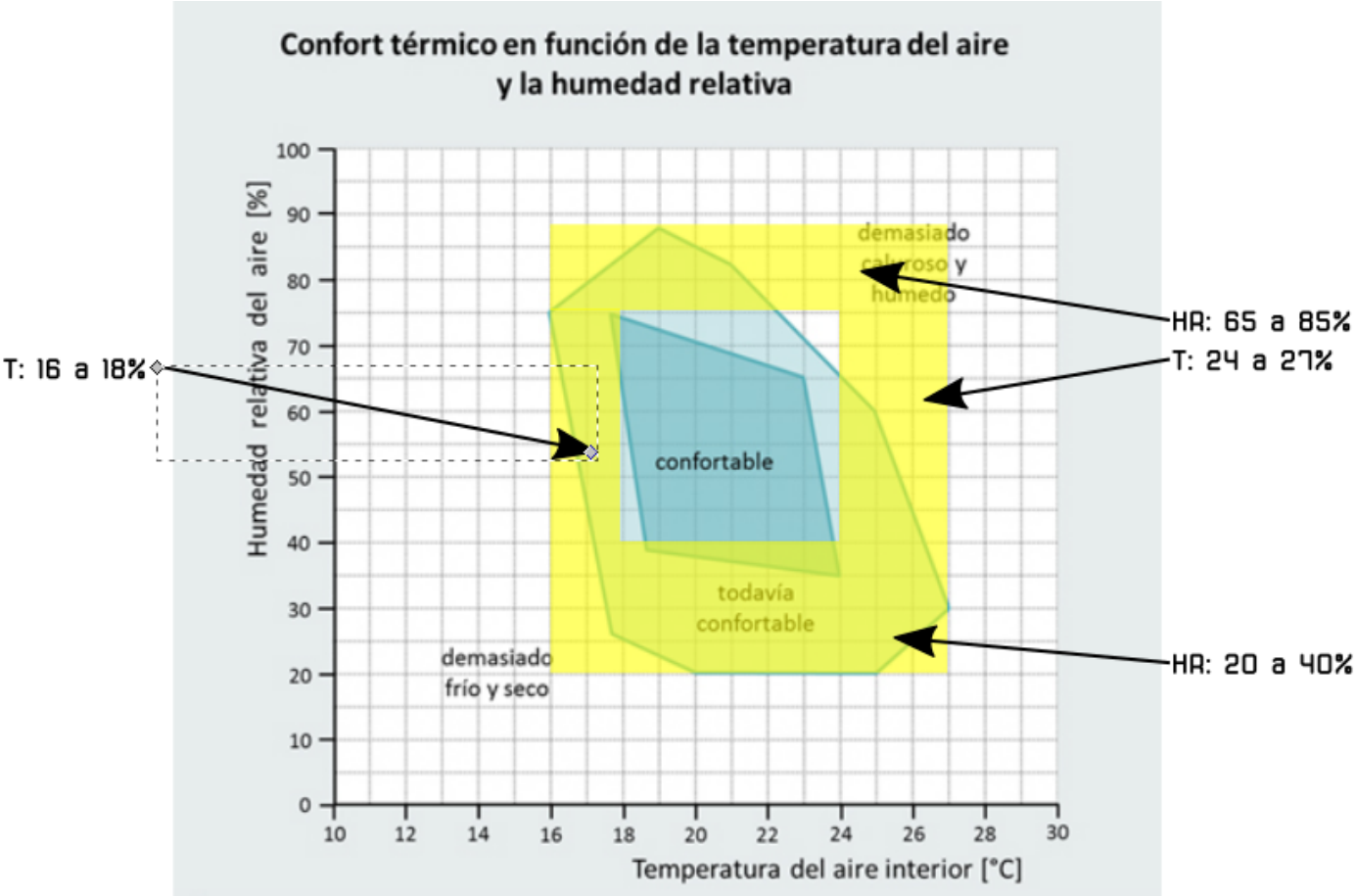
- Humedad Relativa: superior al 85% e inferior al 20%
- Temperatura: superior a 27ºC e inferior a 16ºC



Delimitación color rojo zona de confort *Imagen Federico Coca [Notas sobre ESP32 STEAMakers](#) CC-BY-SA*

2.- Zona Amarilla: en la imagen siguiente tenemos delimitadas las zonas:

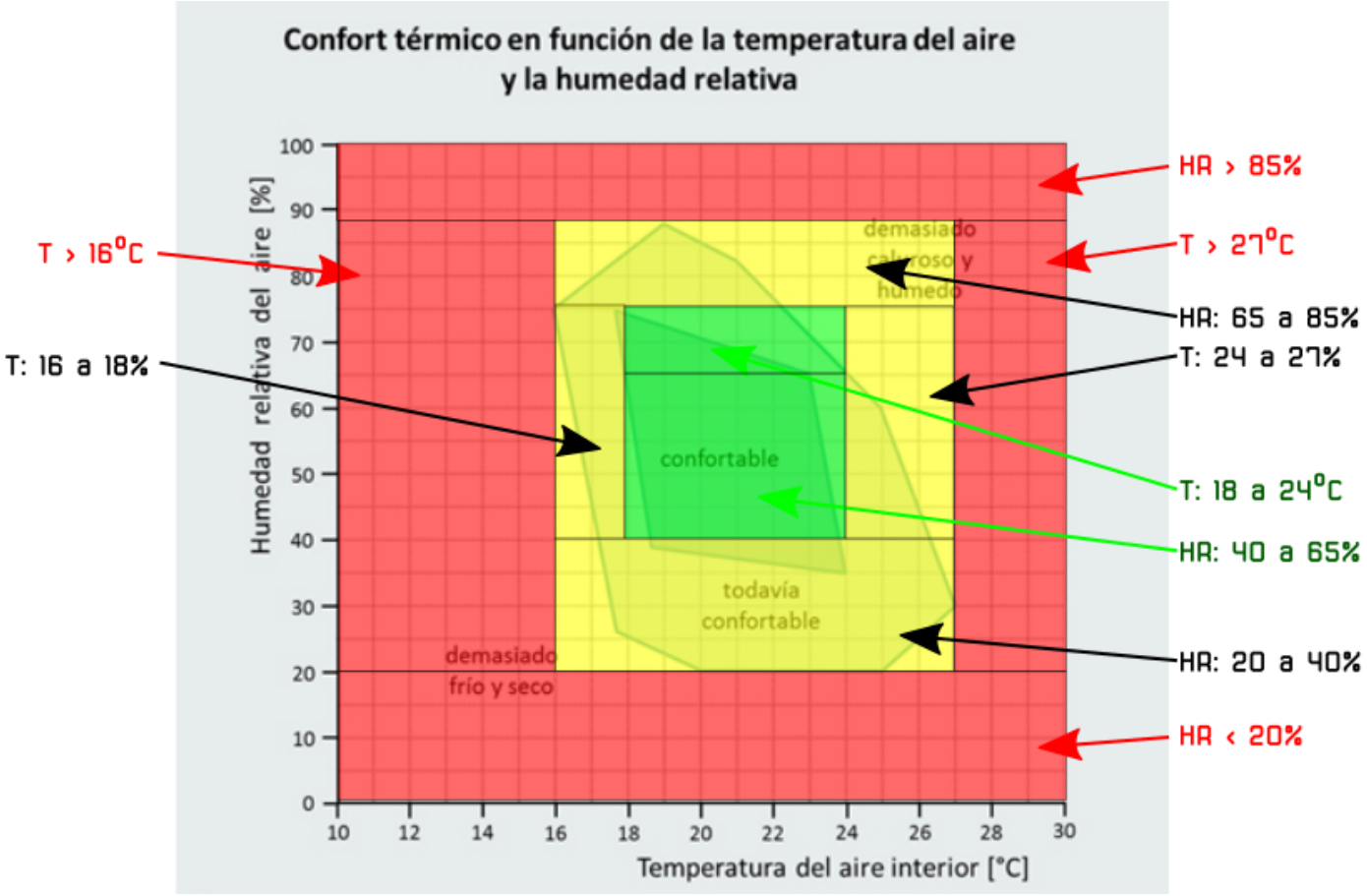
- Humedad Relativa: entre el 20% y el 40% y entre el 65% y el 85%
- Temperatura: entre 16ºC y 18ºC y entre 24ºC y 27ºC



Delimitación color amarillo zona de confort *Imagen Federico Coca* [*Notas sobre ESP32 STEAMakers*]
CC-BY-SA

3.- Zona Verde, rojo y amarillo: en la imagen siguiente tenemos delimitadas todas las zonas, correspondiendo a la verde los siguientes datos:

- Humedad Relativa: entre el 40% y el 65%
- Temperatura: entre 18ºC y 24ºC



Delimitación colores zona de confort *Imagen Federico Coca* [*Notas sobre ESP32 STEAMakers*](#) *CC-BY-SA*

Con este enlace al archivo colores-A10.svg puedes descargarte el archivo vectorial, editarlo con Inkscape y ver como se han realizado estos gráficos.

En la TdR STEAM

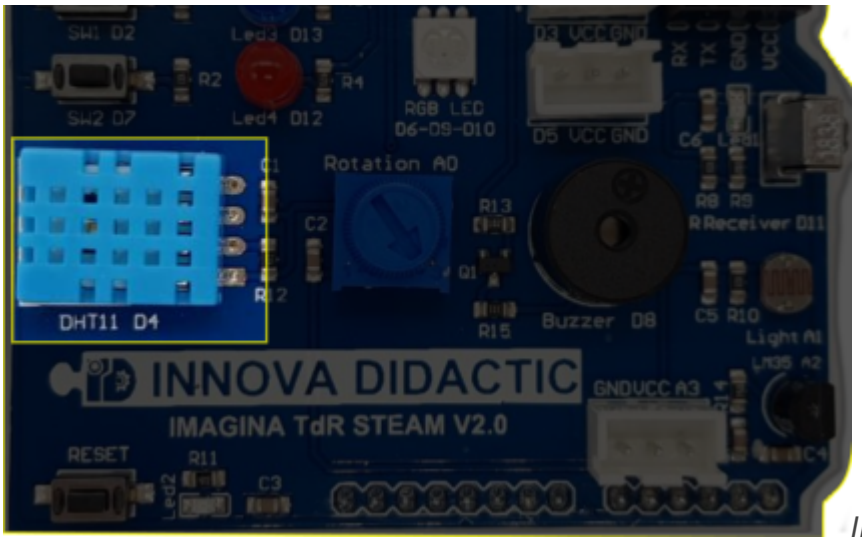
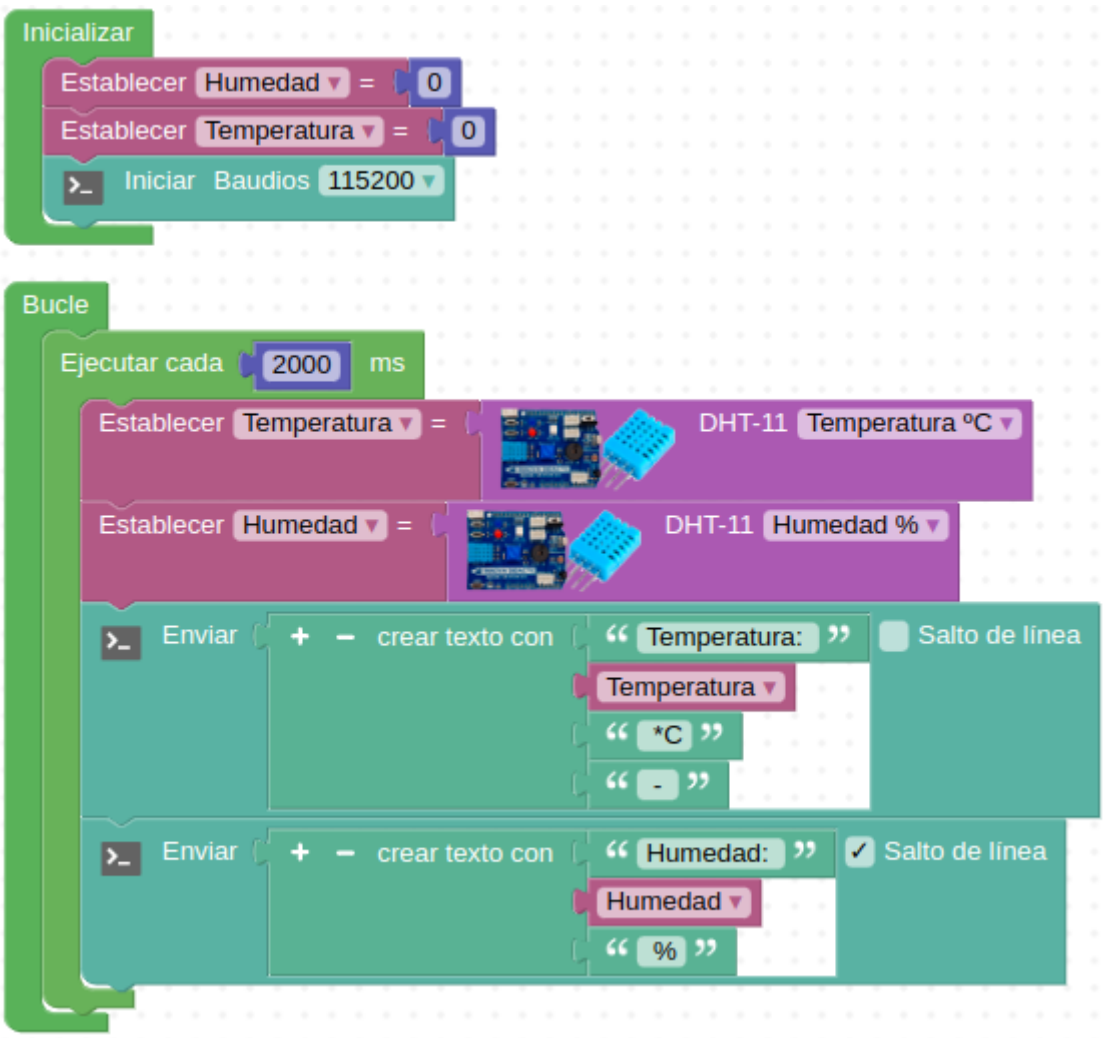


Imagen Federico Coca [Notas sobre ESP32 STEAMakers](#) CC-BY-SA

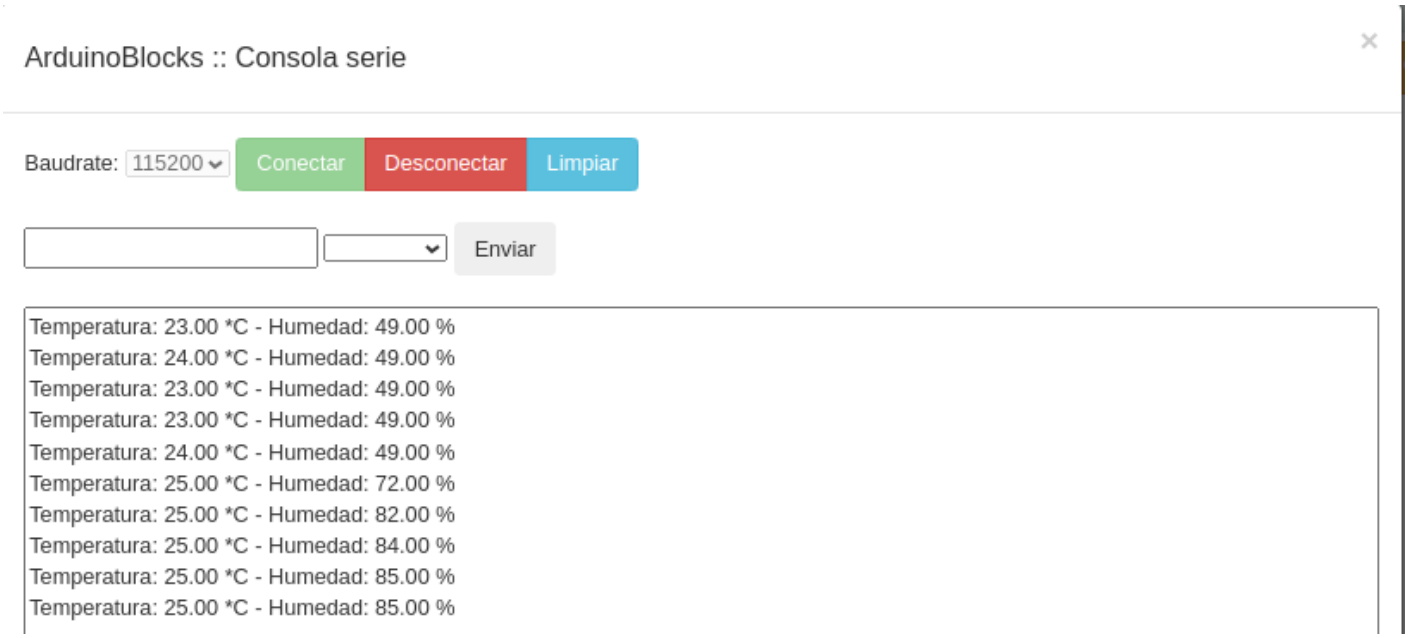
Programando la actividad

De nuevo realizaremos la programación como en las actividades anteriores, utilizando el monitor serie para mostrar los datos. El programa ESP32-SM-Actividad-10 lo vemos en la imagen siguiente:



Actividad-10 Imagen Federico Coca [Notas sobre ESP32 STEAMakers](#) CC-BY-SA

Esto nos generará algo como lo de la imagen siguiente en la consola:



Consola que produce la actividad 10 *Imagen Federico Coca [Notas sobre ESP32 STEAMakers](#) CC-BY-SA*

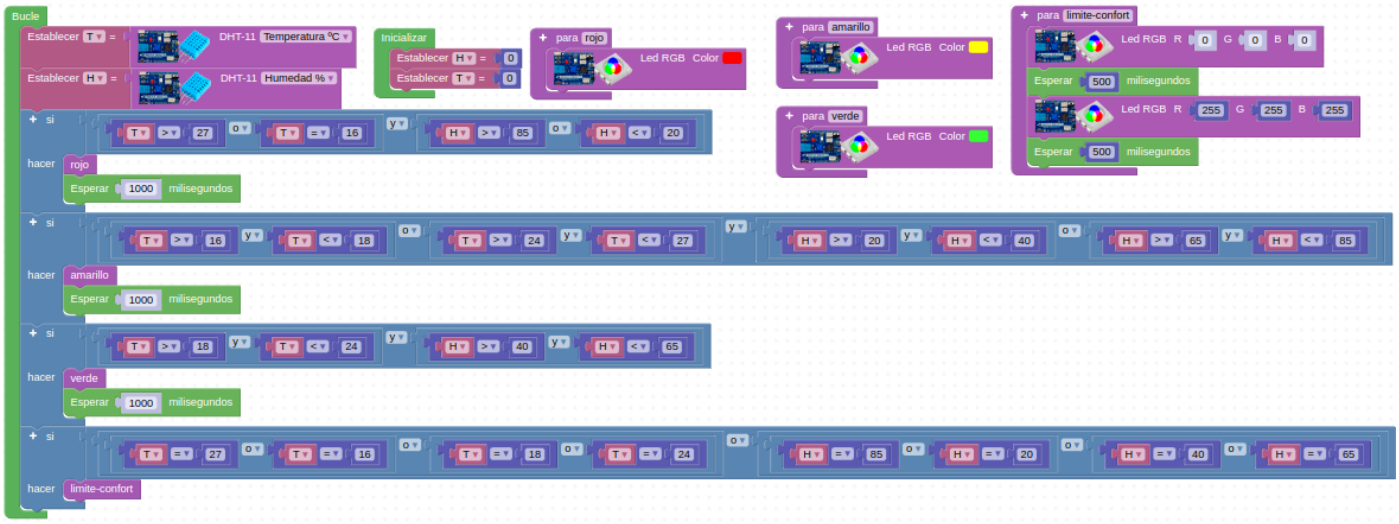
Retos de ampliación

A10.R1. Realizar un programa que nos muestre el estado de confort según las explicaciones vista y la idea de un semáforo que utilice el LED RGB para componer esos colores rojo, verde y amarillo.

Para resolver la actividad vamos a necesitar varios bloques del menú "Lógica" y especialmente combinando funciones AND y OR múltiples. Si necesitamos, por ejemplo, aumentar el número de operadores AND dentro de un bloque simplemente tenemos que combinarlo como vemos en la imagen siguiente, donde se han combinado cuatro bloques AND.

Combinación de 4 bloques AND *Imagen Federico Coca [Notas sobre ESP32 STEAMakers](#) CC-BY-SA*

El Programa es el de la imagen siguiente y está disponible en [ESP32-SM-Reto-1-A10](#).



Reto 1 de la actividad 10 *Imagen Federico Coca* [*Notas sobre ESP32 STEAMakers*](#) *CC-BY-SA*

A10.R2. Idear un método para probar de forma completa la funcionalidad del programa anterior, aunque se requiera modificarlo. También se pide explicar la misión de la función "limite-confort" creada en el reto 1 de la actividad 10.

Actividad-11. Emisor y receptor de infrarrojos

Página extraída de Federico Coca Notas sobre ESP32 STEAMakers CC-BY-SA

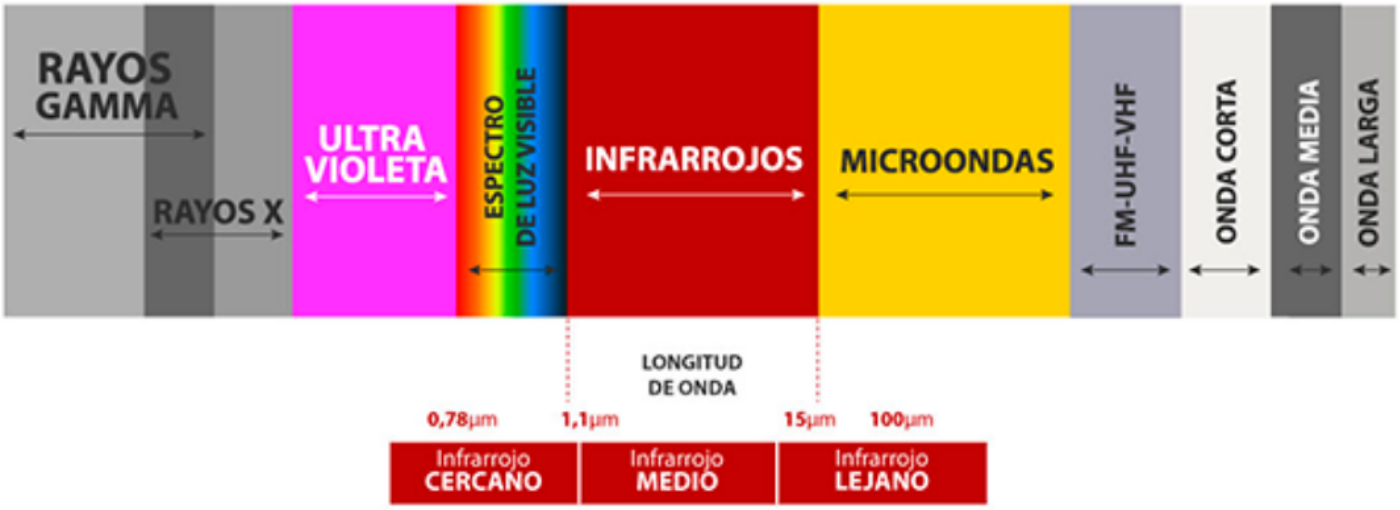
Enunciado

Enviaremos códigos desde el control remoto por infrarrojos que serán recibidos por el LED de infrarrojos para actuar según el código recibido.

Teoría

¿Qué son los infrarrojos?

Son una clase de radiación electromagnética con una longitud de onda que resulta superior a la longitud de onda de la luz visible, siendo su frecuencia superior a las microondas. Dentro del espectro electromagnético, la radiación infrarroja se encuentra comprendida entre el espectro de luz visible y las microondas. Tiene longitudes de onda mayores o más largas que el rojo. En la imagen siguiente, obtenida del blog de Mercedes González Mas vemos caracterizados los infrarrojos dentro del espectro.

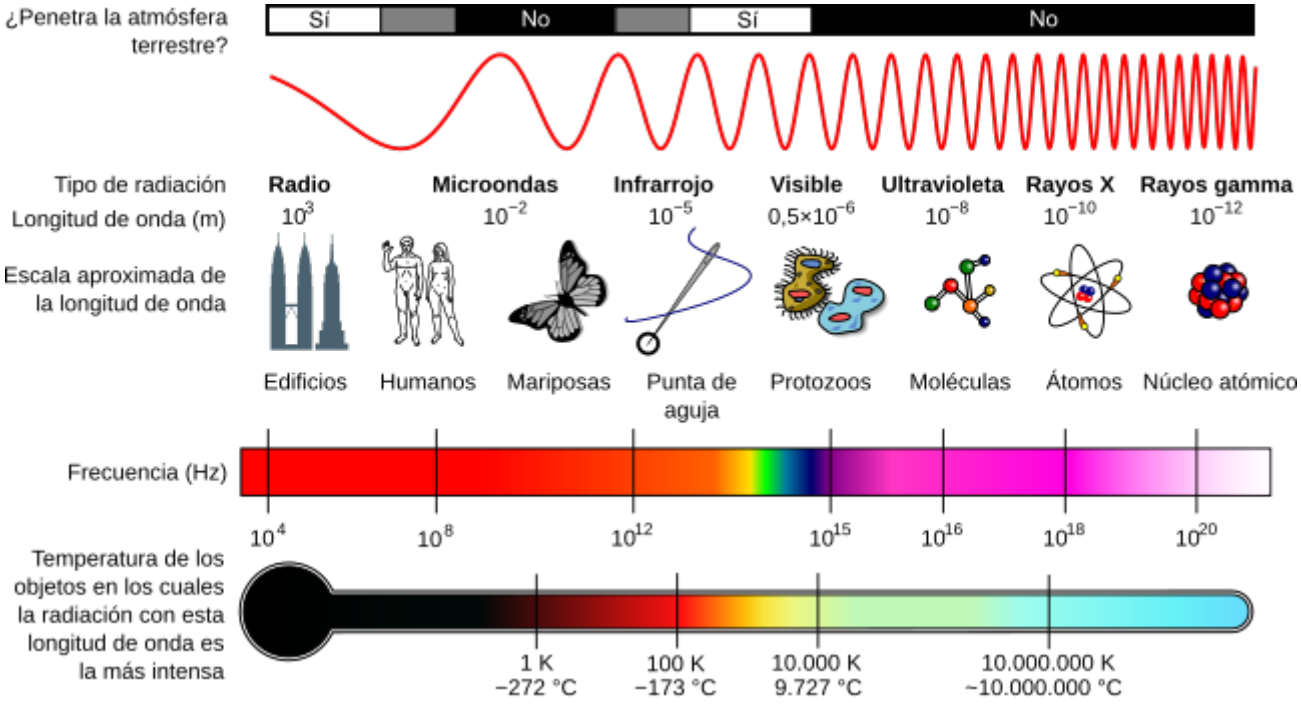


Espectro electromagnético Imagen Federico Coca [Notas sobre ESP32 STEAMakers](#) CC-BY-SA

Como podemos observar en la imagen, los rayos infrarrojos son clasificados, de acuerdo a su longitud de onda, del siguiente modo:

- infrarrojo cercano, con longitud de onda entre 0.7 µm y 1.1 µm, es la parte del espectro infrarrojo que ese encuentra más próximo a la luz visible.
- infrarrojo medio, con longitud de onda entre 1,1 µm y 15 µm.
- infrarrojo lejano o región más cercana a las microondas, con longitud de onda entre 15 µm y 100 µm

En la imagen siguiente, obtenida de Wikipedia, sobre espectro electromagnético podemos ver más información del tema.



Espectro electromagnetico Crates. Original version in English by Inductiveload, Public domain, via Wikimedia Commons

Todos los cuerpos emiten una cierta cantidad de radiación, y aunque esta resulta invisible para el ojo humano, existen dispositivos electrónicos capaces de "verla" por estar diseñados para ello.

Receptor de infrarrojos

Uno de los receptores más universal utilizado en placas tipo Arduino es el receptor de infrarrojos universal TL1838, VS1838B o simplemente 1835 de 38KHz, que es el que lleva montado la placa

TdR STEAM y cuyo aspecto podemos ver en la imagen siguiente:

Imagen Federico Coca Notas sobre ESP32 STEAMakers CC-BY-SA

En el datasheet TL1838 del dispositivo tenemos toda la información sobre el mismo destacando las siguientes características:

- Voltaje de funcionamiento: 2.7V a 5.5V
- Frecuencia: 37.9KHz
- Ángulo de recepción: 90°
- Rango de recepción: 18m

El dispositivo genera una señal de salida que sirve para controles remotos universales y utiliza la codificación NEC. El receptor de infrarrojos permite codificar los protocolos de señales de pulsos infrarrojos utilizados por los mandos a distancia. Los protocolos detectados son los siguientes: RC5, RC6, NEC, SONY, PANASONIC, JVC, SAMSUNG, WHYNTER, AIWA, LG, SANYO, MITSUBISHI y DENON. Es decir, detectaría cualquier señal emitida por cualquiera de esos mandos.

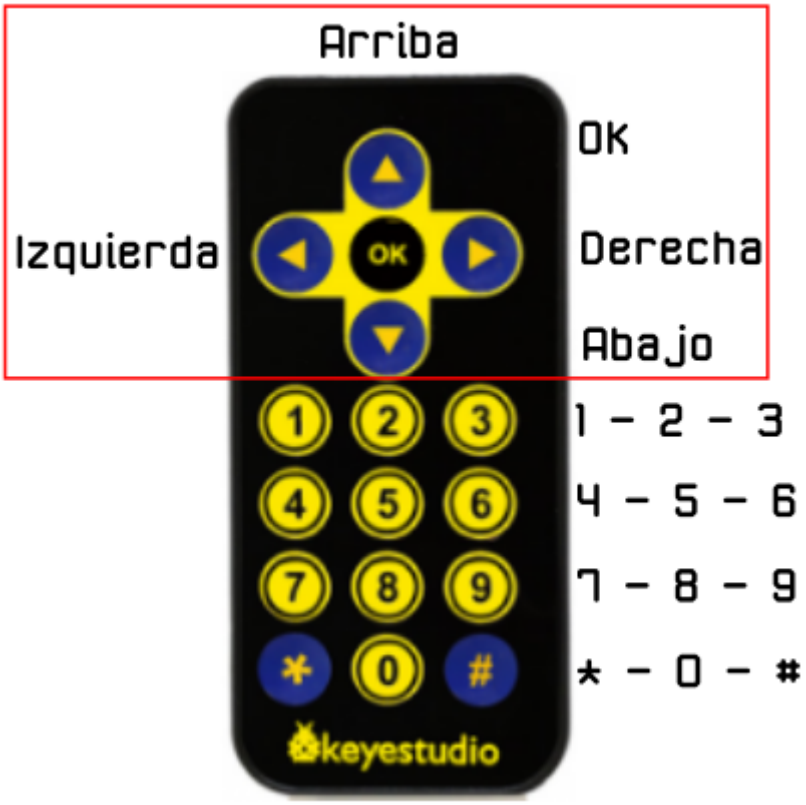
Emisor de infrarrojos

En nuestro caso como emisor de infrarrojos vamos a utilizar el control remoto de Keyestudio que vemos en la imagen siguiente:

El mini control remoto tiene 17 teclas de función y tiene las siguientes especificaciones:

- Batería: pilas de botón CR2025
- Distancia de transmisión: hasta 8 m
- Ángulo efectivo: 60°

El control remoto, o mando a distancia, por IR funciona emitiendo trenes de pulsos de luz infrarroja. Diferentes señales corresponden a botones diferentes La señal infrarroja transmite el código correspondiente al botón del mando a distancia pulsado al dispositivo en forma de una serie de impulsos de luz infrarroja. El receptor recibe la serie de impulsos de infrarrojos y los pasa a un procesador que decodifica y activarán una determinada función del dispositivo. En el reto y las actividades del mismo obtendremos estos códigos. En ArduinoBlocks se han asignado los siguientes nombres a las teclas:



Nombre teclas control remoto en ArduinoBlocks *Imagen Federico Coca [Notas sobre ESP32 STEAMakers](#) CC-BY-SA*

> **TRUCO** En las prácticas con Infrarrojos, cuando las cosas no van, es muy útil saber si el problema es que el emisor de infrarrojos funciona o no funciona para descartar otros problemas.
> El truco es simple, y sirve **para cualquier mando:** Como las cámaras de los móviles **si** detectan el Infrarrojo no visible, utiliza la cámara de tu móvil hacia el led del emisor del infrarrojos y aprieta cualquier botón, si se enciende el led (visto desde la cámara de tu móvil) el emisor **funciona**.

Bloques en ArduinoBlocks

El sensor receptor de infrarrojos permite obtener la cadena de texto con el código en formato hexadecimal correspondiente al tren de pulsos de IR generado al pulsar una determinada tecla. El bloque es el que vemos en la imagen siguiente:

Imagen Federico Coca [Notas sobre ESP32 STEAMakers](#) CC-BY-SA

El valor devuelto por el bloque de recepción será una cadena de texto con valor vacío en caso de no detectar ningún código. Al devolver el bloque una cadena de texto debemos recordar que lo tenemos que almacenar en una variable de tipo texto.

Si utilizamos mandos genéricos RC5 como el modelo de Keyestudio, podemos usar el bloque de la imagen siguiente para comparar los códigos recibidos y así identificar fácilmente cada tecla.

Imagen Federico Coca [Notas sobre ESP32 STEAMakers](#) CC-BY-SA

En la TdR STEAM

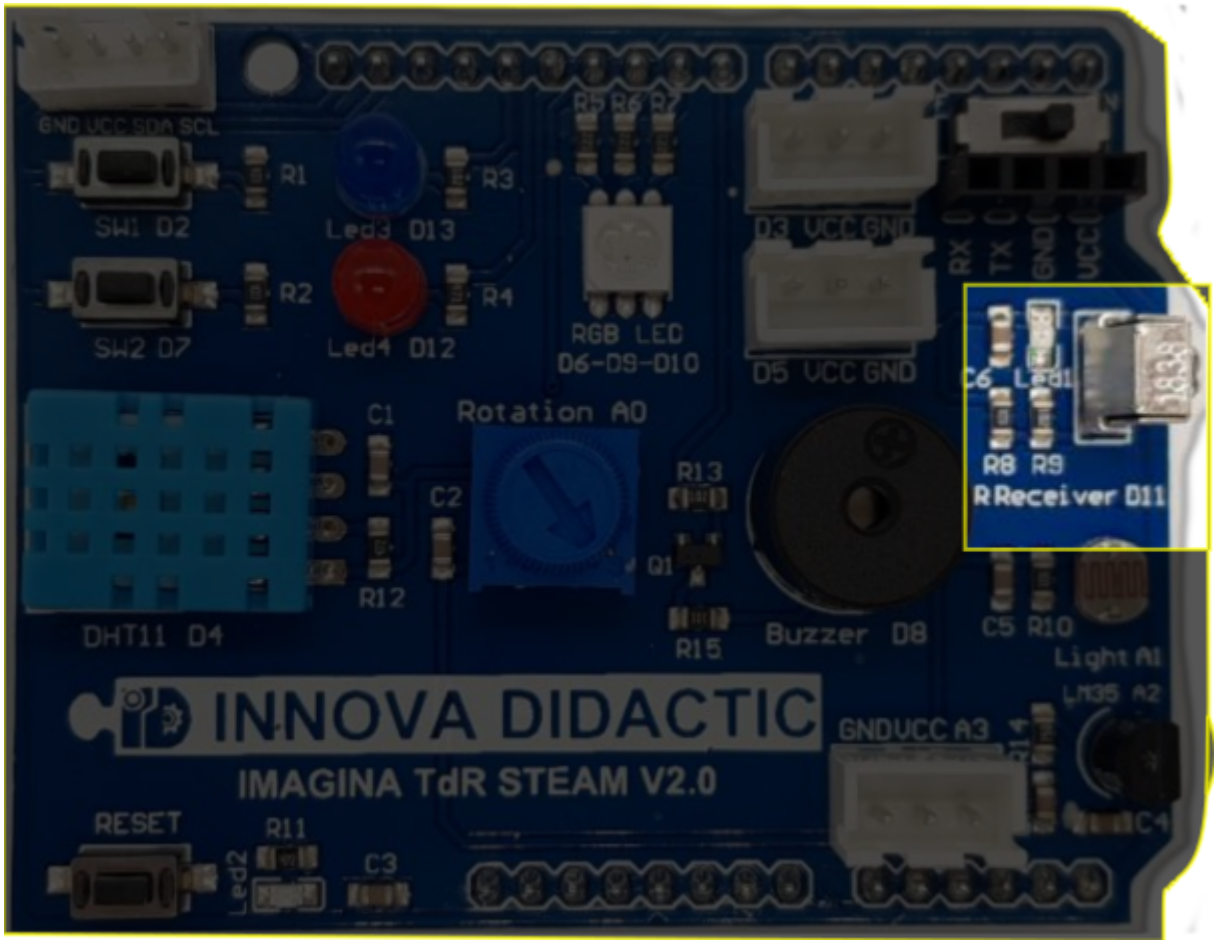
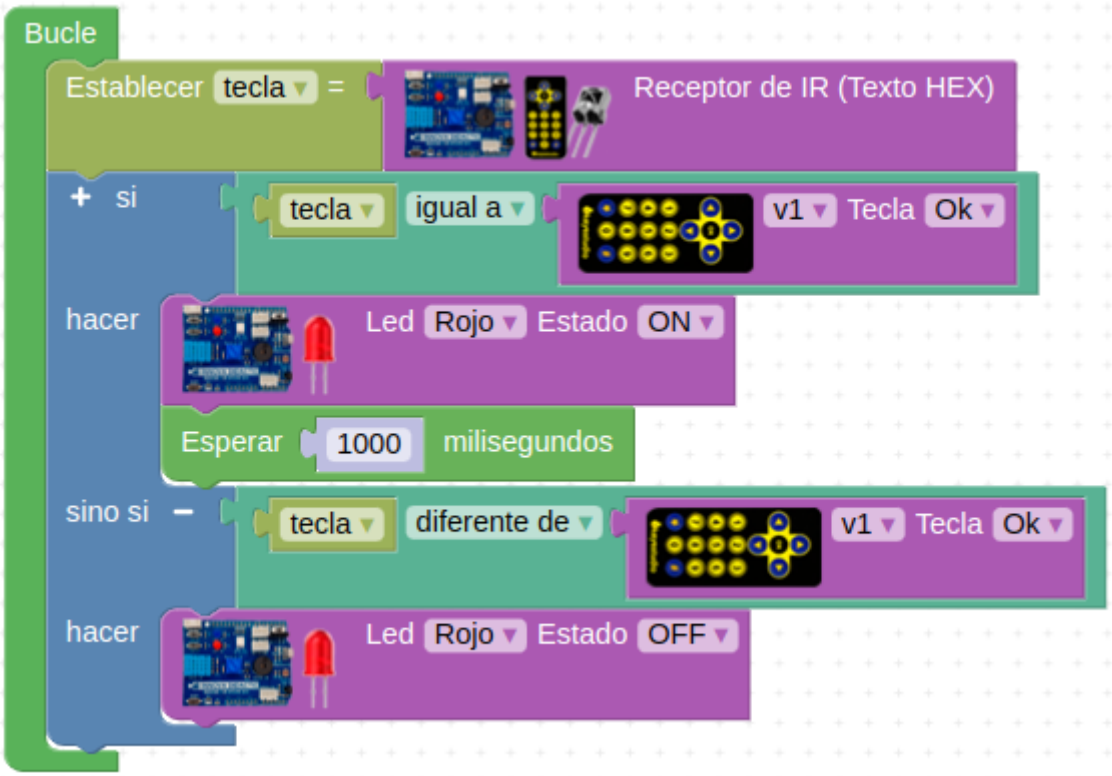


Imagen Federico Coca [Notas sobre ESP32 STEAMakers](#) CC-BY-SA

Programando la actividad

Vamos a realizar un programa en el que si pulsamos la tecla Ok del control remoto se nos encienda el diodo LED rojo. La solución la tenemos disponible en ESP32-SM-Actividad-11 que es el programa que vemos en la imagen siguiente:



Solución Actividad 11 Imagen Federico Coca [Notas sobre ESP32 STEAMakers](#) CC-BY-SA

Sabemos que el diodo IR está recibiendo un código porque junto al mismo hay un diodo LED rojo que parpadea.

Retos de ampliación

- **A11.R1.** Realizar un programa que encienda el LED RGB en los colores establecidos a continuación y según la tecla flecha pulsada.
  - Flecha arriba = Rojo
  - Flecha izquierda = Verde
  - Flecha derecha = Amarillo
  - Flecha abajo = azul
- **A11.R2.** Realizar un programa que nos muestre por consola el código hexadecimal correspondiente a cada una de las teclas pulsadas junto a un texto descriptivo indicador de la tecla pulsada en cada caso, es decir, que mantenga una estructura del tipo: El CODIGO se corresponde con la tecla TECLA pulsada.