

Reto A04. El potenciómetro

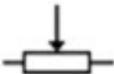
Estos contenidos han sido elaborados por Fernando Hernández García, Ingeniero Técnico Industrial Especialidad Electrónica, formador del profesorado y profesor del Institut Torre del Palau (Terrassa - Barcelona). [Enlace de los contenidos](#). **Licencia CC-BY-NC-ND**.

Permiso

Un potenciómetro es una resistencia cuyo valor es variable ya que son un tipo de resistencias especiales que tienen la capacidad de variar su valor cambiando de forma mecánica su posición. Con ellos indirectamente, se puede controlar la intensidad de corriente que fluye por un circuito si se conecta en paralelo, o  controlar el voltaje al conectarlo en serie. Son adecuados para su uso como elemento de control en los aparatos electrónicos como el control de volumen, brillo, etc.



Símbolo



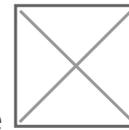
Componente

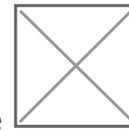
La placa Imagina TDR STEAM  tiene un potenciómetro denominado *Rotation* que van

asociado al pin A0. Las entradas *Anúmero* son entradas  analógicas, así que empezamos con el uso de este tipo de entradas. Este potenciómetro permite realizar un giro de unos 270° entre topes (3/4 de vuelta).



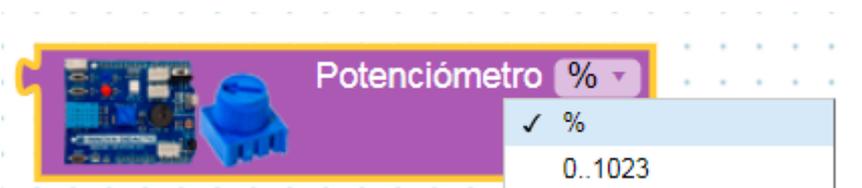
La diferencia entre un sensor **analógico** y **digital** es que mientras este último, el digital, sólo permite dos tipos de entradas, 0-1, *alto-bajo*, *high-low*, *on-off*, un sensor analógico puede tener infinidad de valores. En Arduino, las entradas analógicas pueden tener 210 valores (10 bits de resolución), es decir, valores comprendidos entre 0 y 1023.



En el menú de sensores de ArduinoBlocks, disponemos de un bloque  específico para realizar programas utilizando el potenciómetro de nuestra placa.

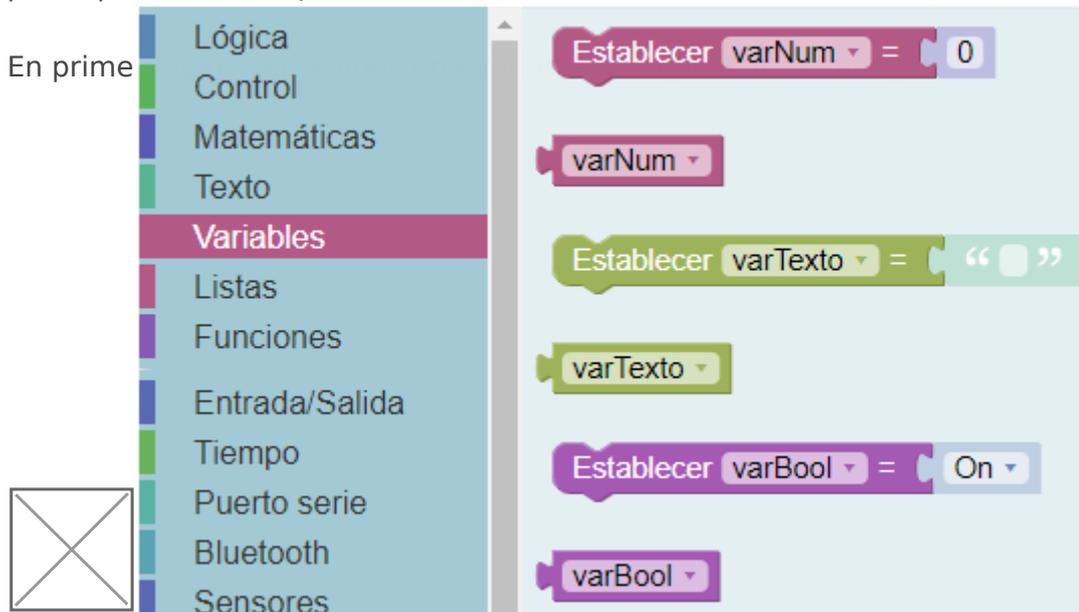


En el desplegable del bloque del sensor, podemos elegir su lectura en porcentaje (%) o en valor (de 0 a 1023).



7.5.1 Lectura de valores con el puerto serie

Para realizar una lectura de los valores del sensor es necesario utilizar la *Consola* (lector de datos por el puerto serie) que nos ofrece ArduinoBlocks, vamos a ver como se hace.



Para cambiar el nombre de la variable pulsaremos sobre el menú desplegable del bloque de la variable y elegiremos *Variable nueva...* nos aparecerá una ventana en la que escribiremos el nuevo nombre y daremos a *Aceptar*. Ahora fijaremos el valor de la variable al valor del potenciómetro, tal y como está en la imagen.



Es importante establecer la variable con el valor del potenciómetro dentro  de *Bucle*, ya que si sólo se hace en *Inicializar* el valor siempre será el mismo a lo largo de todo el programa. En otras ocasiones interesa establecer las variables en el inicio, pero no es este el caso.

Continuando con el programa, ahora nos faltan los bloques del *Puerto Serie*. El primero que debemos utilizar es el *Iniciar Baudios 9.600* que siempre lo colocaremos en el Inicio y después el bloque *Enviar*.



Matemáticas
Texto
Variables
Listas
Funciones
Entrada/Salida
Tiempo
Puerto serie
Bluetooth
Sensores
Actuadores

> Fijar timeout 1000

> Enviar “ ” ✓ Salto de línea

> Enviar byte 0

> ¿Datos recibidos?

Observa cómo queda el programa resultante:

Inicializar

Establecer pot = 0

> Iniciar Baudios 9600

Bucle

Establecer pot = Potenciómetro %

> Enviar pot ✓ Salto de línea

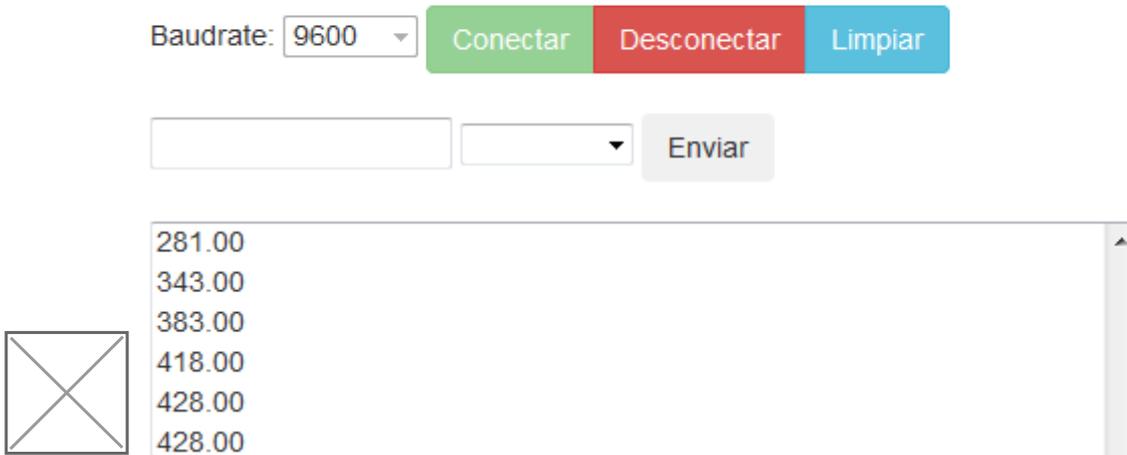
Esperar 500 milisegundos

Sube ahora el programa y después pulsa sobre el botón de la *Consola*.



Home Save Settings Subir Consola Refresh

Se abrirá la siguiente ventana y pulsaremos sobre el botón conectar. De esta manera podremos ver cada **ArduinoBlocks :: Consola serie** observa cómo van caml



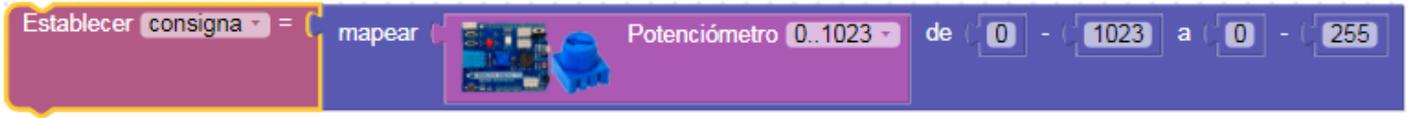
Actividad de ampliación: prueba ahora quitando el tic de *Salto de línea* a ver qué sucede.

7.5.2 Ajuste de valores de entrada y salida: mapear

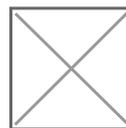
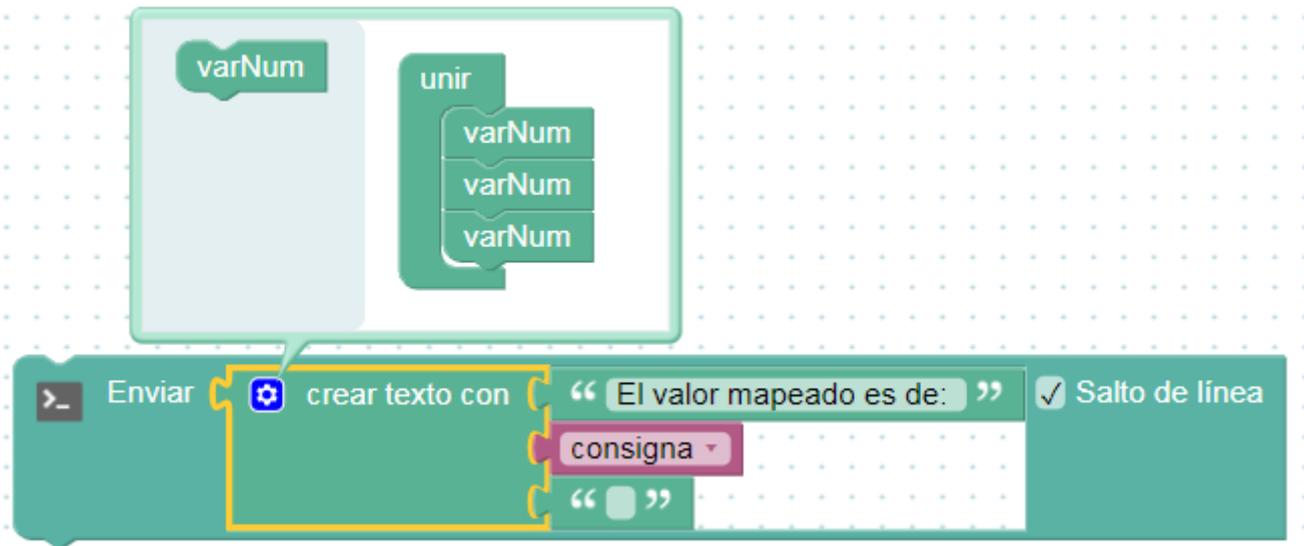
Existe un pequeño “problema” entre las entradas y las salidas en Arduino. Las entradas trabajan con 10 bits (210 valores = 0 a 1023) y las salidas trabajan a 8 bits (28 valores = 0 a 255). Debido a esto, debemos realizar un cambio de escala. A este cambio de escala se le llama “mapear”. En el menú *Matemáticas* existe un bloque llamado *mapear*. Este bloque permite modificar el rango de un valor o variable desde un rango origen a un rango destino. Esta función es especialmente útil para adaptar los valores leídos de sensores o para adaptar valores a aplicar en un actuador.



En esta actividad vamos a imaginar que con el potenciómetro queremos definir un rango de valores entre 0 a 255. Para ello definiremos una variable, llamada *consigna*, que será el valor *mapeado* del potenciómetro. En el potenciómetro cambiaremos su opción para obtener datos 0...1023.



Continuando el programa para poder realizar lecturas por el puerto serie utilizaremos un nuevo bloque de *crear texto con...* Fíjate como al pulsar sobre el símbolo del mecanismo podemos ampliar las líneas añadiendo *varNum* a la parte derecha.



El programa resultante quedará de la siguiente forma:



```
Inicializar
  Establecer consigna = 0
  Iniciar Baudios 9600

Bucle
  Establecer consigna = mapear Potenciometro 0..1023 de 0 - 1023 a 0 - 255
  Enviar crear texto con "El valor mapeado es de: " consigna " " Salto de línea
  Esperar 500 milisegundos
```

Por último, carga el programa, abre la *Consola* y comprueba las lecturas moviendo el potenciómetro.

ArduinoBlocks :: Consola serie

Baudrate: 9600 ▾

Conectar

Desconectar

Limpiar



Enviar

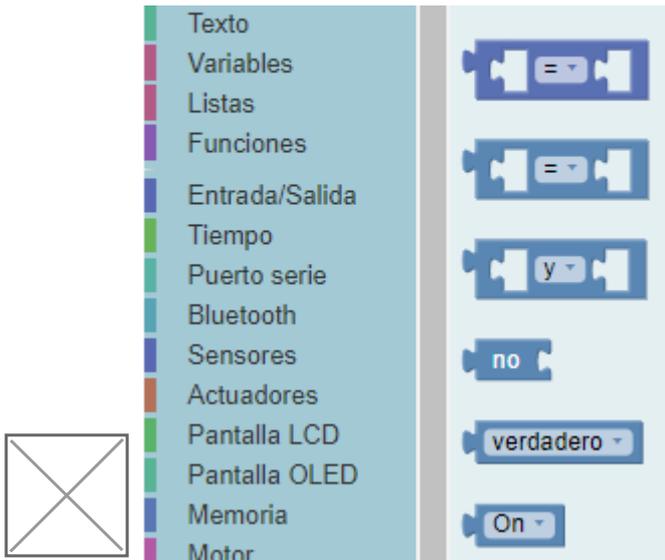
```
El valor mapeado es de: 255.00
El valor mapeado es de: 255.00
El valor mapeado es de: 249.00
El valor mapeado es de: 208.00
El valor mapeado es de: 174.00
El valor mapeado es de: 158.00
El valor mapeado es de: 140.00
El valor mapeado es de: 86.00
El valor mapeado es de: 49.00
El valor mapeado es de: 0.00
El valor mapeado es de: 0.00
El valor mapeado es de: 0.00
El valor mapeado es de: 207.00
El valor mapeado es de: 253.00
El valor mapeado es de: 255.00
El valor mapeado es de: 255.00
```

Actividad de ampliación: cambia ahora el rango de salida y el texto que envía por el puerto serie.

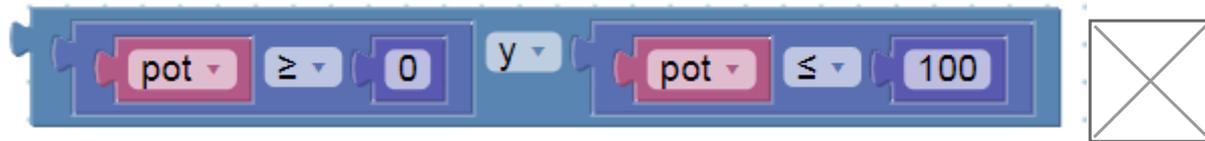
7.5.3 Control del led RGB con el potenciómetro

En la siguiente actividad vamos a controlar los colores del led RGB utilizando el potenciómetro. Vamos a hacer que cambie de color según varíe el valor del potenciómetro. Es decir, cuando el valor del potenciómetro se encuentre entre 0 y 100 que el color del led sea rojo, cuando se encuentre entre 101 y 200 que sea verde y cuando esté entre 201 y 255 que sea azul.

Del menú *Lógica* vamos a necesitar dos nuevos bloques; el bloque de *Evaluar condición* y el bloque de *Conjunción/Disyunción*. Con ellos crearemos estas condiciones:

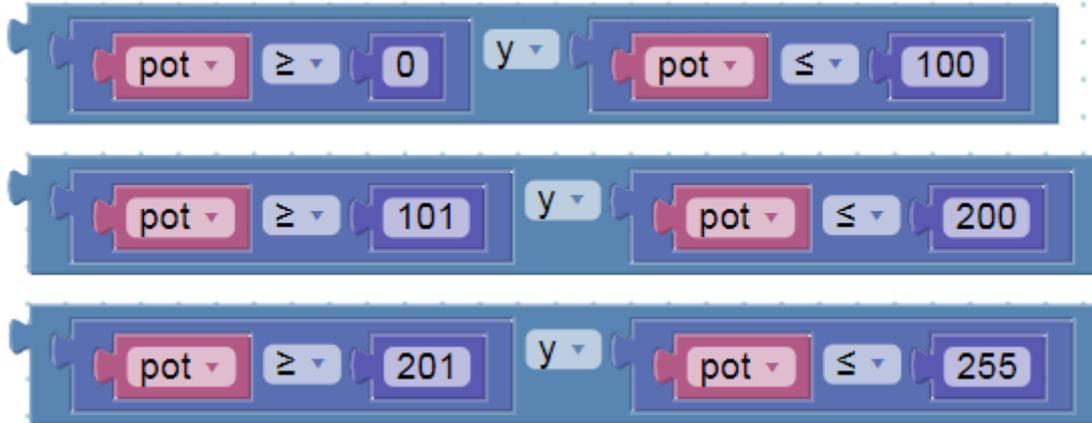


A Scratch block palette with a list of categories on the left and corresponding block icons on the right. The categories are: Texto, Variables, Listas, Funciones, Entrada/Salida, Tiempo, Puerto serie, Bluetooth, Sensores, Actuadores, Pantalla LCD, Pantalla OLED, Memoria, and Motor. The icons include mathematical operators like =, >, <, and logical operators like no, verdadero, and On.



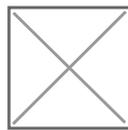
A Scratch block containing a logical AND condition: `pot >= 0 y pot <= 100`. To the right of the block is a square placeholder box with an 'X' inside.

Deberem



Three Scratch blocks stacked vertically, each containing a logical AND condition with different numerical values:

- Block 1: `pot >= 0 y pot <= 100`
- Block 2: `pot >= 101 y pot <= 200`
- Block 3: `pot >= 201 y pot <= 255`



El programa quedaría como muestra la imagen:



```

Inicializar
  Establecer pot = 0

Bucle
  Establecer pot = mapear Potenciómetro 0..1023 de 0 - 1023 a 0 - 255
  si pot ≥ 0 y pot ≤ 100
  hacer Led RGB Color [Red]
  si pot ≥ 101 y pot ≤ 200
  hacer Led RGB Color [Green]
  si pot ≥ 201 y pot ≤ 255
  hacer Led RGB Color [Blue]
  
```

También se puede hacer el mismo programa de la siguiente forma:

```
graph TD
    subgraph Inicializar
        A[Establecer pot = 0]
    end
    subgraph Bucle
        B[Establecer pot = mapear Potenciómetro 0..1023 de 0 - 1023 a 0 - 255]
        C[si pot >= 0 y pot <= 100]
        D[hacer Led RGB R 255 G 0 B 0]
        E[si pot >= 101 y pot <= 200]
        F[hacer Led RGB R 0 G 255 B 0]
        G[si pot >= 201 y pot <= 255]
        H[hacer Led RGB R 0 G 0 B 255]
    end
    A --> B
    B --> C
    C --> D
    D --> E
    E --> F
    F --> G
    G --> H
    H --> B
```



Actividad de ampliación: completa el programa con más condiciones.

Revision #2

Created 3 February 2023 11:38:25 by Javier Quintana

Updated 3 February 2023 12:00:39 by Javier Quintana