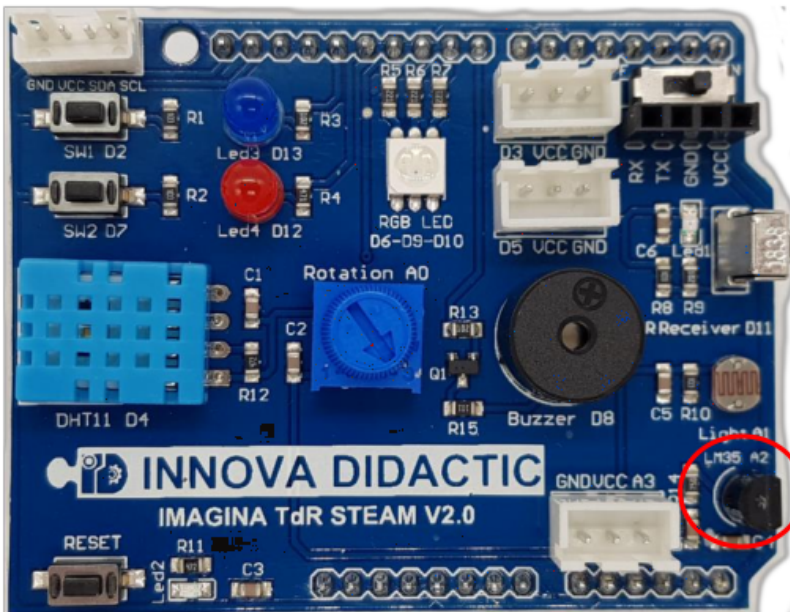


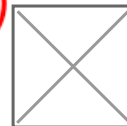
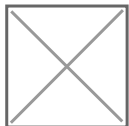
7.7 Reto A07. Sensor de temperatura LM35D

En el siguiente reto vamos a medir la temperatura de una habitación utilizando el sensor de temperatura LM35D. El LM35D tiene un rango de temperatura de 0º a 100º °C y una sensibilidad de 10mV/ºC.

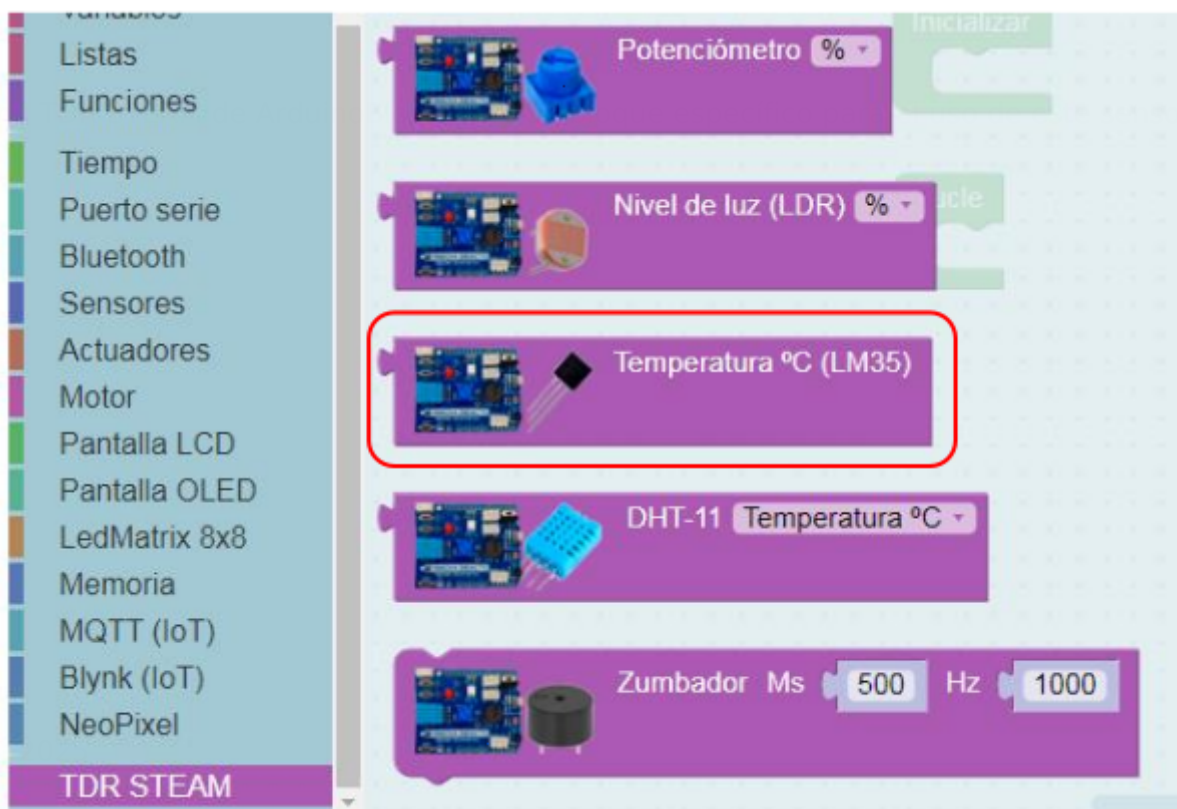
La placa A2.



está conectado en el Pin analógico



En el menú





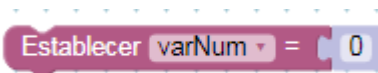
7.7.1 Lectura del valor de la temperatura

Para ello vamos a repasar el concepto de variables. Lo vimos inicialmente con la práctica del LED en la que incrementábamos y disminuíamos su brillo utilizando una variable *i*. En esta ocasión vamos a profundizar un poco más.

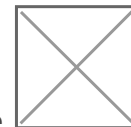
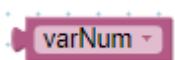
Las variables son elementos muy comunes en programación. Básicamente, crear una variable es darle un nombre a un dato o a una lectura. Por ejemplo, las mediciones de valores de temperatura las podemos guardar en una variable que se llame “Temperatura” o las del sensor de ultrasonidos en una llamada “Distancia”, etc. No es obligatorio su uso, pero nos permiten trabajar más cómodamente, además, como podemos personalizar su nombre, ayudan a clarificar el código y utilizar un lenguaje más natural.

Al trabajar con variables vamos a tener dos tipos de bloques principales:

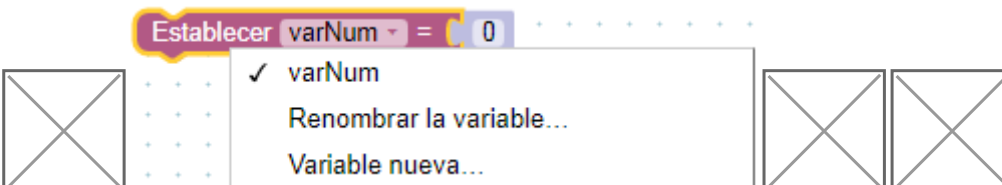
- El bloque en el que le damos valor a la variable:



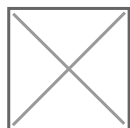
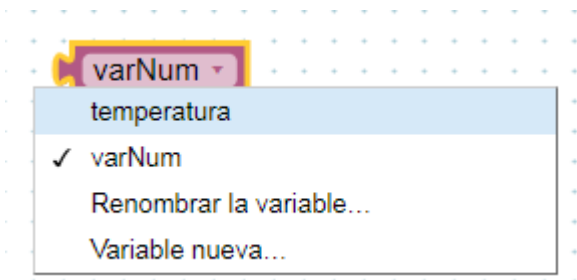
- Y el bloque de la propia variable creada, para poder insertarla y combinarla con otros bloques:



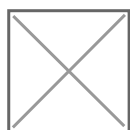
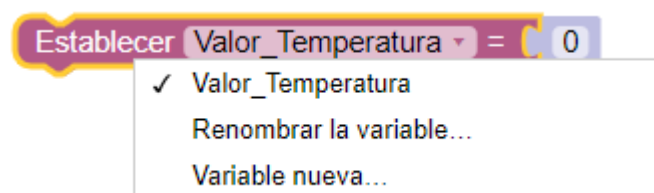
También podemos personalizar el nombre de la variable, de la siguiente forma:



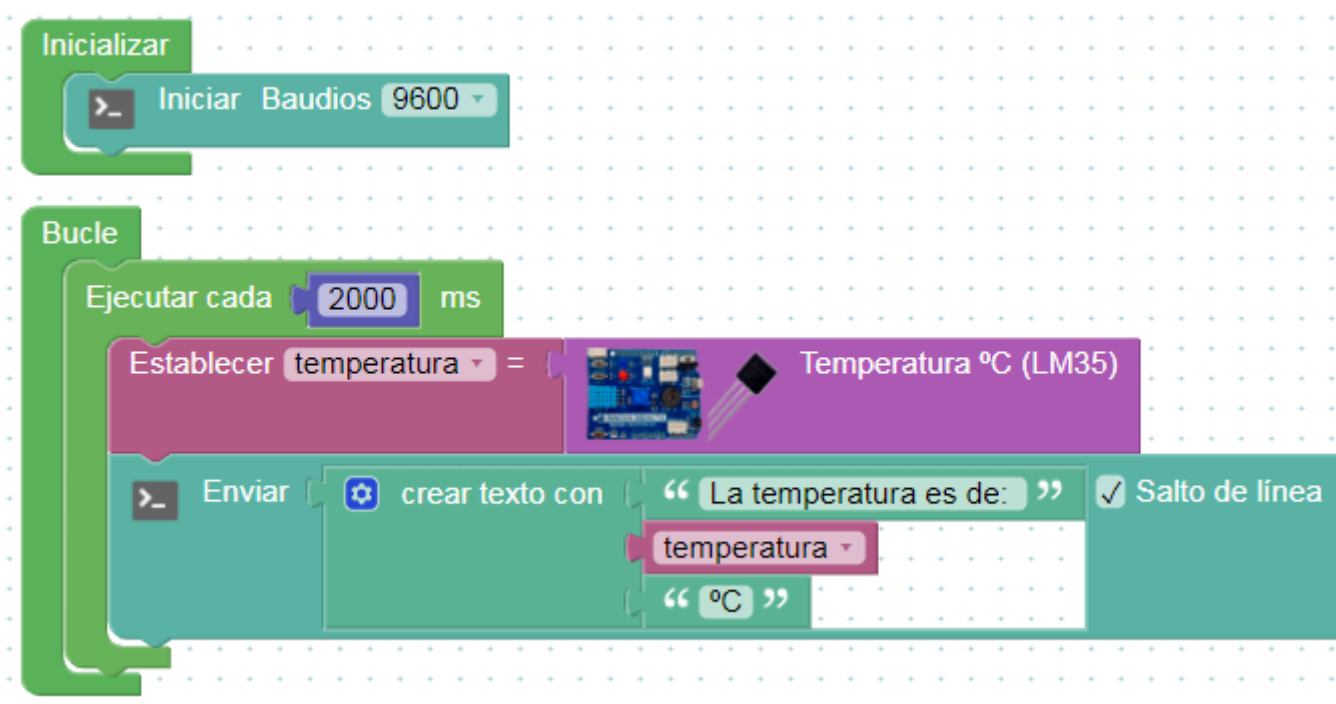
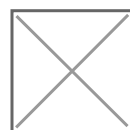
Una vez creada la nueva variable, podemos seleccionarla pulsando sobre el desplegable:



Hay que tener en cuenta que las variables solo pueden estar formadas por una palabra. Si quieres incluir varias palabras, puedes usar el truco de separarlas con una barra baja “_”, como en el ejemplo, “Valor_Temperatura”.



Ahora vamos a realizar el programa que mida la temperatura y la muestre por la consola.





Envía el código a **ArduinoBlocks :: Consola serie** terminal.

Lee la información que aparece en el

Baudrate: 9600 ▼

Conectar

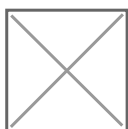
Desconectar

Limpiar

 ▼

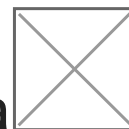
Enviar

La temperatura es de 23.44 C
 La temperatura es de 23.44 C
 La temperatura es de 23.44 C
 La temperatura es de 23.44 C
 La temperatura es de 24.90 C
 La temperatura es de 24.90 C
 La temperatura es de 25.39 C
 La temperatura es de 25.88 C
 La temperatura es de 25.88 C
 La temperatura es de 26.37 C
 La temperatura es de 25.88 C



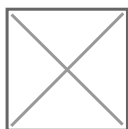
Actividad de ampliación: haz un programa que muestre datos más rápido y que muestre otro texto.

7.7.2. Alarma por exceso de temperatura

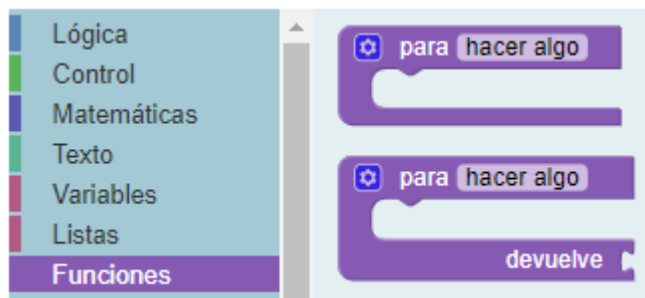


Ahora vamos a hacer una alarma sonora y acústica. El programa consistirá en hacer que el led rojo y el zumbador se accionen a la vez cuando el sensor de temperatura detecte una temperatura superior a 28°C.

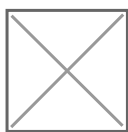
Para ello vamos a utilizar las *Funciones*. Las funciones permiten agrupar bloques de código. Esto es útil cuando un bloque de código se repite en varias partes del programa y así evitamos escribirlo varias veces o cuando queremos dividir el código de nuestro programa en bloques funcionales para realizar un programa más entendible.



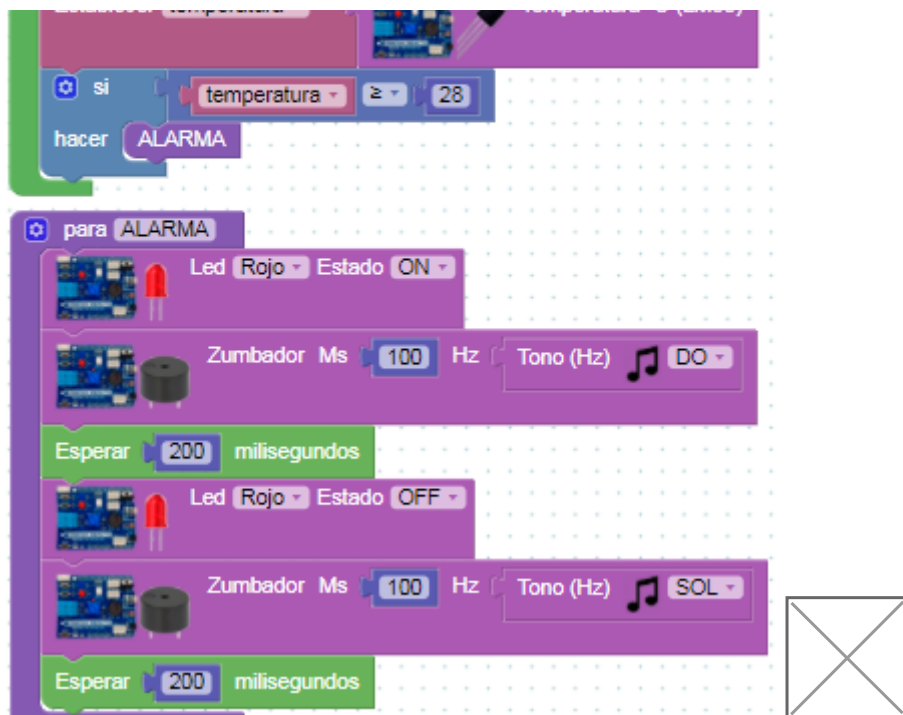
Las funciones nos permiten realizar tareas que se repiten a lo largo del programa. En el menú “Funciones” tenemos el bloque “*para () hacer algo*”. Lo utilizaremos para crear nuestra función como la siguiente imagen.



Debes escribir el nombre de ALARMA dentro de la función. Al hacer esto automáticamente en el menú "Función" aparecerá un nuevo bloque llamado ALARMA.



El programa final sería el siguiente:



Actividad de ampliación: haz un programa que varíe el programa anterior para que encienda de color verde el led RGB si la temperatura inferior a los 28°C y rojo si la temperatura es superior a los 35°C. En el rango intermedio se indicará de color azul.

Revision #2

Created 1 June 2022 11:22:14 by Equipo CATEDU

Updated 1 June 2022 11:27:47 by Equipo CATEDU