

PROGRAMANDO CON BITBLOQ

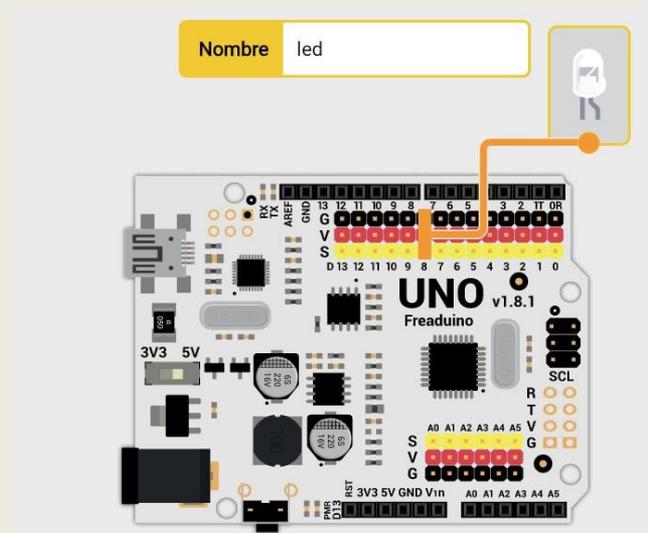
VIII

FUNCIONES

Recordamos que las variables eran un elemento que me sirve para guardar un dato, por ejemplo, la lectura de un sensor, un contador para un bucle, etc.

Las FUNCIONES son parecidas, pero en vez de almacenar datos, lo que guardo son acciones. Recordemos algo que ya hemos hecho: el parpadeo; encender, esperar, apagar, esperar... Pues esto mismo lo puedo hacer mediante una función que se llame parpadeo y que guarda dentro estas acciones.

Vamos como ejemplo a hacer una función que parpadee una vez.



Vamos al software:

Trabajaremos sobre el bloque variables globales, funciones y clases, de la parte superior, porque las funciones son siempre globales.

Vamos a la derecha, a las funciones y vemos que hay de dos tipos: declarar función que no devuelve nada y declarar función que devuelve algo. En esta ocasión vamos a usar la primera.





Bloques

Código



— Variables globales, funciones y clases

Declarar función `parpadeo_`

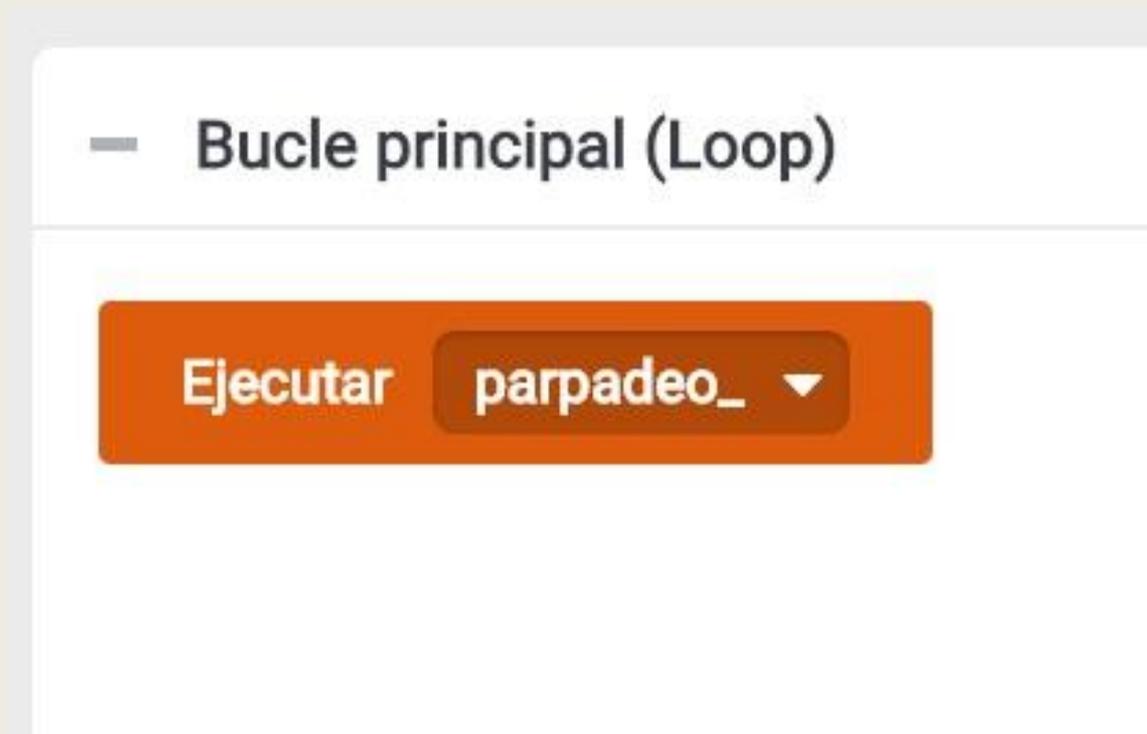
Encender ▼ el LED `led` ▼

Esperar `1000` ms

Apagar ▼ el LED `led` ▼

Esperar `1000` ms

Con esto estaría programada la función. Si ahora quiero que mi programa parpadee, lo que hago, como las otras veces, es en el bloque bucle llamar a la función.



De este modo, siempre que queramos hacer un parpadeo, solo tenemos que llamar a la función, y no es necesario copiar otra vez la programación.

FUNCIONES CON PARÁMETROS

Hemos hecho una función que no devuelve nada, enciende y apaga el led. Ahora queremos que encienda y apague el led varias veces, pero sin un número fijo, una función a la que yo le diga cuantas veces quiero que parpedee el led.

Vamos a hacer una función con un parámetro: función parpadeo del led y el parámetro es el número de veces que yo decida.

Ya tenemos la placa con el led conectado, y voy a funciones / Avanzados.



— Variables globales, funciones y clases

- Declarar función **parpadeo** con los siguientes argumentos : Variable **Decimal** **numero**

- Declarar función **parpadeo** con los siguientes argumentos : Variable **Decimal** **numero**

Declarar variable **contador** = **0**

- Contar con Variable **contador** desde **0** hasta Variable **numero** sumando **1**

Encender el LED **led**

Esperar **1000** ms

Apagar el LED **led**

Esperar **1000** ms

Qué hace esta función ???

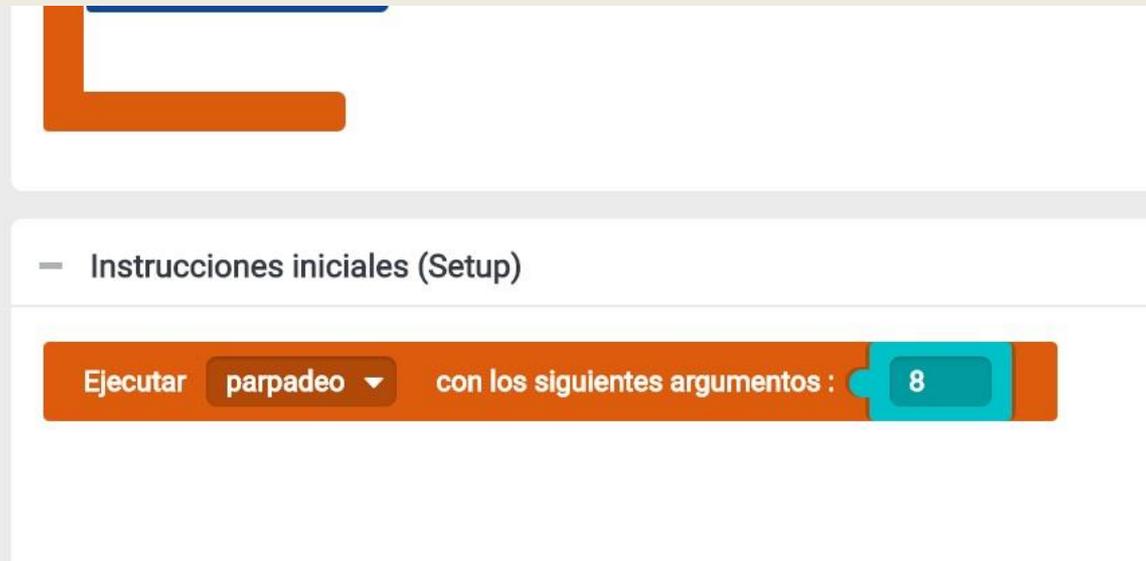
Lo primero que hace es que me pregunta como parámetro cuantas veces quiero que el led parpadee.

Y luego con el bucle contar con, que ya hemos utilizado, en lugar de contar desde 0 a un número fijo, lo hace desde 0 al parámetro número.

Así tenemos una función que dependiendo del parámetro, parpadea más o menos veces.

Cómo ejecutamos la función ??

Pues con la función parpadeo y el número de veces que queremos que parpadee, y en las instrucciones iniciales para que lo haga una sola vez.



Qué pasaría si en lugar de un parámetro le programamos dos parámetros , para decir el número de veces que parpadee y la pausa, es decir, las veces que le digamos y con el tiempo que le digamos.

LAS FUNCIONES CON RETORNO

Son funciones que además de hacer una tarea, nos devuelven cierta información. Por ejemplo una función que nos dice si un número es par o impar.

Conectamos placa y puerto serie, y pasamos a la programación. Trabajamos sobre el bloque variables globales, funciones y clases, y buscamos en avanzado una función que nos devuelva un parámetro.



Llamamos la función par, porque nos va a leer los números pares del 1 al 100, y como argumento recibe un número entero con cualquier valor que queremos comprobar si es par o impar.

— Variables globales, funciones y clases



Scratch code block for function declaration. The block is orange and has a circular handle on the left. It contains the text "Declarar función" followed by a dropdown menu with "par" selected, then "con los siguientes argumentos :", a "Variable" dropdown menu with "Entero" selected, and a text input field with "numero". Below the main block is a smaller orange block with the text "Devuelve".

Ahora ponemos las instrucciones que se tienen que ejecutar: comprobar si es par o impar, guardar el valor, y ese es el valor que devolveremos.

— Variables globales, funciones y clases



Scratch code block for variable declaration. The block is orange and has a circular handle on the left. It contains the text "Declarar función" followed by a dropdown menu with "par" selected, then "con los siguientes argumentos :", a "Variable" dropdown menu with "Entero" selected, and a text input field with "numero". Below the main block is a yellow block with the text "Declarar variable" followed by a dropdown menu with "resultado" selected, then "con tipo" followed by a dropdown menu with "valor lógico" selected, an equals sign, and a purple dropdown menu with "Verdadero" selected. Below the main block is a smaller orange block with the text "Devuelve".

Ahora lo que hacemos es decir: si es par el resultado es verdadero, de lo contrario el resultado es falso.

Para saber si el número es par, dividimos por dos y comprobamos el resto. Tenemos una función que nos da directamente el resto de una división, el operador tanto por ciento.

Si el resto de dividir el número por dos es cero,

– Variables globales, funciones y clases



Entonces la variable resultado es verdadero

Declarar función **par** con los siguientes argumentos : Variable Entero numero

Declarar variable **resultado** con tipo **valor lógico** = **Verdadero**

Si Variable **numero** % **2** = **0** ejecutar:

Variable **resultado** = **Verdadero**

De lo contrario

Si Variable **numero** % **2** = **0** ejecutar:

Variable **resultado** = **Verdadero**

de lo contrario, ejecutar:

Variable **resultado** = **Falso**

Y devolvemos la variable resultado



Y mostramos en las instrucciones iniciales para que sea una sola vez.



Y contamos

https://www.youtube.com/watch?v=Kw_XJCqG6PM&index=15&list=PL_AaWt7YXUYiU99BB3175JN9TK4lc4Q-m