Pico Bricks

Pico Bricks es una placa con sensores y actuares básicos integrados para hacer proyectos maker

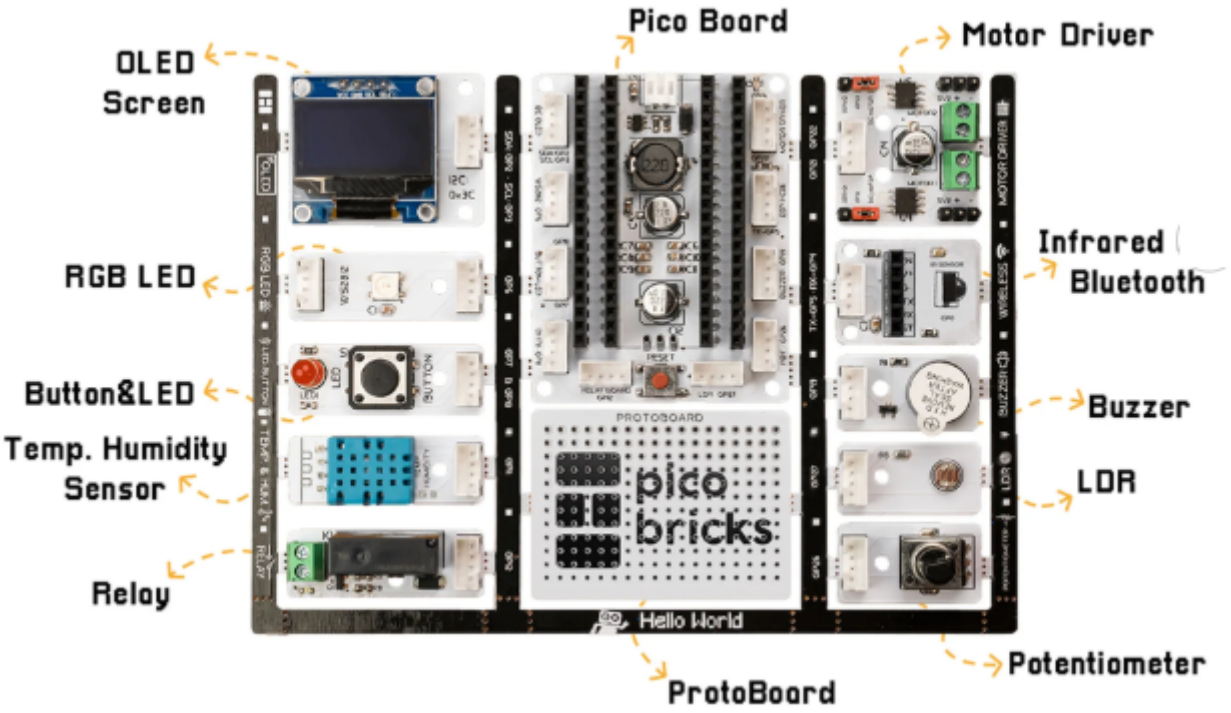
Introducción

¿Qué es Pico Bricks?

Pico Bricks pertenece a las placas electrónicas con sensores y actuadores básicos integrados, pero también preparado para poner externos y así poder hacer proyectos maker.

2024-12-26 09_23_28-Pico Bricks IDE book - PDF-XChange Viewer.png

Hardware



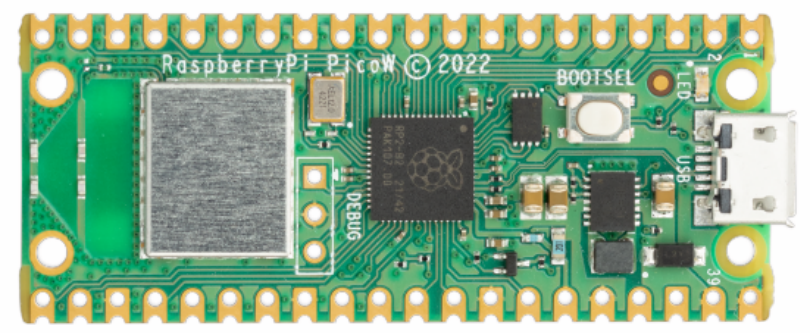
- **Placa microcontroladora**: Raspberry pi Pico W.
- Sensores o **entradas**
  - Botón
  - Sensor Temperatura y Humedad
  - Potenciómetro
  - Sensor IR con mando
  - LDR
- Actuadores o **salidas**
  - OLED Screen
  - Led rojo
  - RGB Led
  - Relé
  - Driver motor
  - Buzzer

¿Qué es la Raspberry Pi Pico W?

Es una placa controladora con las siguientes características: (*extraido de Bricogeek Licencia CC-BY*)

- **Chip RP2040:** Un microcontrolador con un procesador ARM Cortex-M0 de doble núcleo funcionando a 133 MHz. Este es el corazón de la placa y le da su potencia.
- **Memoria**: 264 KB de RAM SRAM y 2 MB de memoria Flash. Aunque no es tanto como un Raspberry Pi tradicional, es más que suficiente para la mayoría de los proyectos de microcontroladores.
- **Conectividad**: 2.4GHz Wifi 802.11n
- **Pines**: 26 pines GPIO (General Purpose Input Output), de los cuales muchos son multifunción, incluyendo UART, SPI, I2C, PWM, y pines analógicos (ADC).
- **Alimentación**: Puede funcionar con una fuente de alimentación de entre 1.8V y 5.5V, lo que la hace versátil para distintas fuentes de energía, como baterías o conexiones USB.
- **Interfaz de programación:** Puedes programarla usando C/C , MicroPython, o el Arduino IDE, lo que la hace- extremadamente accesible para la comunidad de makers.

P: ¿Qué diferencia tiene con respecto a otros microcontroladores?
R: Con respecto al de las placas Arduino UNO y similares (Atmegaxxx...) es más potente, permite la programación MicroPython, y sobre todo **tiene wifi incorporado**
P: ¿Igual que el ESP32?
R: No, el ESP32 tiene el Bluetooh y el Raspberry Pi Pico W **NO.**
P: ¿Entonces el ESP32 es mejor que el Raspberry Pi Pico W?
R: Pues si tus proyectos no necesitan Bluetooth, ni sensores de tacto que tiene el ESP32, el Raspberry Pi Pico W es muy buena opción pues sólo cuesta 8€, además Raspberry tiene PIO (Programable Input/Output) que permite visualizar en pantallas VGA +info

> **Importante**: Tienes que localizar el botón **BOOTSEL** que lo hablaremos en este curso
>
> 
>
>
> Fuente Datasheet Pico W https://datasheets.raspberrypi.com/picow/pico-w-datasheet.pdf

Software

Puede programarse con multitud de plataformas de código o de bloques gráficos

Fuente https://picobricks.com/products/raspberry-pi-pico-w-kit

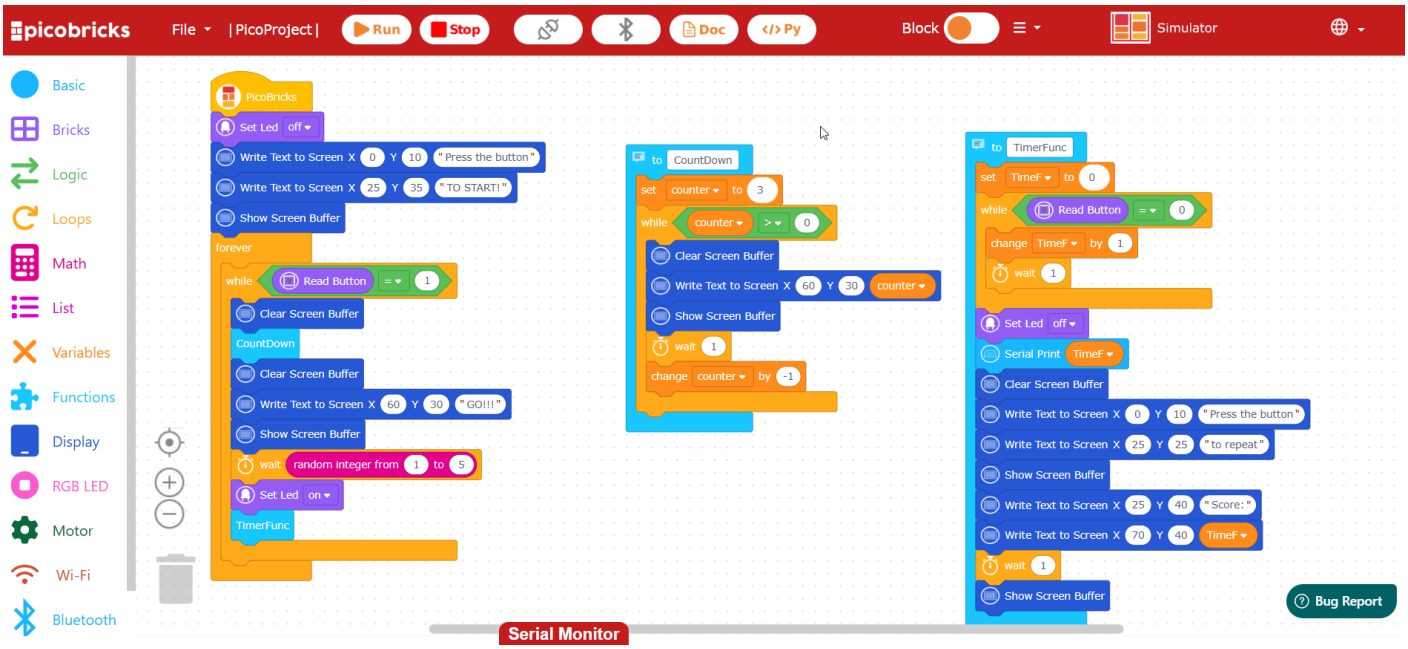
¿Qué software se va a utilizar en este curso?

Vamos a ver cuatro programas pertenecientes a dos formas de programar :

- **PROGRAMACIÓN POR BLOQUES** adecuado para primaria y primeros cursos de secundaria
  - **PICOBLOCKLY** perteneciente a los 4 programas de Pico Bricks IDE
  - **MICROBLOCKS** de software libre, multi tarjeta y popularmente extendido
- **PROGRAMACIÓN POR CÓDIGO** adecuado para secundaria
  - **THONNY IDE** con el lenguaje micropython
  - **ARDUINO IDE** con el lenguaje de código Arduino (C++)

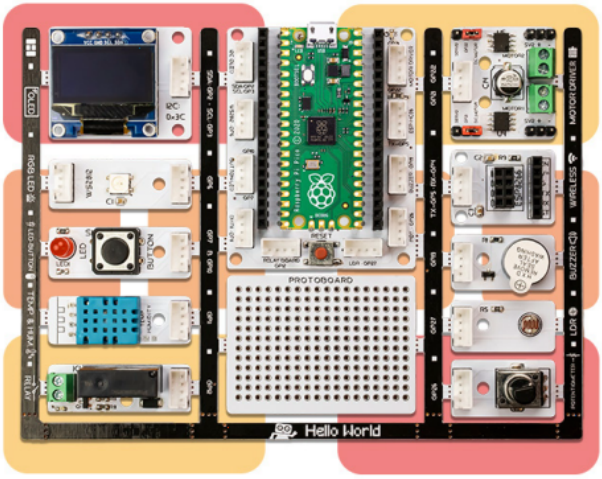
¿Cuál es el recomendable?

Para programación con bloques: **PICOBLOCKLY** de **Pico Bricks IDE** que es un software propio y tiene unos tutoriales muy cómodos (en inglés)



Luego para código el que recomendamos es **MicroPython con Thonny**

Opciones de compra



Fuente Pico Bricks IDE Book CC-BY-SA https://picobricks.com/pages/idebook ver créditos

Picobricks se venden en muchas tiendas y con muchas opciones y accesorios. Tienes que tener en cuenta que :

- **Lo más barato es ...**
  - Pack básico imprescindible:
    - comprar **solo la placa** unos 35€
    - comprar el microcontrolador **Raspbery Pico W** aparte pero ojo **que este soldado** los pines para poderlos insertar en el socket de la placa 8.60€
  - Puedes añadirle componentes externos. Recomendamos :
    - sensor ultrasónico HC-SR04 2€,
    - servo 2.20€,
    - kit de coche pues permite la conexión de dos motores 11€
- **Lo más cómodo es ....**
  - Comprar kits ya hechos ver https://picobricks.com/collections

Proyectos

Vamos a ver los siguientes proyectos y vamos a ver que se pueden desarrollar de cuatro formas diferentes:

- **PROGRAMACIÓN POR BLOQUES** adecuado para primaria y primeros cursos de secundaria
  - **PICOBLOCKLY** perteneciente a los 4 programas de Pico Bricks IDE
  - **MICROBLOCKS** de software libre, multi tarjeta y popularmente extendido
- **PROGRAMACIÓN POR CÓDIGO** adecuado para secundaria
  - **THONNY IDE** con el lenguaje micropython
  - **ARDUINO IDE** con el lenguaje de código Arduino (C++)

Los proyectos son

- PROYECTO **BLINK**
  - https://github.com/Robotistan/PicoBricks/tree/main/Software/Activities/Blink
- PROYECTO **ACTION-REACTION**
  - https://github.com/Robotistan/PicoBricks/tree/main/Software/Activities/Action-Reaction
- PROYECTO **Autonomous Lighting**
  - https://github.com/Robotistan/PicoBricks/tree/main/Software/Activities/Autonomous%20Lighting
- PROYECTO **Thermometer**
  - https://github.com/Robotistan/PicoBricks/tree/main/Software/Activities/Thermometer
- PROYECTO **Graphic Monitor**
  - https://github.com/Robotistan/PicoBricks/tree/main/Software/Activities/Graphic%20Monitor
- PROYECTO **Dominate the Rhythm**
  - https://github.com/Robotistan/PicoBricks/tree/main/Software/Activities/Dominate%20the%20Rhythm
- PROYECTO **Show Your Reaction**
  - https://github.com/Robotistan/PicoBricks/tree/main/Software/Activities/Show%20Your%20Reaction
- PROYECTO **My Timer**
  - https://github.com/Robotistan/PicoBricks/tree/main/Software/Activities/My%20Timer
- PROYECTO **Alarm Clock**

    - https://github.com/Robotistan/PicoBricks/tree/main/Software/Activities/Alarm%20Clock
- PROYECTO **Know Your Color**
    - https://github.com/Robotistan/PicoBricks/tree/main/Software/Activities/Know%20Your%20Color

Además de algún proyecto que creemos que completa la formación en este robot.

Hay más proyectos en https://github.com/Robotistan/PicoBricks/tree/main/Software/Activities pero implican usar actuadores y sensores externos pero interesantes y fáciles de conseguir y baratos, por ejemplo un servo, un sensor de distancia ultrasónico, etc..

> *Mentirijilla*: el proyecto Buzz Wire Game no hace falta ningún componente externo especial, sólo unos simples cables

El mismo repositorio facilita los proyectos ordenándolos de **más fácil a más dificil dificultad,** marcamos los que vamos a dar en este curso

1. - Blink
2. - Action-Reaction
3. - Autonomous Lighting
4. - Thermometer
5. - Graphic Monitor
6. - Dominate the Rhythm
7. - Show Your Reaction
8. - My Timer
9. - Alarm Clock
10. - Know Your Color
11. - Magic Lamp
12. - Smart Cooler
13. - Buzz Wire Game
14. - Dinosaur Game
15. - Night and Day
16. - Voice Controlled Robot Car
17. - Two Axis Robot Arm
18. - Smart House
19. - Piggy Bank

20. - RFID Smart Door
21. - Automatic Trash Bin
22. - Digital Ruler
23. -Air Piano
24. - Maze Solver Robot
25. - Smart Greenhouse

Libros

Project Book

Un libro completo (en inglés) que lo puedes conseguir aquí https://picobricks.com/pages/projectbook que pena que no tiene licencia CC

https://drive.google.com/file/d/1PDql_GYyxcz68JqmQAGOLs0YE6SXgPCm/preview

IDEBOOK

Este libro esta especializado en realizar proyectos con el software PICOBRICKS IDE lo puedes conseguir aquí https://picobricks.com/pages/idebook y sí que tiene licencia CC

https://drive.google.com/file/d/1plad6bjn87FcgHb3cpd1vI-B_A25rnfF/preview

Teacher Book

Con 20 actividades STEM, https://picobricks.com/pages/education

No es descargable, pero puedes solicitarlo gratis en contacto https://picobricks.com/pages/contact-information-picobricks

Pensamiento computacional

¿Dónde encaja Picobricks dentro de la oferta de equipos robóticos para la educación? Cómo puedes ver entra tanto en primaria como secundaria gracias a sus dos modos de programación:

- Programación en bloques (Primaria)
- Programación en código (Secundaria)

Unido al bajo precio y a las prestaciones que tiene con actuadores y sensores, es un buen producto con buena relación calidad/precio

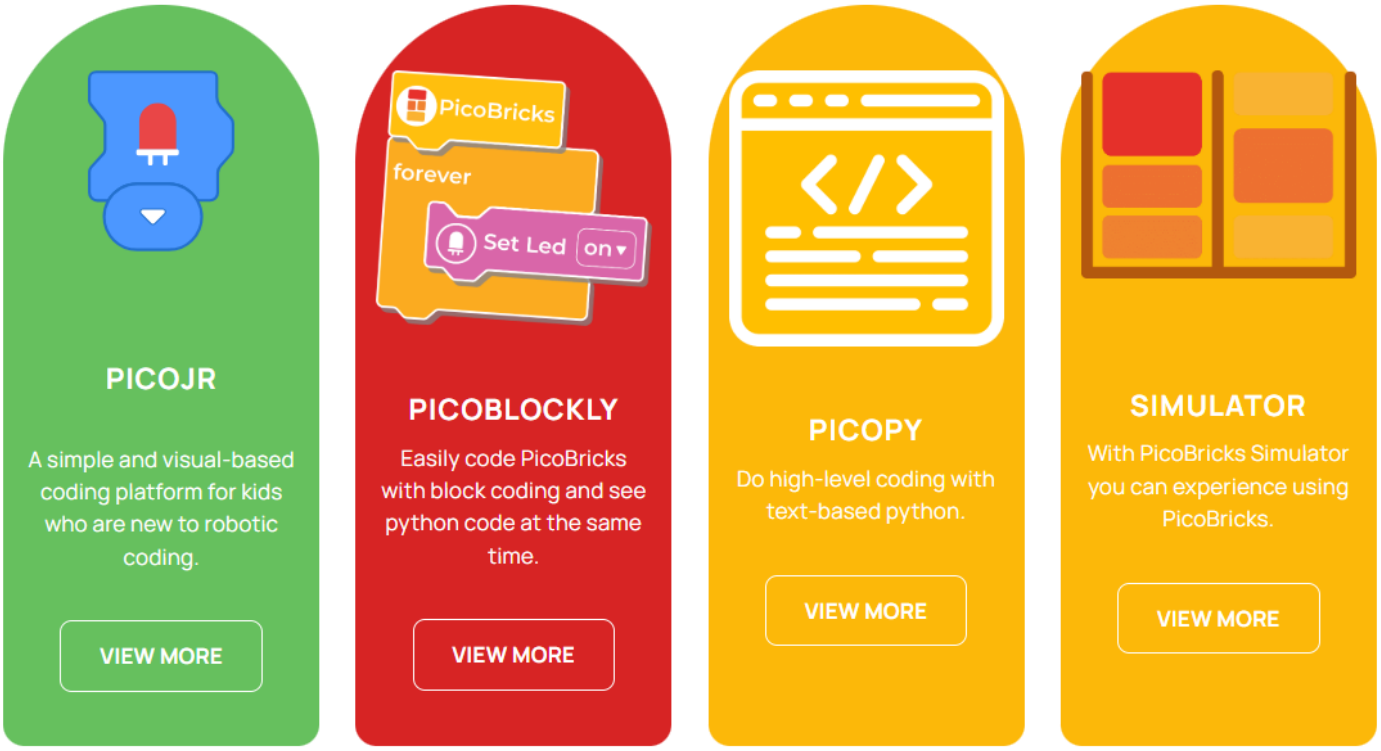
Guía orientativa

> Tenemos un **grupo Telegram Robótica Educativa en Aragón**, si estás interesado en unirte, envía un mensaje por Telegram (obligatorio) a CATEDU 623197587
>
> https://t.me/catedu_es y te añadimos en el grupo

PicoBlockly

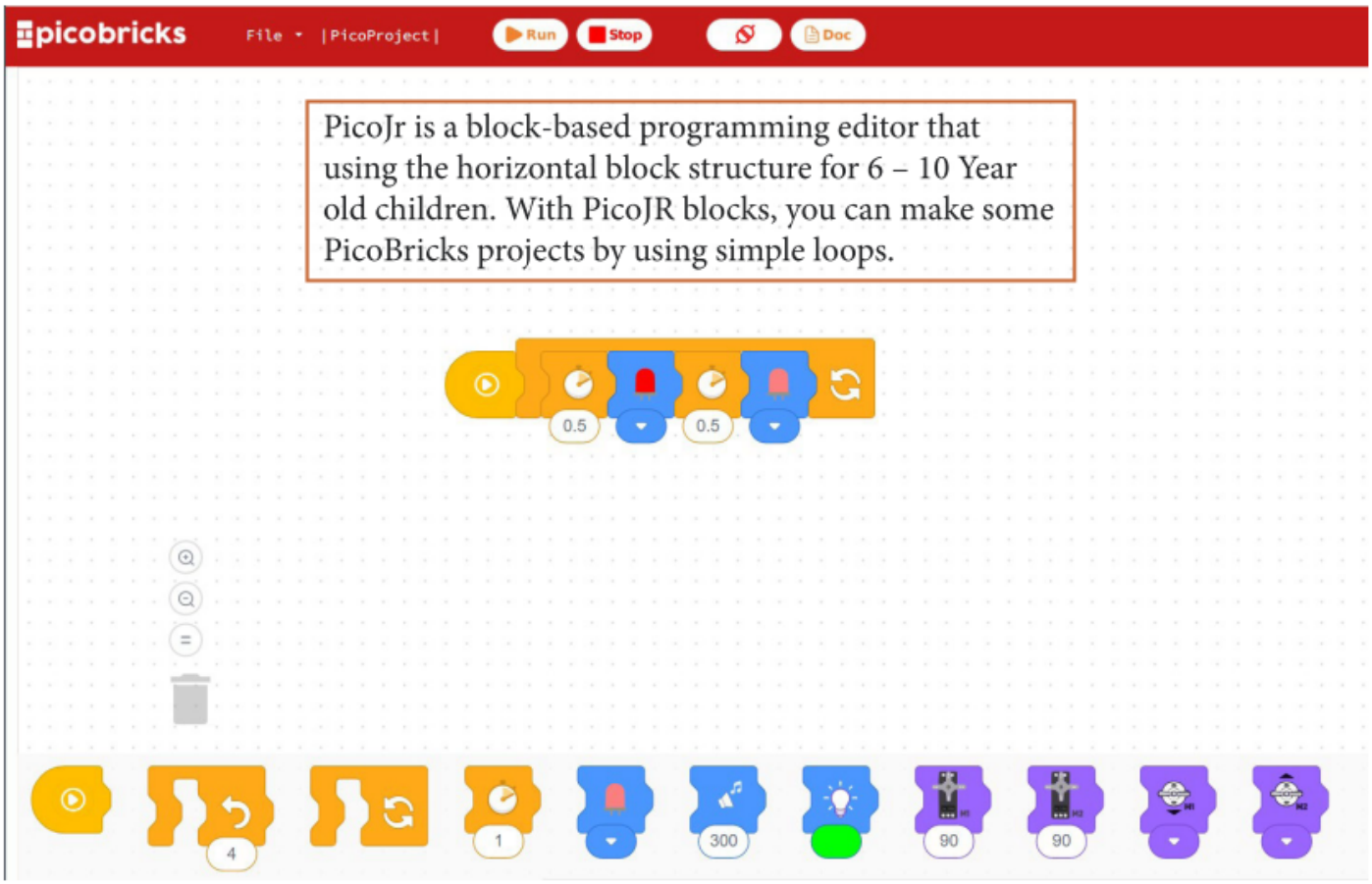
Cuatro programas a elegir

Si entramos en http://rbt.ist/ide podemos ver cuatro opciones



PICOJR

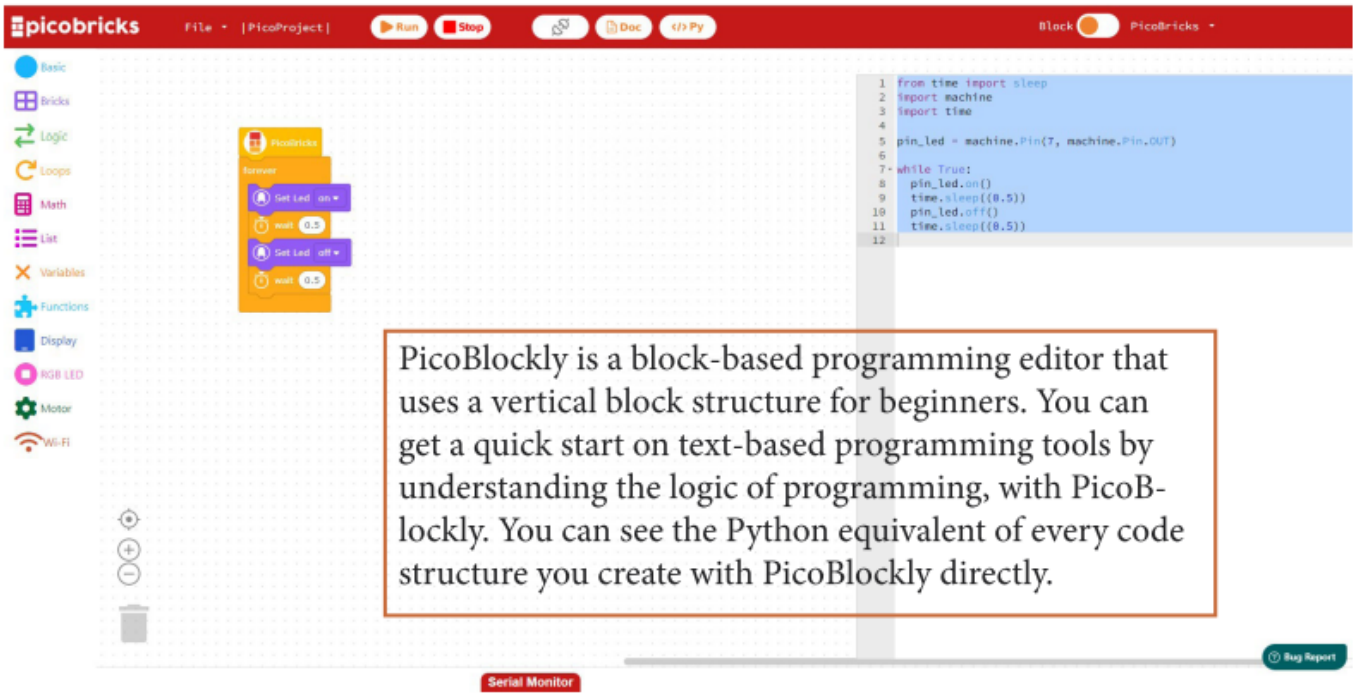
Pensado para programar Picobriks con bloques para etapas de 8 a 10 años con un **mínimo de instrucciones**



Fuente Pico Bricks IDE Book CC-BY-SA https://picobricks.com/pages/idebook ver créditos

PicoBlockly

Es la opción más recomendada para la mayoría de las etapas



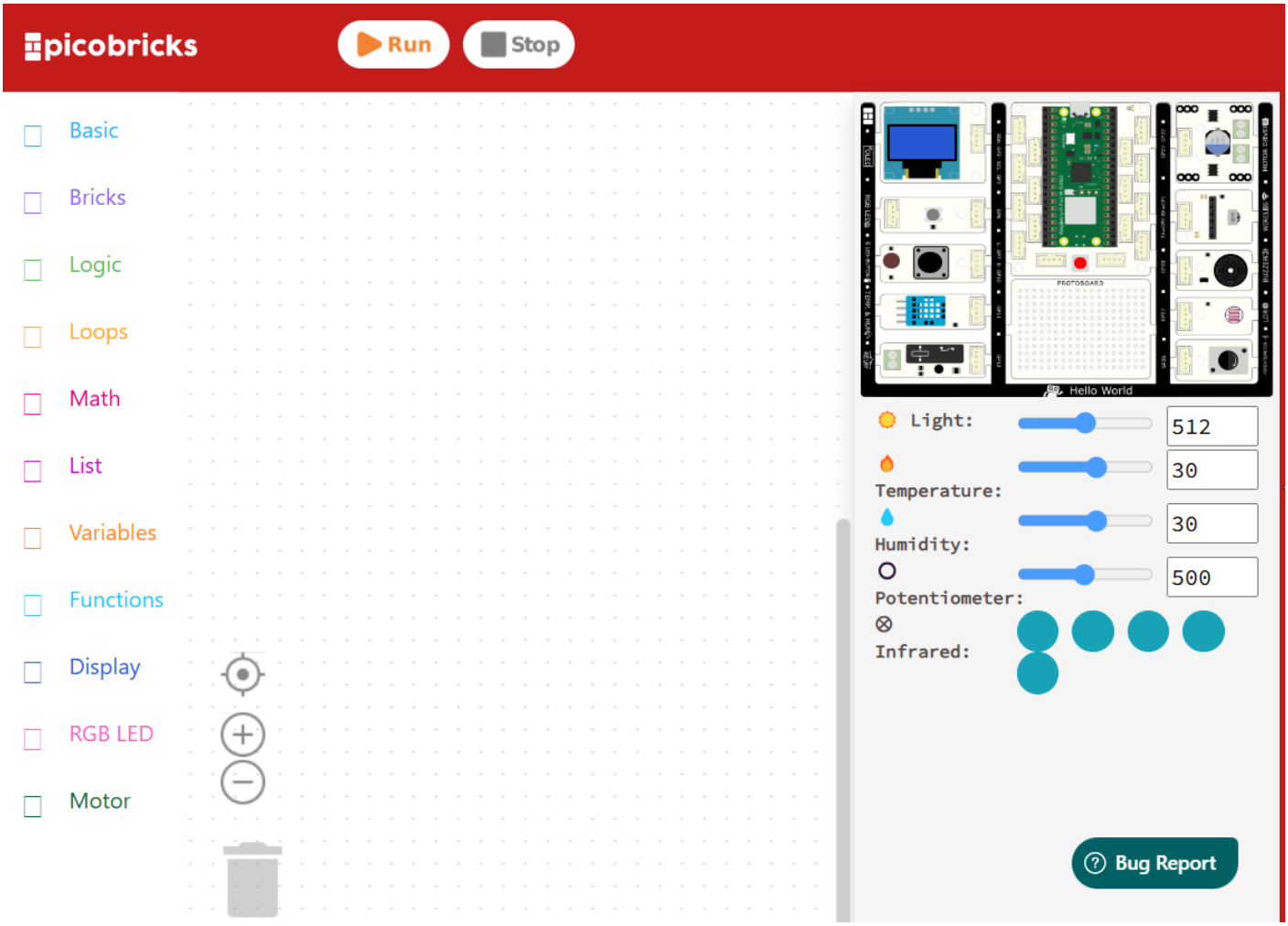
Fuente Pico Bricks IDE Book CC-BY-SA https://picobricks.com/pages/idebook ver créditos

PicoPy

Para poder editar en Python, lo trataremos en la sección de código

Pico simulator

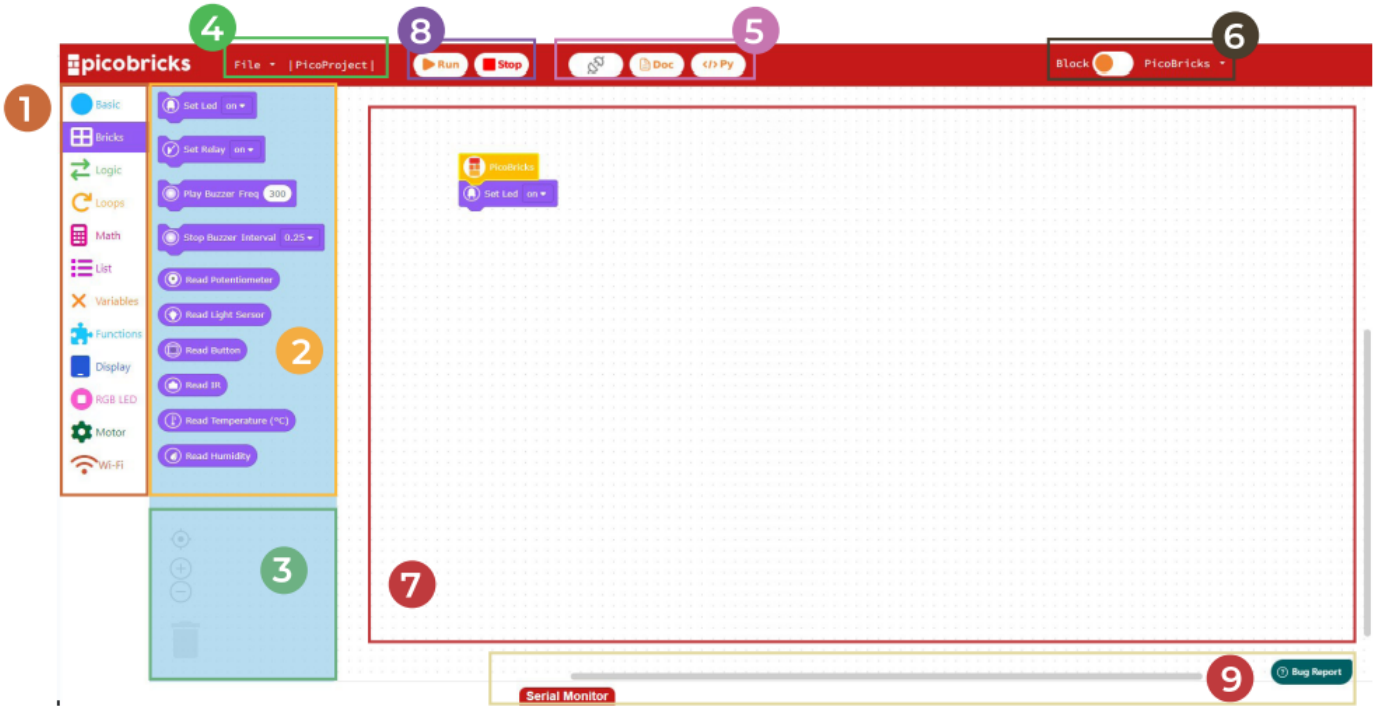
Es un simulador online que permite realizar proyectos sin tener físicamente la Pico bricks



> Ojo el simulador no permite gestión de ficheros, es decir, no puedes ni grabar proyectos ni abrirlos, cuando cierras el navegador se pierde todo

Interface

Cuando abrimos Picoblockly tenemos la siguiente ventana:



Fuente Pico Bricks IDE Book CC-BY-SA https://picobricks.com/pages/idebook ver créditos

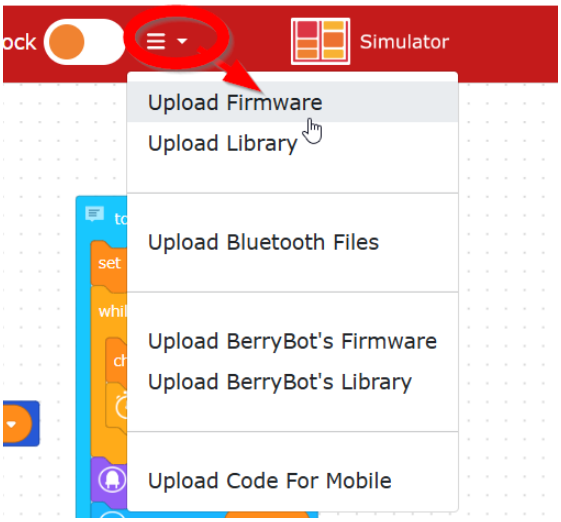
1. Donde encontramos las diferentes instrucciones ordenadas por categorías
2. La paleta de instrucciones preparados para elegir y arrastrar a 7
3. Herramienta de zoom, borrar
4. Menú de fichero para grabar los proyectos o abrirlos (todo localmente)
5. Panel operaciones
    1. Botón de conectar, por cable (recomendado) o bluetooh
    2. Botón de proyectos ya preinstalados
    3. Vista de código Python (también en 6 hay una pestaña para pasar a esta vista)
6. Menú de configuración para descargar los firmwares necesarios para la conexión
7. Área donde programamos
8. Start stop tu programa
9. Área del puerto serie donde podemos ver los valores que desemos

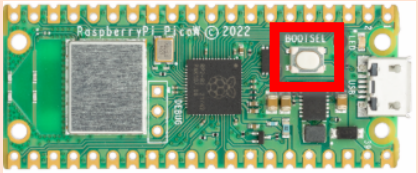
Conexión

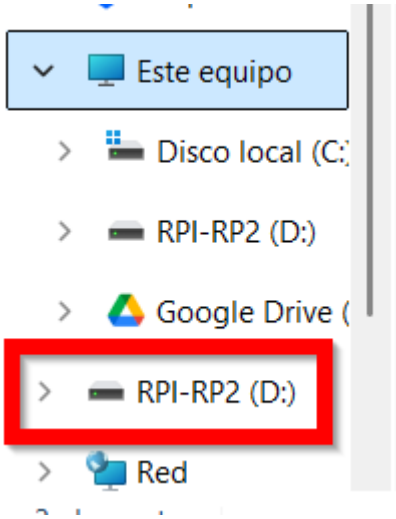
Lo primero que tenemos que hacer es poner el firmware para podernos conectar con Picobriks

> **P: ¿Qué es eso de "firmware"?**
> R: No es más que un software que se graba en los chips de la placa.
> **P ¿Y por qué se llama así, y no se llama software o programa y en paz?**
> R: Digamos que como se graba en los chips, es un medio camino entre software y hardware, para diferenciarlo del software habitual.

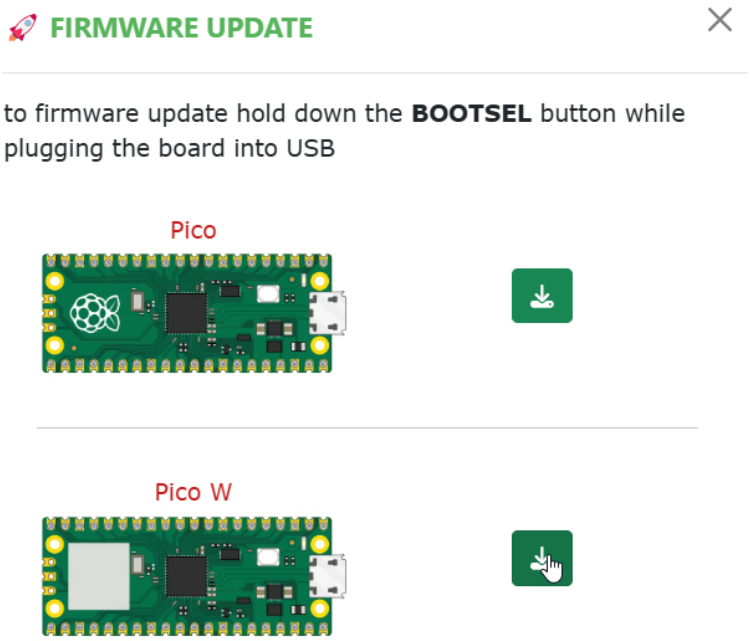
Entramos en el menú y descargamos el firmware

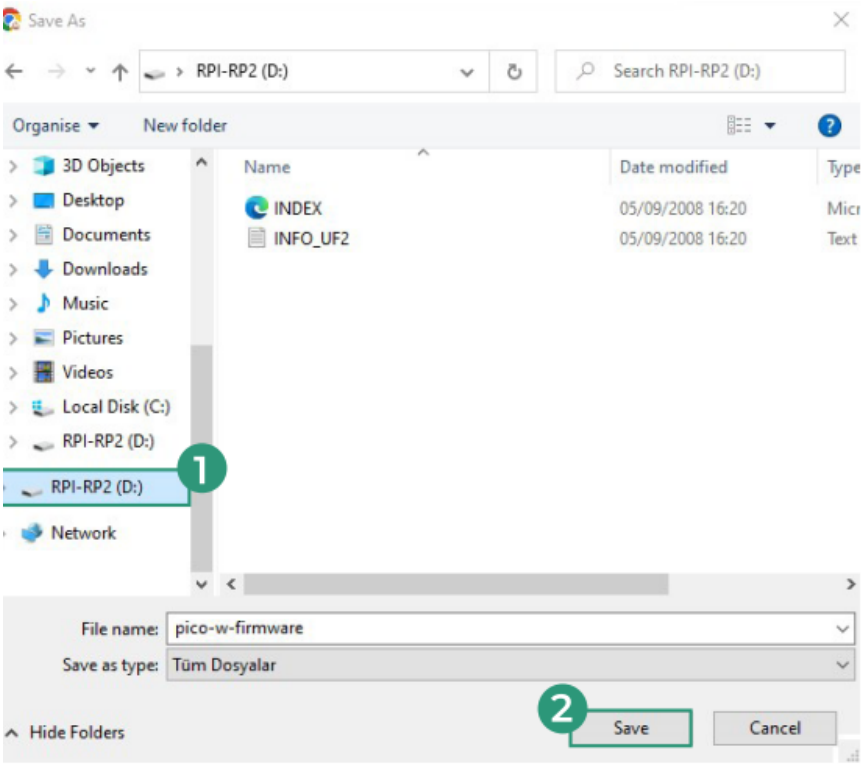


> **ATENCIÓN, poner PicoBricks en modo Bootloader**
> 1.-Desconectamos PicoBricks de nuestro ordenador
> 2.- Apretamos el botón BOOTSEL **mientras** lo volvemos a conectar al puerto USB
> 
> 3.- Automáticamente aparecerá una nueva unidad de disco en nuestro ordenador (ya puedes soltar BOOTSEL)



Descargamos el correspondiente al PicoW **Y LO GRABAMOS EN LA UNIDAD NUEVA** en mi caso RPI-RP2 (D:)



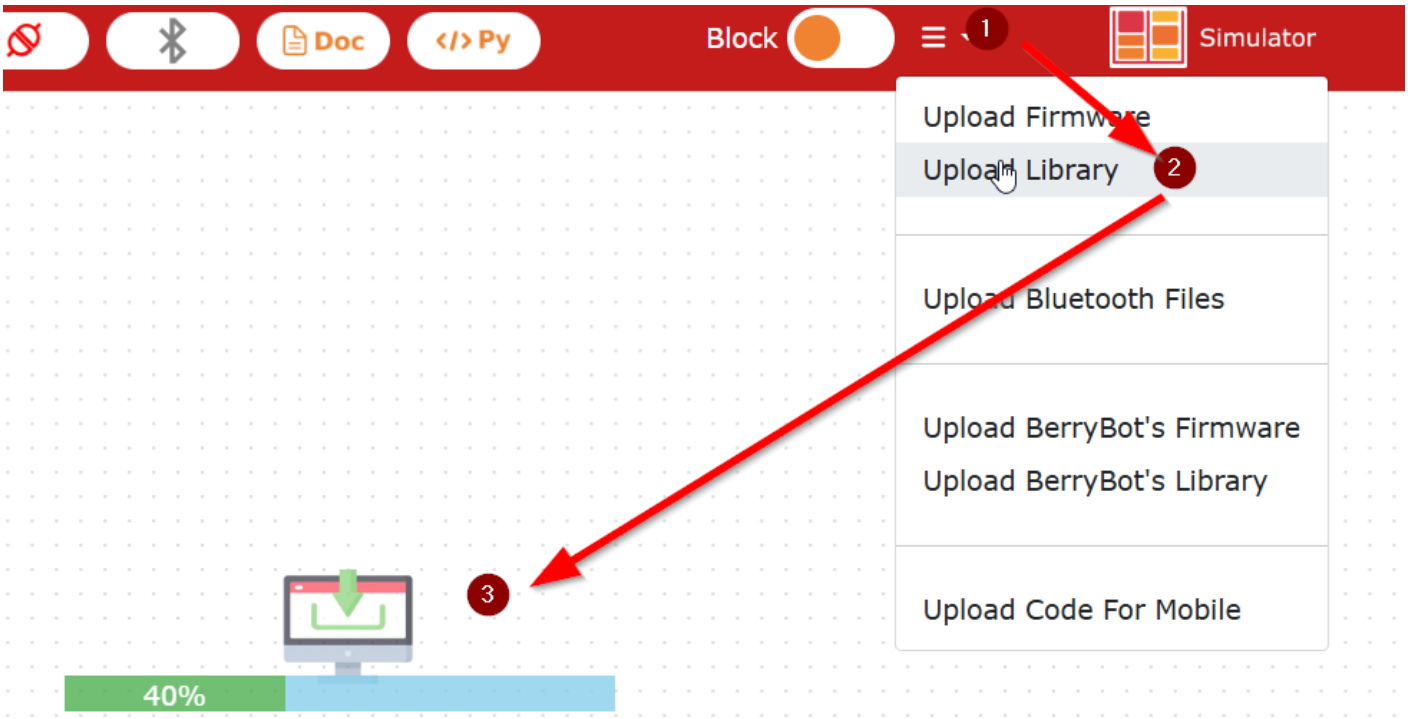


Fuente Pico Bricks IDE Book CC-BY-SA https://picobricks.com/pages/idebook ver créditos

Una vez grabado el firmware, esperamos a que nos salga un mensaje: *Please conect to the board*

Entonces dar a conectar y seleccionar la placa

Una vez conectado, descargamos las librerías en el PicoBricks para poder usar todas las funciones



> YA ESTA, esto lo tienes que hacer **SOLO UNA VEZ** mientras uses PicoBlockly, si te pasas a otro programa y te cargas su firmware, tendrás que volverlo a poner.

Dos formas de ejecutar los programas

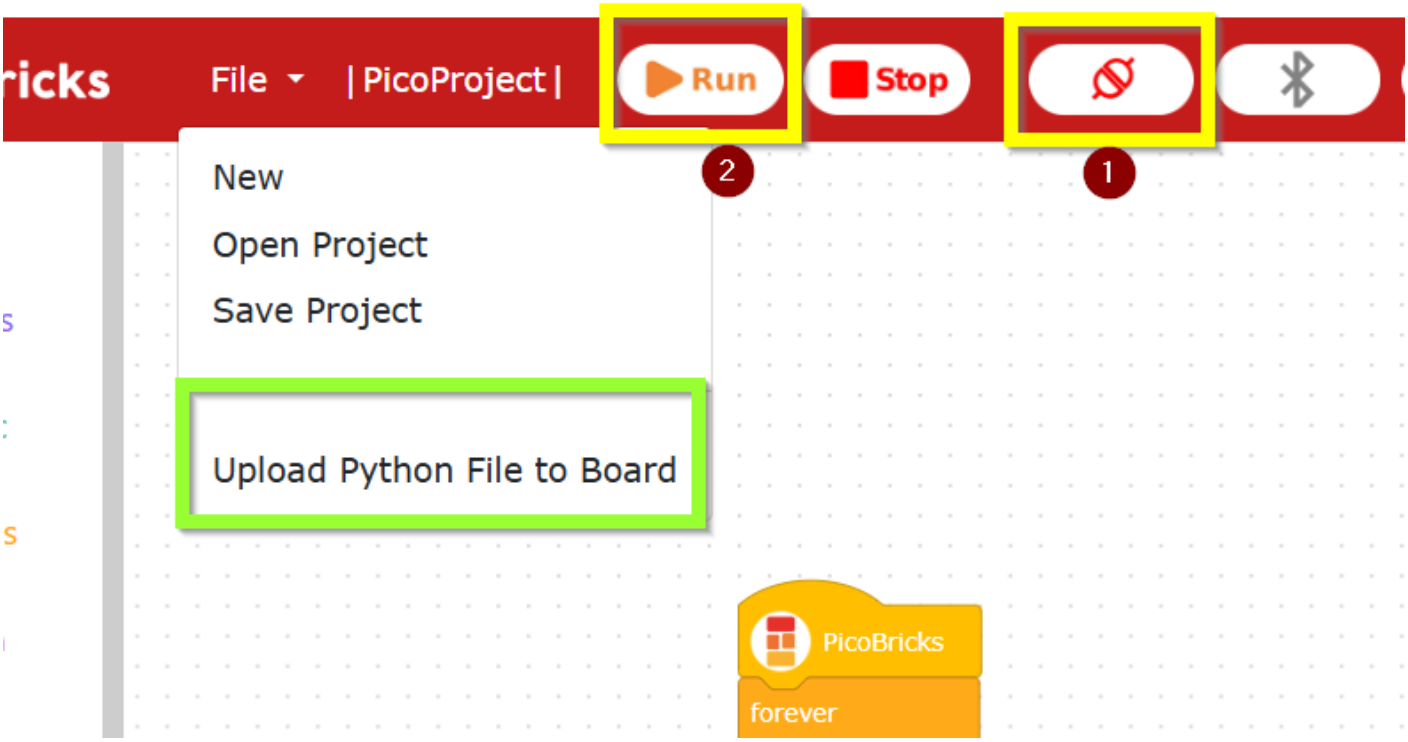
La forma más fácil de trabajar es **EN VIVO** es decir, que los programas se ejecuten **desde nuestro ordenador** es la más rápida y para ello necesita que el PicoBricks tenga el Firmware correspondiente dentro (tal y como hemos visto)

La otra forma de trabajar es **EN CARGA** es decir que los programas se ejecutan **desde dentro de PicoBricks** tiene la ventaja que el programa funciona si necesidad de ordenador. Eso sí, hay que alimentar Picobriks por el cable USB (usando un Powerbank o un cargador de móvil por ejemplo)

> **ATENCIÓN** si trabajamos EN CARGA nos "cargamos" el Firmware, por lo que si queremos volver a trabajar EN VIVO tenemos que volverlo a poner tal y como hemos visto

> Recomendamos EN VIVO por la rapidez y sencillez. Sólo es aconsejable EN CARGA cuando sean proyectos que precisen que el ordenador no esté.

- Para trabajar **EN VIVO** tenemos que estar **conectados** (1) y darle al **Run** (2) (recuadros amarillos)
- Para trabajar **EN CARGA** entramos en archivo y cargamos el programa dentro de Picobricks (recuadro verde) **Upload Python File to Board**

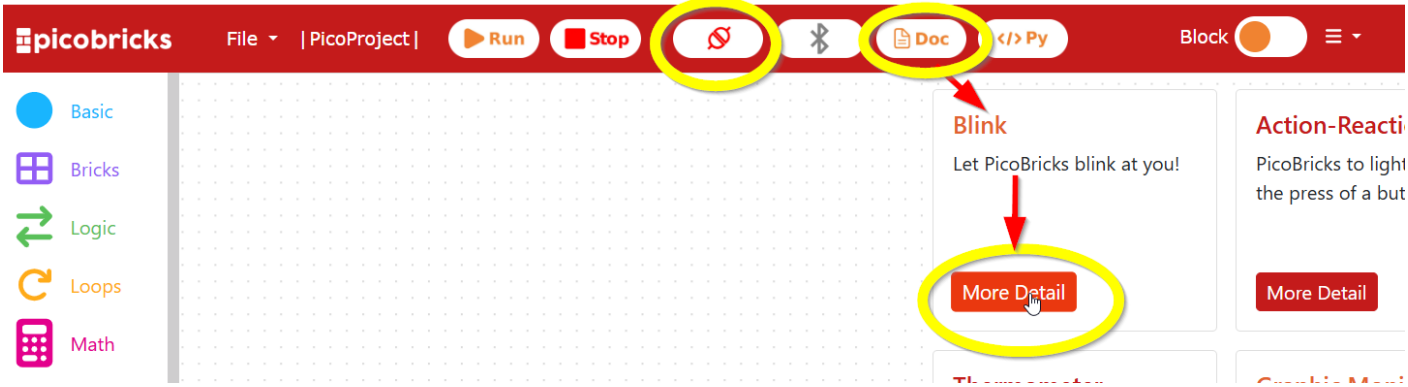

ricks
File
| PicoProject |
Run
Stop
New
Open Project
Save Project
Upload Python File to Board
2
1
PicoBricks
forever

PROYECTO BLINK

> Extraido de *Pico Bricks IDE Book CC-BY-SA* *https://picobricks.com/pages/idebook* ver créditos

Vamos a realizar nuestro primer proyecto, parpadear el led rojo

Como es un programa predeterminado, lo más cómodo es ir los tutoriales que lo explican bien



Vamos al código y si apretamos en este botón, nos aparece en nuestro panel **si necesidad de hacerlo** pero ojo que a veces está escondido tras la ventana, usar el zoom y navegar

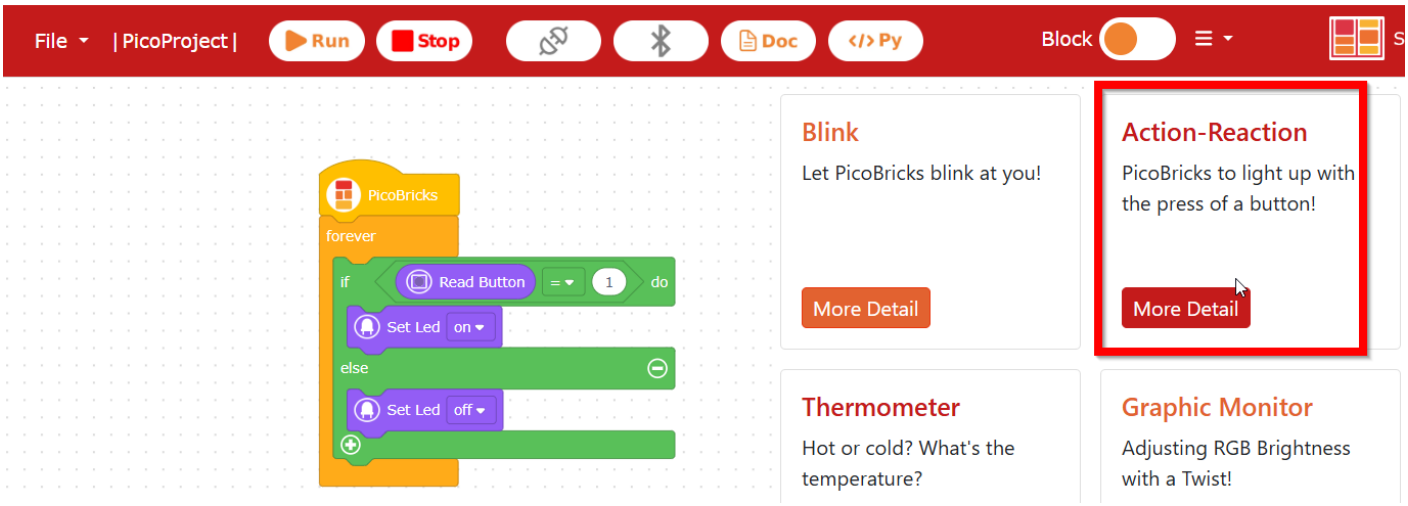
al dar a **RUN** tenemos

https://www.youtube.com/embed/nYPWAC_SHWA

PROYECTO ACTION-REACTION

> Extraido de *Pico Bricks IDE Book CC-BY-SA https://picobricks.com/pages/idebook* ver créditos

Ahora será con la interacción del botón. Repetimos los pasos pero con este proyecto:



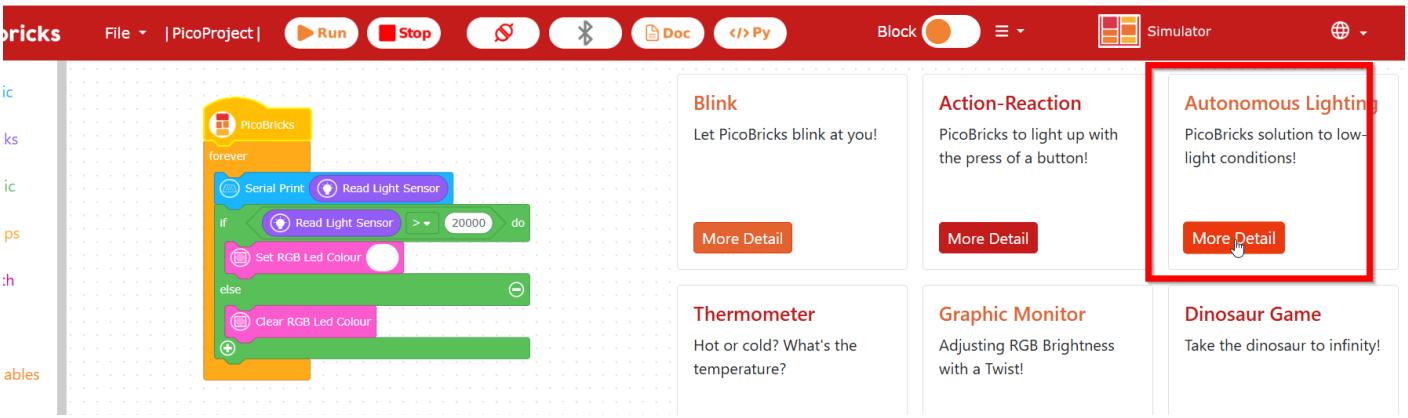
Resultado

https://www.youtube.com/embed/6cZ-Hk3dnJ8

PROYECTO Autonomous Lighting

> Extraido de *Pico Bricks IDE Book CC-BY-SA* *https://picobricks.com/pages/idebook* ver créditos

Lo mismo con el siguiente proyecto

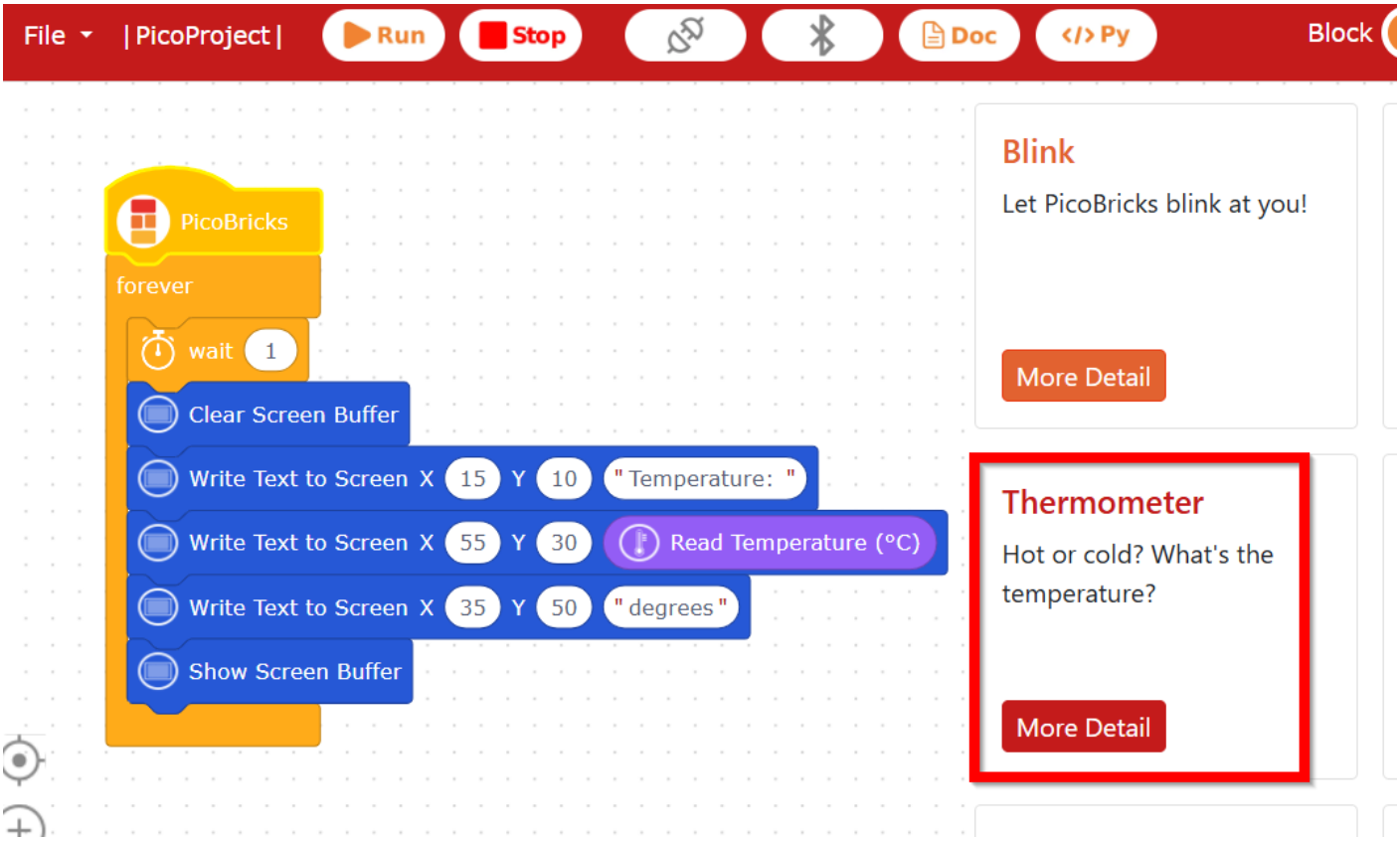


Resultado

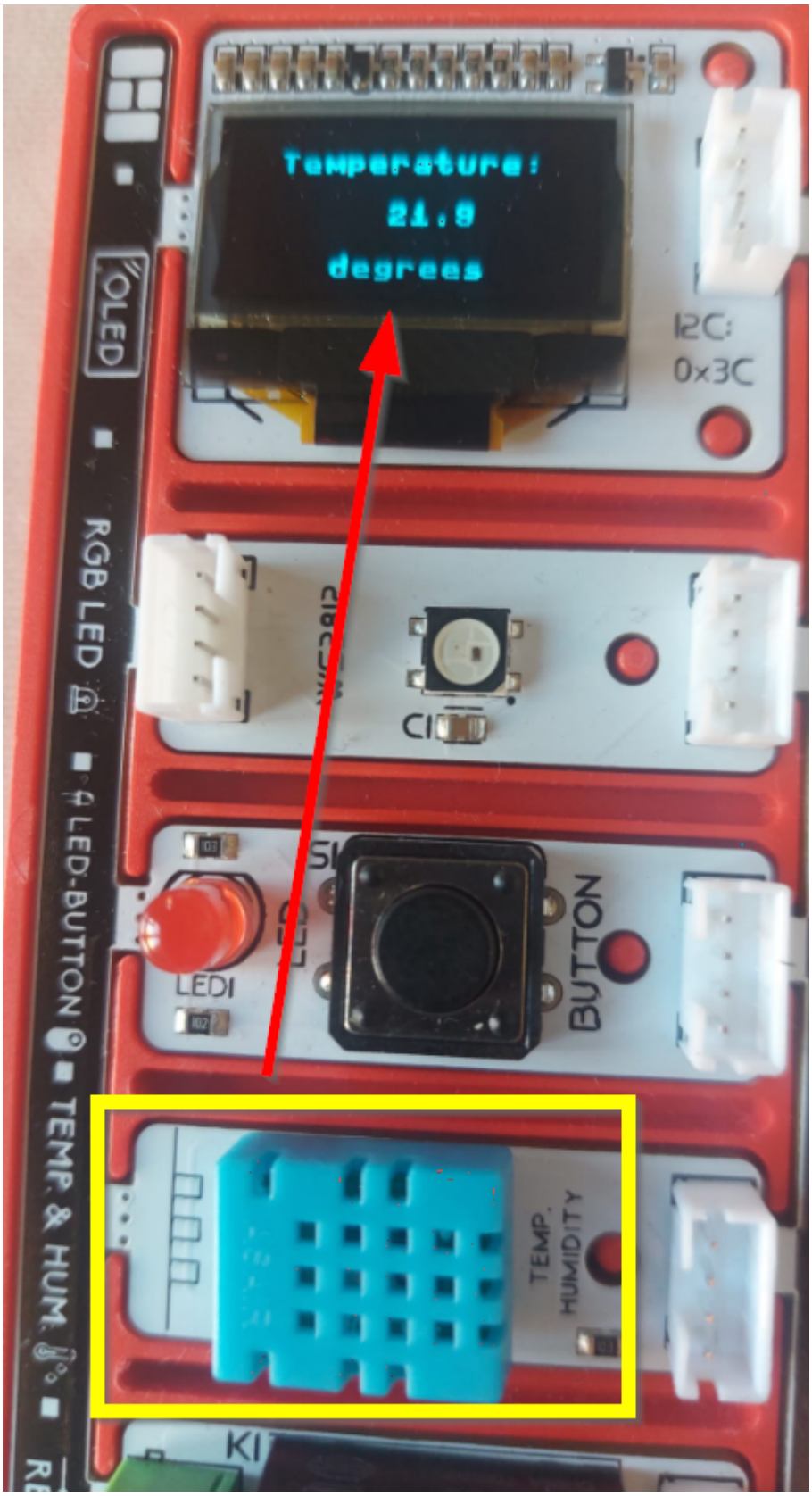
https://www.youtube.com/embed/sN8Y3boBPAg

PROYECTO Thermometer

> Extraído de *Pico Bricks IDE Book CC-BY-SA* *https://picobricks.com/pages/idebook* ver créditos



Si soplamos el aliento sobre el sensor podemos ver como sube la temperatura



> Recomendamos este proyecto cargarlo en el PicoBricks y así funciona autónomo sin necesidad de PC, con lo que se puede colocar en el exterior y ver la temperatura que hace simplemente alimentandolo con un PowerBank en el cable USB

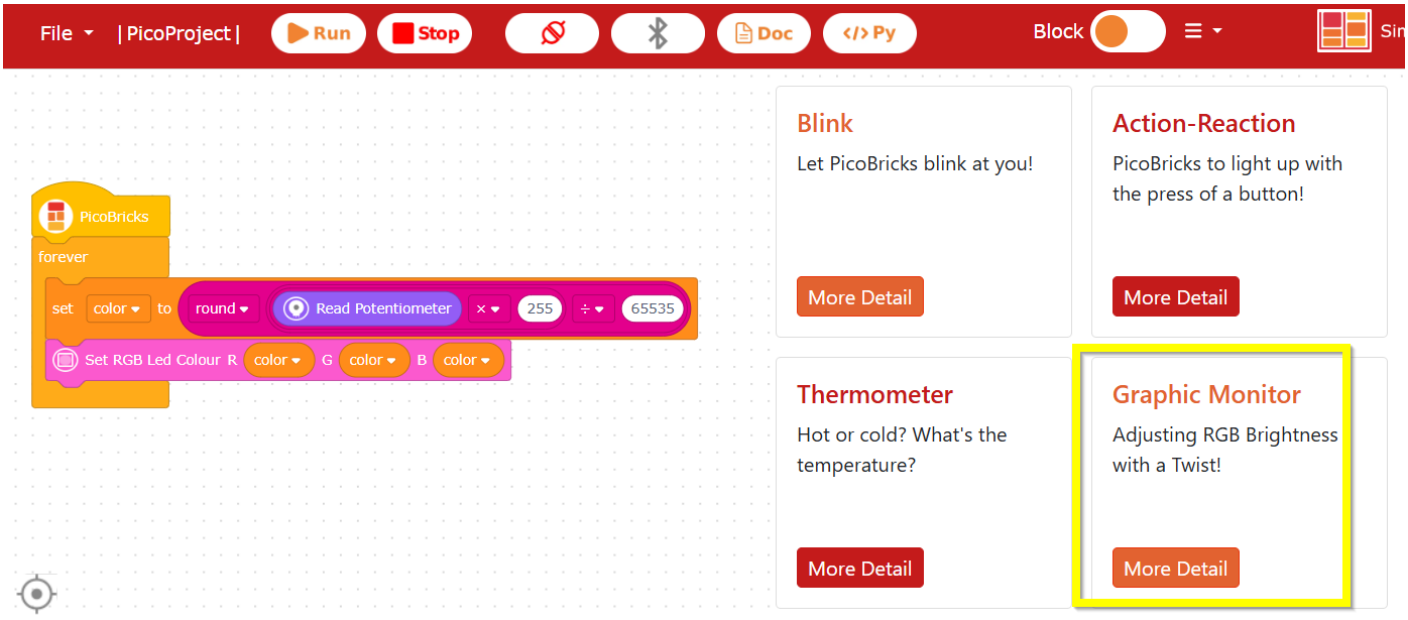
>

P: ¿No sabes cómo se carga el programa en PicoBricks?

R: Porque no te has leído https://libros.catedu.es/books/pico-bricks/page/dos-formas-de-ejecutar-los-programas

PROYECTO Graphic Monitor

> Extraido de *Pico Bricks IDE Book CC-BY-SA https://picobricks.com/pages/idebook* ver créditos



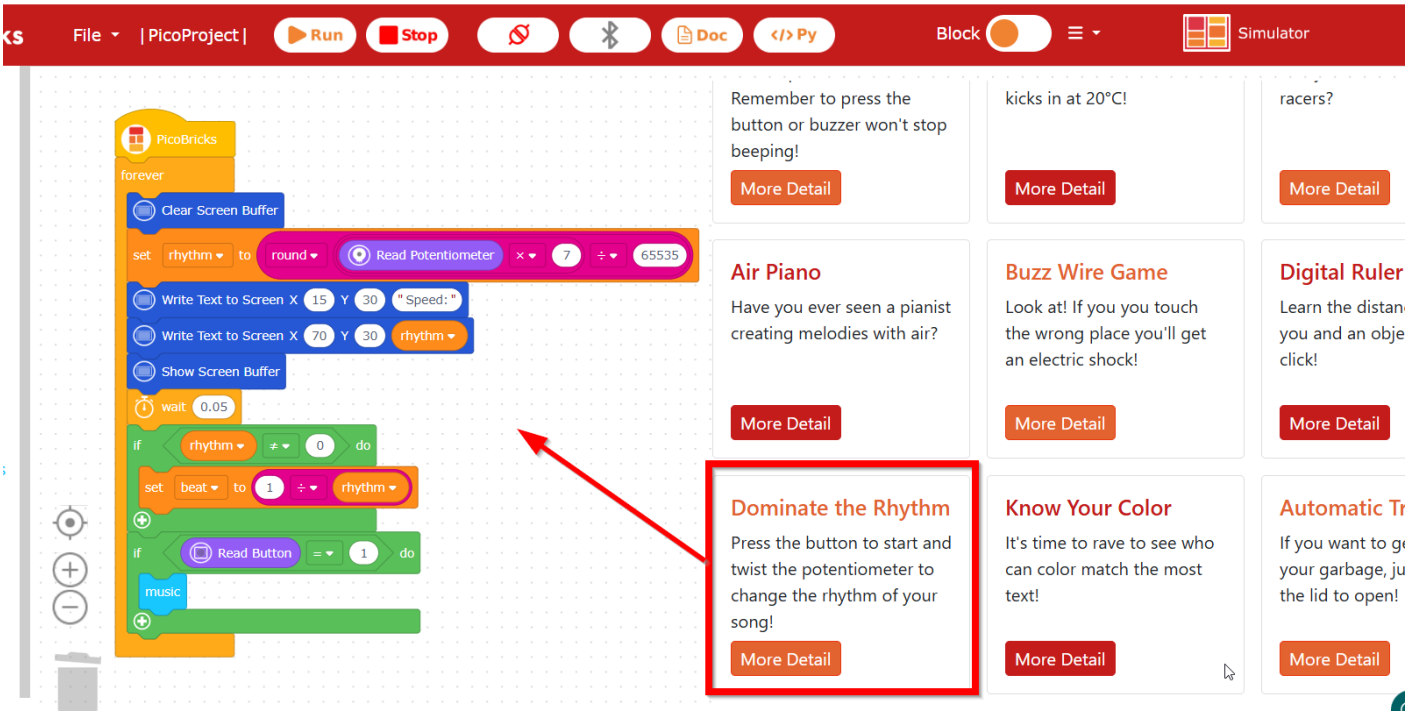
Resultado

https://www.youtube.com/embed/NqovcB6RImA

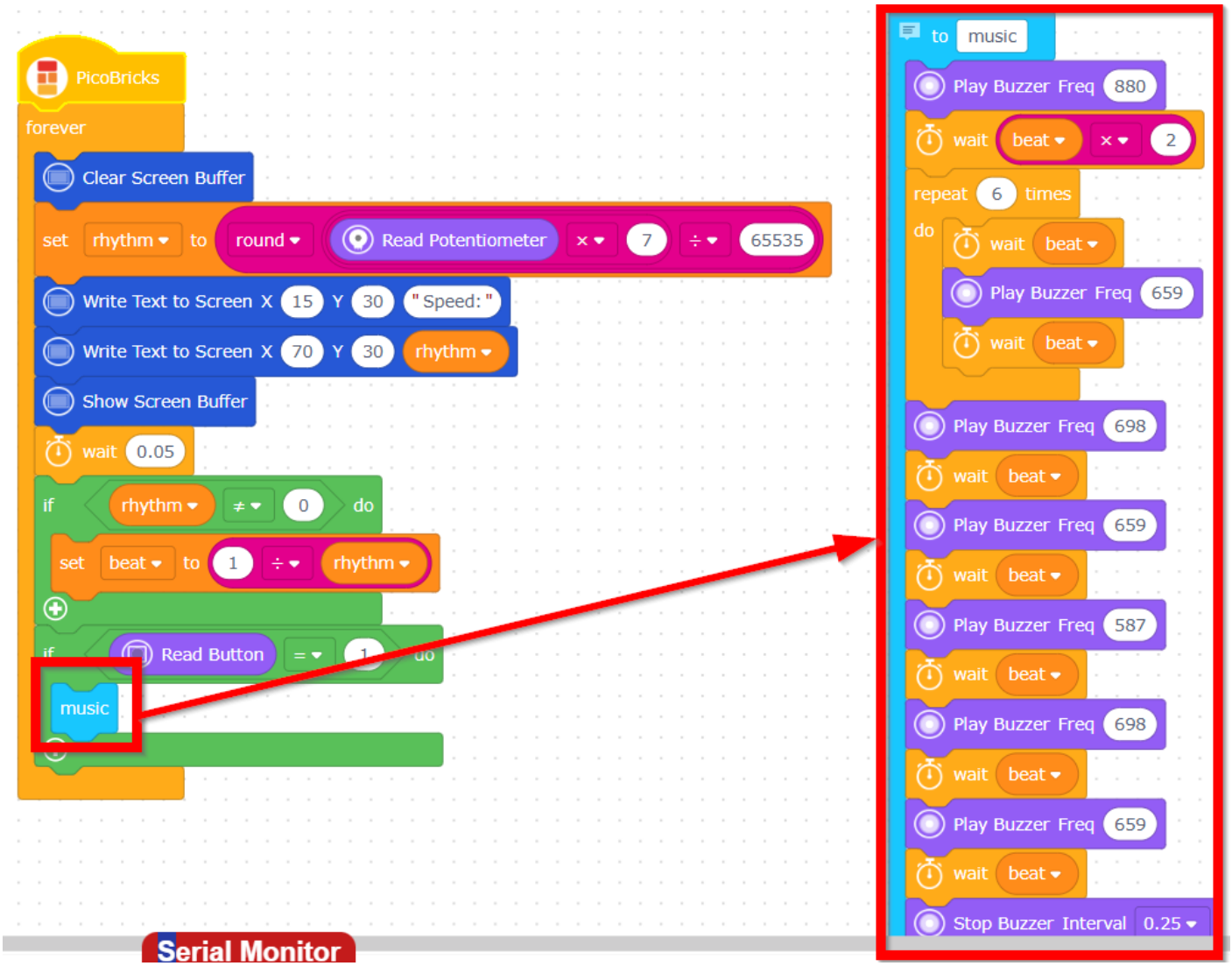
PROYECTO Dominate the Rhythm

> Extraído de *Pico Bricks IDE Book CC-BY-SA https://picobricks.com/pages/idebook* ver créditos

Este proyecto ya es más complejo y recomendamos importarlo desde el tutorial como siempre pues es más largo



Implica la utilización de FUNCIONES



Y recomendamos leer el tutorial, esta bien explicado en el libro en la página 34;

https://drive.google.com/file/d/1plad6bjn87FcgHb3cpd1vI-B_A25rnfF/preview

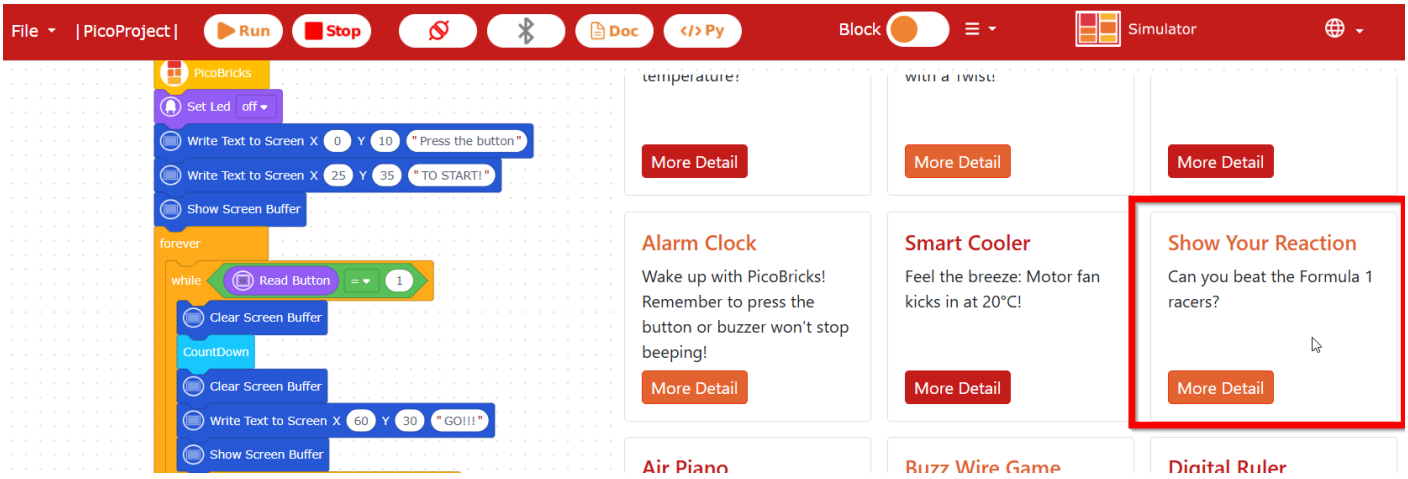
Como se puede ver en el resultado, la primera vez suena la música a un ritmo número 4 pero en la segunda vez subimos con el potenciómetro al ritmo máximo 7 y la música suena más deprisa

https://www.youtube.com/embed/WynkqehvWuw

PROYECTO Show Your Reaction

> Extraido de *Pico Bricks IDE Book CC-BY-SA https://picobricks.com/pages/idebook* ver créditos

Cuando nuestro proyecto comience a funcionar, mostraremos un mensaje de bienvenida en el OLED pantalla. A continuación imprimiremos en la pantalla lo que el usuario tiene que hacer para iniciar el juego. Para comenzar el juego, le pediremos al jugador que se prepare contando hacia atrás desde 3 en la pantalla después de presionar el botón. Después del final de la cuenta regresiva, el El LED rojo se encenderá en un tiempo aleatorio entre 2 y 10 segundos. Reiniciaremos el temporizador inmediatamente después se enciende el LED rojo. Mediremos el temporizador tan pronto como el se vuelve a pulsar el botón. Este valor que obtengamos estará en milisegundos. Mostraremos esto en la pantalla como el tiempo de reacción del jugador.



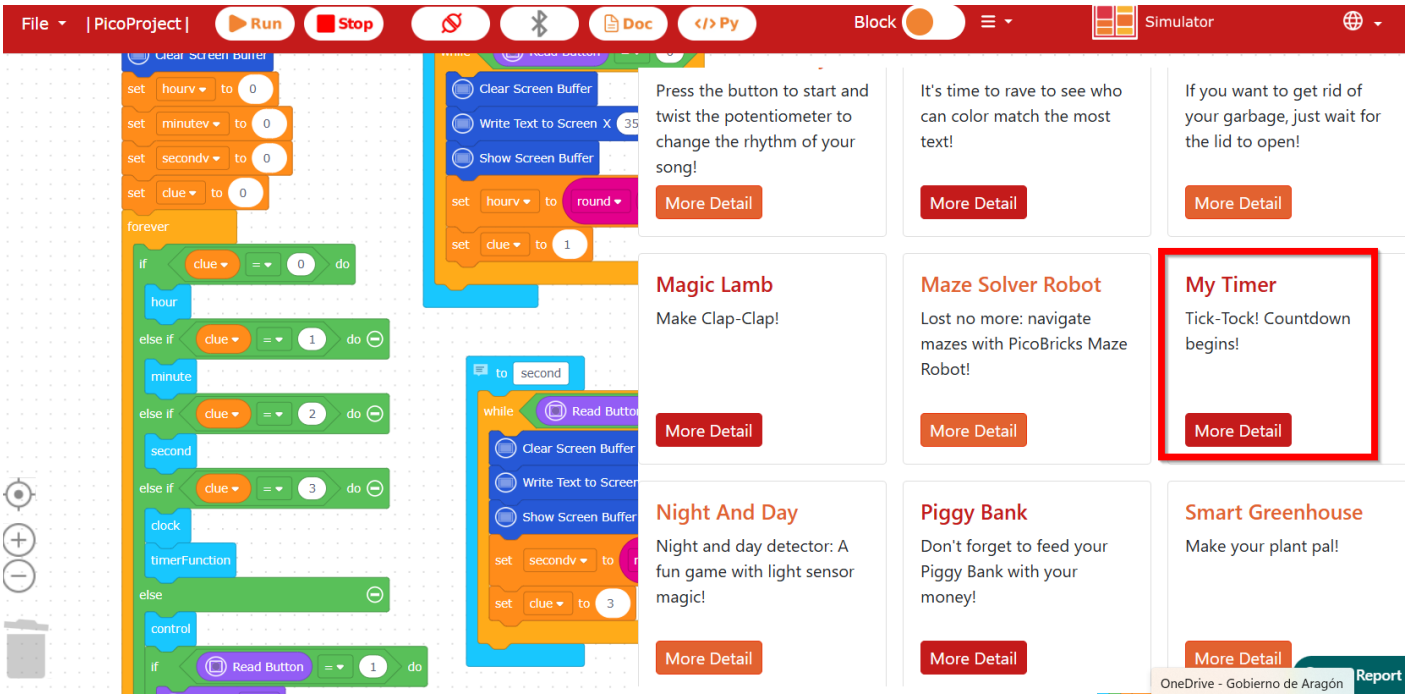
Aquí he ganado pues sólo he tardado 1ms en pulsar el botón

https://www.youtube.com/embed/CLEUZPzl2vI

PROYECTO My Timer

> Extraído de *Pico Bricks IDE Book CC-BY-SA* *https://picobricks.com/pages/idebook* ver créditos

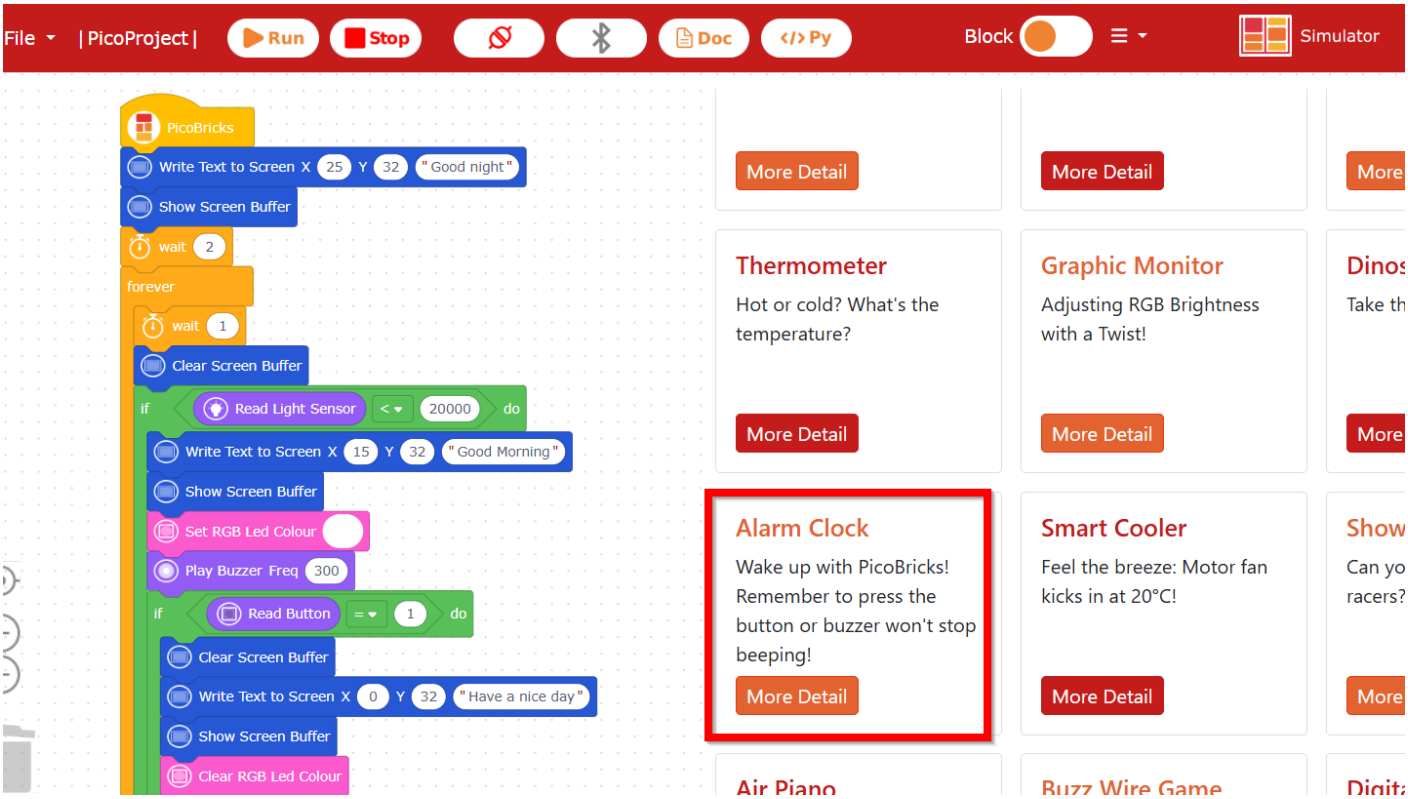
El clásico cuenta atrás pero con la peculiaridad que es fácil de programar con el potenciómetro, hasta las horas !



https://www.youtube.com/embed/QA7Oe8KibCo

PROYECTO Alarm Clock

> Extraído de *Pico Bricks IDE Book CC-BY-SA https://picobricks.com/pages/idebook* ver créditos



https://www.youtube.com/embed/8Drcl_YEsFs

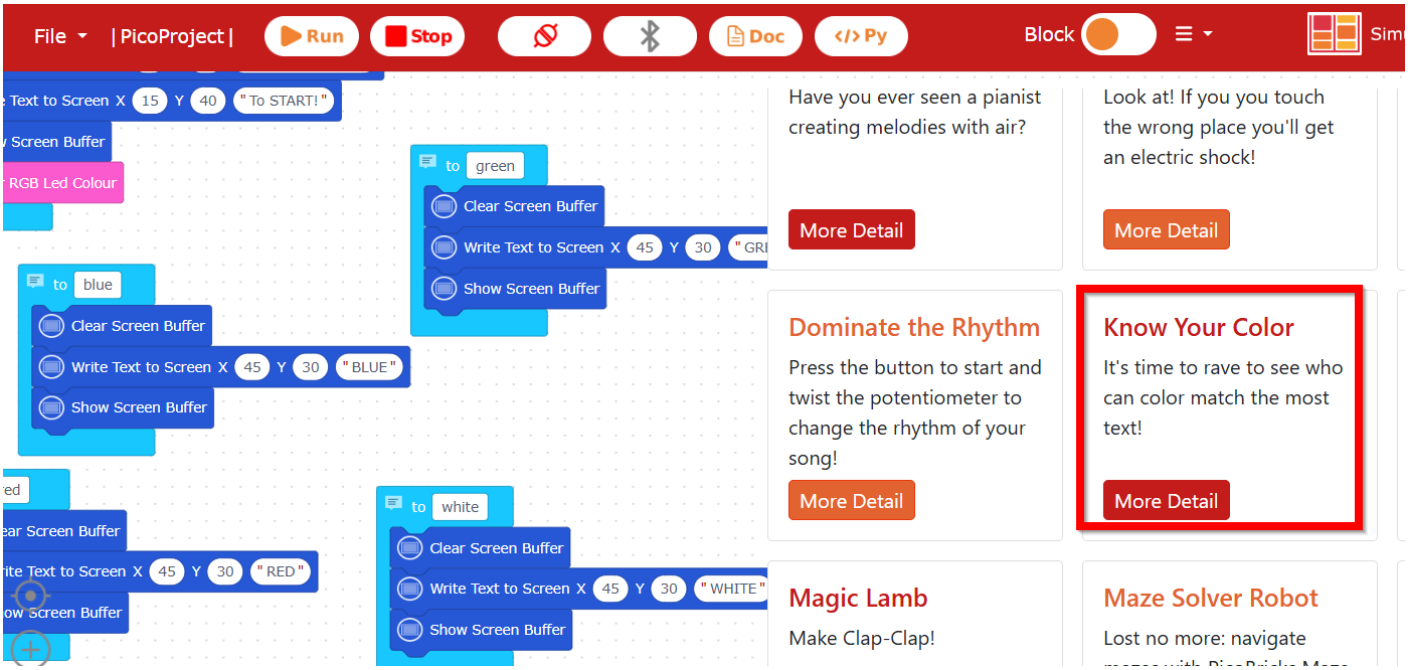
PROYECTO Know Your Color

> Extraido de *Pico Bricks IDE Book CC-BY-SA* *https://picobricks.com/pages/idebook* ver créditos

El juego que construiremos en el proyecto se construirá en base a que el usuario conozca los colores correcta o incorrectamente. Uno de los colores rojo, verde, azul y blanco se iluminará aleatoriamente en el LED RGB de Picobricks, y el nombre de uno de estos cuatro colores se escribirá aleatoriamente en la pantalla OLED al mismo tiempo. El usuario debe pulsar el botón de Picobricks en 1,5 segundos para utilizar el derecho de réplica.

- El juego se repetirá 10 veces, cada repetición obtendrá
  - 10 puntos si el usuario presiona el botón cuando los colores coinciden
  - -10 puntos si no coinciden

Después de diez repeticiones, la puntuación del usuario se mostrará en el OLED pantalla. Si el usuario lo desea, no podrá hacer uso de su derecho de réplica no pulsando el botón botón.



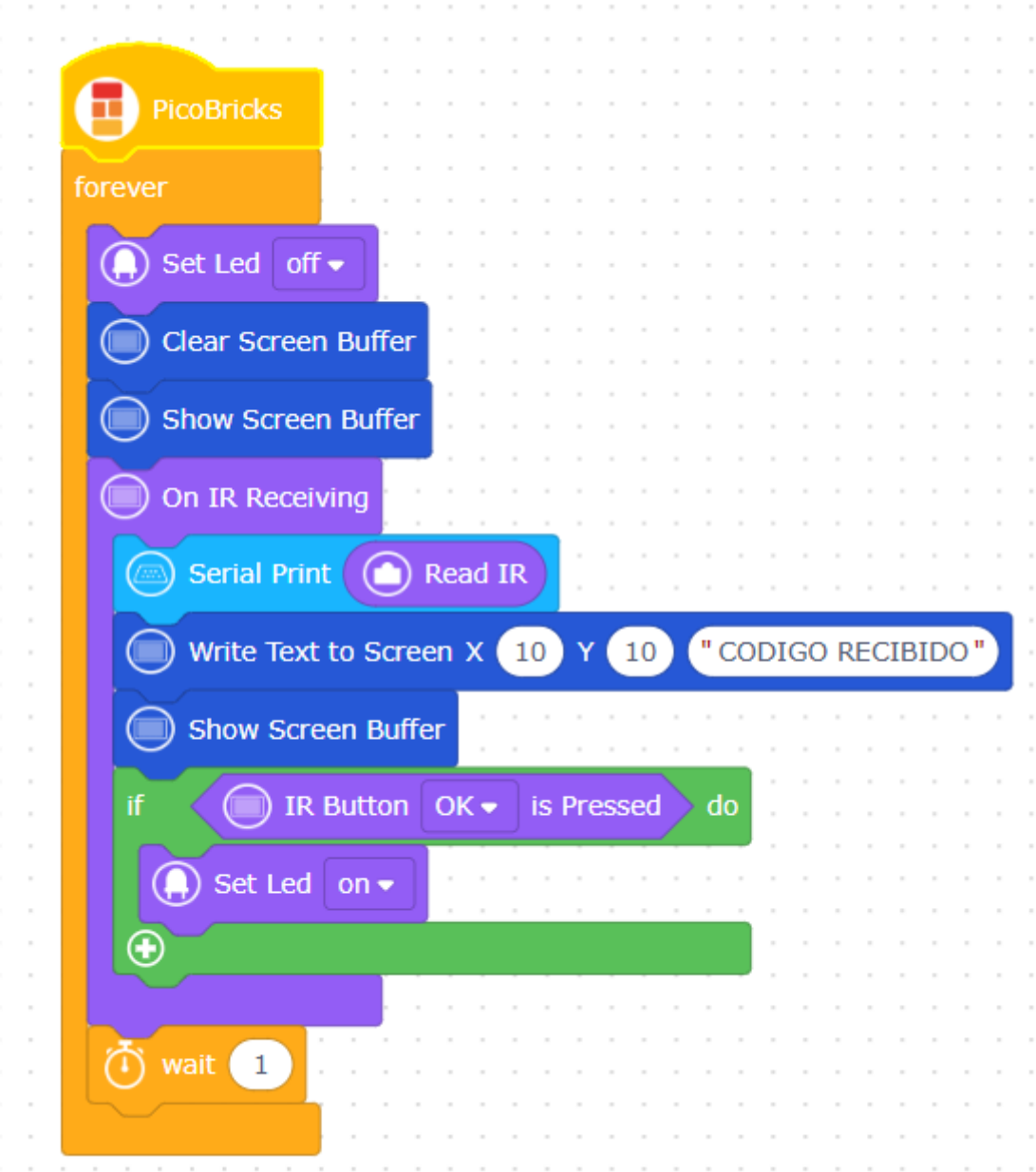
https://www.youtube.com/embed/Tv9z8krs26g

Algo diferente PROYECTO IR

No hay en los tutoriales ningún proyecto para usar el mando IR, luego este proyecto no pertenece a ninguno de los tutoriales que predetermina PicoBricks. Proponemos el siguiente enunciado

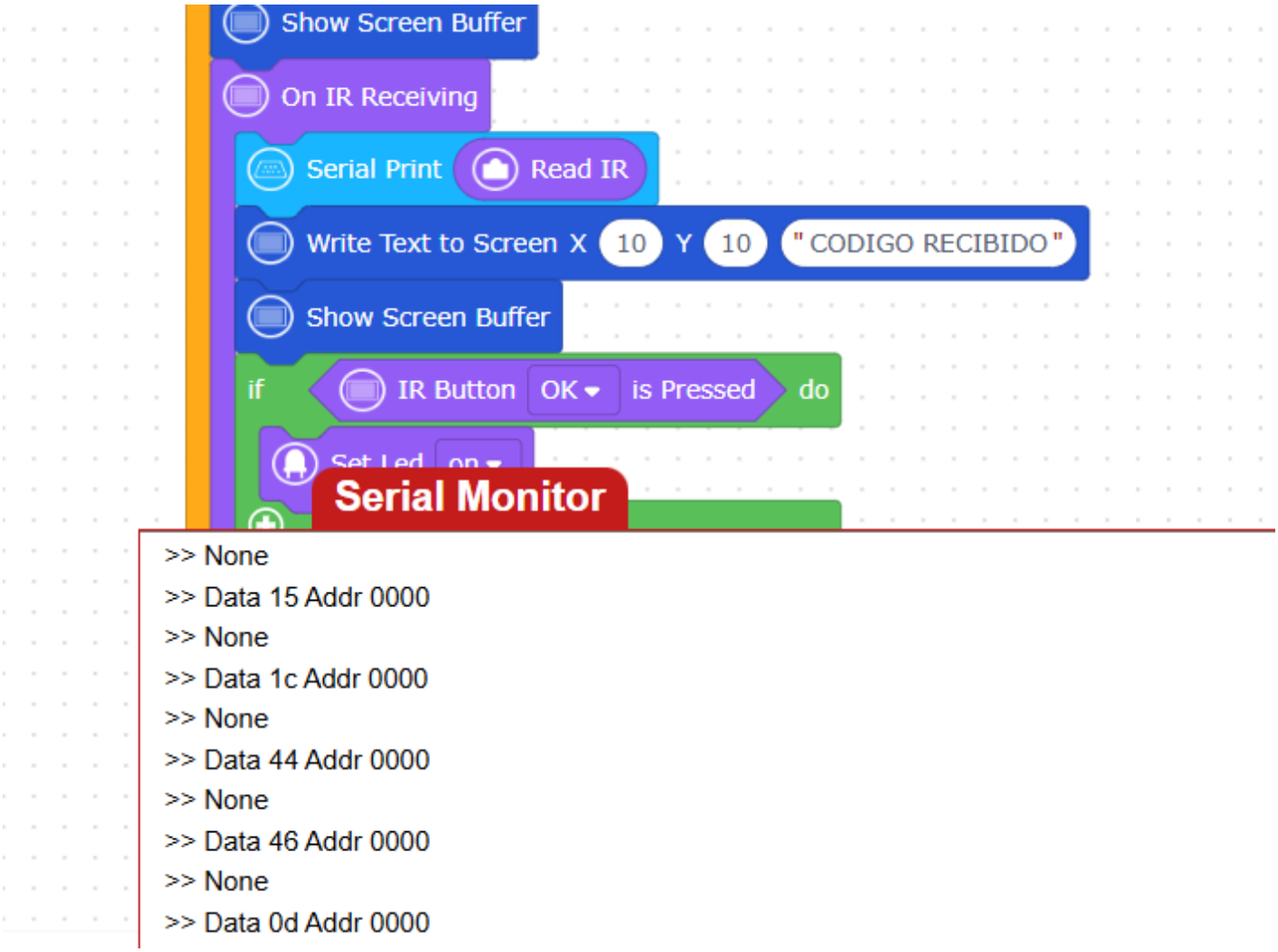
> Realizar un programa que:
> -si se aprieta un botón del mando IR que visualice por la pantalla OLED que ha recibido un código
> - visualizará por el puerto serie el código recibido
> - si la tecla es OK se encenderá el led rojo

Solución



Resultado

Por el puerto serie van apareciendo los códigos de las teclas apretadas en el mando IR



y en la pantalla OLED se visualizaba que se había recibido un código y si era OK se enciende el led rojo:

https://www.youtube.com/embed/6kSRjpbTSDg

Microblocks

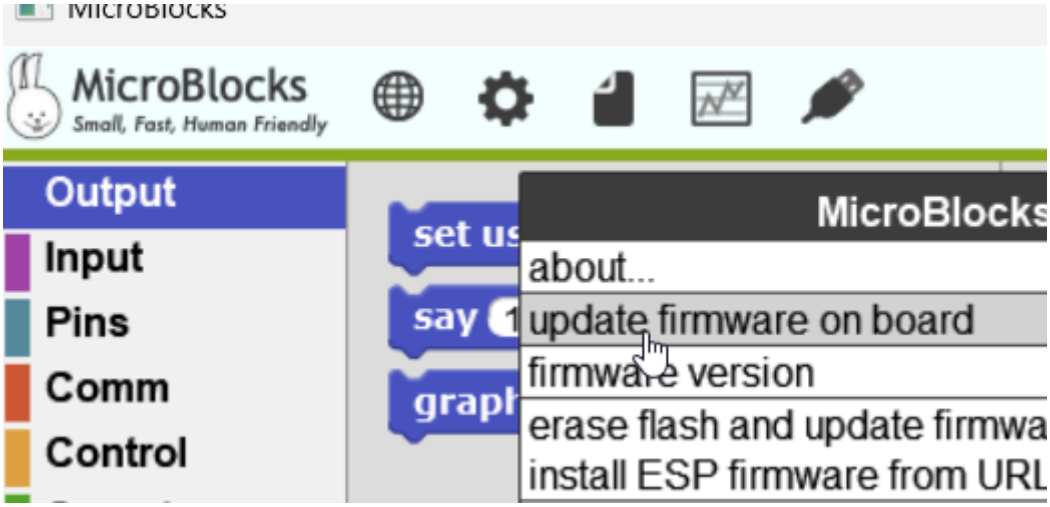
Conexión con Microblocks

Microblocks es un software extendido de uso libre para programar en bloques para diferentes placas. Se puede trabajar online o también descargarse e instalarlo localmente en el ordenador en https://microblocks.fun/

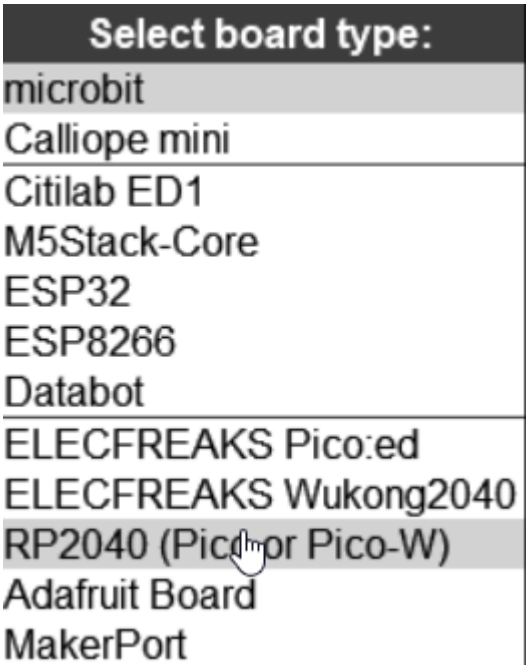
INSTALAR EL FIRMWARE

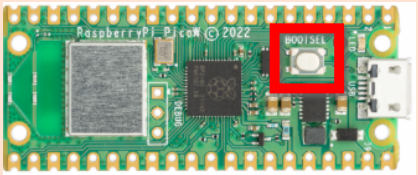
> **ATENCIÓN** al instalar el firmware de MicroBlocks **te cargas** el firmware de PicoBlockly por lo tanto si quiere volver a programar con PicoBlockly tienes que poner su firmware (y viceversa)

Entramos en Micrboblocks y vamos al menú de la **rueda dentada** a **Update firmware on board**

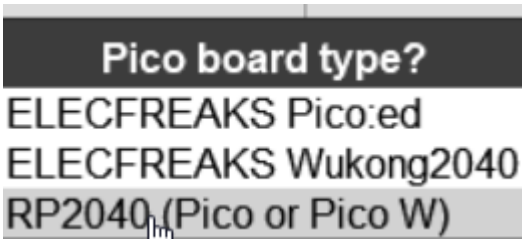


Nos vamos a RP2040 PicoW

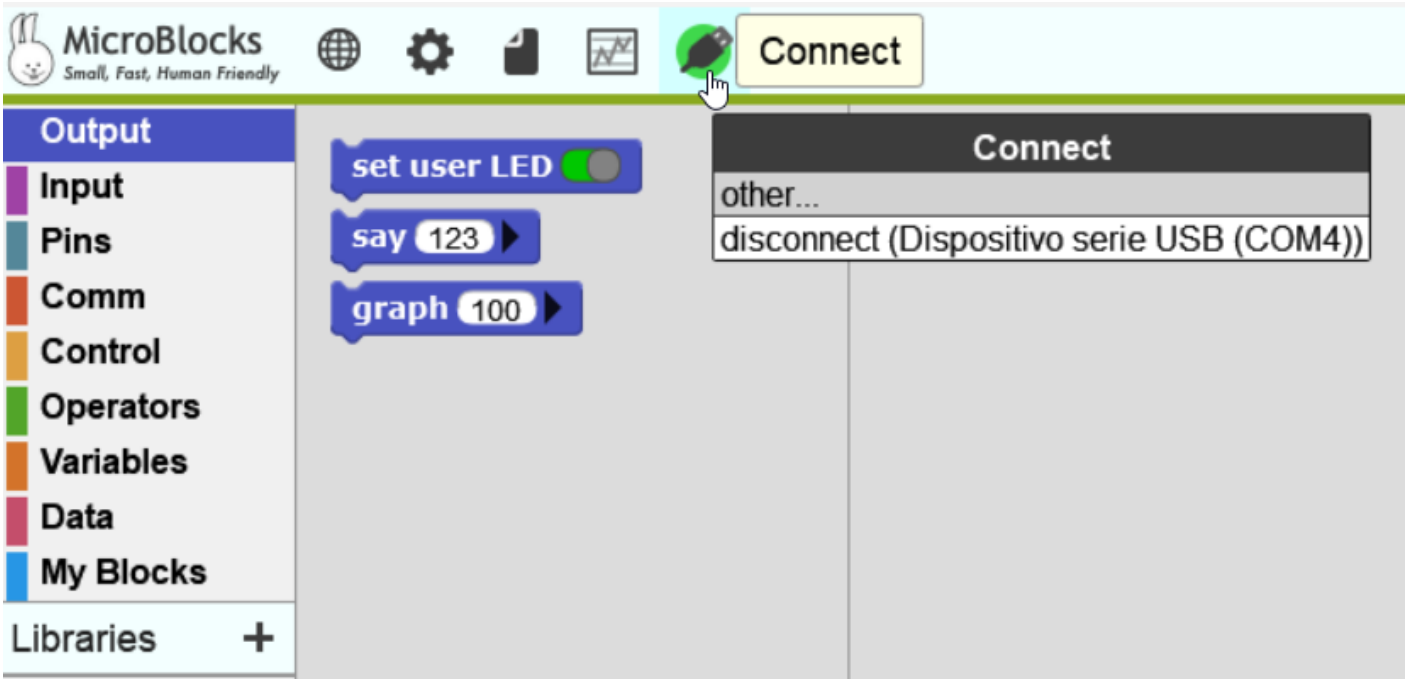


> **ATENCIÓN, poner PicoBricks en modo Bootloader**
> 1.-Desconectamos PicoBricks de nuestro ordenador
> 2.- Apretamos el botón BOOTSEL **mientras** lo volvemos a conectar al puerto USB
> 
> 3.- Automáticamente aparecerá una nueva unidad de disco en nuestro ordenador (ya puedes soltar BOOTSEL)

Y en la siguiente pantalla volvemos a elegir RP2040

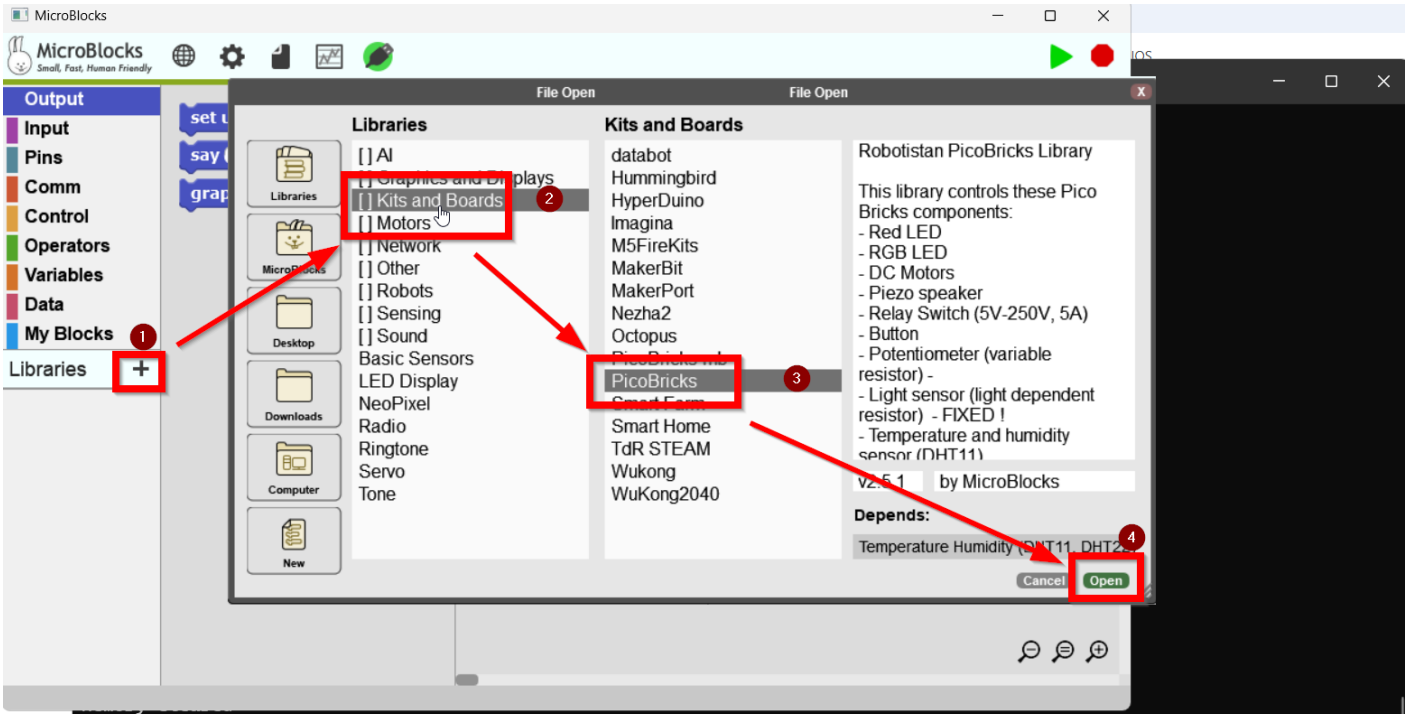


Entonces aparecerá CONECTADO el icono del USB

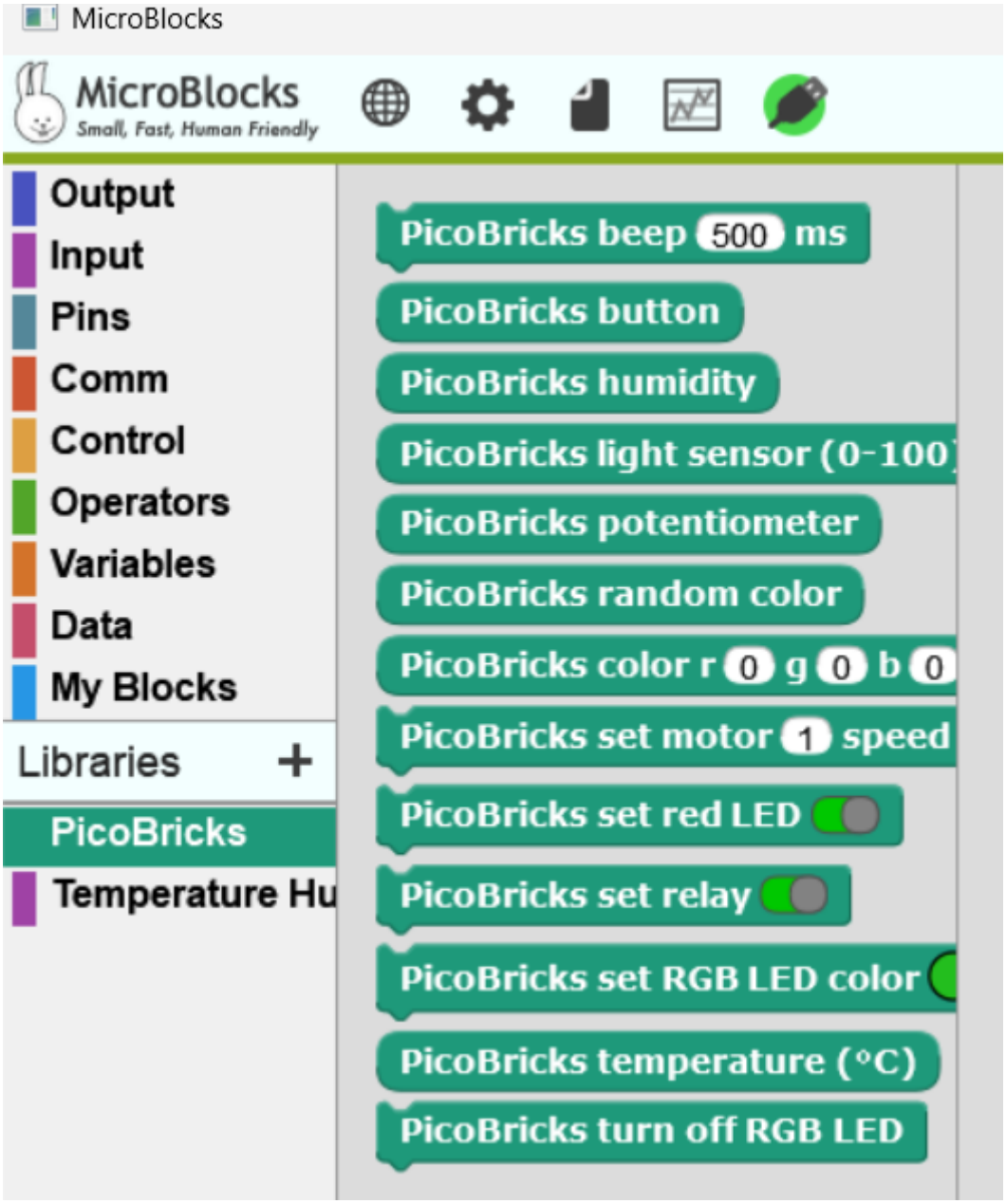


INSTALACIÓN DE LAS LIBRERÍAS

Entramos en la siguiente ruta

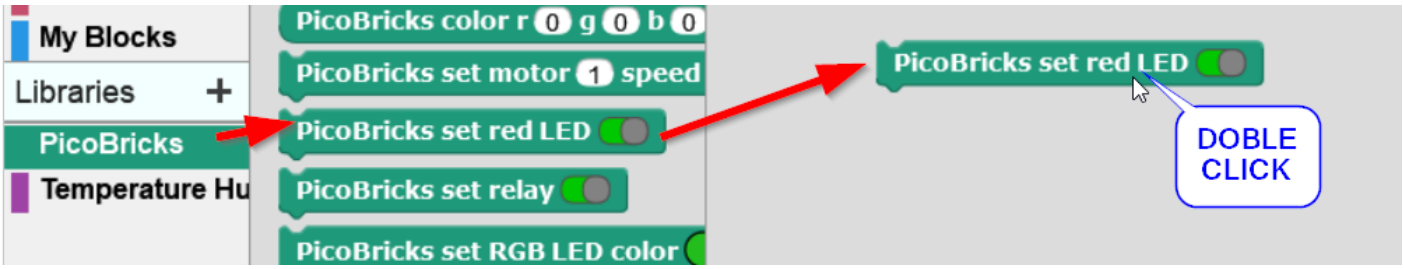


Y entonces se instala una librería para poder manejar Picobricks



PRUEBALO

Microblocks trabaja en vivo, luego arrastra esta sencilla instrucción y da dos clicks **comprueba que se enciende el led rojo**



haz dos clicks con el selector en rojo para apagarlo


PicoBricks set red LED

PROYECTOS

Los proyectos vistos con PicoBlockly se pueden hacer igual con Microblocks.

Los tienes todos desarrollados **paso a paso en este libro** (en inglés) que lo puedes conseguir aquí https://picobricks.com/pages/projectbook

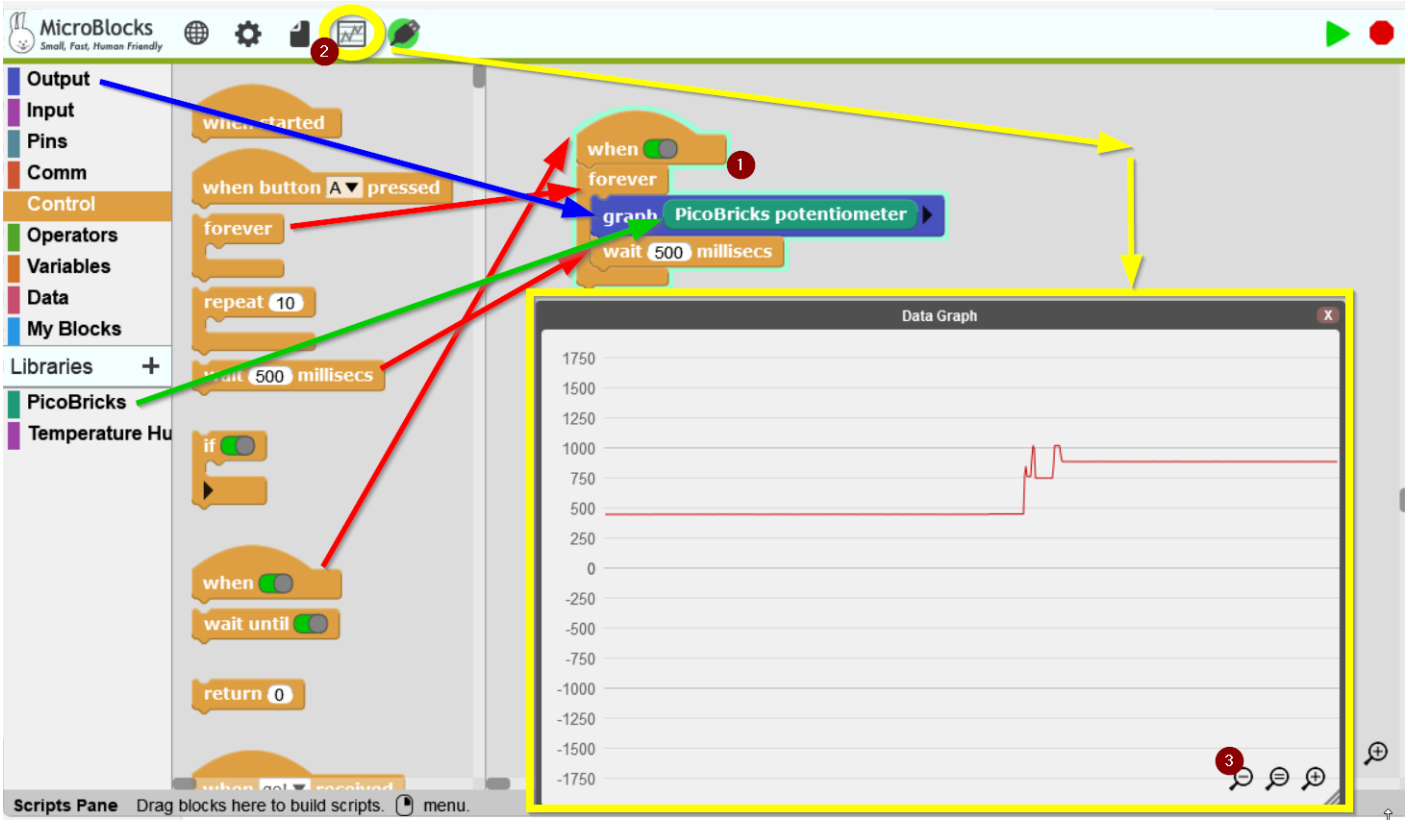
https://drive.google.com/file/d/1PDql_GYyxcz68JqmQAGOLs0YE6SXgPCm/preview

Al no tener licencia CC no los podemos reproducir aquí en este tutorial

Algo diferente: Data Graph

Microblocks tiene algo diferente a PicoBlockly y es la posibilidad de visualizar gráficamente variables

Haz el siguiente programa y pulsa en Graph y verás que puedes visualizar gráficamente los valores del potenciómetro (que van de 0 a 1023)

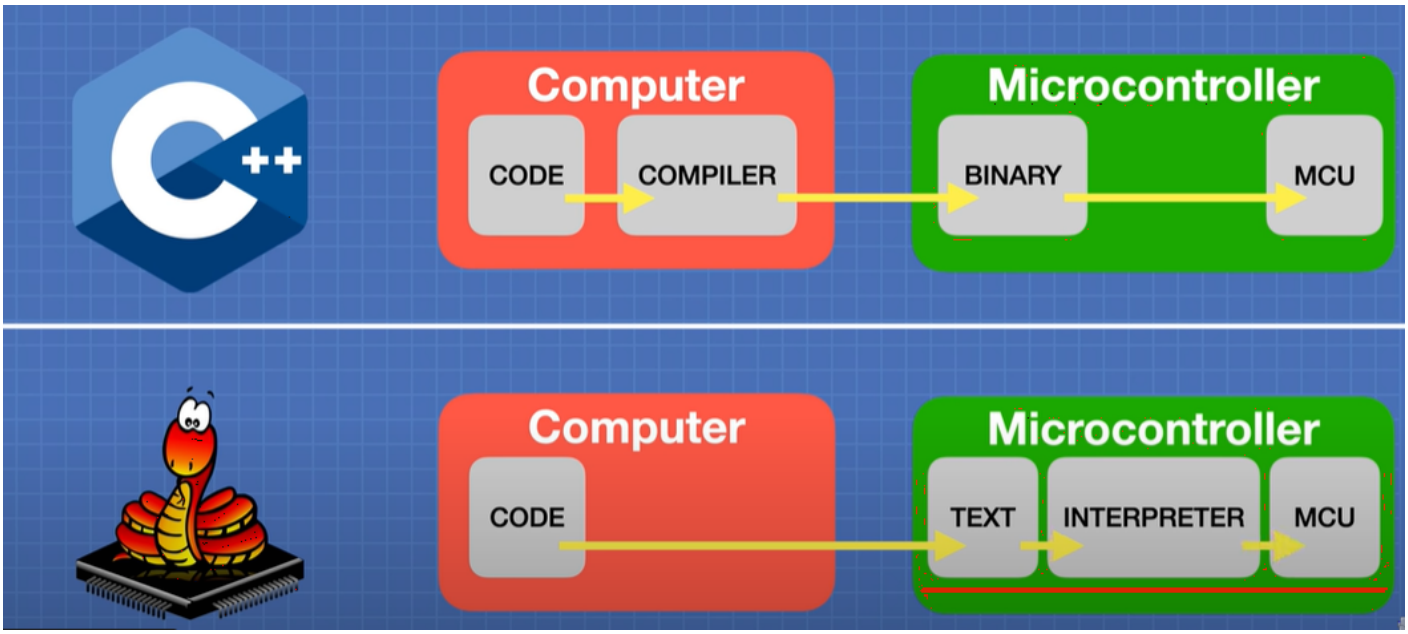


MicroPython con Thonny

Instalación de micropython

¿Dónde se instala el Micropython?

Como puedes ver en este vídeo en 21:20 Python se compila **dentro del microcontrolador** es decir, dentro del ESP32. A diferencia con otros lenguajes, como el C++, el ordenador tiene el compilador, y se lo da ya en binario.



Fuente vídeo Exploring the Arduino Nano ESP32 | MicroPython & IoT

¿Qué programa vamos a usar?

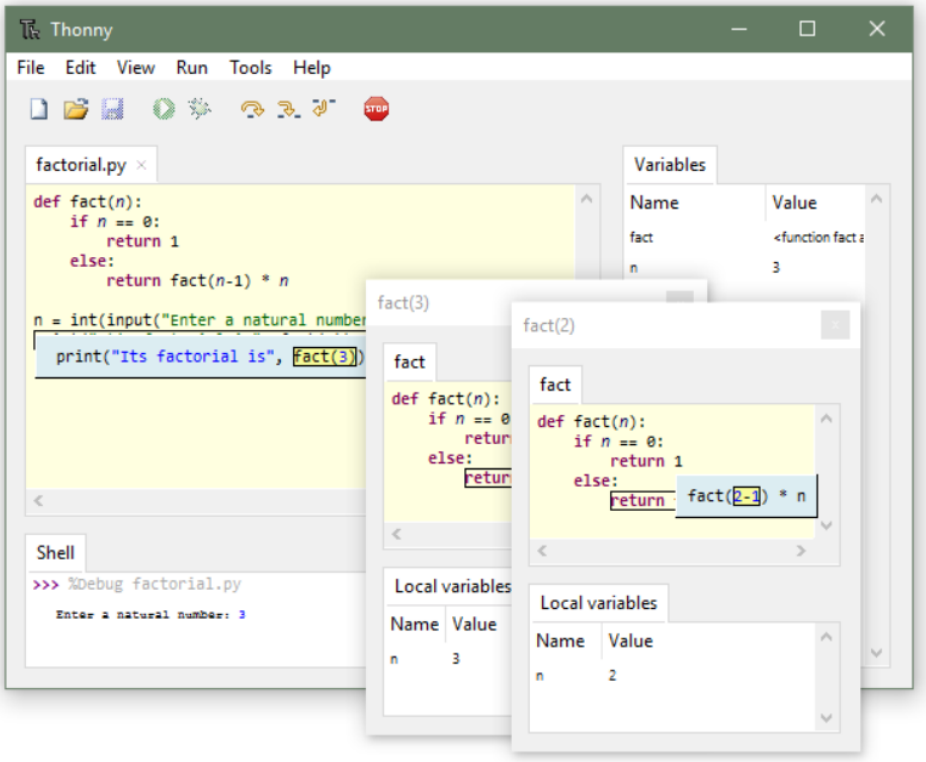
Usaremos el Thonny https://thonny.org/ que lo puedes descargar e instalar de esta página:

https://thonny.org/

Thonny

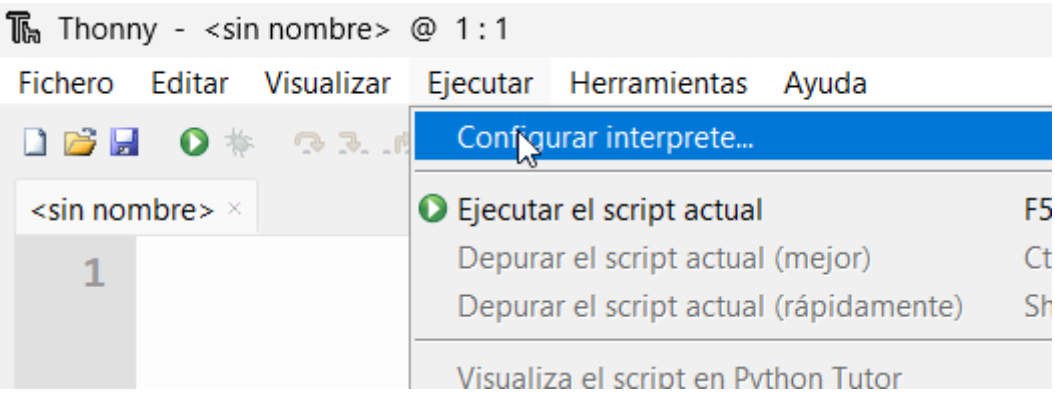
Python IDE for beginners

Download version 4.1.7 for Windows • Mac • Linux

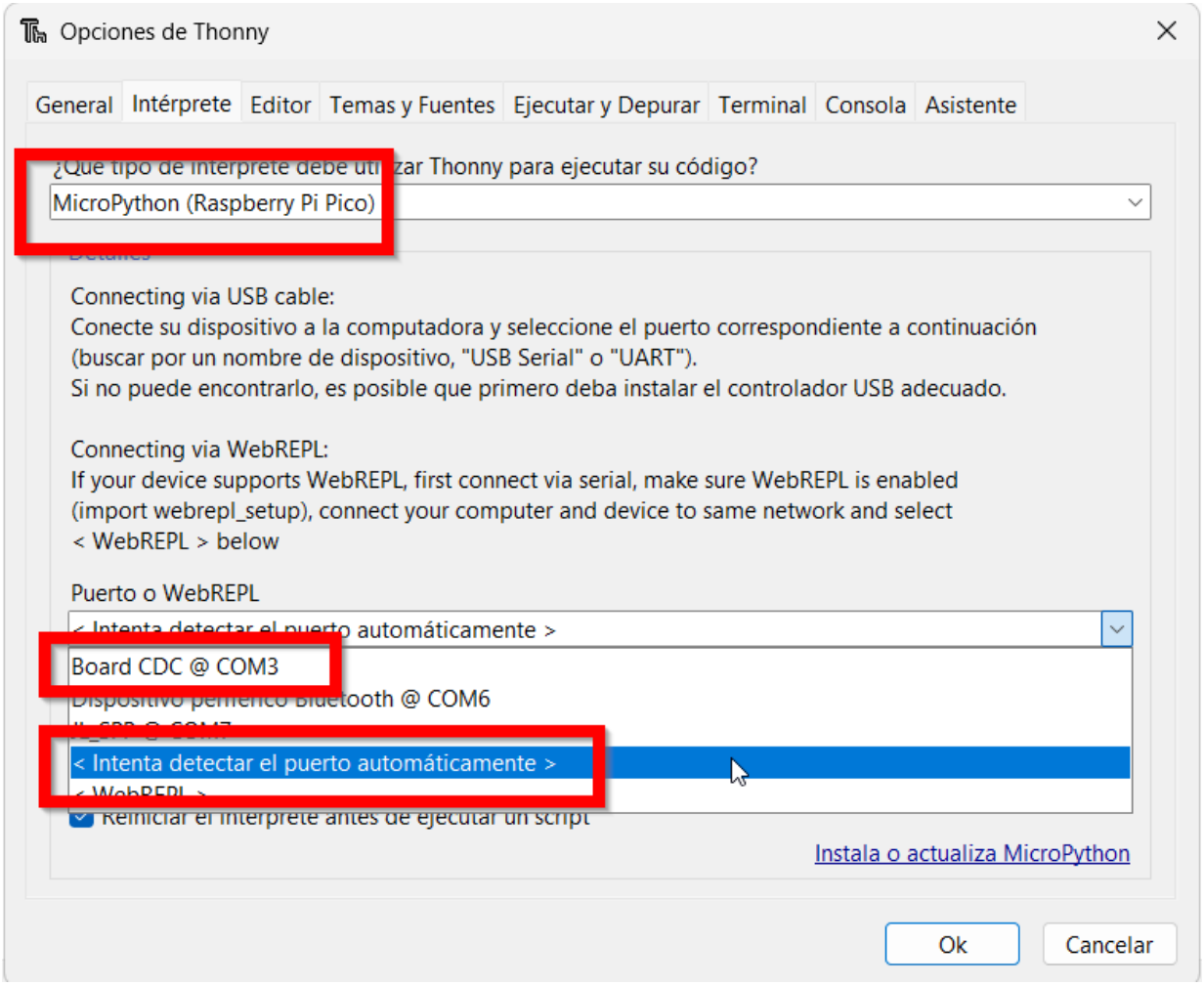


¿Cómo se instala micropython con Thonny en Picobricks?

Entramos en ejecutar-configurar intérprete



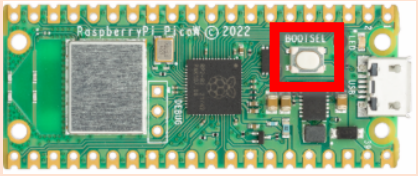
Seleccionamos en ¿Qué tipo de interprete ...? le decimos que **Rasbperry Pi pico** y el puerto si lo sabemos lo seleccionamos o si no lo sabemos que lo detecte automáticamente



ATENCIÓN, poner PicoBricks en modo Bootloader

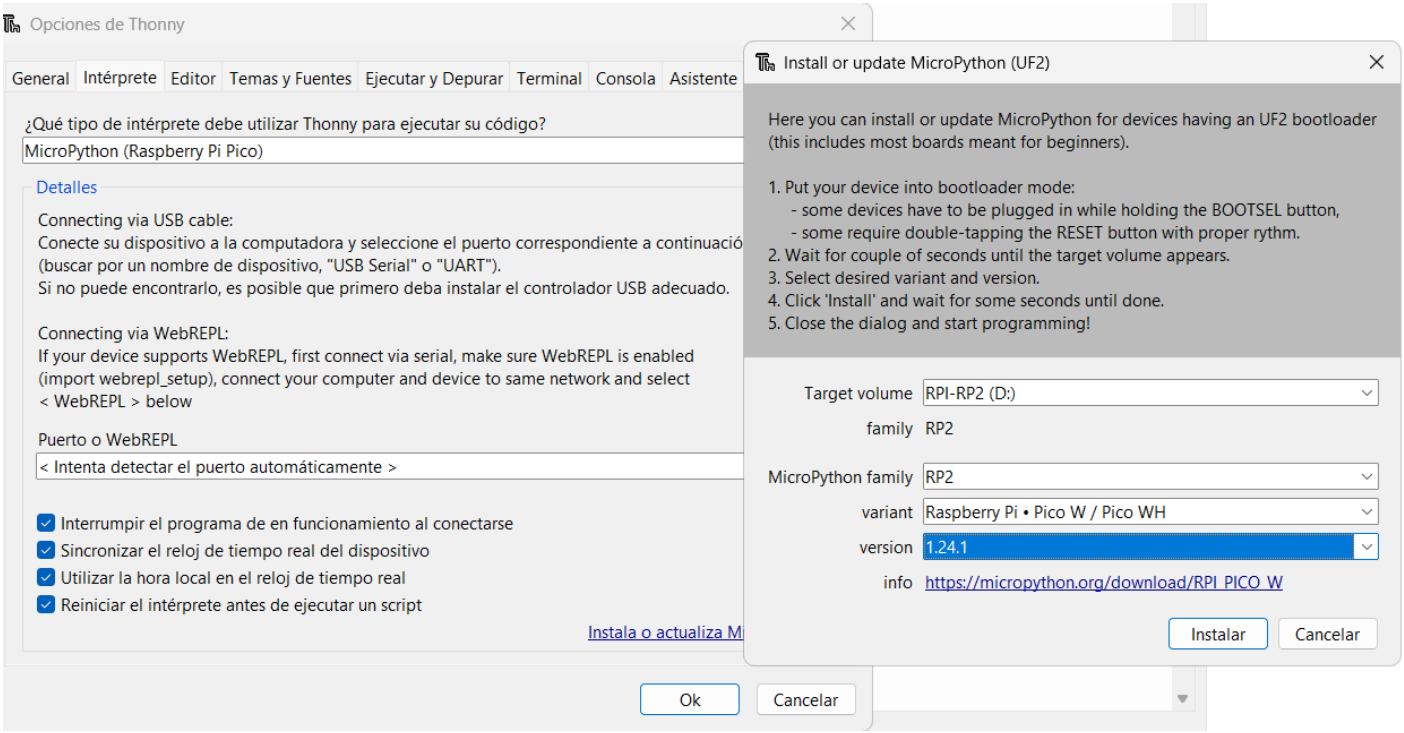
1.-Desconectamos PicoBricks de nuestro ordenador

2.- Apretamos el botón BOOTSEL **mientras** lo volvemos a conectar al puerto USB

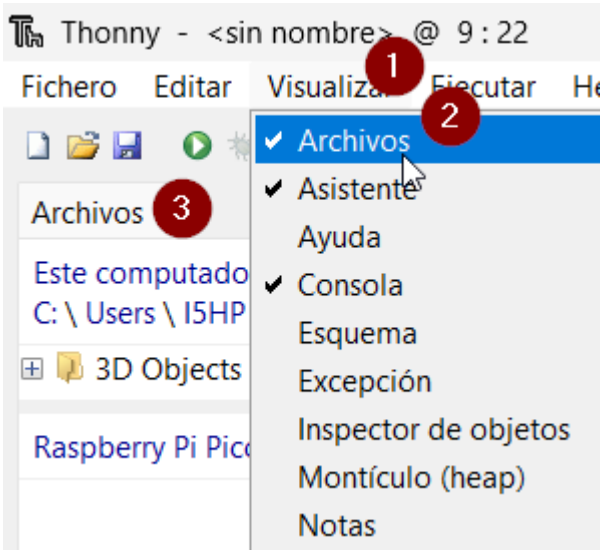


3.- Automáticamente aparecerá una nueva unidad de disco en nuestro ordenador (ya puedes soltar BOOTSEL)

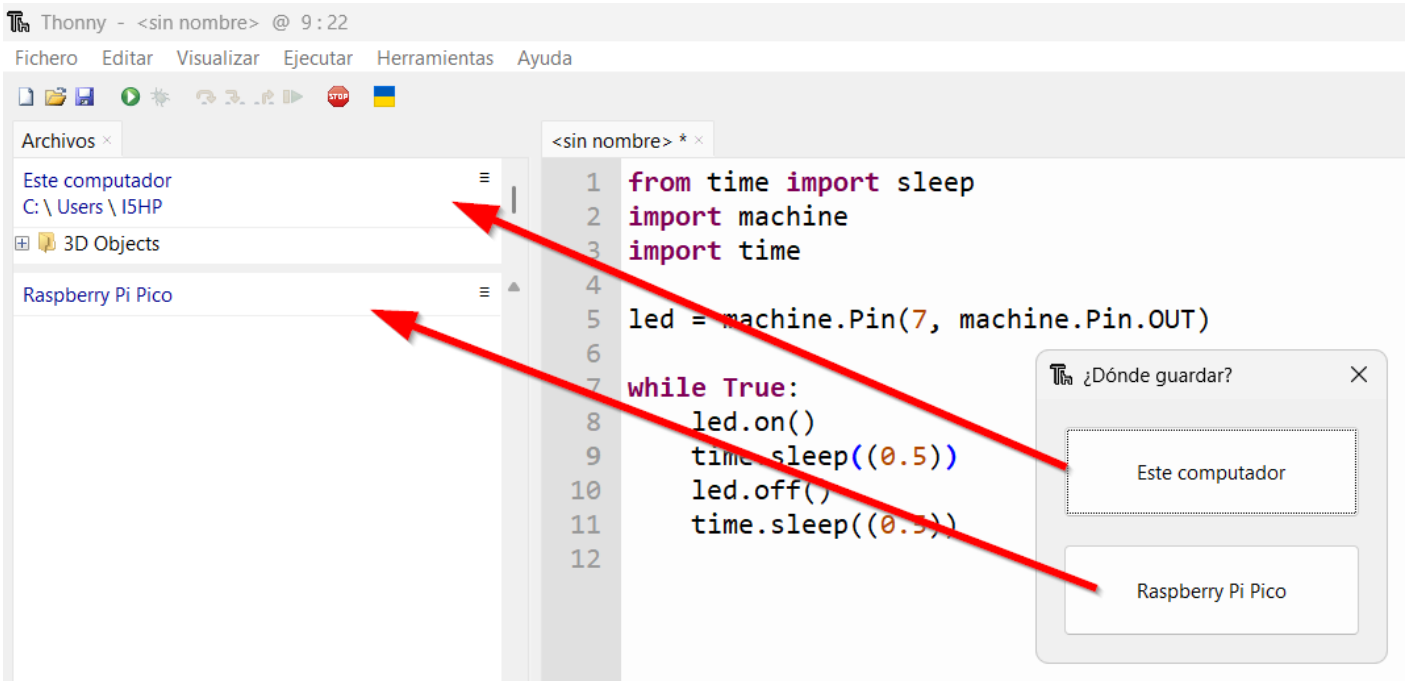
Entonces le damos a Instalar y lo instala en la unidad nueva que ha detectado, en el siguiente diálogo seleccionamos v**ariante Raspberry pico W:**



Si visualizamos la ventana de archivos



Podemos ver que a la hora de guardar nos pregunta si lo queremos guardar en el chip de PicoBricks o en tu ordenador

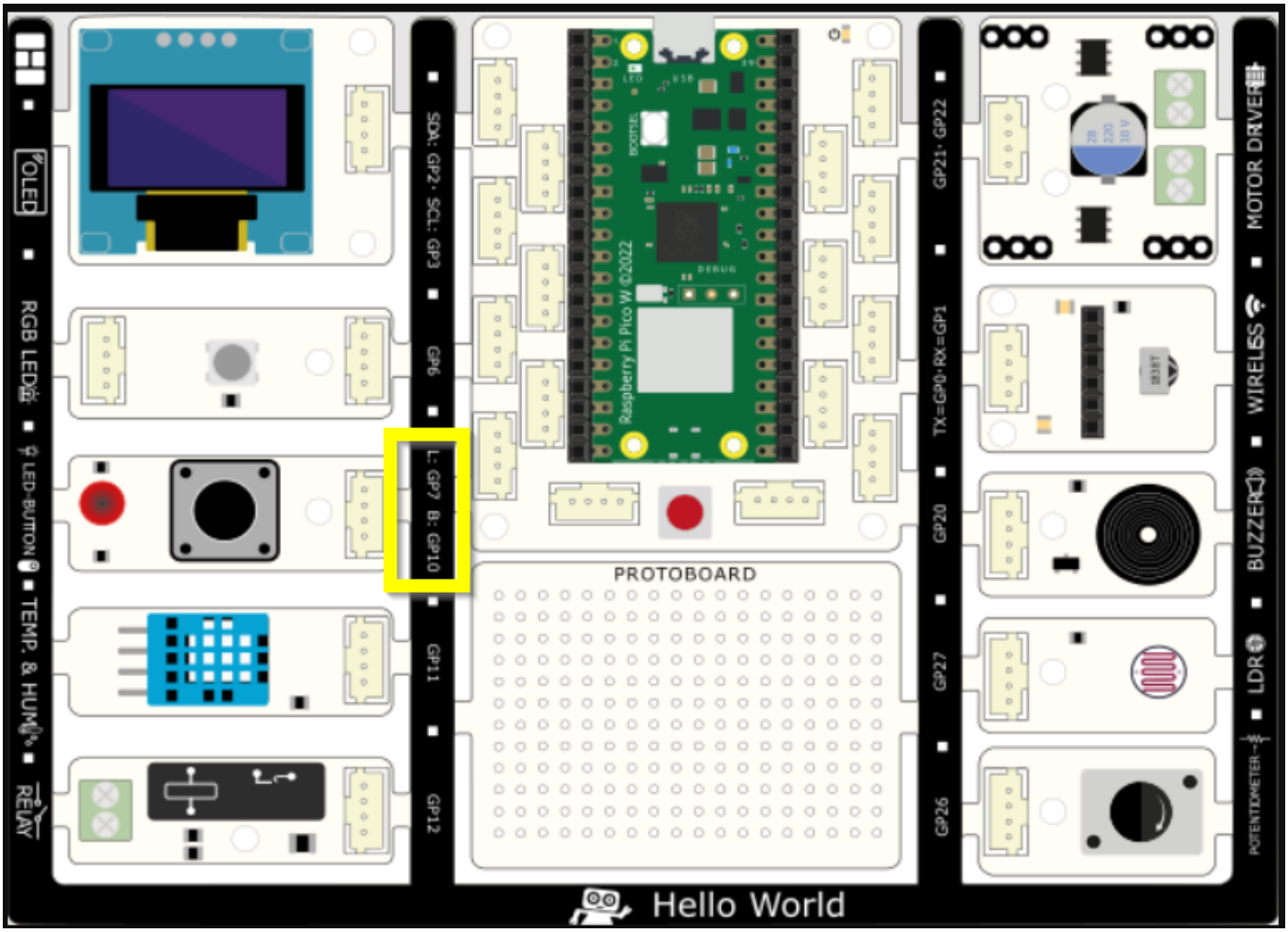


¿SABIAS QUE ...?

Si lo grabas en Raspberry Pi Pico con el nombre de **main.py**, entonces cuando enciendas el Picobricks, se ejecutará automáticamente sin necesidad de ningún ordenador

El primer programa con Python: Blink

El led rojo está en el pin GPI7 tal y como lo indica en la placa

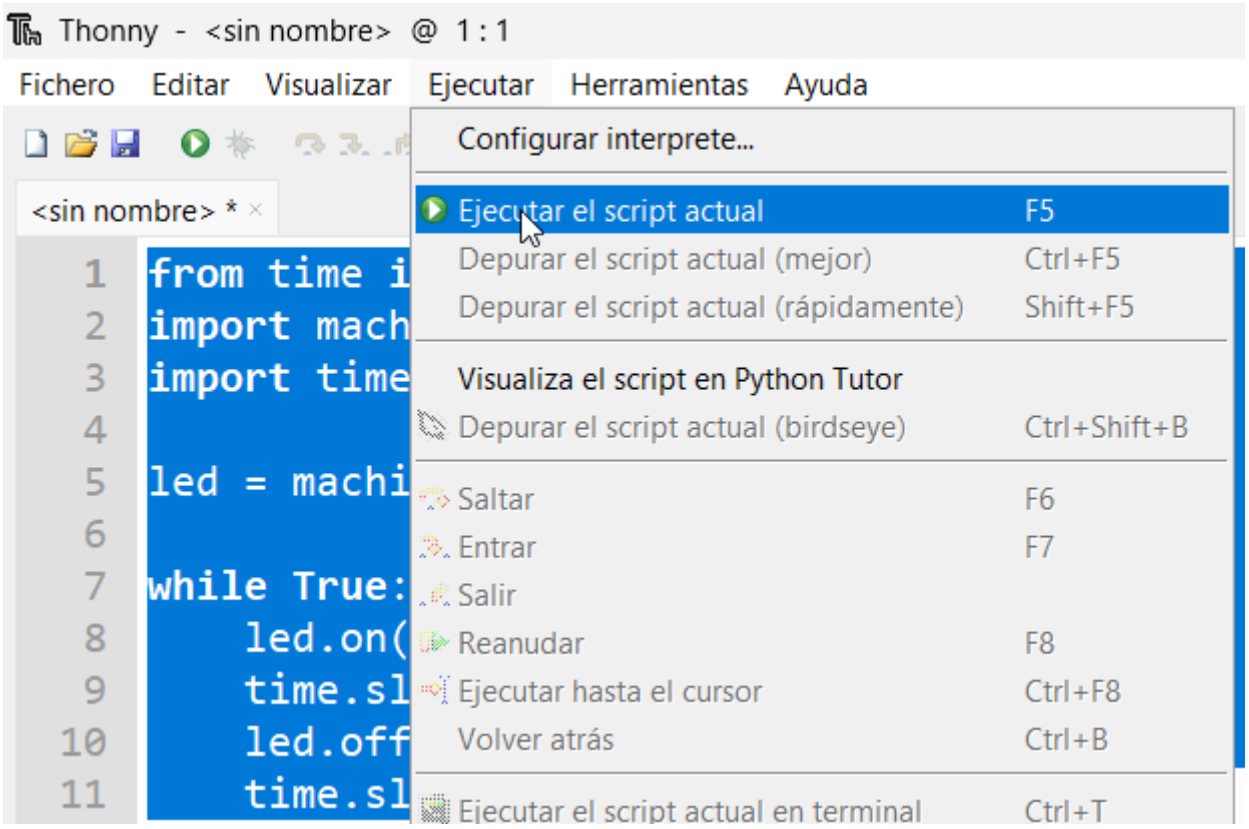


Luego ponemos en el Thonny el siguiente programa

```
from time import sleep
import machine
import time

led = machine.Pin(7, machine.Pin.OUT)
```

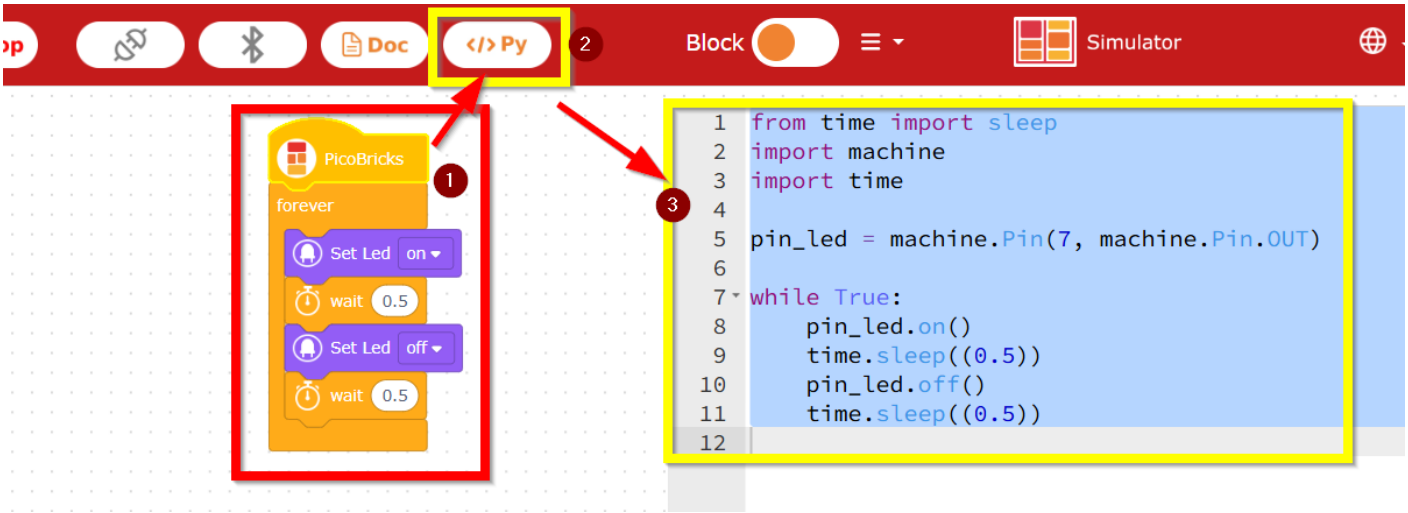
```
while True:
    led.on()
    time.sleep((0.5))
    led.off()
    time.sleep((0.5))
```

Ejecutamos con F5



y el led parpadea como estaba previsto

Otra forma de conseguir el programa es con la ventana de Python de PicoBlockly



Otra manera de ver el mismo programa, está en la página 25 del libro

https://picobricks.com/pages/projectbook

se encuentra el mismo código pero usando la instrucción

```
led.toggle()
```

https://drive.google.com/file/d/1PDql_GYyxcz68JqmQAGOLs0YE6SXgPCm/preview

Proyectos

Los mismos proyectos vistos con PicoBlockly se pueden hacer igual con código.

Repositorio Github

En la ruta https://github.com/Robotistan/PicoBricks/tree/main/Software/Activities los tienes listos los programas para copiar y pegar

1. - Blink
2. - Action-Reaction
3. - Autonomous Lighting
4. - Thermometer
5. - Graphic Monitor
6. - Dominate the Rhythm
7. - Show Your Reaction
8. - My Timer
9. - Alarm Clock
10. - Know Your Color

Libro Projectbook

Los tienes en este libro (en inglés) que lo puedes conseguir aquí

https://picobricks.com/pages/projectbook

- PROYECTO BLINK ver pag 25
- PROYECTO ACTION-REACTION ver pag 29
- PROYECTO Autonomous Lighting ver pag 35
- PROYECTO Thermometer ver pag 41
- PROYECTO Graphic Monitor ver pag 46
- PROYECTO Dominate the Rhythm ver pag 55
- PROYECTO Show Your Reaction ver pag 63
- PROYECTO My Timer ver pag 71
- PROYECTO Alarm Clock ver pag 81
- PROYECTO Know Your Color ver pag 90

> A diferencia de Microblocks, no los explica paso a paso, por lo que es mejor copiar y pegarde los repositorios de Github

https://drive.google.com/file/d/1PDql_GYyxcz68JqmQAGOLs0YE6SXgPCm/preview

Al no tener licencia CC no los podemos reproducir aquí en este tutorial

Un proyecto diferente: Encender y apagar led por wifi

En la lista de proyectos que propone PicoBricks sólo hay uno que usa la Wifi SmartHome, pero **no utiliza la wifi de Raspberry Pi** sino que utiliza un módulo wifi ESP8266 auxiliar.

Proponemos uno que no use elementos auxiliares

> **Enunciado**: Encender y apagar el led rojo conectado en GPI7 a través de una página web puesto en el servidor que se instala en la Raspberry

Solución

> La explicación del programa está en https://peppe8o.com/getting-started-with-wifi-on-raspberry-pi-pico-w-and-micropython/
> La fuente del programa en https://github.com/raspberrypi/pico-micropython-examples/blob/master/wireless/webserver.py

> **Recuerda** que tienes que poner los datos de tu wifi en las líneas 35 y 36

```
import socket
###################################33
import network, rp2
import time

def connectWiFi(ssid,password,country):
  rp2.country(country)
  wlan = network.WLAN(network.STA_IF)
  wlan.config(pm = 0xa11140)
```

```
  wlan.active(True)
  wlan.connect(ssid, password)
  # Wait for connect or fail
  max_wait = 10
  while max_wait > 0:
    if wlan.status() < 0 or wlan.status() >= 3:
      break
    max_wait -= 1
    print('waiting for connection...')
    time.sleep(1)

  # Handle connection error
  if wlan.status() != 3:
    raise RuntimeError('network connection failed')
  else:
    print('connected')
    status = wlan.ifconfig()
    print( 'ip = ' + status[0] )
  return status
##########################################################333
from machine import Pin

led = Pin(7, Pin.OUT)

country = 'ES'
ssid = 'pon aqui el nombre de tu wifi'
password = 'pon aqui el password de tu wifi'

wifi_connection = connectWiFi(ssid,password,country)
#################################################################
######33333
html = """<!DOCTYPE html>
<html>
<head> <title>Pico W</title> </head>
<body> <h1>Pico W</h1>
<p>Current status: %s</p>
<p><a href="http://"""+wifi_connection[0]+"""/light/on">Turn ON</a></p>
<p><a href="http://"""+wifi_connection[0]+"""/light/off">Turn OFF</a></p>
<p>by <a href="https://peppe8o.com">peppe8o.com</a></p>
</body>
```

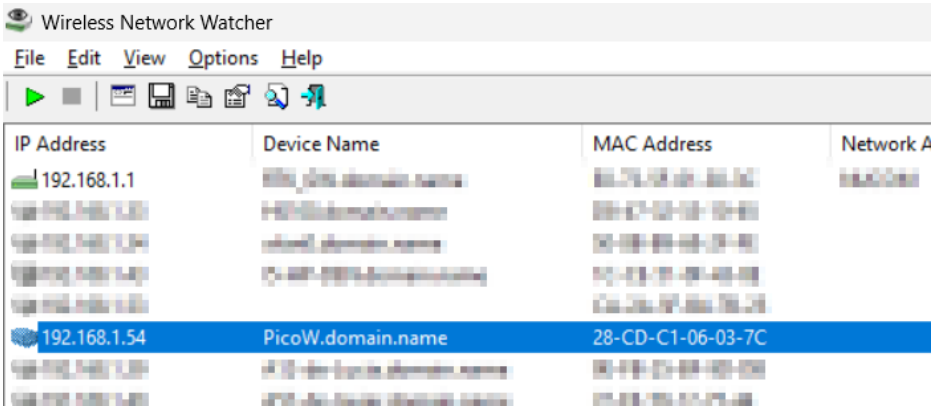
```
</html>
"""
#################################################################################

# Open socket
addr = socket.getaddrinfo('0.0.0.0', 80)[0][-1]
s = socket.socket()
s.bind(addr)
s.listen(1)

print('listening on', addr)

# Initialize LED status
led.value(0)
stateis = "LED is OFF"

# Listen for connections
while True:
  try:
    cl, addr = s.accept()
    print('client connected from', addr)
    request = cl.recv(1024)
    print(request)

    request = str(request)[0:50] # The [0:50] avoids getting the url directory from referer
    led_status = request.find('GET / HTTP')
    led_on = request.find('/light/on')
    led_off = request.find('/light/off')
    print( 'led on = ' + str(led_on))
    print( 'led off = ' + str(led_off))

    if led_status >0:
      print("LED status request") # No LED action

    if led_on >0:
      print("led on")
      led.value(1)
      stateis = "LED is ON"
```

```
  if led_off >0:
    print("led off")
    led.value(0)
    stateis = "LED is OFF"

  response = html % stateis

  cl.send('HTTP/1.0 200 OK\r\nContent-type: text/html\r\n\r\n')
  cl.send(response)
  cl.close()

except OSError as e:
  cl.close()
  s.close()
  print('connection closed')
```

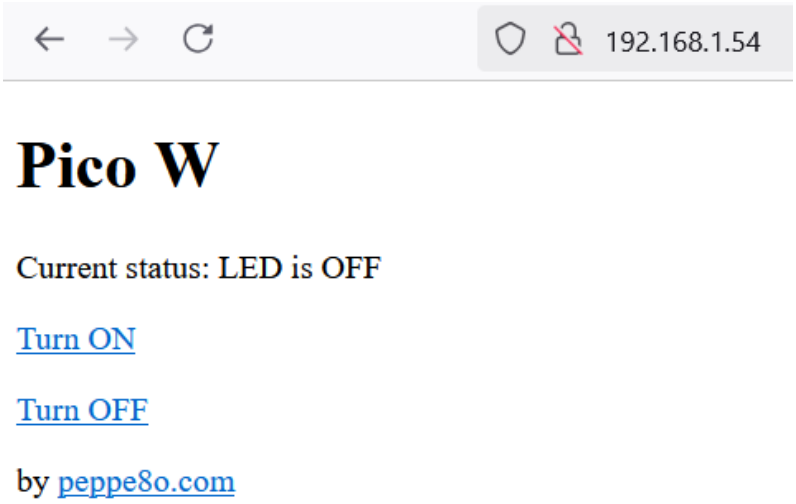
Ejecución del programa

Para encender y apagar el led tienes que entrar en la IP de la Raspberry Pi, puedes verlo en la ventana del puerto serie (cónsola) que puedes ver en el programa Thonny:

Otro truco es ejecutar un programa de rastreo de IPs como el Wireless Network Watcher y ver la IP de la Raspberry. O poner una IP estática ver aquí



Abrimos un navegador y ponemos la IP de la Raspberry en mi caso 192.168.1.54



https://www.youtube.com/embed/pRtXEg_cCCI

> Si os sale el error OSError: [Errno 98] EADDRINUSE es porque no se ha cerrado bien la conexión, desconectar PicoBrikcs y volverlo a conectar y solucionado

En cambio, el envío de Telegram no funciona

En la anterior página, PicoBricks hacía de servidor, alojaba una página web y desde el exterior, se llamaba a su página web para encender y apagar un led.

Al revés, es decir, la llamada de PicoBricks a una web externa **no funciona**

Esto sería útil para llamar a la API de Telegram y que Picobricks pudiese enviar información al usuario por Telegram

- Primero creando un bot de Telegram y consiguiendo su Token
- Segundo identificar nuestro ID de usuario a donde enviar el mensaje
- Tercero utilizar la instrucción urequest.get(laurl) de la librería urequests

> Pero según este foro la instrucción **urequest ya no funciona**. Lo hemos probado de muchas maneras y efectivamente. Si sabes cómo poder enviar a Telegram con Picobricks por favor ponte en contacto con nosotros en www.catedu.es en información y pondremos el código correcto.

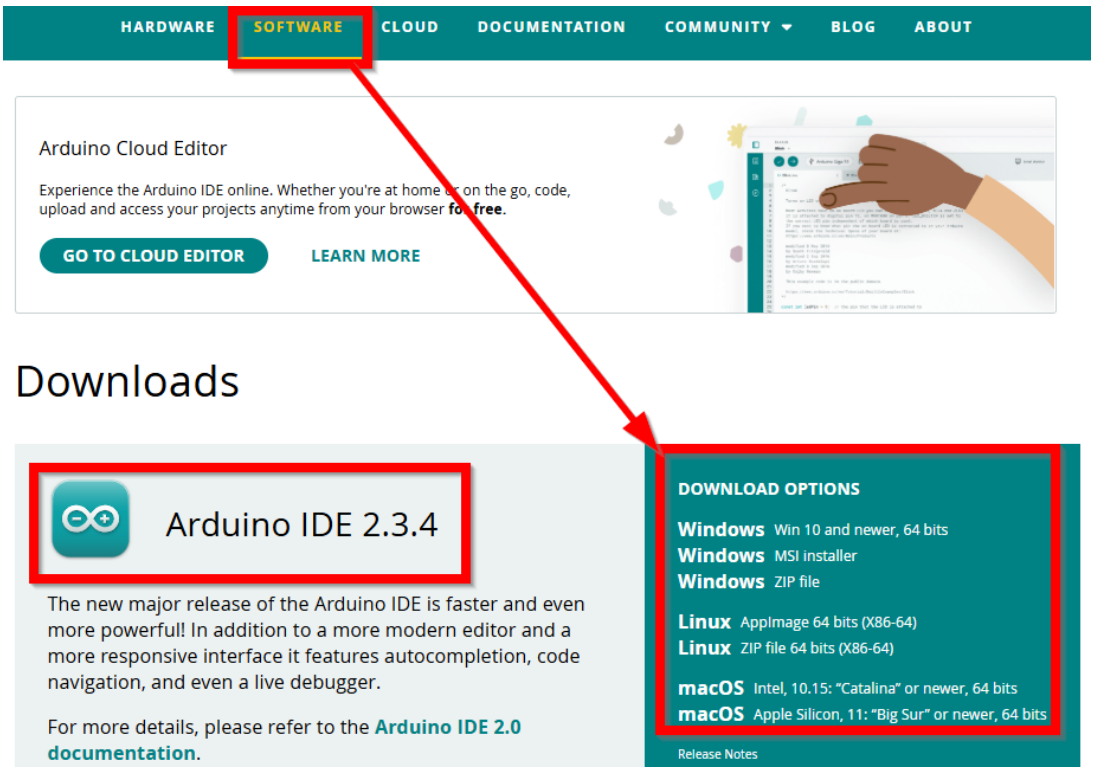
```
import urequests
import network
import time

TOKEN="----------"     ## Pon aquí el Token que sale de @BotFather
CHAT_ID="---------"    ## Pon aquí el ID del usuario de Telegram destinatario lo da @myidbot
SSID = '------------'  ## Pon aquí la red wifi
PASSWORD = '---------' ## Pon aquí la contraseña de la red wifi

def conectar_wifi():
    wlan = network.WLAN(network.STA_IF)
    wlan.active(True)
    wlan.connect(SSID, PASSWORD)
    print("Conectando a Wi-Fi...")
```

```
    while not wlan.isconnected():
        time.sleep(1)
    print("Conectado a Wi-Fi:", wlan.ifconfig())
    return True

def enviarmensaje (mensaje):

url="https://api.telegram.org/bot"+telegramBot+"/sendMessage?chat_id="+telegramChatId+"&text="+mensaj
e
    respuesta = urequests.get(url)
    print('Mensaje Enviado')
    return respuesta

########### PROGRAMA PRINCIPAL, Conexión Wi-Fi y envío del mensaje
conectar_wifi()
enviarmensaje("Prueba")  #Poner el mensaje a enviar, podría ser la temperatura, la humedad, el estado del
led,...
#############################################################
```

Arduino IDE

Conexión con Arduino IDE

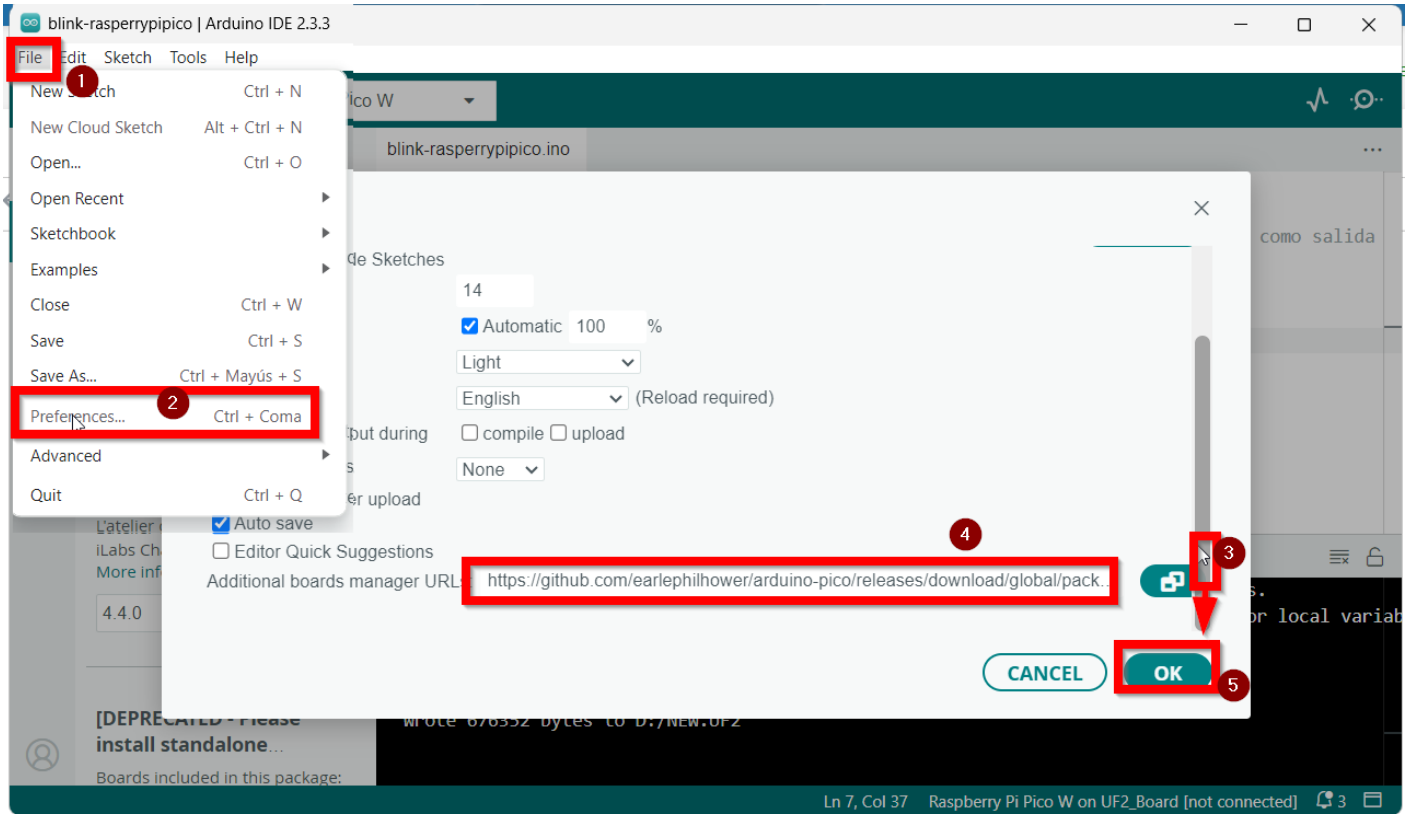
El software Arduino IDE lo puedes descargar en https://www.arduino.cc/



Una vez instalado vamos a ver cómo podemos programar con el software de Arduino nuestro Picobriks

> No sigas las instrucciones de Project Book aquí https://picobricks.com/pages/projectbook **NO funcionan**.
>
> Las siguientes instrucciones **sí que funcionan** y son de ***Bricogeek*** *Licencia CC-BY*
> *(hay muchas páginas con las mismas instrucciones pon en un buscador raspberry pico arduino ide)*

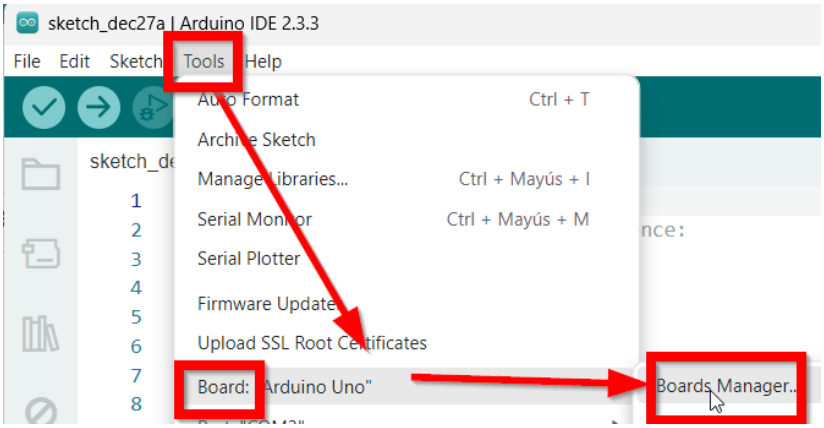
Nos vamos a **Archivo-Preferencias** y le decimos que ponga el siguiente directorio para las librerías :



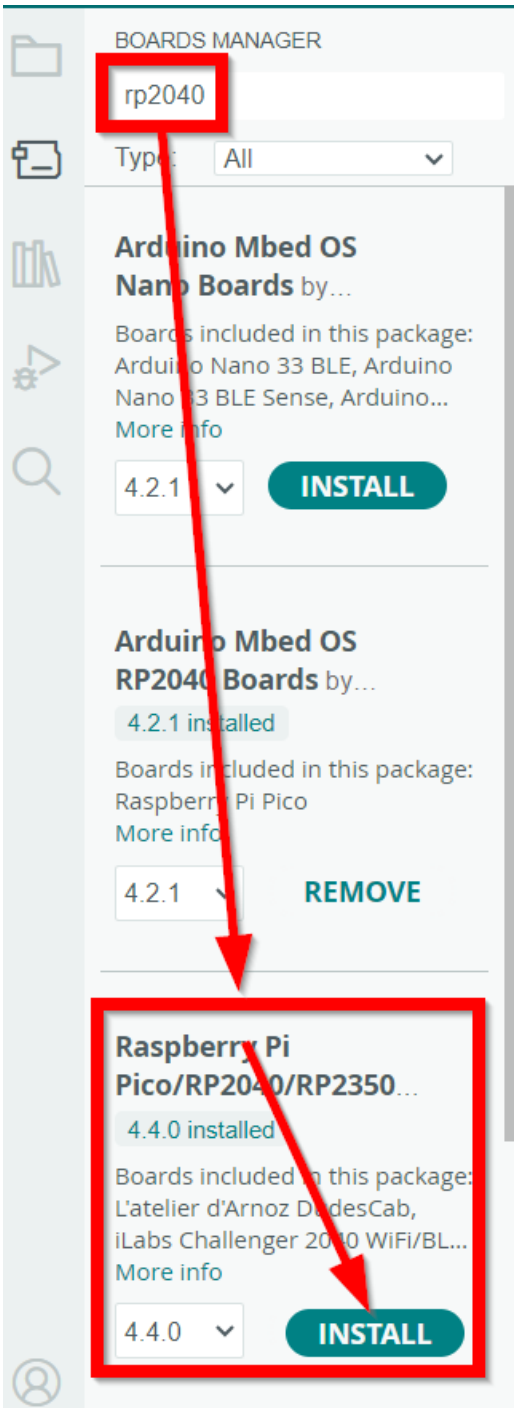
Esta es la URL que pegar para que cargue placas que no vienen por defecto :

```
https://github.com/earlephilhower/arduino-pico/releases/download/global/package_rp2040_index.json
```

Una vez puesto nos vamos a **Boards manager**..



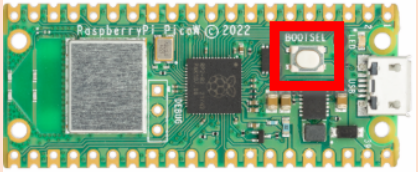
Y ponemos **RP2040** en el buscador, aparecerá este software, **Raspberry Pi Pico/RP2040** Lo instalamos



ATENCIÓN, poner PicoBricks en modo Bootloader

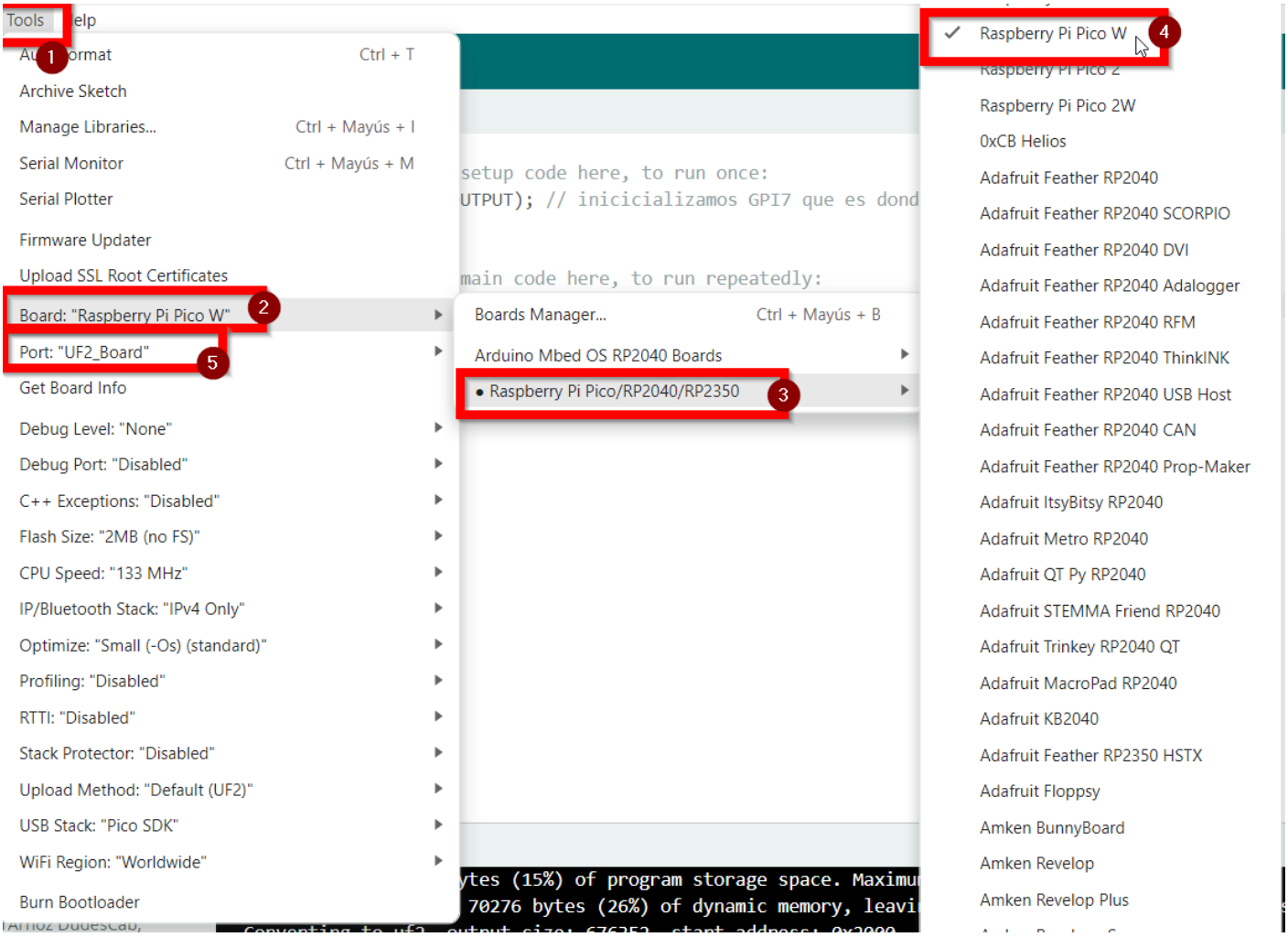
1.-Desconectamos PicoBricks de nuestro ordenador

2.- Apretamos el botón BOOTSEL **mientras** lo volvemos a conectar al puerto USB

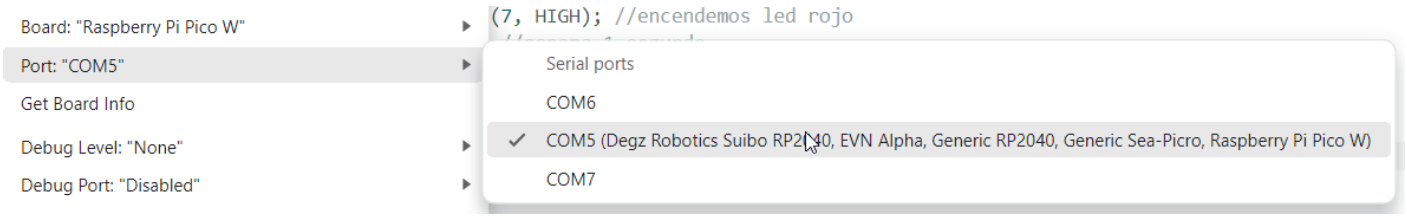


3.- Automáticamente aparecerá una nueva unidad de disco en nuestro ordenador (ya puedes soltar BOOTSEL)

Una vez puesto en modo Booloader seleccionamos la placa **Raspberry Pi Pico W** (4) y también seleccionamos el puerto (5) **UF2_Board**

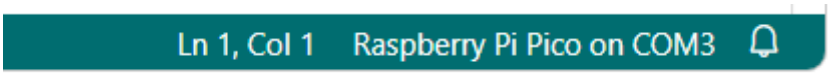


También puede salir otro tipo de puerto como este que dice que es el RP2040 Rasbberry Pi Pico W



.

En resumen tiene que salir abajo a la derecha que esta conectado

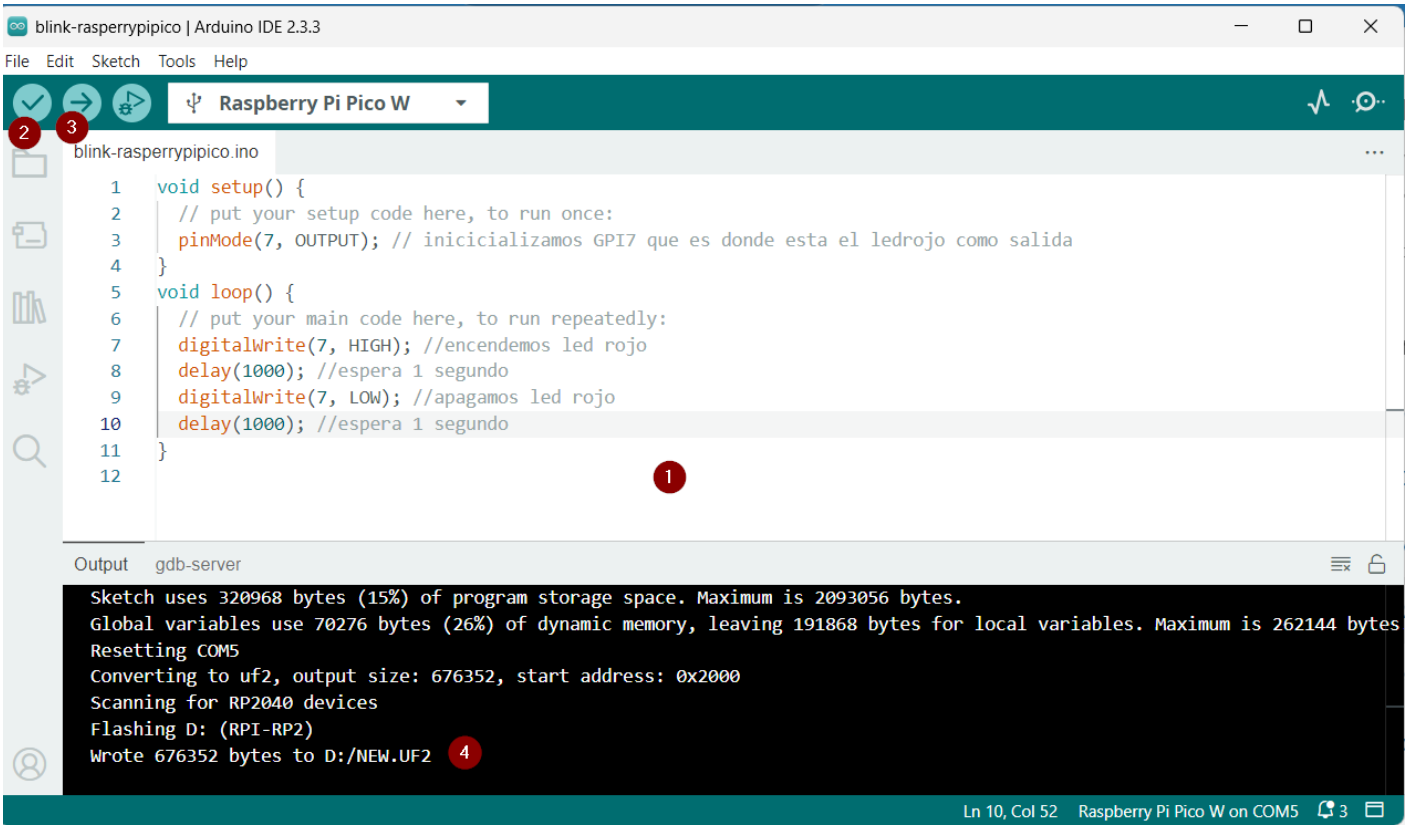


El primer programa con Arduino IDE: Blink

Entramos en Arduino IDE configurado según la página anterior (Board: Raspberry Pi y el puerto COM que corresonda) y pegamos el siguiente código:

```
void setup() {
  // put your setup code here, to run once:
  pinMode(7, OUTPUT); // inicicializamos GPI7 que es donde esta el ledrojo como salida
}
void loop() {
  // put your main code here, to run repeatedly:
  digitalWrite(7, HIGH); //encendemos led rojo
  delay(1000); //espera 1 segundo
  digitalWrite(7, LOW); //apagamos led rojo
  delay(1000); //espera 1 segundo
}
```

Ponemos el código en el área **1,** pulsamos **2** para compilar, y finalmente **3** para que lo suba a PicoBricks, cuando aparezca el mensaje **wrote...** en 4 ya podemos ver que el led rojo empieza a parpadear



> Por cierto, el programa **se queda cargado** pruébalo! desenchúfalo del ordenador, alimenta PicoBricks con un PowerBank o un cargador de móvil con el cabla USB y ¡¡ **sigue funcionando !!!**

Proyectos

Los mismos proyectos vistos con PicoBlockly se pueden hacer igual con código.

Repositorio Github

En la ruta https://github.com/Robotistan/PicoBricks/tree/main/Software/Activities los tienes listos los programas para copiar y pegar

1. - Blink
2. - Action-Reaction
3. - Autonomous Lighting
4. - Thermometer
5. - Graphic Monitor
6. - Dominate the Rhythm
7. - Show Your Reaction
8. - My Timer
9. - Alarm Clock
10. - Know Your Color

Libro Projectbook

Los tienes en este libro (en inglés) que lo puedes conseguir aquí

https://picobricks.com/pages/projectbook

- PROYECTO BLINK ver pag 25
- PROYECTO ACTION-REACTION ver pag 29
- PROYECTO Autonomous Lighting ver pag 35
- PROYECTO Thermometer ver pag 41
- PROYECTO Graphic Monitor ver pag 46
- PROYECTO Dominate the Rhythm ver pag 55
- PROYECTO Show Your Reaction ver pag 63
- PROYECTO My Timer ver pag 71
- PROYECTO Alarm Clock ver pag 81
- PROYECTO Know Your Color ver pag 90

> A diferencia de Microblocks, no los explica paso a paso, por lo que es mejor copiar y pegarde los repositorios de Github

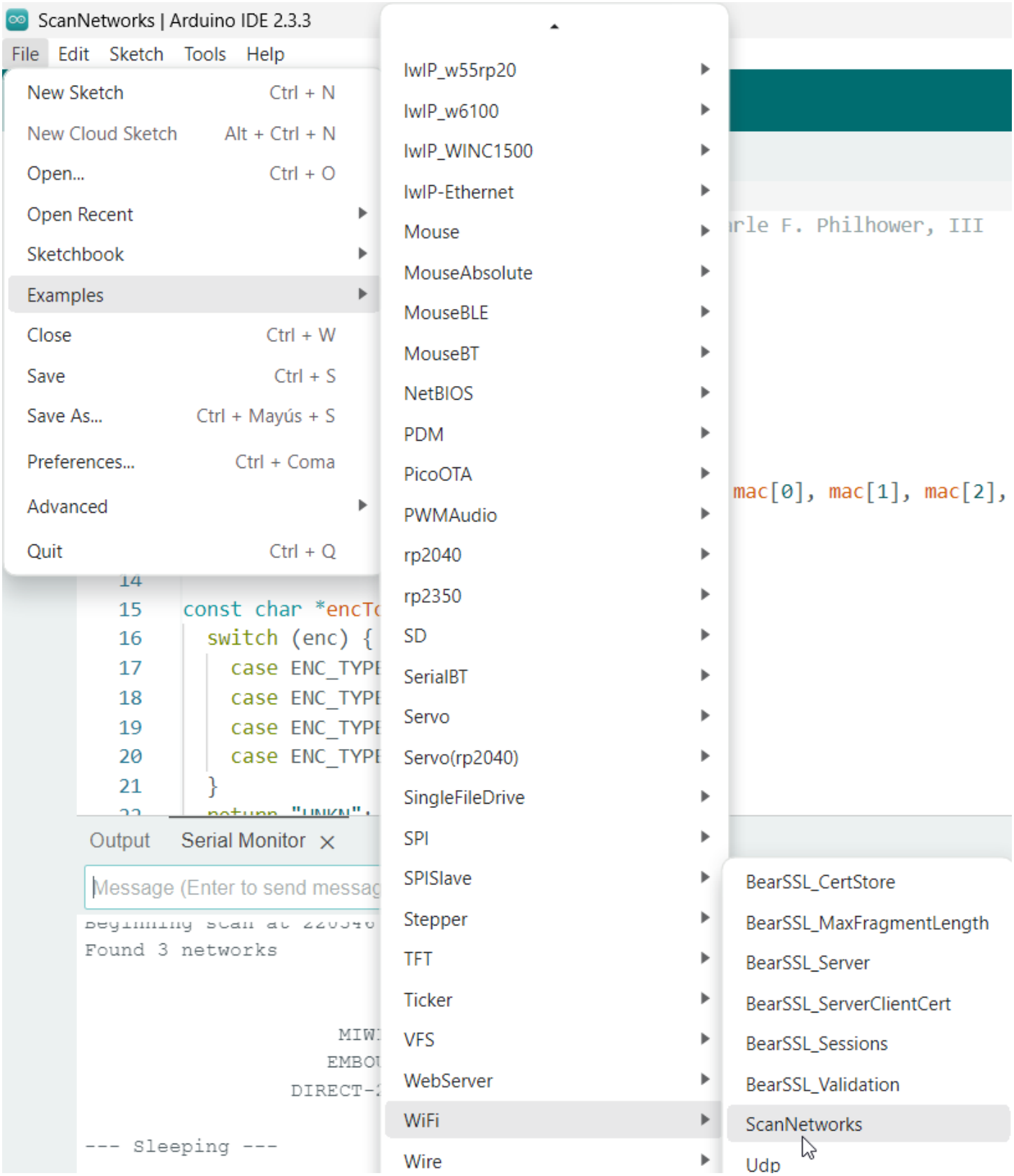
https://drive.google.com/file/d/1PDql_GYyxcz68JqmQAGOLs0YE6SXgPCm/preview

Al no tener licencia CC no los podemos reproducir aquí en este tutorial

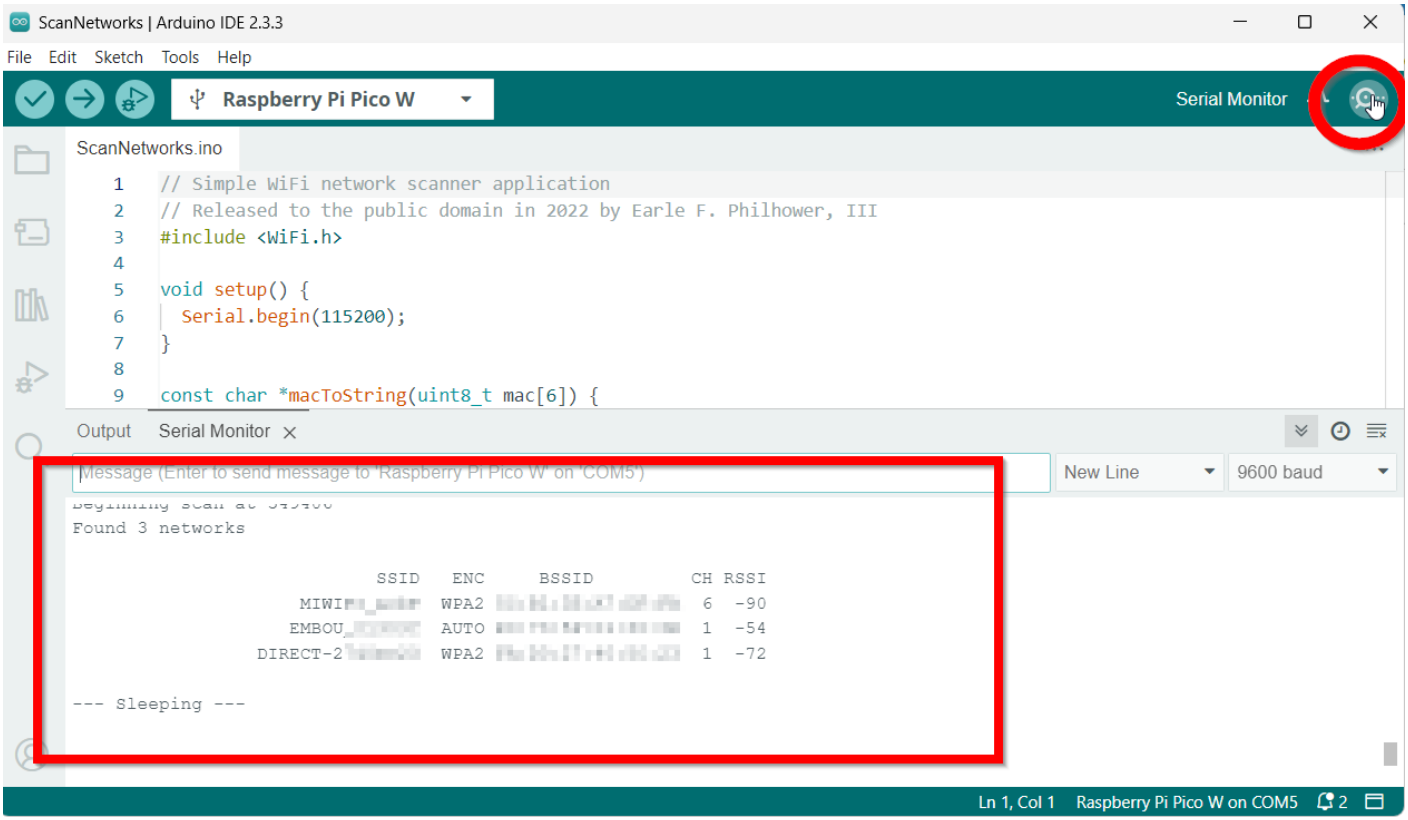
Proyectos con Wifi

ScanNetworks

Podemos ir a Ejemplos y vamos a ejecutar el **ScanNetworks** que no requiere contraseñas de wifi

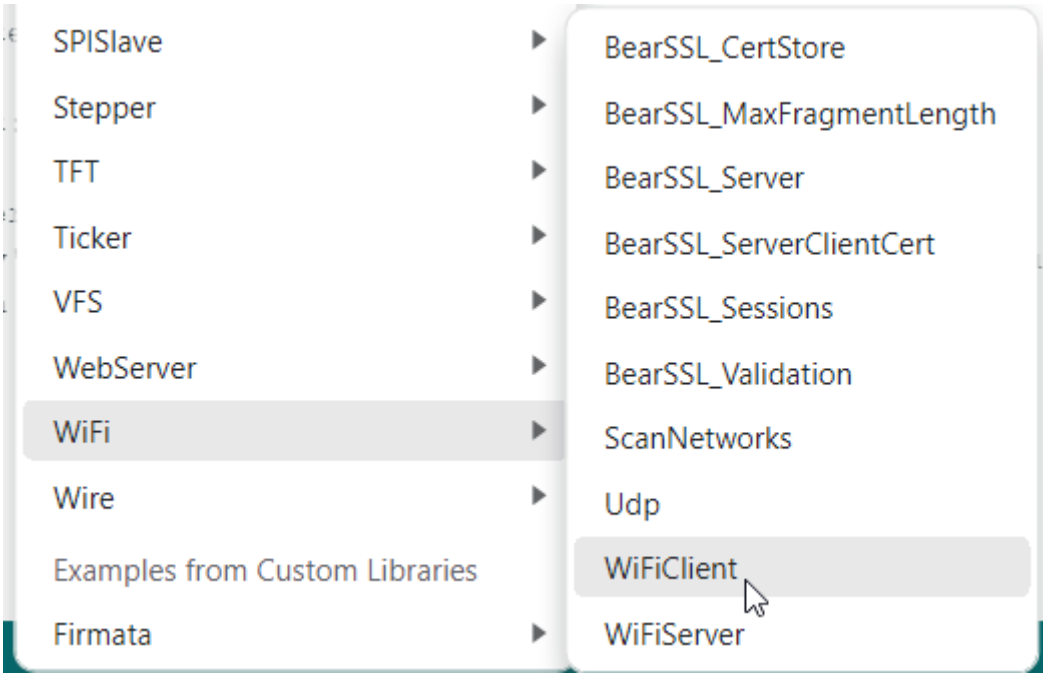


Al ejecutarlo nos sale en la ventana de monitor serie las wifis que encuentra



WifiClient

Si nos vamos a este ejemplo:



Tenemos que ir a las líneas 9 y 10 para poner nuestra wifi y contraseña:

```
#define STASSID "aquituwifi"
#define STAPSK "aqui tu contraseña"
```

Al ejecutarlo llama a esta dirección definida en las líneas 16 y 17

```
const char* host = "djxmmx.net";
const uint16_t port = 17;
```

Y sale **la cita del día** cada 5 minutos en el puerto serie

```
WiFi connected
IP address:
192.168.1.48
connecting to djxmmx.net:17
sending data to server
receiving from remote server
"Thomas wants to get it in quickly, and...Now there's a steal by Bird!
 Underneath to DJ, lays it in!
 1 second left, what a play by Bird!"
 - 1987 NBA Eastern Conference Finals

closing connection
connecting to djxmmx.net:17
sending data to server
receiving from remote server
"Oh oh oh oh oh ooh, why'd you have to go oh, away from all, me love, why you leave me, w-why you leave me?..."
 - Sean Kingston (Me Love)

closing connection
connecting to djxmmx.net:17
sending data to server
receiving from remote server
"I wanna run away, with you, cuz baby you're my everything..."
 - Frankie J. (Run Away)

closing connection
```

Encender y apagar un led

De https://dronebotworkshop.com/picow-arduino/ hemos conseguido este código, donde Raspberry actúa como cliente, pero esta pensado para encender y apagar el led integrado en la Rasberry Pi Pico W, así que le hemos añadido el Led rojo de PicoBrick que esta en GPI7

- En la línea 43 el pin 7 como salida **pinMode(7,OUTPUT);**
- En la línea 97 que encienda el pin 7 también **digitalWrite(7, HIGH);**
- En la línea 102 que apague el pin 7 también **digitalWrite(7, LOW);**
- Acuérdate de poner en las líneas 17 y 18 tu wifi

```
/*
  Pico W Web Interface Demo
  picow-web-control-demo.ino
  Web Interface & WiFi Connection
  Control the onboard LED with Pico W

  Adapted from ESP32 example by Rui Santos - https://randomnerdtutorials.com

  DroneBot Workshop 2022
  https://dronebotworkshop.com
*/

// Load Wi-Fi library
#include <WiFi.h>

// Replace with your network credentials
const char* ssid = "pontuwifi";
const char* password = "pontucontraseña";

// Set web server port number to 80
WiFiServer server(80);

// Variable to store the HTTP request
String header;

// Variable to store onboard LED state
String picoLEDState = "off";

// Current time
unsigned long currentTime = millis();
// Previous time
unsigned long previousTime = 0;
// Define timeout time in milliseconds (example: 2000ms = 2s)
const long timeoutTime = 2000;
```

```
void setup() {

  // Start Serial Monitor
  Serial.begin(115200);

  // Initialize the LED as an output
  pinMode(LED_BUILTIN, OUTPUT);
  pinMode(7,OUTPUT);//initialize digital pin 7 as an output

  // Set LED off
  digitalWrite(LED_BUILTIN, LOW);

  // Connect to Wi-Fi network with SSID and password
  WiFi.begin(ssid, password);

  // Display progress on Serial monitor
  while (WiFi.status() != WL_CONNECTED) {
    delay(500);
    Serial.print(".");
  }

  // Print local IP address and start web server
  Serial.println("");
  Serial.print("WiFi connected at IP Address ");
  Serial.println(WiFi.localIP());

  // Start Server
  server.begin();
}

void loop() {

  WiFiClient client = server.available();   // Listen for incoming clients

  if (client) {                             // If a new client connects,
    currentTime = millis();
    previousTime = currentTime;
    Serial.println("New Client.");          // print a message out in the serial port
    String currentLine = "";                // make a String to hold incoming data from the client
    while (client.connected() && currentTime - previousTime <= timeoutTime) {  // loop while the client's
```

```
connected
      currentTime = millis();
      if (client.available()) {             // if there's bytes to read from the client,
        char c = client.read();             // read a byte, then
        Serial.write(c);                    // print it out the serial monitor
        header += c;
        if (c == '\n') {                    // if the byte is a newline character
          // if the current line is blank, you got two newline characters in a row.
          // that's the end of the client HTTP request, so send a response:
          if (currentLine.length() == 0) {
            // HTTP headers always start with a response code (e.g. HTTP/1.1 200 OK)
            // and a content-type so the client knows what's coming, then a blank line:
            client.println("HTTP/1.1 200 OK");
            client.println("Content-type:text/html");
            client.println("Connection: close");
            client.println();

            // Switch the LED on and off
            if (header.indexOf("GET /led/on") >= 0) {
              Serial.println("LED on");
              picoLEDState = "on";
              digitalWrite(LED_BUILTIN, HIGH);
              digitalWrite(7, HIGH);
            } else if (header.indexOf("GET /led/off") >= 0) {
              Serial.println("LED off");
              picoLEDState = "off";
              digitalWrite(LED_BUILTIN, LOW);
              digitalWrite(7, LOW);
            }

            // Display the HTML web page
            client.println("<!DOCTYPE html><html>");
            client.println("<head><meta name=\"viewport\" content=\"width=device-width, initial-scale=1\">");
            client.println("<link rel=\"icon\" href=\"data:,\">");

            // CSS to style the on/off buttons
            client.println("<style>html { font-family: Helvetica; display: inline-block; margin: 0px auto; text-align:
center;}");
            client.println(".button { background-color: #4CAF50; border: none; color: white; padding: 16px 40px;");
            client.println("text-decoration: none; font-size: 30px; margin: 2px; cursor: pointer;}");
```

```
            client.println(".button2 {background-color: #F23A3A;}</style></head>");

            // Web Page Heading
            client.println("<body><h1>Pico W LED Control</h1>");

            // Display current state, and ON/OFF buttons for Onboard LED
            client.println("<p>Onboard LED is " + picoLEDState + "</p>");

            // Set buttons
            if (picoLEDState == "off") {

              //picoLEDState is off, display the ON button
              client.println("<p><a href=\"/led/on\"><button class=\"button\">ON</button></a></p>");
            } else {

              //picoLEDState is on, display the OFF button
              client.println("<p><a href=\"/led/off\"><button class=\"button button2\">OFF</button></a></p>");
            }

            client.println("</body></html>");

            // The HTTP response ends with another blank line
            client.println();
            // Break out of the while loop
            break;
          } else { // if you got a newline, then clear currentLine
            currentLine = "";
          }
        } else if (c != '\r') {  // if you got anything else but a carriage return character,
          currentLine += c;      // add it to the end of the currentLine
        }
      }
    }
    // Clear the header variable
    header = "";
    // Close the connection
    client.stop();
    Serial.println("Client disconnected.");
    Serial.println("");
}
```

```
}
```

Al ejecutarlo, nos aparece por el puerto serie la IP que se ha conectado:

Entramos en un navegador en un ordenador conectado a la misma wifi en la dirección 192.168.1.48 y este es el resultado:

https://www.youtube.com/embed/haE4GdOd4zo

¿Y con un servo?

Si tienes un servo puedes conectarlo y también puedes utilizar el código que aparece en De https://dronebotworkshop.com/picow-arduino/ con las conexiones que indican

Créditos

Capitulo PicoBlocky Extraido de Pico Bricks IDE Book

- Autores
  - Contents: Mustafa Kemal Avcı, Abdullah Kaya, Selim Gayretli
  - Translation: Naze Gizem Özer
  - Design: Ahmet Gürsu, Elanur Tokalak
  - Pico Bricks Developer Team
    - Yasir Çiçek - Project Manager
    - Yusuf Gündoğdu - Software Developer
    - Mehmet Suat Morkan - Chief Developer
    - Mehmet Ali Dağ - Hardware Developer

- Descarga del libro https://picobricks.com/pages/idebook

Resto de capítulos: Autor Javier Quintana CATEDU Diciembre 2024

Cualquier observación o detección de error en soporte.catedu.es

GOBIERNO DE ARAGON
Departamento de Educación, Cultura y Deporte

CATEDU
CENTRO ARAGONÉS de TECNOLOGÍAS para la EDUCACIÓN

CC BY NC SA