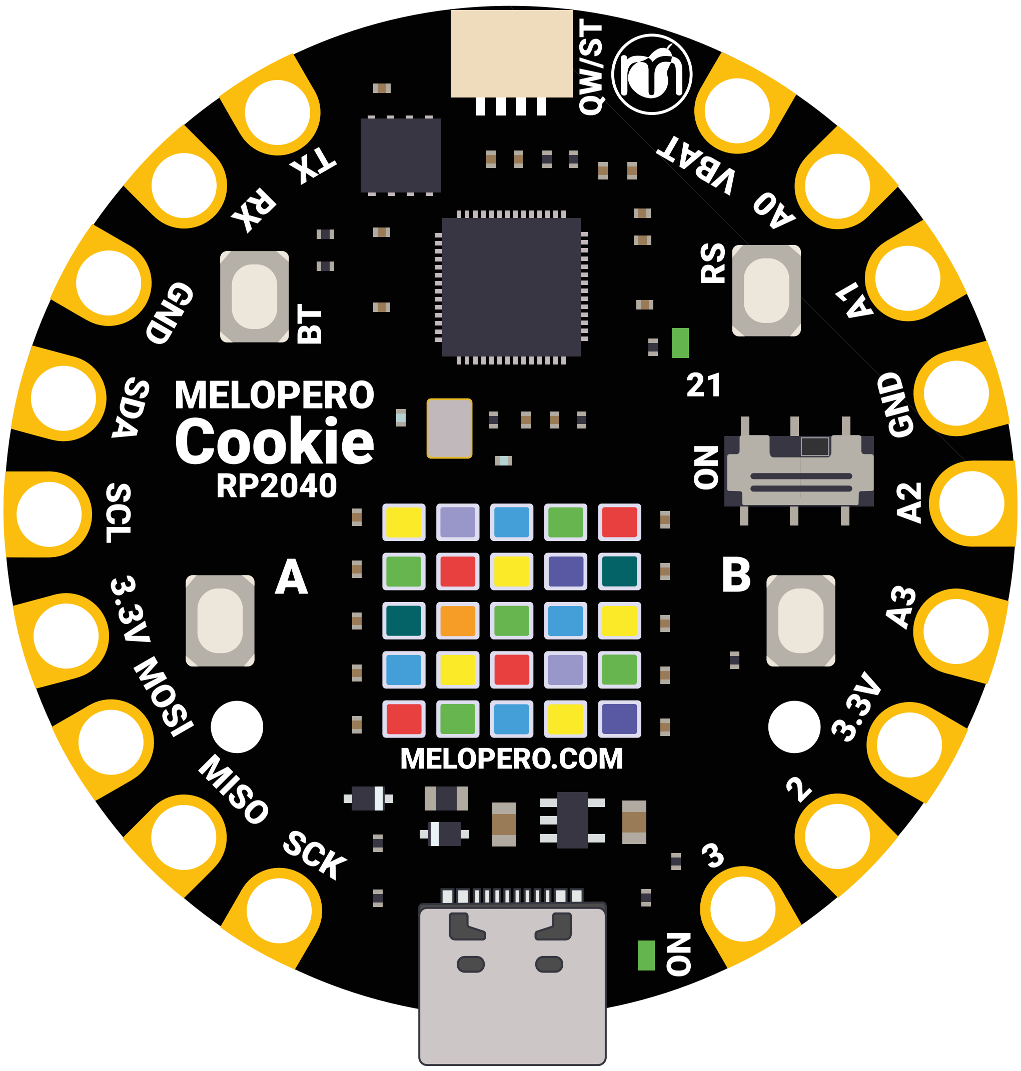
Getting started with **Melopero Cookie RP2040**



MELOPERO®
WWW.MELOPERO.COM

RoHS Compliant

This guide is constantly updated with corrections and new content.
When a new version is released, we also update the version number:

Version **2.0.0**
12 April 2023

Contents

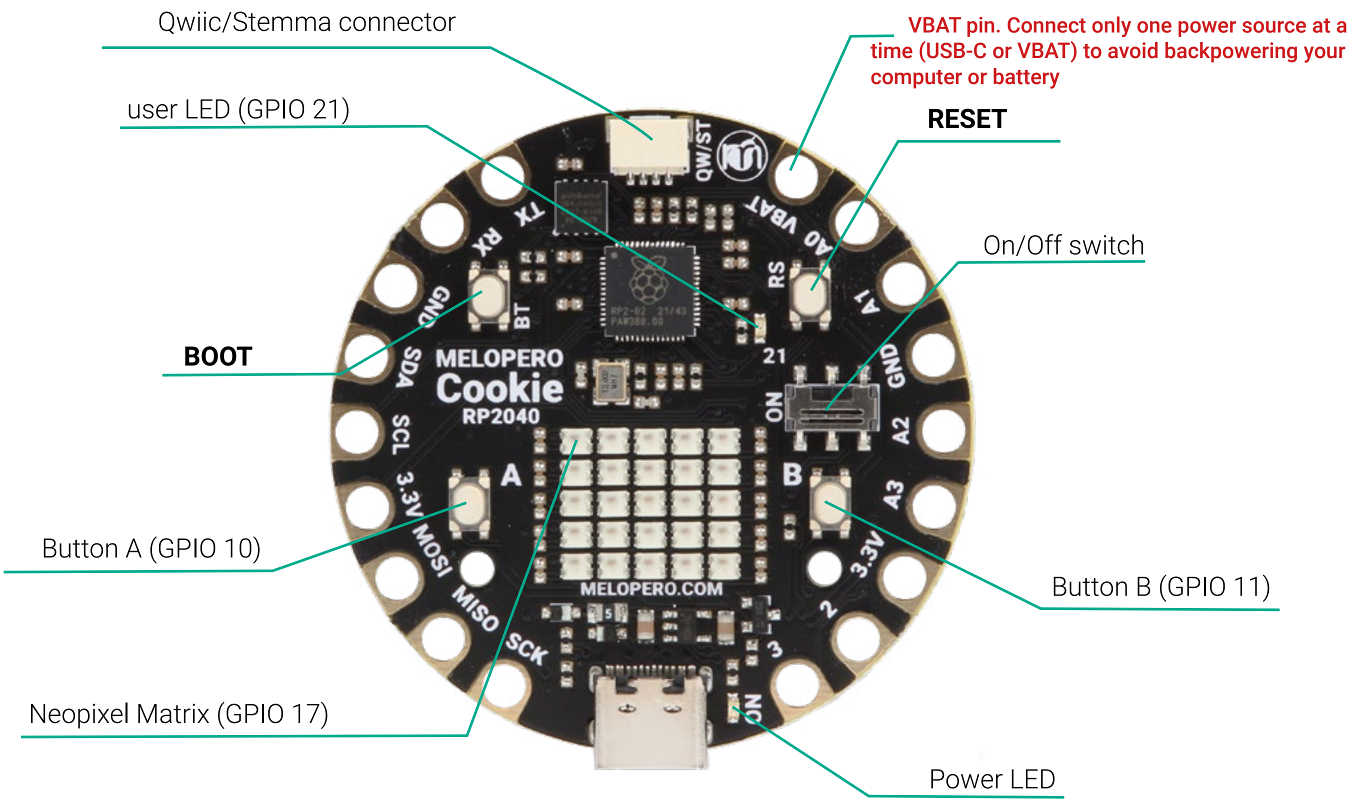
1. Hardware overview

The Melopero Cookie RP2040 is a development board based on the Raspberry Pi RP2040 micro-controller, programmable in C/C++, MicroPython, CircuitPython and with the Arduino IDE.
The board features:

- 8MB of FLASH Memory
- Reset and boot buttons (no need to detach/attach the board to enter boot mode)
- Qwiic/Stemma QT connector for attaching lots of Melopero, Adafruit and SparkFun sensors
- USB-C connector for powering, programming and charging
- Green user LED on pin 21
- WS2812 (aka NeoPixel) Matrix Display
- Green Power LED
- 2 mounting holes
- Programmable in C/C++, MicroPython, CircuitPython and Arduino language.

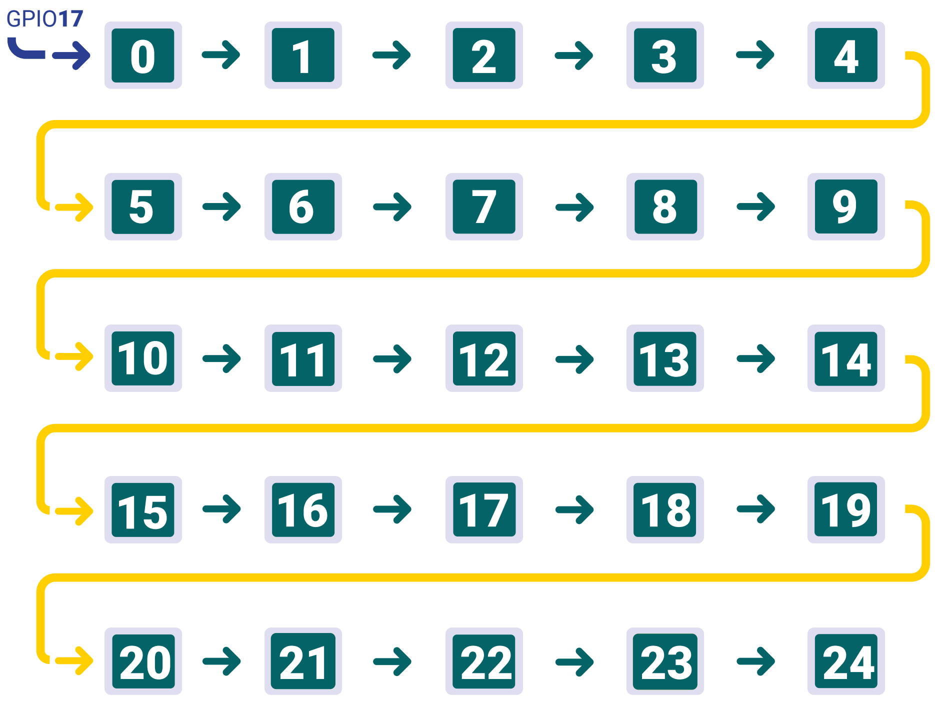
Dimensions: 50.8mm x 50.8mm

The Melopero Cookie RP2040 is in conformity with the requirements of the RoHS and EMC EU and UK Directives (see European and UK Declarations of Conformity for more information).


Qwiic/Stemma connector
user LED (GPIO 21)
VBAT pin. Connect only one power source at a time (USB-C or VBAT) to avoid backpowering your computer or battery
RESET
On/Off switch
BOOT
Button A (GPIO 10)
Button B (GPIO 11)
Neopixel Matrix (GPIO 17)
Power LED

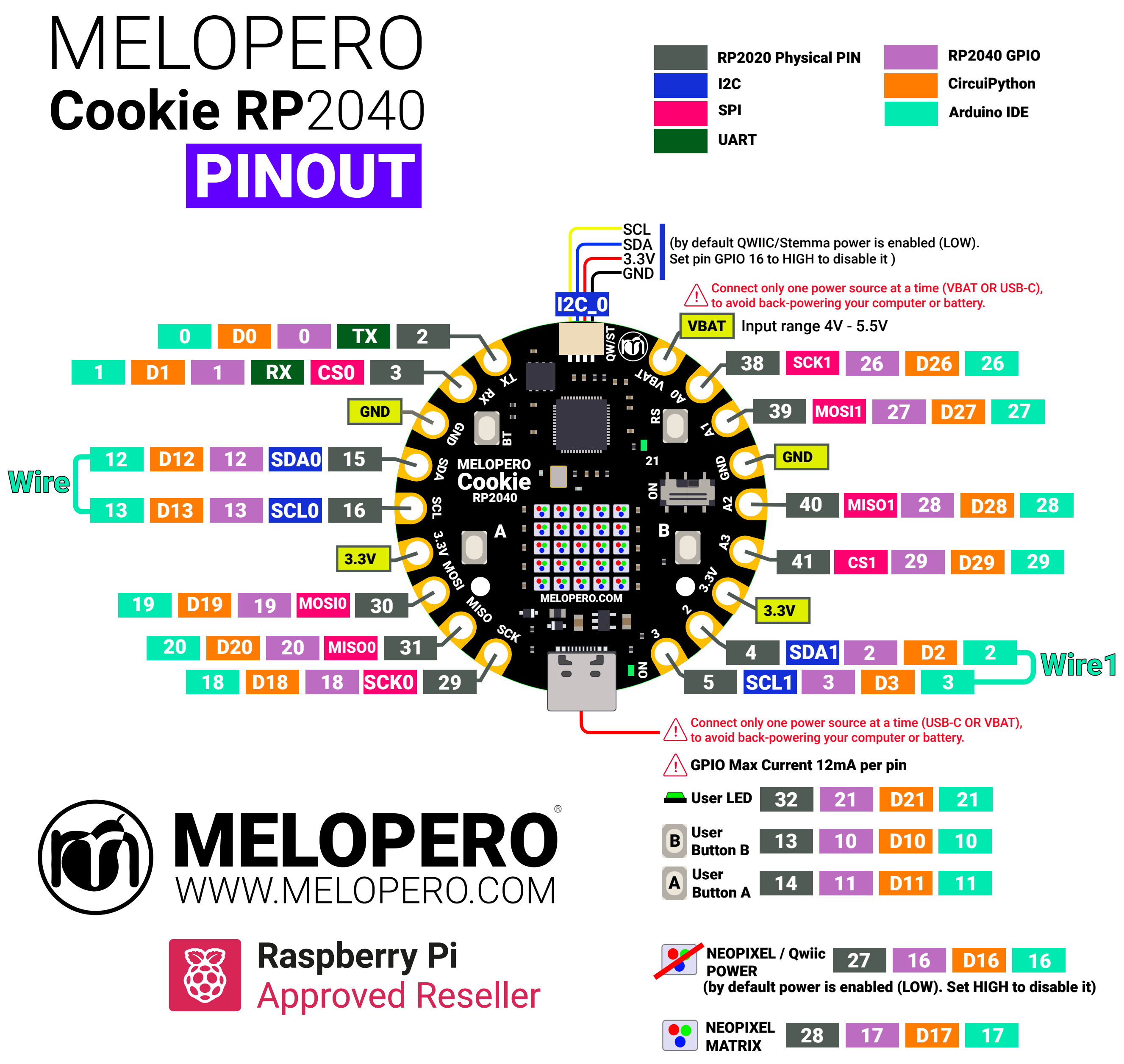
1.1 Neopixels (WS2812) matrix display

Melopero Cookie RP2040 is equipped with a 5x5 neopixels matrix display, which can be accessed through pin GPIO17.
Neopixels are hard-wired in a row-major configuration, as shown in the following picture:



We developed libraries in Micropython, CircuitPython, Arduino Language and C++ for controlling the display, A-B buttons and user LED.
These libraries are detailed in the relative sections.

2. Pinout


MELOPERO
Cookie RP2040
PINOUT
RP2020 Physical PIN
I2C
SPI
UART
RP2040 GPIO
CircuiPython
Arduino IDE
SCL
SDA
3.3V
GND
(by default QWIIC/Stemma power is enabled (LOW).
Set pin GPIO 16 to HIGH to disable it)
I2C_0
Connect only one power source at a time (VBAT OR USB-C),
to avoid back-powering your computer or battery.
VBAT Input range 4V - 5.5V
0 D0 0 TX 2
1 D1 1 RX CS0 3
GND
12 D12 12 SDA0 15
13 D13 13 SCL0 16
Wire
3.3V
19 D19 19 MOSI0 30
20 D20 20 MISO0 31
18 D18 18 SCK0 29
38 SCK1 26 D26 26
39 MOSI1 27 D27 27
GND
40 MISO1 28 D28 28
41 CS1 29 D29 29
3.3V
4 SDA1 2 D2 2
5 SCL1 3 D3 3
Wire1
Connect only one power source at a time (USB-C OR VBAT),
to avoid back-powering your computer or battery.
GPIO Max Current 12mA per pin
User LED 32 21 D21 21
User Button B 13 10 D10 10
User Button A 14 11 D11 11
NEOPIXEL / Qwiic POWER 27 16 D16 16
(by default power is enabled (LOW). Set HIGH to disable it)
NEOPIXEL MATRIX 28 17 D17 17
MELOPERO
WWW.MELOPERO.COM
Raspberry Pi
Approved Reseller

The pinout chart above is useful to quickly find the right name of a specific pin, depending on the language and IDE. As example, note how the pins labeled as 28 and 29 change name depending on which platform you are using to program the board:

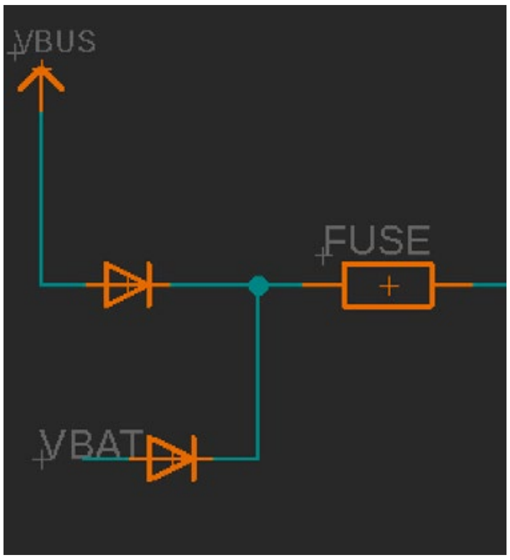
- CircuitPython refer to those pins as D28 and D29
- In the Arduino IDE they are 28 and 29 (check the green labels)
- For use with MicroPython, they are called with the number you find on the GPxx labels (only the number)

3. How to power the Cookie RP2040

There are two input connectors that can be used, **one at a time and not together**, to supply power to the Cookie RP2040:

1. USB-C connector (also used for programming the board)
2. VBAT pad

The applied input voltage must be in the range 4-5.5V.
Both inputs are protected with diodes and a resetable fuse limits the input current to 500mA:

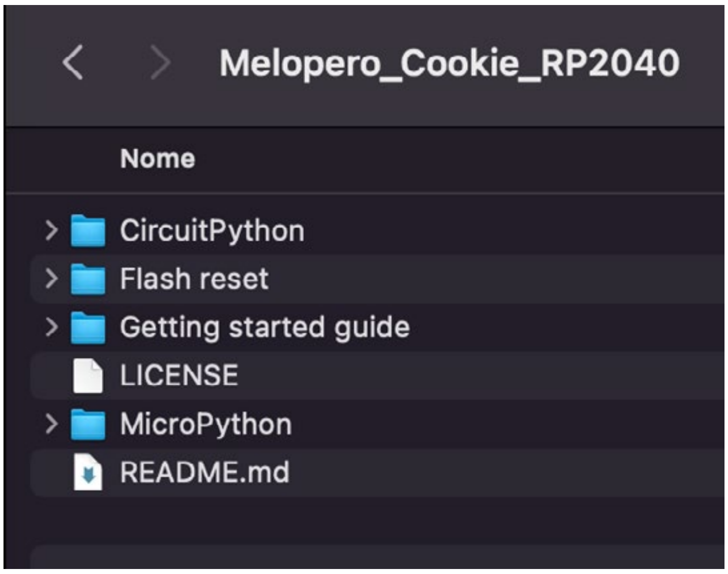


WARNING

Despite the presence of diodes, we strongly reccomend to power only one of the above connectors at a time, for best safety and to avoid backpowering your battery or computer

4. Cookie RP2040 software pack

The Cookie software pack contains all the files needed to install MicroPython, CircuitPython, along with C++ library files, a Getting started guide, flash reset file and examples.
The Arduino library can be downloaded directly from the Arduino IDE library manager (see the relevant section).



Melopero_Cookie_RP2040

Nome

- CircuitPython
- Flash reset
- Getting started guide
- LICENSE
- MicroPython
- README.md

4.1 Download the software pack

First of all, downlad the zip file with all the software and demo code for your Cookie:

www.melopero.com/Melopero_Cookie_RP2040.zip

5. Install MicroPython

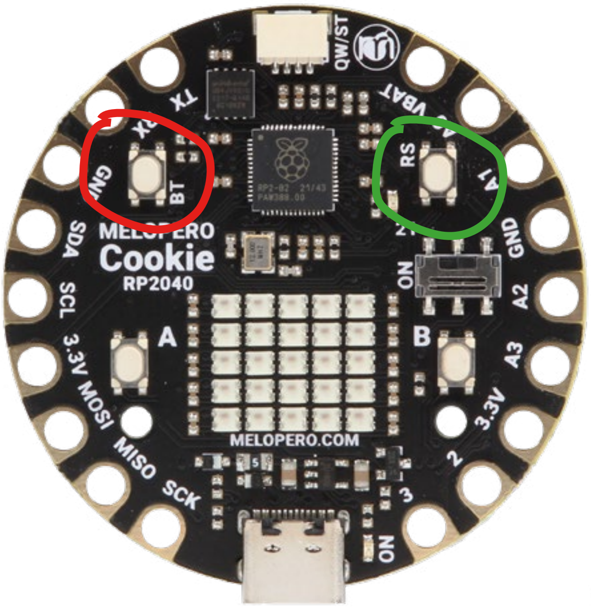
MicroPython is a lean and efficient implementation of the Python 3 programming language that includes a small subset of the Python standard library and is optimised to run on microcontrollers and in constrained environments.
MicroPython is a full Python compiler and runtime that runs on the bare-metal. You get an interactive prompt (the REPL) to execute commands immediately, along with the ability to run and import scripts from the built-in filesystem. The REPL has history, tab completion, auto-indent and paste mode for a great user experience.

5.1 Download MicroPython

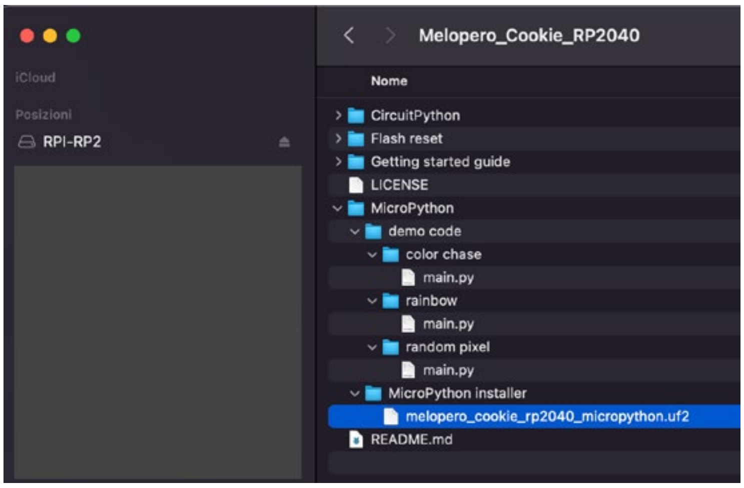
After downloading the latest version of the **Cookie software pack** (see chapter 3 of this guide), activate the bootloader mode on your board and copy the micropython installer .uf2 file into it.
We have packed the Cookie RP2040 library into the installer so no additional installation is required.

To activate the bootloader mode, when the Cookie is already connected to your computer's USB port, press and hold the BOOT / BT button (circled in red in the image below), then press and release the reset button (circled in green). Continue holding the BOOT / BOOTSEL button until the RPI-RP2 drive appears.

You can also start with your board unplugged from USB, press and hold the BOOTSEL button on your Cookie RP2040. While holding the button, connect the other end of the USB cable to the



Drag (or copy and paste) the melopero_cookie_rp2040_micropython.uf2 file to RPI-RP2.
The RPI-RP2 drive will disappear and you'll be ready to write your micropython code using Thonny (see next chapter).



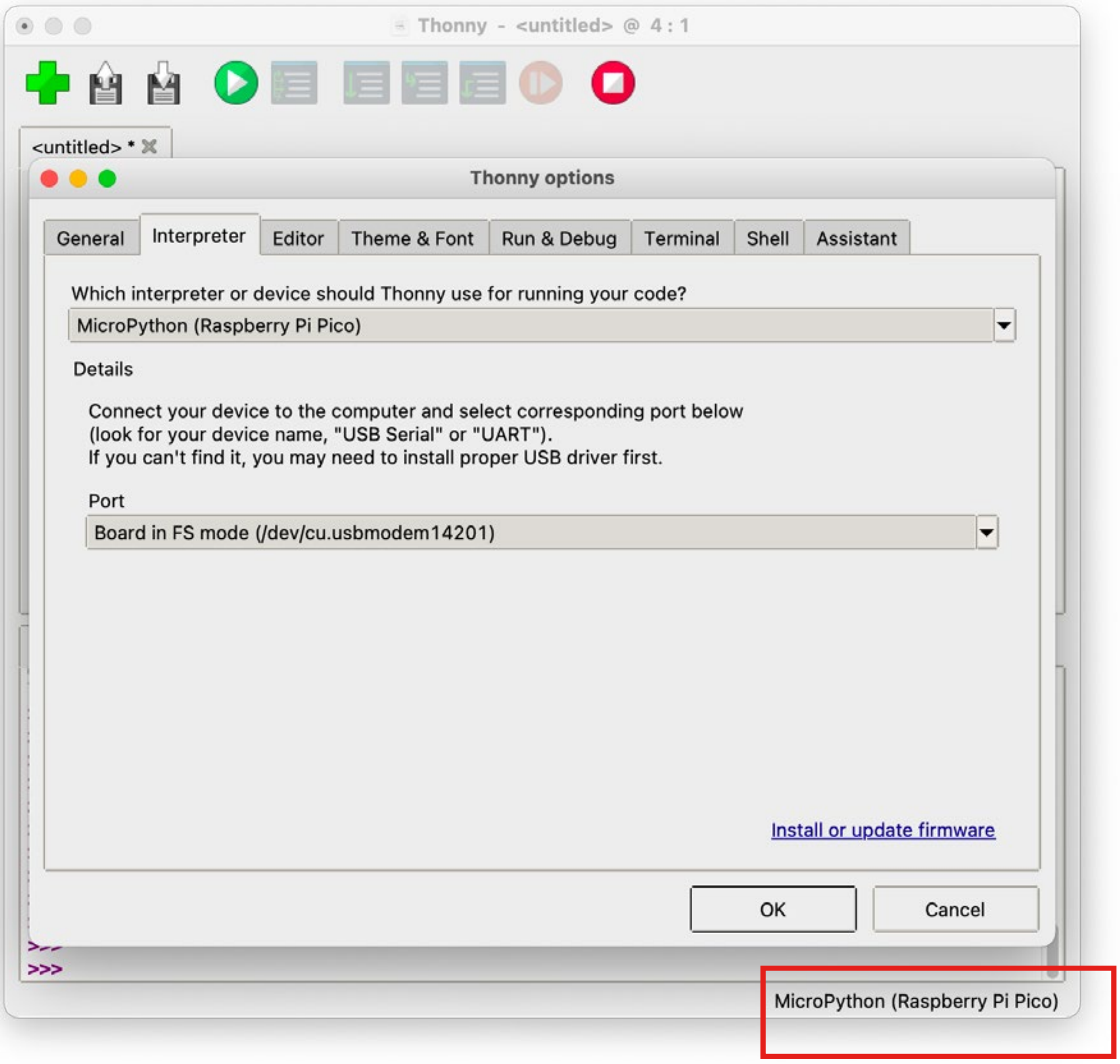
```
Melopero_Cookie_RP2040

iCloud                 Nome
Posizioni              > CircuitPython
RPI-RP2                > Flash reset
                       > Getting started guide
                         LICENSE
                       v MicroPython
                         v demo code
                           v color chase
                               main.py
                           v rainbow
                               main.py
                           v random pixel
                               main.py
                         v MicroPython installer
                             melopero_cookie_rp2040_micropython.uf2
                         README.md
```

5.2 Install Thonny editor

Thonny is a Python code editor for beginner programmers and it’s the recommended editor for programming in RP2040 based boards with MicroPython.
It’s available for Windows, Mac and Linux at the following address:
https://thonny.org

The first time you’ll run Thonny editor, you should set MicroPython on Raspberry Pi Pico and select the right port fo your connected device.
Go to Preferences (should be options/settings on windows), click “Interpreter” tab and then select Micropython (Raspberry Pi Pico) and the right port from the port menu (your board must be connected and with MicroPython already installed)
If you are using Thonny in simple mode, you can access the interpreter configuration panel clicking on the lower right corner of Thonny window (circled in red below).



Which interpreter or device should Thonny use for running your code?
MicroPython (Raspberry Pi Pico)

Details

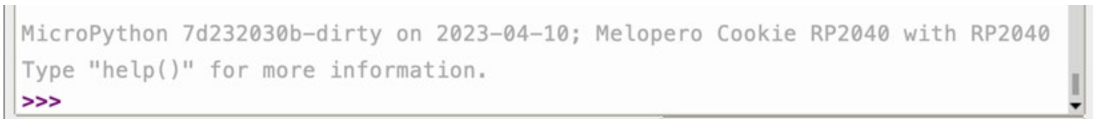
Connect your device to the computer and select corresponding port below
(look for your device name, "USB Serial" or "UART").
If you can't find it, you may need to install proper USB driver first.

Port
Board in FS mode (/dev/cu.usbmodem14201)

MicroPython (Raspberry Pi Pico)

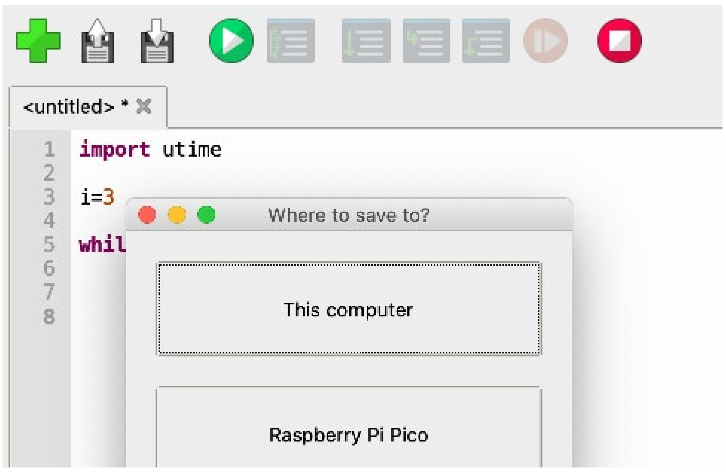
5.3 Thonny quick start

After installing MicroPython onto the Cookie, open Thonny and click on STOP/RESTART, you'll read the following text in the shell at the bottom of the window when the Cookie is ready to receive instructions:



```
MicroPython 7d232030b-dirty on 2023-04-10; Melopero Cookie RP2040 with RP2040
Type "help()" for more information.
>>>
```

Write some code in the script area of Thonny or open the micropython example code included in the Software pack (see chapter 3 to download the software pack).
When ready, click RUN or SAVE, Thonny will promt you to select where you want to run or save the script: select "Raspberry Pi Pico". In case you are saving the file, give it a name including the .py extension, for example myfile**.py** . If you save the file as main.py, your code will be executed automatically everytime you power the Cookie.
Now that your file has been saved, execute your program clicking on RUN (green PLAY button).
If your code includes some prints, you'll see them in the Shell.
Before saving a new file, remember to click on the "stop/restart" button if your Cookie is already running a program.



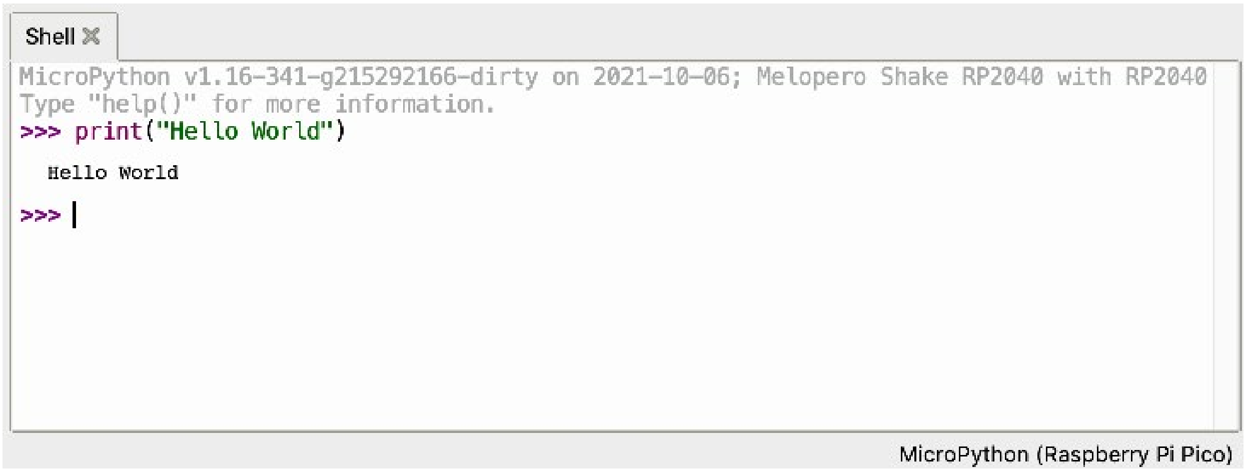
```
import utime

i=3

whil
```

5.4 The REPL

The REPL, Read-Evaluate-Print-Loop, allows you to execute lines of code directly in the console and get an immediate result.
In the Shell, try to run the command print("hello world") and press enter.
The REPL will interpret the line of code and get you the result, in this case it'll print "hello world".



```
MicroPython v1.16-341-g215292166-dirty on 2021-10-06; Melopero Shake RP2040 with RP2040
Type "help()" for more information.
>>> print("Hello World")

  Hello World

>>> |
```

5.5 MicroPython Library overview

The fastest way to start using the Cookie RP2040 is loading the main example, which includes comments to the code.
The MicroPython library uses the RP2040's PIO, DMA and interrupts to keep the main processor free as long as possible. For example, after calling **`show_message(...)`**, even though the text is still scrolling, the main processor is not blocked and the next code is executed.
The library includes functions to control the neopixels, user buttons and user LED:

```
#create a new cookie_rp2040 object
Cookie_RP2040()


#set a pixel to an rgb color
#pass as parameters the pixel position (0 to 24), red, green, blue values and
#brightness (optional)
#if not specified, the default brightness is 0.2
#after all the desired pixels have been set, call show_pixels() to write the
values on the screen
set_pixel(pixel,red,green,blue,brightness)


#set all the 25 neopixels at once
#pass as parameters red, green, blue values along with brightness (optional)
#if not specified, the default brightness is 0.2
#to write the pixels on screen, call show_pixels()
set_matrix(red,green,blue,brightness)


#write the pixels previously set on the screen
show_pixels()


#set the scrolling text color
#pass as parameters red, green, blue values along with brightness (optional)
#if not specified, the default brightness is 0.2
set_rgb_color(red,green,blue,brightness)


#set the scrolling text background color
#pass as parameters red, green, blue values along with brightness (optional)
#if not specified, the default brightness is 0.2
set_rgb_background(red,green,blue,brightness)


#scroll a message on the screen
show_message(message)

#stop a scrolling text before it has finished, without clearing the screen
stop_message()
```

```
#set the scrolling direction, the default is LEFT
#takes as parameter 0 for LEFT or 1 for RIGHT
set_direction(direction)

#enable auto re-start when scrolling a message
enable_repeated_start()

#disable auto re-start when scrolling a message
disable_repeated_start()

#clear the screen with a rgb color
#pass as parameters red, green, blue values along with brightness (optional)
#if not specified, the default brightness is 0.2
clear_screen(red,green,blue,brightness)

#clear the screen and turn off all the pixels
clear_screen()

#Initialize the user LED on GPIO21
led_init()

#Turn on the user LED
led_on()

#Turn off the user LED
led_off()

#change value to the user LED
led_toggle()

#Initialize Buttons A and B respectively on GPIO10 and GPIO11
buttons_init()

#read the button value (takes as parameter 10 or 11)
read_button(button)
```

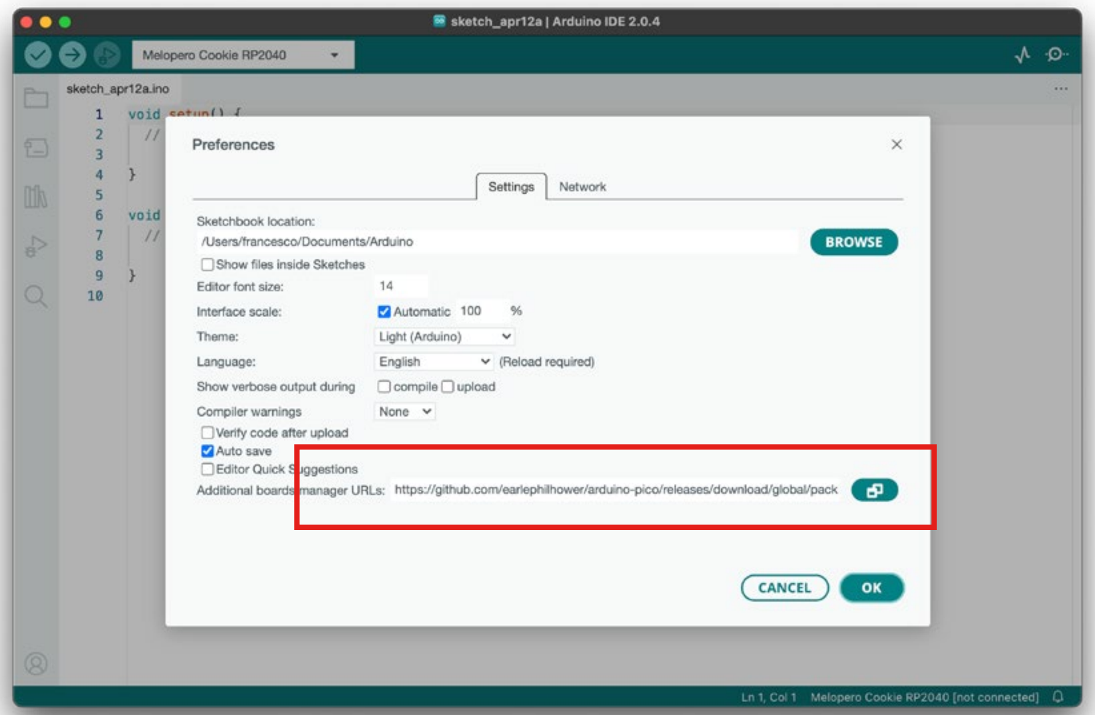
6. Set up the Arduino IDE 2.0

6.1 Download the Arduino IDE

To download the Arduino IDE for your favourit OS go to:
https://www.arduino.cc/en/software

6.2 Add the Cookie RP2040 board

We use the port of the RP2040 developed by Earle F. Philhower, III (earlephilhower on GitHub)
Go to File>Preferences and enter the following URL in the "Additional Boards Manager URLs" field:
https://github.com/earlephilhower/arduino-pico/releases/download/global/package_rp2040_index.json



sketch_apr12a | Arduino IDE 2.0.4

Melopero Cookie RP2040

sketch_apr12a.ino

```
void setup() {
  //
}

void
  //
}
```

Preferences

Settings | Network

Sketchbook location:
/Users/francesco/Documents/Arduino
BROWSE
☐ Show files inside Sketches

Editor font size:	14
Interface scale:	☑ Automatic 100 %
Theme:	Light (Arduino)
Language:	English (Reload required)
Show verbose output during	☐ compile ☐ upload
Compiler warnings	None

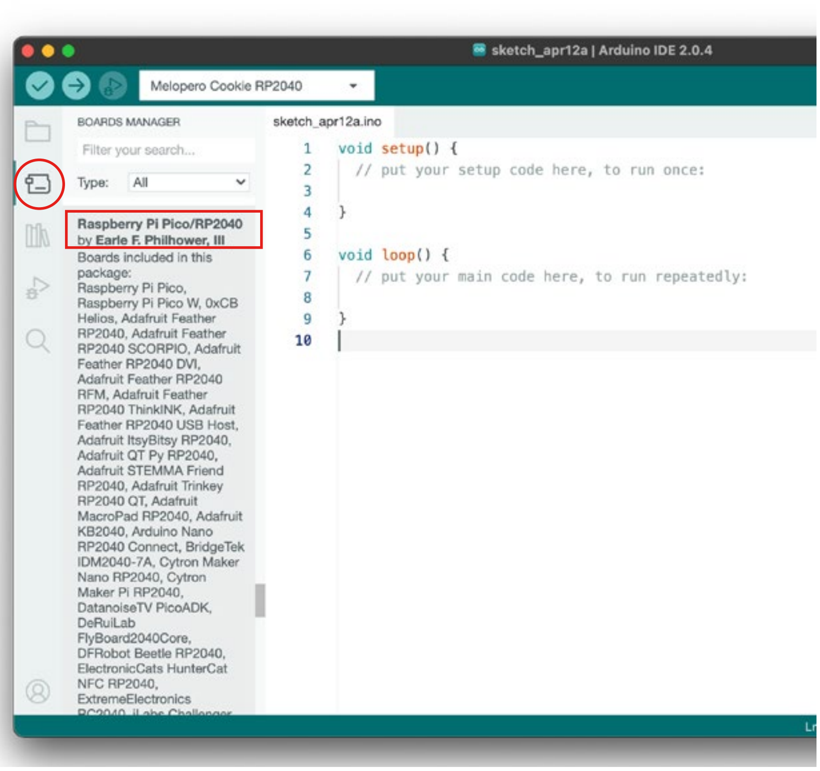
☐ Verify code after upload
☑ Auto save
☐ Editor Quick Suggestions

Additional boards manager URLs: https://github.com/earlephilhower/arduino-pico/releases/download/global/pack

CANCEL | OK

Ln 1, Col 1   Melopero Cookie RP2040 [not connected]

Once the URL with our board has been added, go to the main Arduino window, click on the boards manager icon on the left, scroll down untill you read **Raspberry Pi Pico/RP2040 by Earle F. Philhower, III** and click on Install at the end of the package description:



Raspberry Pi Pico/RP2040
by **Earle F. Philhower, III**
Boards included in this package:
Raspberry Pi Pico, Raspberry Pi Pico W, 0xCB Helios, Adafruit Feather RP2040, Adafruit Feather RP2040 SCORPIO, Adafruit Feather RP2040 DVI, Adafruit Feather RP2040 RFM, Adafruit Feather RP2040 ThinkINK, Adafruit Feather RP2040 USB Host, Adafruit ItsyBitsy RP2040, Adafruit QT Py RP2040, Adafruit STEMMA Friend RP2040, Adafruit Trinkey RP2040 QT, Adafruit MacroPad RP2040, Adafruit KB2040, Arduino Nano RP2040 Connect, BridgeTek IDM2040-7A, Cytron Maker Nano RP2040, Cytron Maker Pi RP2040, DatanoiseTV PicoADK, DeRuiLab FlyBoard2040Core, DFRobot Beetle RP2040, ElectronicCats HunterCat NFC RP2040, ExtremeElectronics

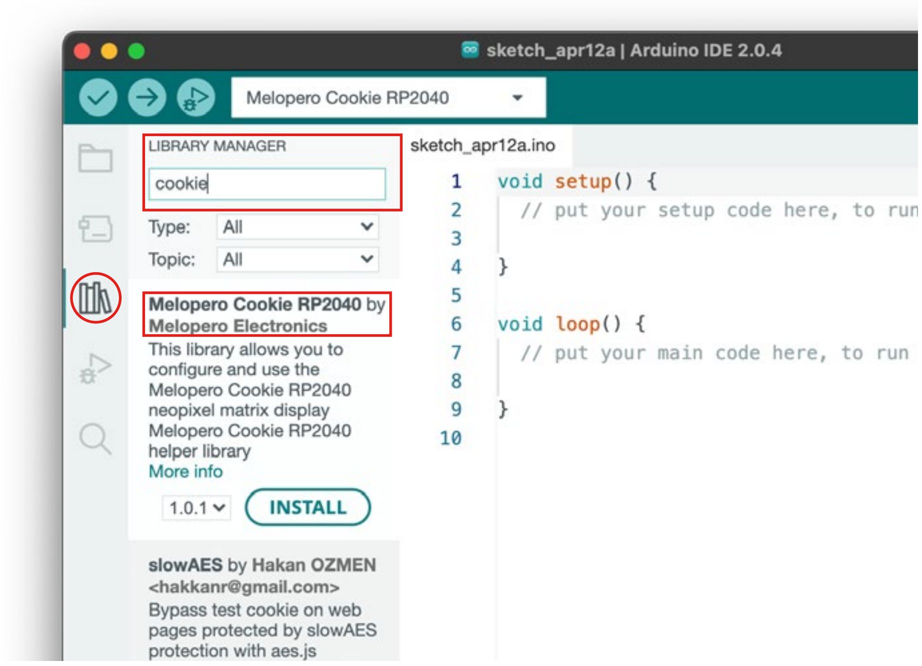
```
void setup() {
  // put your setup code here, to run once:

}

void loop() {
  // put your main code here, to run repeatedly:

}
```

Finally, install the Cookie RP2040 library, click on the library manager icon on the left, search for **Melopero Cookie** and click on Install:



Melopero Cookie RP2040 by **Melopero Electronics**
This library allows you to configure and use the Melopero Cookie RP2040 neopixel matrix display Melopero Cookie RP2040 helper library

slowAES by Hakan OZMEN <hakkanr@gmail.com>
Bypass test cookie on web pages protected by slowAES protection with aes.js

```
void setup() {
  // put your setup code here, to run

}

void loop() {
  // put your main code here, to run

}
```

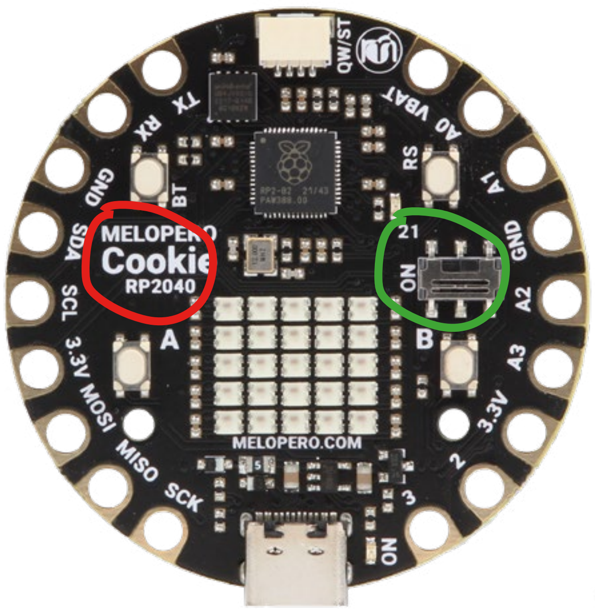
6.3 Prepare the Cookie for the upload

First of all, if you have previously installed MicroPython erase the falsh memory follwing the instruction in section 8.

In order to upload a sketch from the Arduino IDE, the Cookie must be in bootloader mode.
If no software has been installed onto the board, this should happen automatically when Cookie is connected.
If this not happen, you shall activate the bootloader mode manually.

To activate the bootloader mode, when the Cookie is already connected to your computer's USB port, press and hold the BOOT / BT button (circled in red in the image below), then press and release the reset button (circled in green). Continue holding the BOOT / BOOTSEL button until the RPI-RP2 drive appears.

After using the Cookie with Arduino IDE, if you want to go back programming it with MicroPython or CircuitPython, a flash memory reset is necessary, see section 8.



6.4 Select the port and upload a sketch

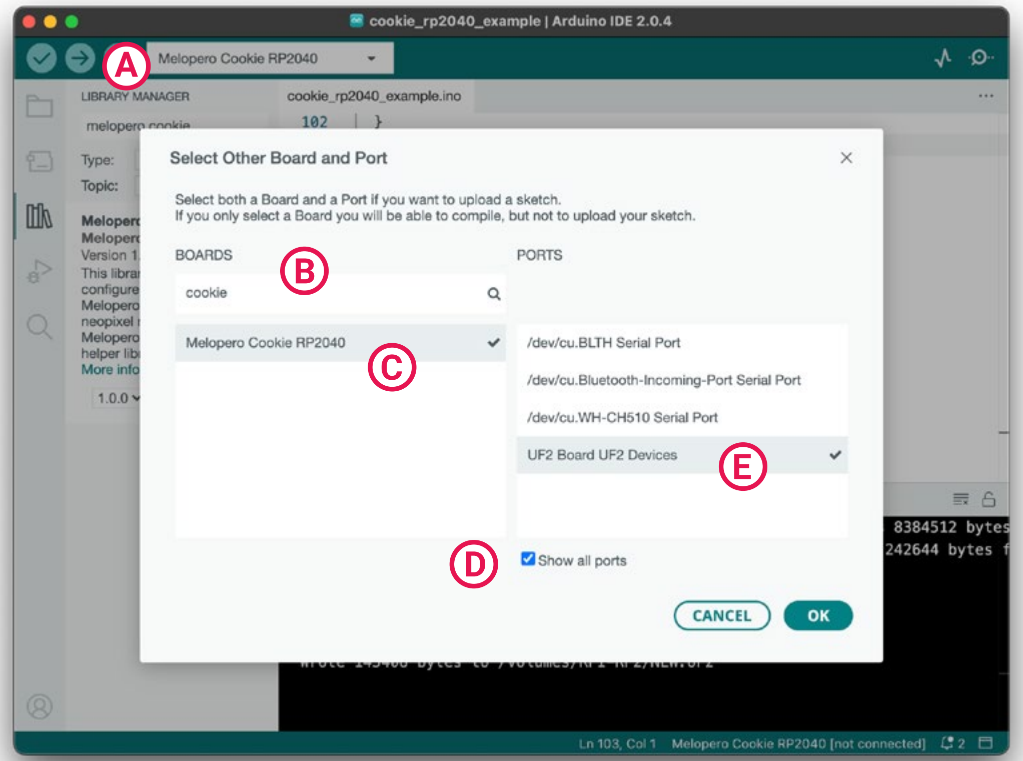
A Open the board menu and choose **Select Other Board and Port**

B Search for **Cookie RP2040**

C Select Melopero **Cookie RP2040**

D Check **Show all ports**

E Select **UF2 Board UF2 Devices** and click **OK**



Select Other Board and Port

Select both a Board and a Port if you want to upload a sketch.
If you only select a Board you will be able to compile, but not to upload your sketch.

BOARDS: cookie — Melopero Cookie RP2040 ✓

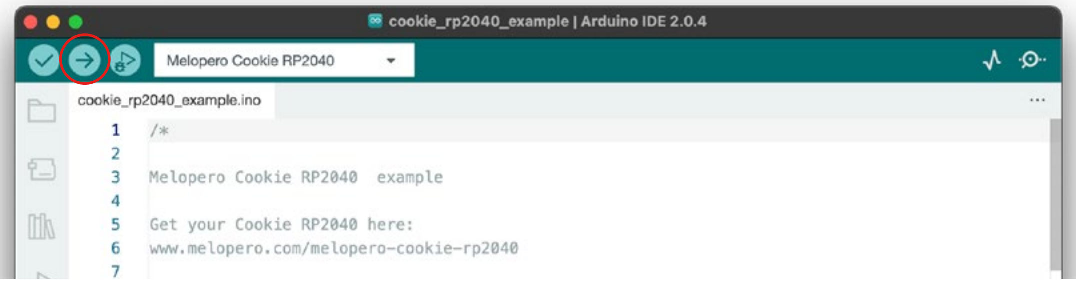
PORTS:
/dev/cu.BLTH Serial Port
/dev/cu.Bluetooth-Incoming-Port Serial Port
/dev/cu.WH-CH510 Serial Port
UF2 Board UF2 Devices ✓

☑ Show all ports

```
8384512 bytes
242644 bytes f
```

Ln 103, Col 1 Melopero Cookie RP2040 [not connected]

finally go to File->Examples, scroll down to Melopero Cookie RP2040 and open the co-okie_rp2040_example.ino, then click on upload


cookie_rp2040_example | Arduino IDE 2.0.4
Melopero Cookie RP2040
cookie_rp2040_example.ino

```
/*

Melopero Cookie RP2040  example

Get your Cookie RP2040 here:
www.melopero.com/melopero-cookie-rp2040

```

6.5 Arduino Library overview

```
//create a new cookie_rp2040 object
```
`Melopero_Cookie_RP2040()`

```
//set a pixel to an rgb color
//pass as parameters the pixel position (0 to 24), red, green, blue values and
//brightness (optional)
//if not specified, the default brightness is 0.3
//after all the desired pixels have been set, call showPixels() to write the //
values on the screen
```
`setPixel(int pix, uint8_t r, uint8_t g, uint8_t b, double brightness)`

```
//set all the 25 neopixels at once, passing an array of 25 colors.
//colors must be formatted as uin32_t, this can be done using formatColor(...)
//and passing as parameters r, g, b values along with brightness
```
`setMatrix(uint32_t pixels[25])`

```
//show all the previously set pixels on screen (use setMatrix or setPixels
//to set the desired pixels)#this functions must be called after setting the
pixels with set_pixel(...) or set_matrix(...)
```
`showPixels()`

```
//convert RGB + Brightness values to 32-bit neopixel format
//pass as parameters red, green, blue values along with brightness (optional)
//if not specified, the default brightness is 0.3
//return a uint32_t
```
`formatColor(uint8_t r, uint8_t g, uint8_t b, double brightness)`

```
//set the scrolling text color
//pass as parameters red, green, blue values along with brightness (optional)
//if not specified, the default brightness is 0.3
```
`setRgbColor(uint8_t r, uint8_t g, uint8_t b, double brightness)`

```
//set the scrolling text background color
//pass as parameters red, green, blue values along with brightness (optional)
//if not specified, the default brightness is 0.3
```
`setRgbBackground(uint8_t r, uint8_t g, uint8_t b, double brightness)`

```
//Show a message on the matrix display, if not specified, the default scroll
//delay is 200ms.
//Default scrolling direction is LEFT, to change it use setDirection()
```
`showMessage(String &message, uint32_t timeMS);`

```
//stop a scrolling text before it has finished, without clearing the screen
```
`stopMessage()`

```
//set the scrolling direction, the default is LEFT
```
`setDirection(Direction direction);`

```
//enable or disable auto re-started scrolling text
```
`setRepeatedStart(bool enable)`

```
//clear the screen with a 32-bit color
//colors must be formatted as uin32_t, this can be done using formatColor(...)
//and passing as parameters r, g, b values along with brightness
```
`clearScreen(uint32_t background)`

```
//clear the screen and turn off all the pixels
```
`clearScreen()`

```
//initialize the built-in LED on pin 21, this function has the same result as
//pinMode(21, OUTPUT)
```
`ledInit()`

```
//turn on the built-in LED on pin 21, this function has the same result as
//digitalWrite(21, HIGH)
```
`ledOn()`

```
//turn off the built-in LED on pin 21, this function has the same result as
//digitalWrite(21, LOW)
```
`ledOff()`

```
//change value to the user LED to the opposite
```
`ledToggle()`

```
//initialize the A B user buttons on pins 10 and 11, this function has the
//same result as pinMode(10, INPUT) and pinMode(11, INPUT)GPIO11
```
`buttonsInit()`

```
//read the current value on the specified pin, this function has the same
//result as digitalRead(10) and digitalRead(11)
```
`read_button(button)`

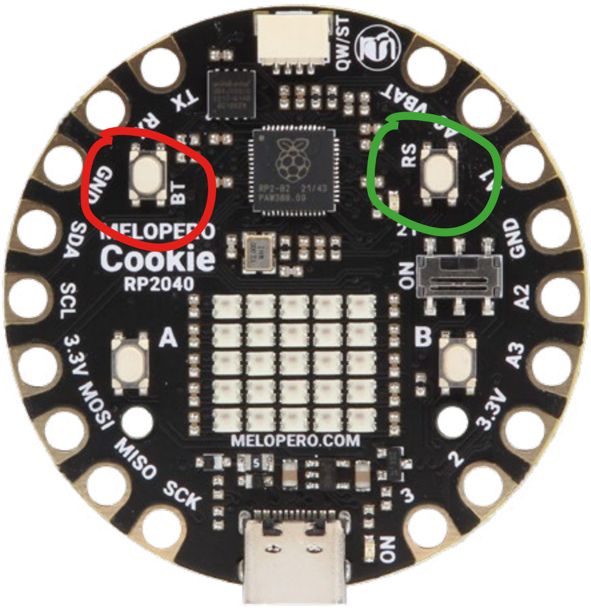
7. Install CircuitPython

CircuitPython is a derivative of MicroPython designed to simplify experimentation and education on low-cost microcontrollers. Simply copy and edit files on the CIRCUITPY drive to iterate. CircuitPython is developed and maintained by Adafruit Industries, along with many sensors libraries to start your project in the blink of an eye.

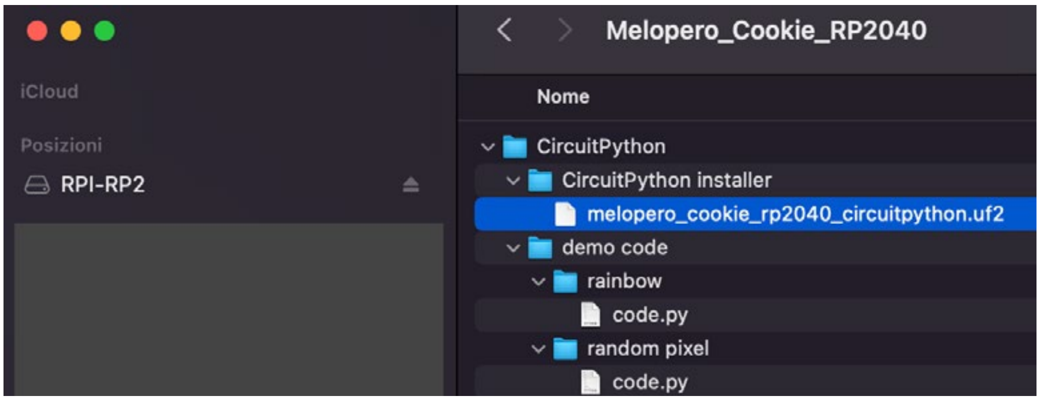
After downloading the latest version of Cookie software pack (see section 3 of this guide), activate the bootloader mode on your Cookie (a drive called RPI-RP2 will appear on your computer) and copy the circuitpython .uf2 file to it.

To activate the bootloader mode, when the Cookie is already connected to your computer's USB port, press and hold the BOOT / BT button (circled in red in the image below), then press and release the reset button (circled in green). Continue holding the BOOT / BOOTSEL button until the RPI-RP2 appears.

You can also start with your board unplugged from USB, press and hold the BT button on your Cookie RP2040. While holding the button, connect the other end of the USB cable to the Cookie board. This will cause Cookie to load his bootloader. You should see RPI-RP2 appearing as a new drive on your computer.



Drag and drop (or copy and paste) melopero_cookie_rp2040_circuitpython.uf2 file to RPI-RP2. The RPI-RP2 drive will disappear and after a few seconds a new disk drive called CIRCUITPY will appear.



Melopero_Cookie_RP2040

iCloud

Posizioni

RPI-RP2

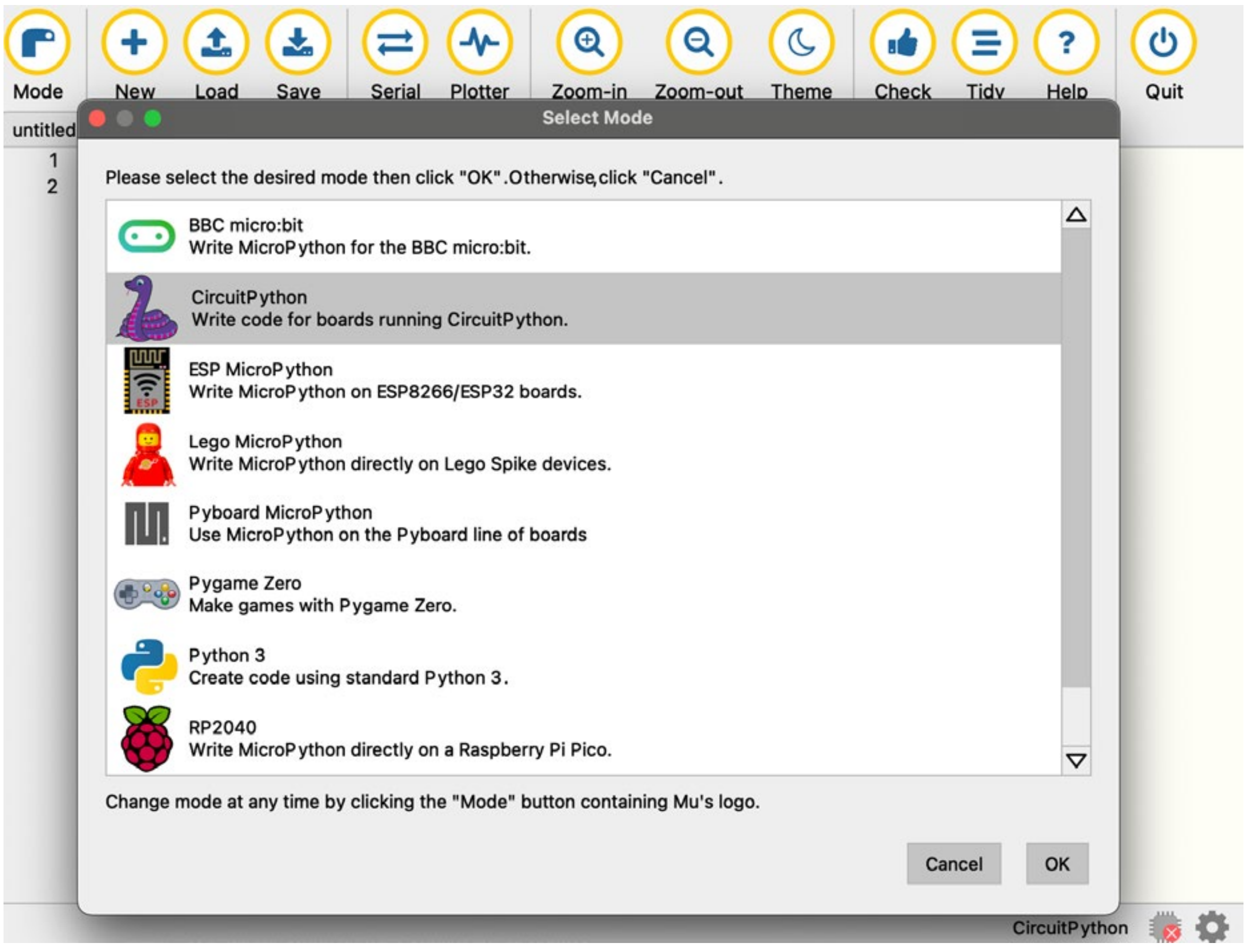
Nome

- CircuitPython
  - CircuitPython installer
    - melopero_cookie_rp2040_circuitpython.uf2
  - demo code
    - rainbow
      - code.py
    - random pixel
      - code.py

7.1 Install Mu editor

Mu is a Python code editor for beginner programmers and it's the recommended editor for programming in CircuitPython.
It's available for Windows, Mac and Linux at the following address:
https://codewith.mu/en/download

The first time you'll run Mu editor, it should ask for which mode you want to load.
Select "CircuitPython". You can always change mode by clicking on "mode" in the upper menu and selecting your favourite one.



Please select the desired mode then click "OK". Otherwise, click "Cancel".

BBC micro:bit
Write MicroPython for the BBC micro:bit.

CircuitPython
Write code for boards running CircuitPython.

ESP MicroPython
Write MicroPython on ESP8266/ESP32 boards.

Lego MicroPython
Write MicroPython directly on Lego Spike devices.

Pyboard MicroPython
Use MicroPython on the Pyboard line of boards

Pygame Zero
Make games with Pygame Zero.

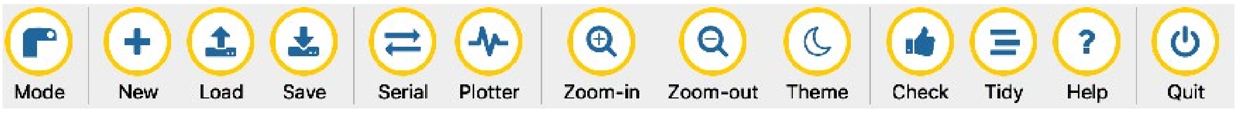
Python 3
Create code using standard Python 3.

RP2040
Write MicroPython directly on a Raspberry Pi Pico.

Change mode at any time by clicking the "Mode" button containing Mu's logo.

7.2 Mu quick start

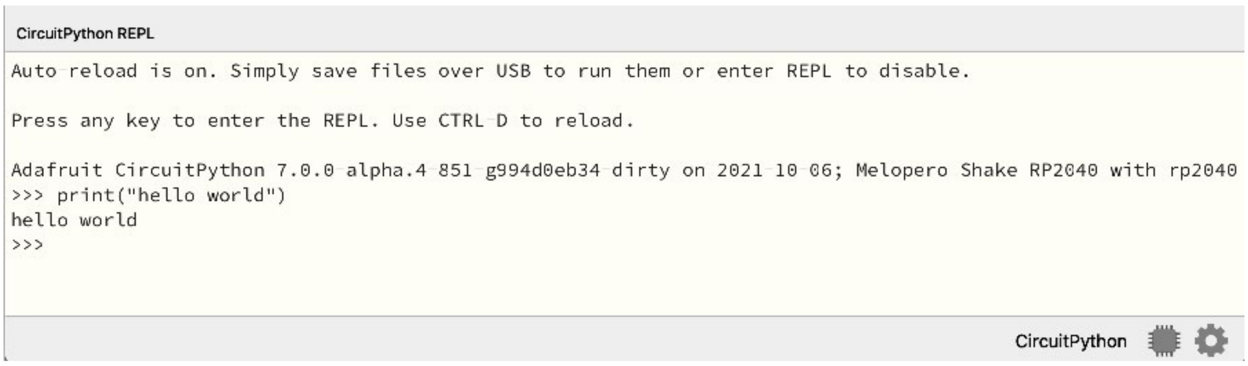
Mu will auto-detect your CircuitPython board.
On the upper menu, clicking “new”, you’ll create a new file.
Once you have connected the Melopero Cookie RP2040, click “Load“, then select the CIRCUITPY driver, and open code.py. After editing this file, click “Save“, and it’ll be loaded on your board.
The script should run automatically, otherwise click on the RESET button.



5.3 The REPL

The REPL, Read-Evaluate-Print-Loop, allows you to execute lines of code directly in the console and get an immediate result.
Click “Serial“ to open the serial console and then press any key to enter the REPL.
Use CTRL-D to reload.

Try to run the command print(“hello world“) and press enter.
The REPL will interpret the line of code and get you the result, in this case it’ll print “hello world”.



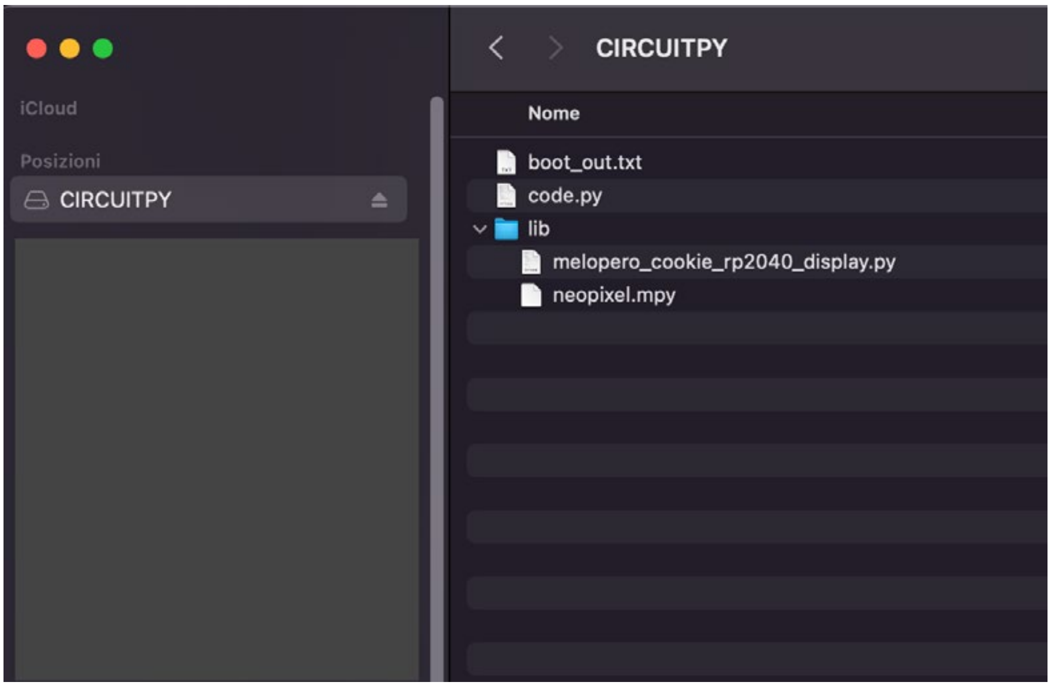
```
Auto reload is on. Simply save files over USB to run them or enter REPL to disable.

Press any key to enter the REPL. Use CTRL D to reload.

Adafruit CircuitPython 7.0.0 alpha.4 851 g994d0eb34 dirty on 2021 10 06; Melopero Shake RP2040 with rp2040
>>> print("hello world")
hello world
>>>
```

7.3 How to run demo code for Circuitpyhon

In order to run circuitython demo code, you must copy the realtive demo **code.py** file in the root directory of your Cookie and the library files into the **/lib** directory, as shown in the picture below. After copying all the files, if the demo doesn't start automatically, push the reset button once.

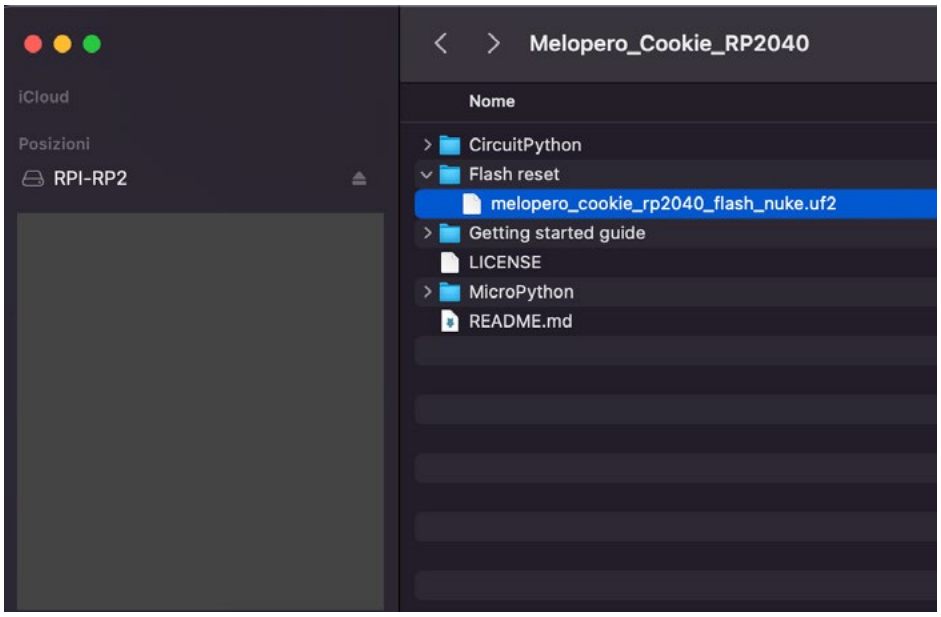


```
boot_out.txt
code.py
lib
    melopero_cookie_rp2040_display.py
    neopixel.mpy
```

8. Clear the Flash memory

If you need to do a deep clean of the flash memory, active the bootloader mode on the Co-okie and drag and drop (or copy and paste) the flash nuke file available in the Cookie Software pack (see chapter 3 to downalod it) onto the RPI-RP2 drive as you did for CircuitPython or Mi-croPython (see sections 5.1 and 4):

To activate the bootloader mode, when the Cookie is already connected to your computer's USB port, press and hold the BOOT / BT button (circled in red in the image below), then press and release the reset button (circled in green). Continue holding the BOOT / BOOTSEL button until the RPI-RP2 drive appears.



CircuitPython
Flash reset
melopero_cookie_rp2040_flash_nuke.uf2
Getting started guide
LICENSE
MicroPython
README.md