Getting Started with Raspberry Pi Pico 2 and Pico 2 W

Raspberry Pi Pico 2 W Pinout

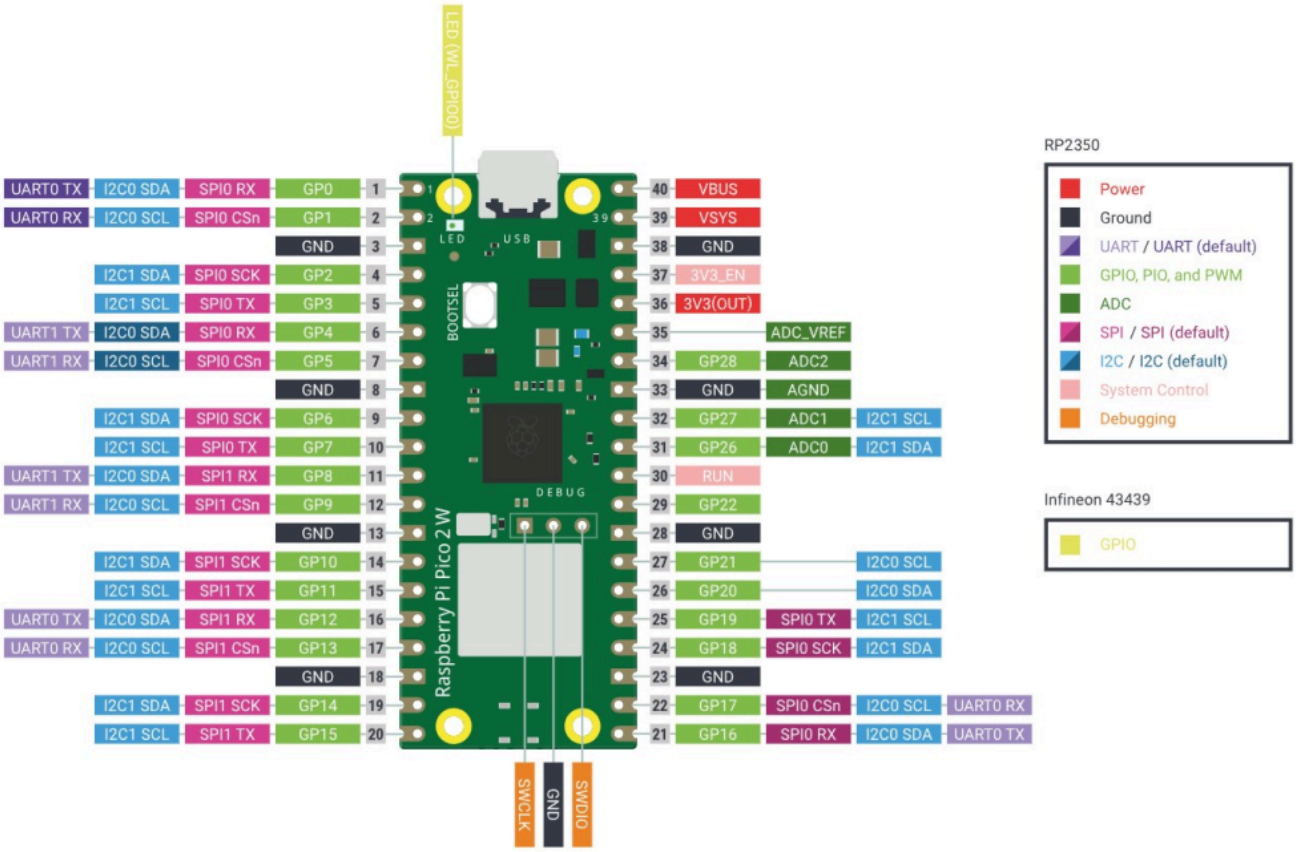


Image source: raspberrypi.com

The pins marked in red are power pins that output 3.3V or 5V. The black pins are GND pins. All pins in light green can be used as “regular” GPIOs (input and output), and all can generate PWM signals.

SPI, I2C, UART communication protocols, and ADC are supported on the pins with the corresponding labels.

Programming Raspberry Pi Pico 2 with MicroPython

MicroPython is a Python 3 programming language re-implementation targeted for microcontrollers and embedded systems. MicroPython is very similar to regular Python. Apart from a few exceptions, the language features of Python are also available in MicroPython. The most significant difference between Python and MicroPython is that MicroPython was designed to work under constrained conditions.

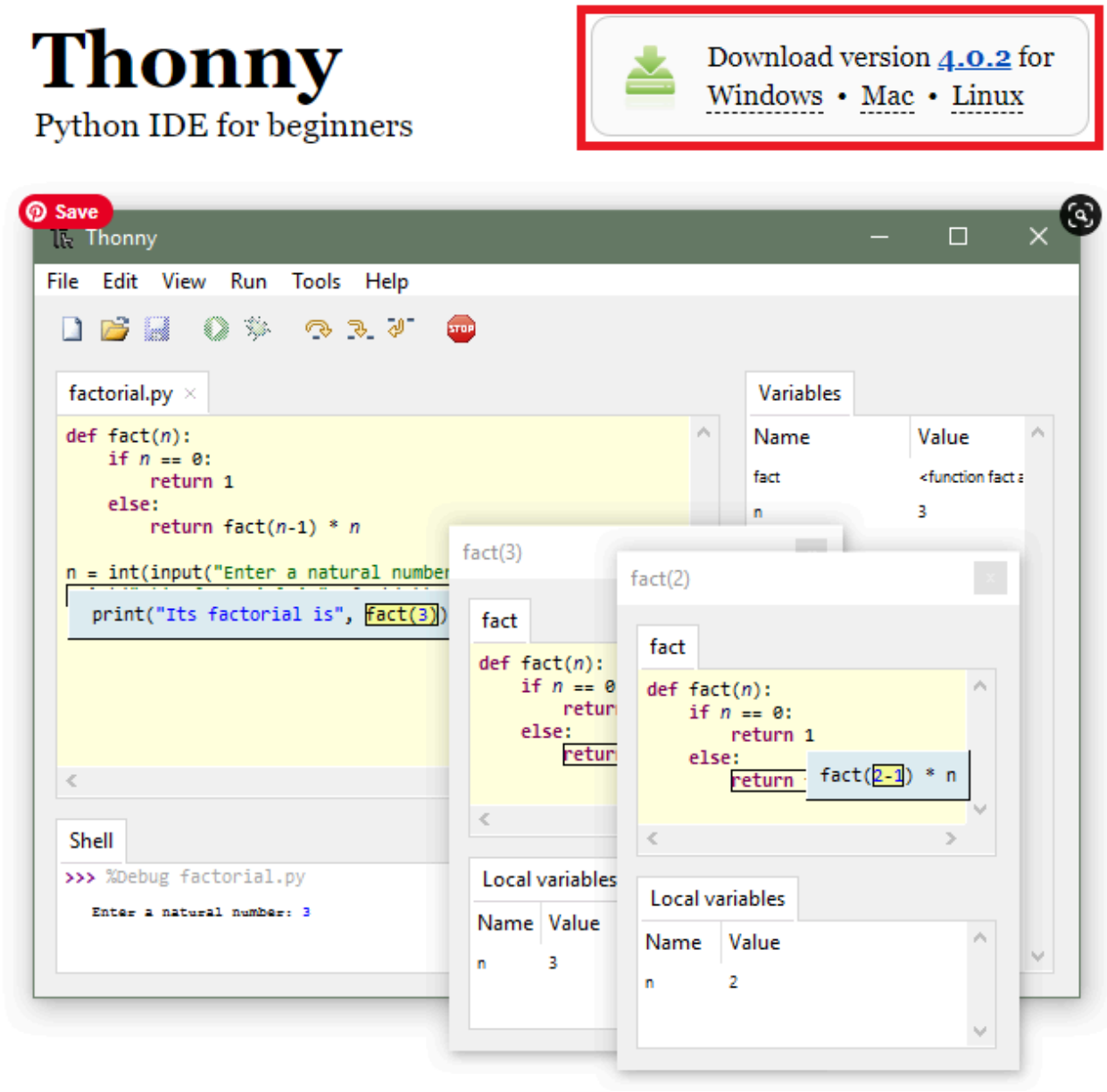
Because of that, MicroPython does not come with the entire pack of standard libraries. It only includes a small subset of the Python standard libraries, but it includes modules to easily control and interact with the GPIOs, use Wi-Fi, and other communication protocols.

Installing Thonny IDE

Thonny IDE is the recommended software to program the Raspberry Pi Pico board using MicroPython, and that's the one we'll use in this tutorial.

> Thonny IDE can also be used to program the ESP32 and ESP8266 using MicroPython firmware. Learn more here: Getting Started with Thonny MicroPython (Python) IDE for ESP32 and ESP8266.

Go to this link https://thonny.org/ and click on the right link to download the package for your operating system.



Run the downloaded executable file and follow the installation procedure (use all the default settings).

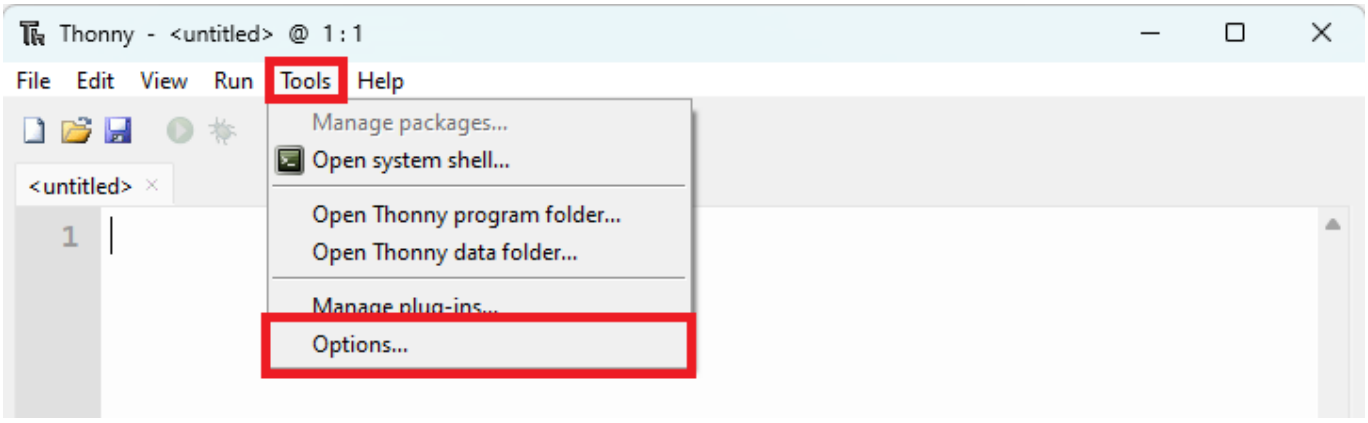
Flashing MicroPython Firmware on Raspberry Pi Pico

Connect the Raspberry Pi Pico to your computer while holding the BOOTSEL button at the same time so that it goes into bootloader mode to flash the firmware.

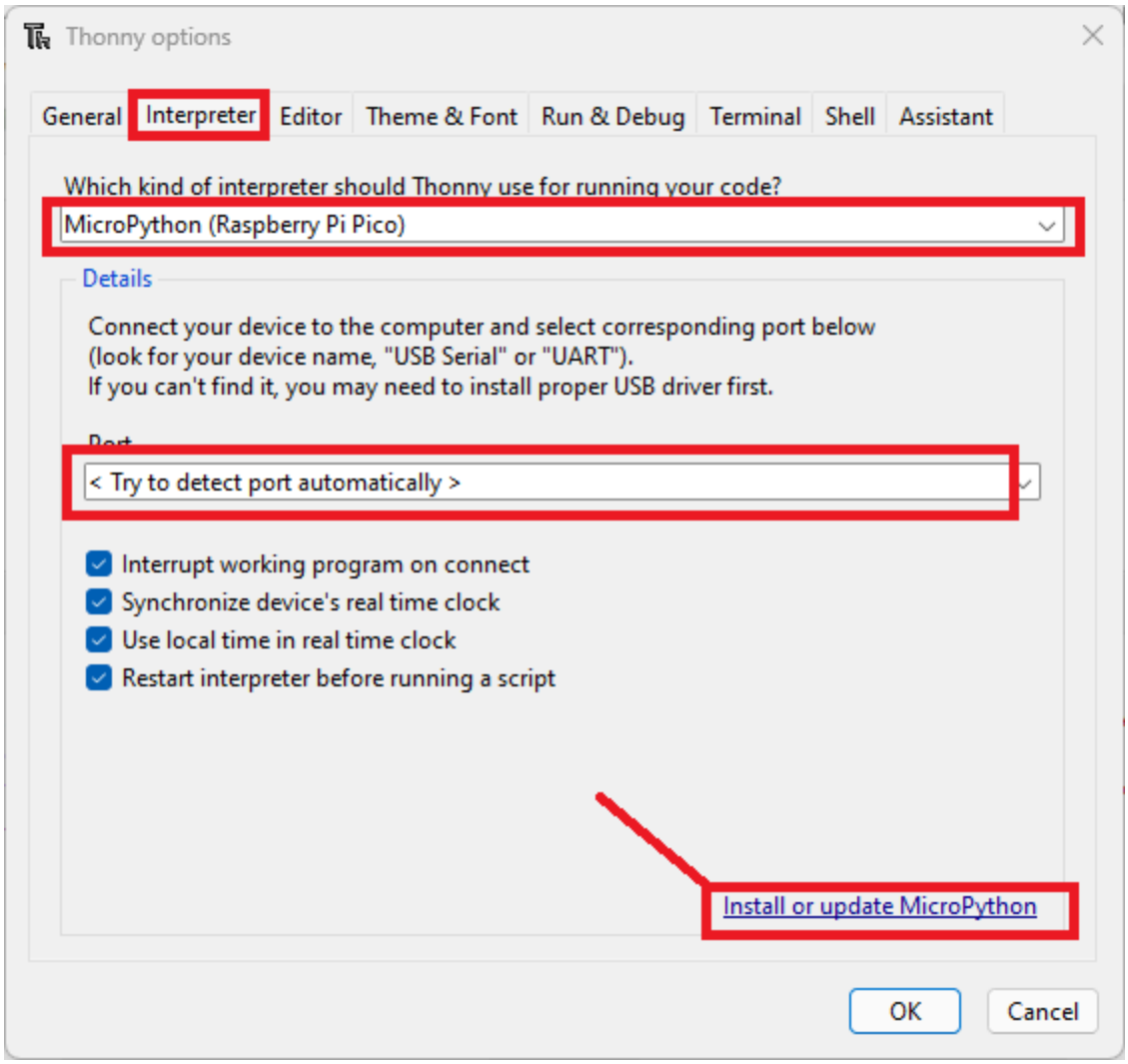
It will open a new window on your computer as a new USB device, you can ignore and close that window.

Open Thonny IDE.

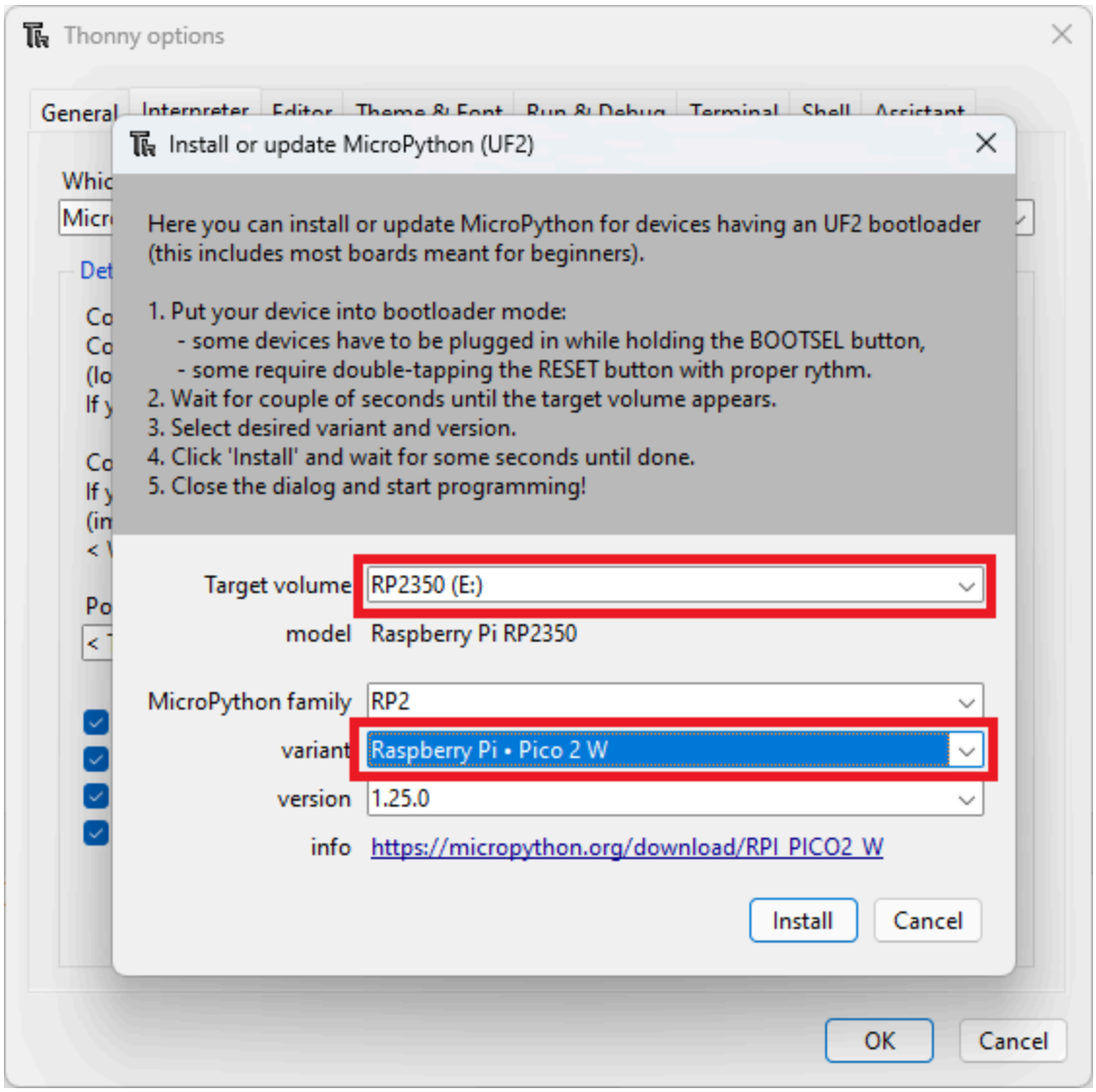
Go to **Tools** > **Options**. Select the **Interpreter** tab on the new window that opens.



Select MicroPython (Raspberry Pi Pico) for the interpreter, and the **Try to detect port automatically** for the Port. Then, click on Install or update MicroPython.



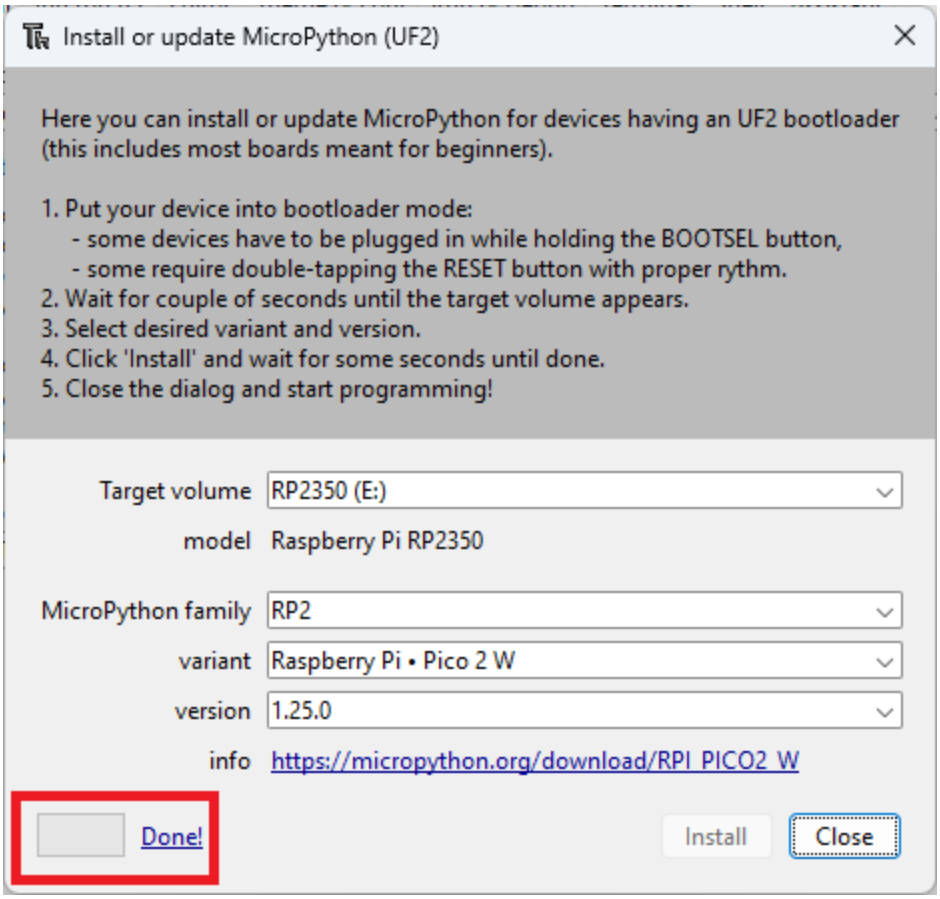
The following window will open.



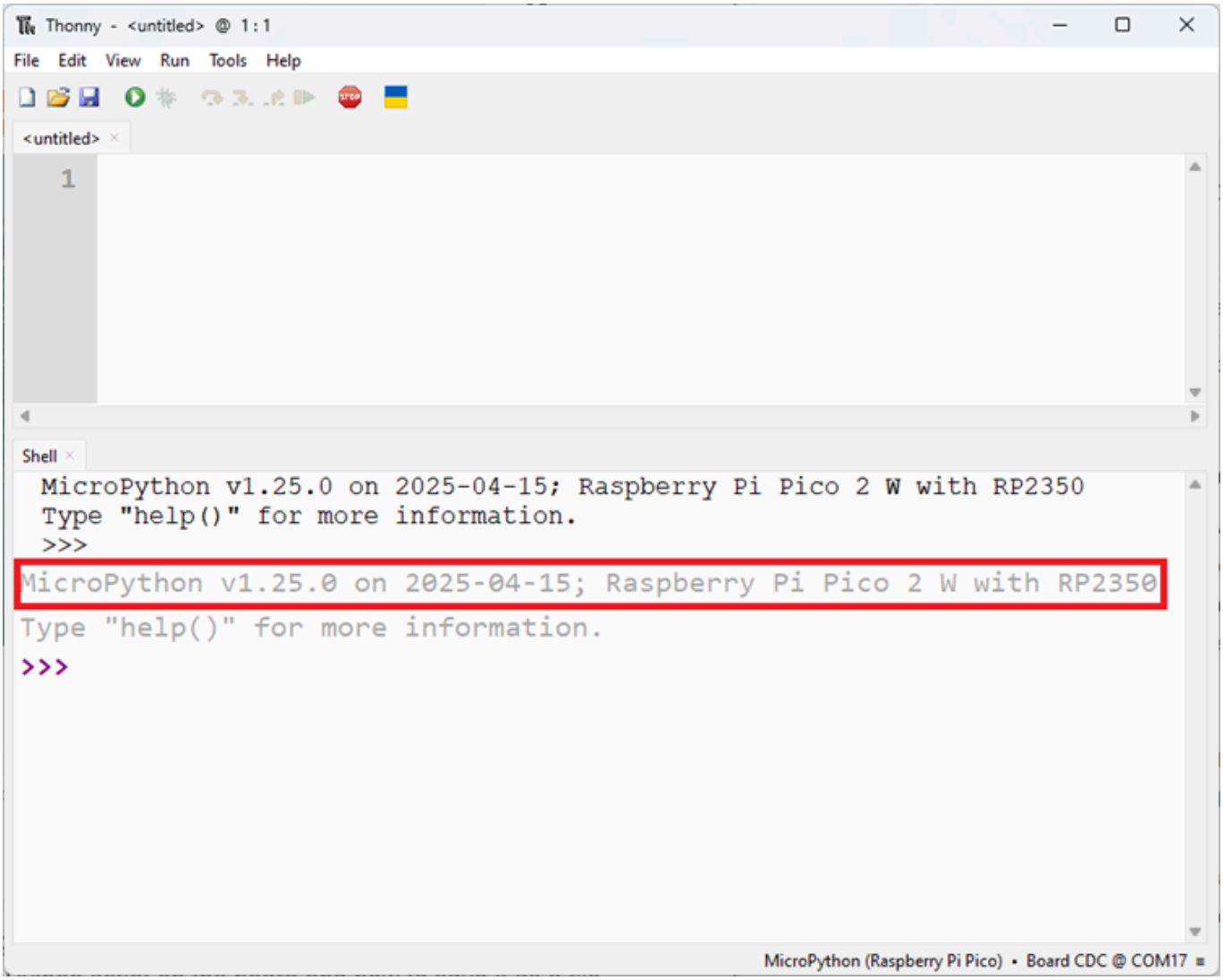
Select the MicroPython variant accordingly to the board you're using. In our case, we're using the Pico 2 W. Select a different option if you're using the Pico 2 (without W).

Finally, click **Install**.

After a few seconds, the installation should be completed.



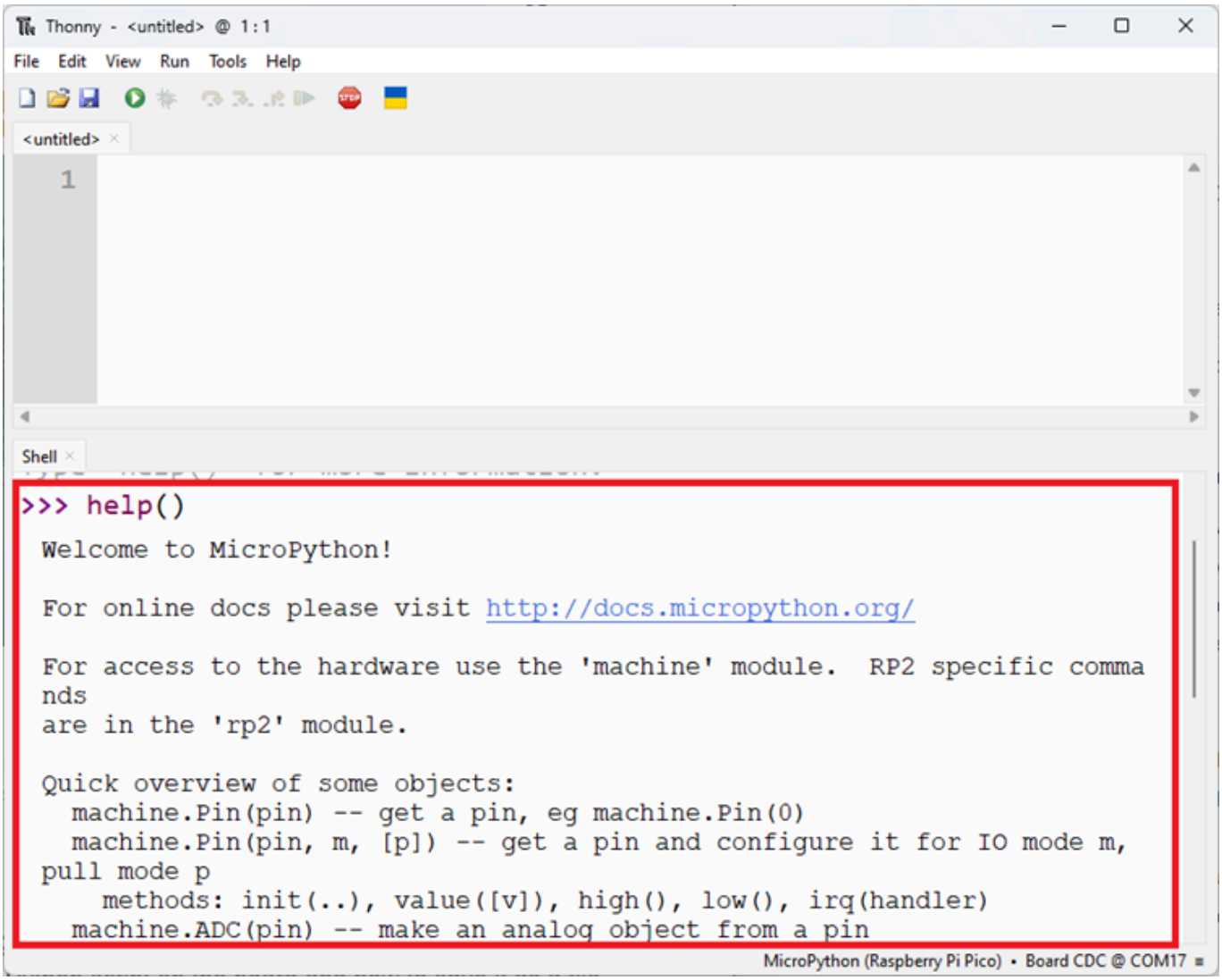
You can close the window. You should get the following message on the Shell, and at the bottom right corner, it should have the Interpreter it’s using and the COM port.



If it doesn't recognize the device, and you have a message saying "no backend" in the bottom-right corner, it may be the case that it is not detecting the COM port automatically. If that's the case, go to **Tools** > **Options** > **Interpreter** and select the COM port manually. After that, it should recognize the MicroPython device, and you should get the previous message.

Testing the Installation

Type `help()` in the Shell and see if your device responds back.



It should print some useful information about programming the Raspberry Pi Pico board using MicroPython.

Type the following:

```
print('Hello')
```

It should print Hello on the Shell.

```
>>> print('Hello')
  Hello
```

If you've followed until this step, you should have Thonny IDE successfully installed, as well as MicroPython firmware on your Raspberry Pi Pico board.

Running Your First Script

To get you familiar with the process of writing a file and executing code on the Raspberry Pi Pico 2 board, we'll upload a new script that simply blinks the on-board LED.

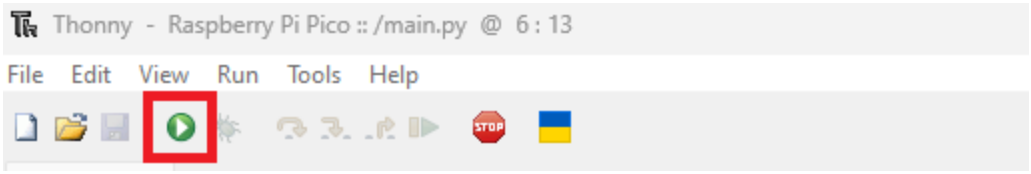
Blinking an LED Script

When you open Thonny IDE for the first time, the Editor shows an untitled file. Copy one of the following scripts to that file:

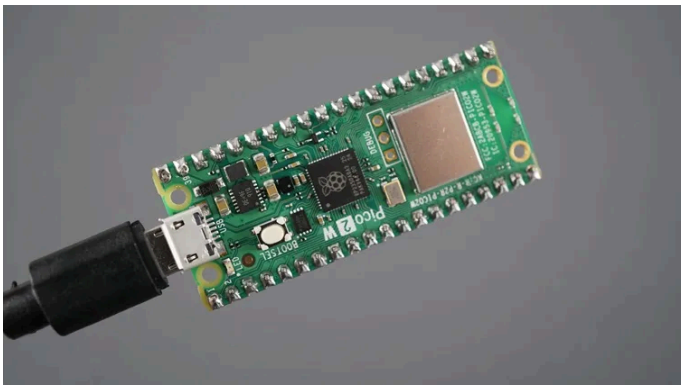
```
from machine import Pin
from time import sleep
led = Pin('LED', Pin.OUT)
while True:
  led.value(not led.value())
  sleep(0.5)
```

Running the Script

To run that script on your board, simply click on the **Run** icon in Thonny IDE.



The onboard LED will start blinking.



To stop the execution of the program you can hit the STOP button or simply press **CTRL**+**C**.

Important note: just running the file with Thonny doesn't copy it permanently to the board's filesystem. This means that if you unplug it from your computer and apply power to the board, nothing will happen because it doesn't have any Python file saved on its filesystem. The Thonny IDE *Run* function is useful to test the code, but if you want to upload it permanently to your board, you need to create and save a file to the board filesystem. See the next section.